Parcial practico

Sofia Bello Dueñas & Julieta Ruiz Castiblanco

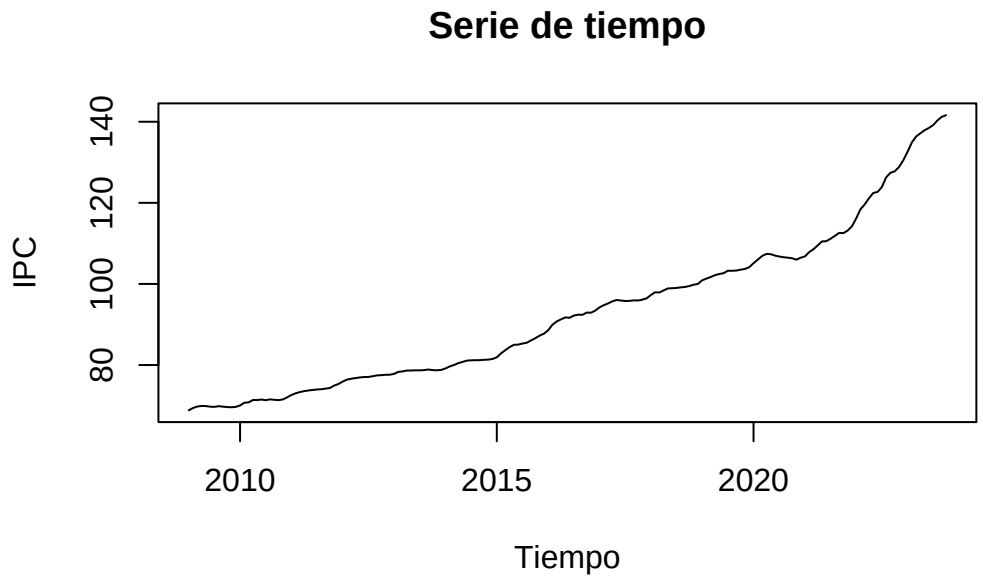
```
# Librerías
library(TSstudio)
library(readxl)
library(dplyr)
library(lubridate)
library(astsa)
library(feasts)
library(fable)
library(timetk)
library(tsibble)
library(zoo)
library(xts)
library(tidyverse)
library(forecast)
library(plotly)
library(urca)
library(forecast)
library(tseries)
library(lmtest)
library(uroot)
library(fUnitRoots)
library(tsoutliers)
library(astsa)
library(feasts)
library(fable)
library(timetk)
library(tsibble)
library(zoo)
library(xts)
library(readxl)
library(tidyverse)
```

```r
library(forecast)
library(plotly)

# Cargar los datos
datos<-read_excel("C:/Users/LENOVO/Desktop/Universidad/DatosParcial.xlsx")

# Datos de Barranquilla
datos<-datos[,c(1,3)]
```

Análisis Descriptivo

A continuación se presenta el gráfico de la serie de tiempo, aparentemente presenta tendencia.

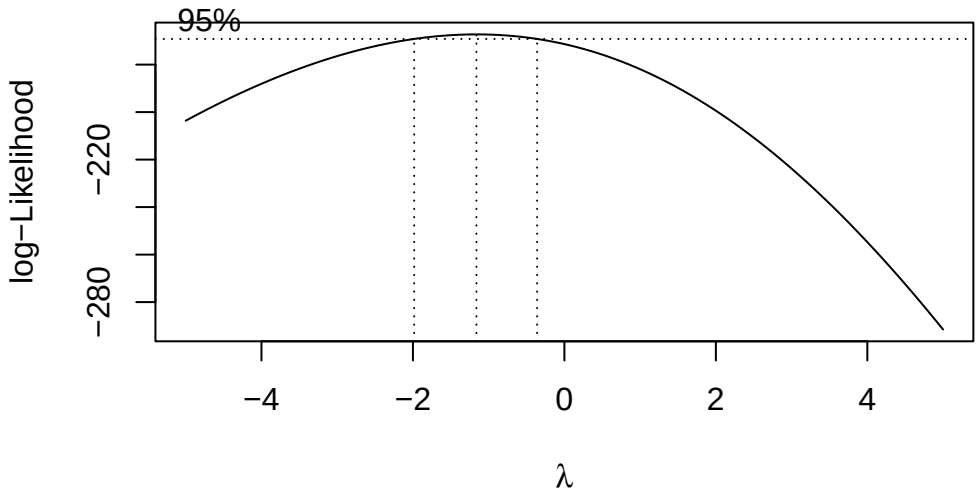
```r
# Pasar a objeto ts
datos1<-ts(datos$Barranquilla,start=c(2009,01),frequency=12)
plot(datos1, main="Serie de tiempo",ylab="IPC",xlab="Tiempo")
```



Estabilización de la varianza

Aunque la serie no parezca tener varianza marginal no constante, se realiza una prueba para saber si es necesario estabilizarla. En la siguiente gráfica se puede ver que no se captura el 1 y que el λ sugerido es -1 por lo que se hará una transformación logarítmica a la serie.

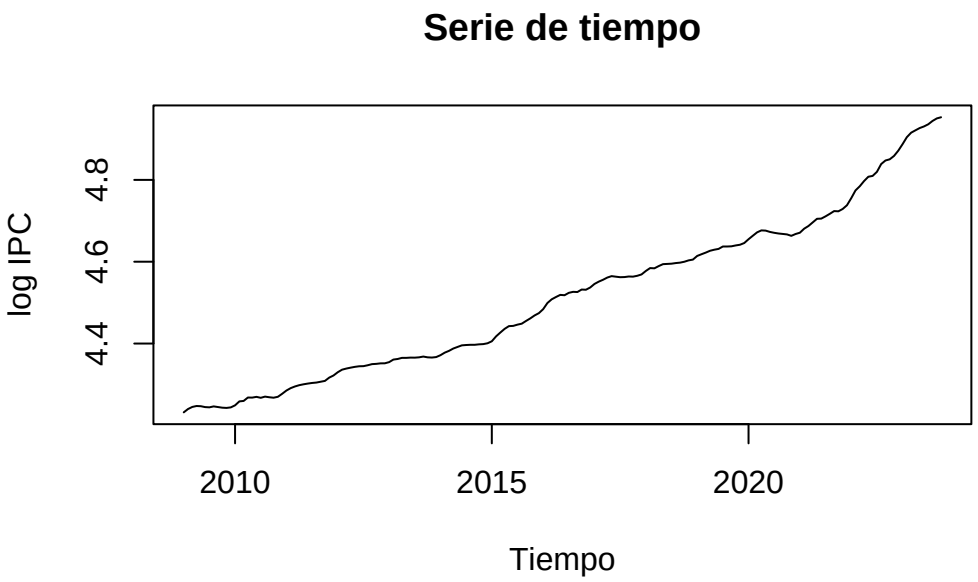
```
MASS::boxcox(lm(datos1 ~ 1),seq(-5, 5, length = 50))
```




```
forecast::BoxCox.lambda(datos1, method ="loglik", lower = -1, upper = 3)
```

```
[1] -1
```

```
# Transformación logarítmica a la serie
datos1=log(datos1)
plot(datos1,main="Serie de tiempo",ylab="log IPC",xlab="Tiempo")
```



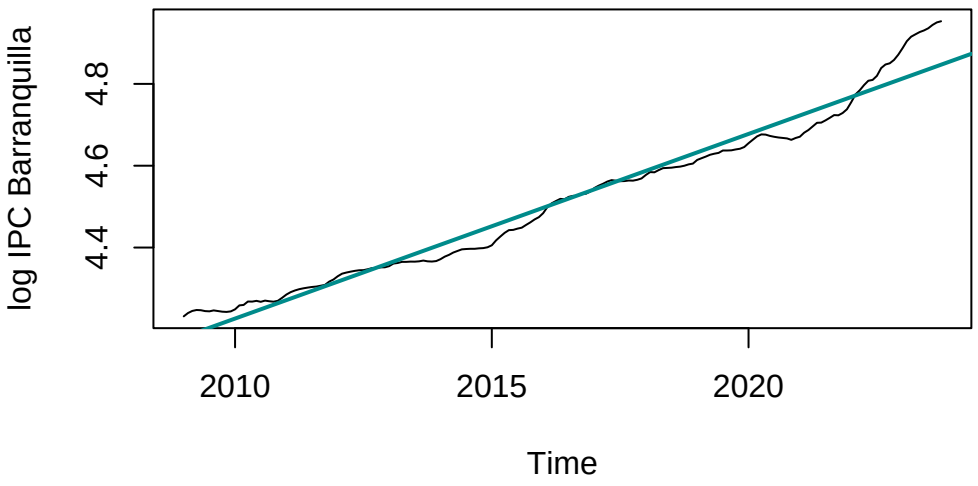
Eliminación de la tendencia

Estimamos la tendencia de diferentes formas como se puede ver a continuación

Regresión lineal

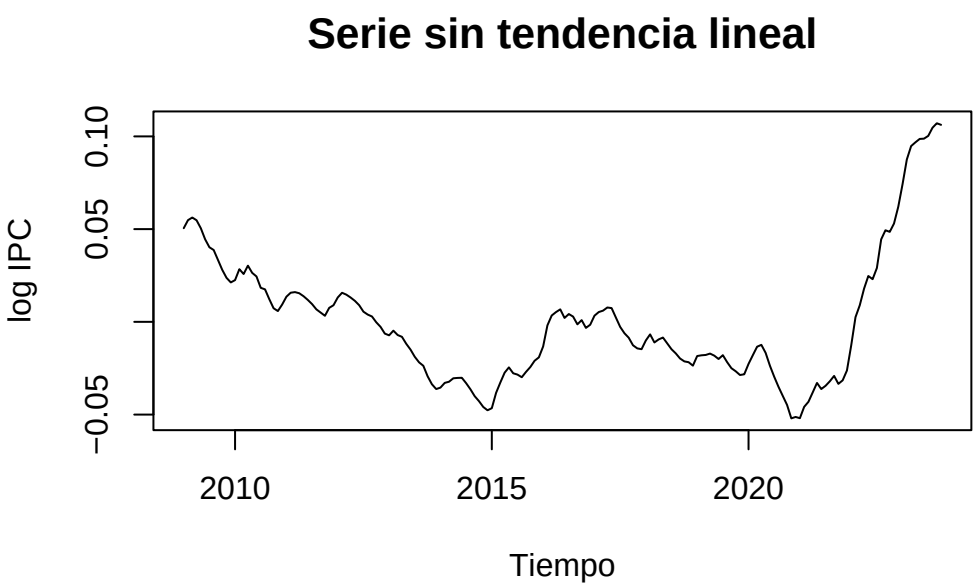
En la siguiente gráfica se puede ver la estimación de la tendencia con regresión lineal, se puede notar que no estima bien la tendencia dado que no se ajusta bien la tendencia al final de la serie.

```
fit<-lm(datos1~time(datos1),na.action=NULL)
plot(datos1,ylab="log IPC Barranquilla")
abline(fit,col="darkcyan",lwd=2)
```



El siguiente es el gráfico de la serie sin tendencia usando regresión lineal. Vemos que la serie aún presenta tendencia.

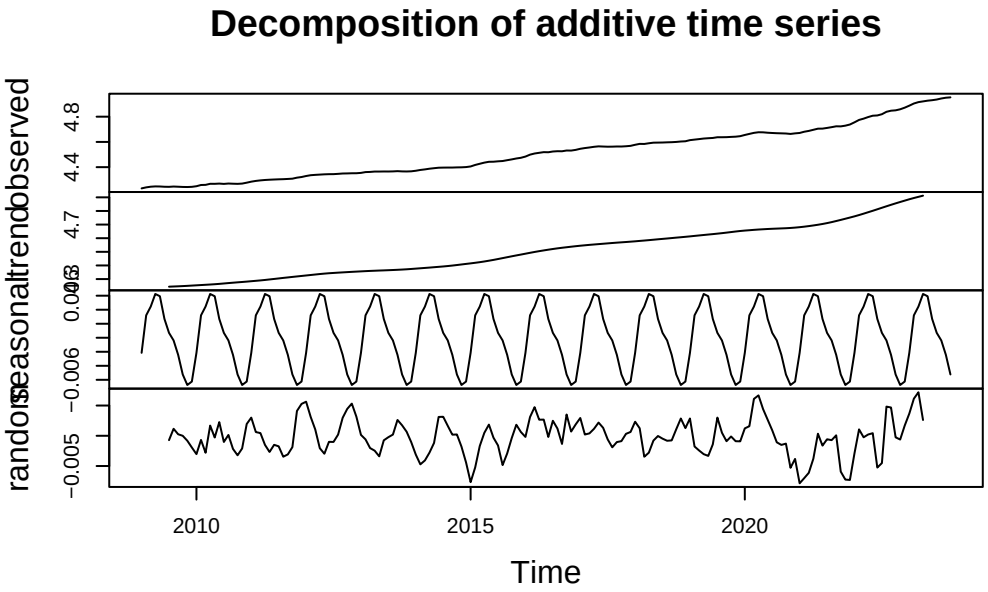
```
ElimiTenddatos1<-datos1-predict(fit)
plot(ElimiTenddatos1,main="Serie sin tendencia lineal",
     cex.main=1.3,
     xlab="Tiempo",
     ylab="log IPC",
     cex.lab=0.4)
```



Promedios móviles

A continuación se presenta la estimación de la tendencia via promedios móviles, esta no la usamos para evitar que la serie tuviera datos faltantes.

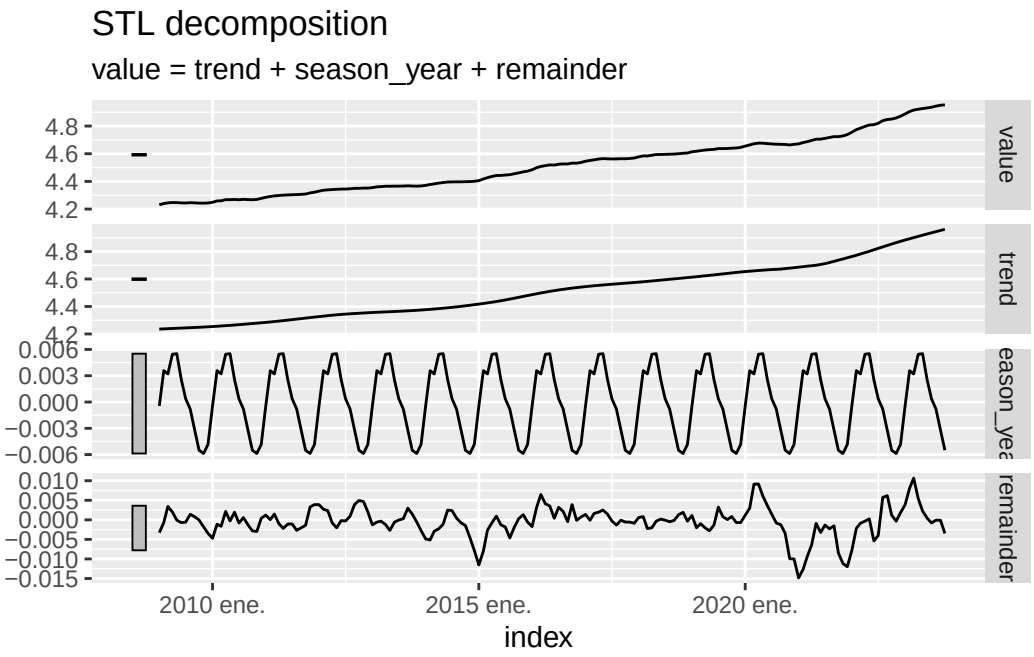
```
# Tendencia promedios moviles
datos1_decompo=decompose(datos1)
plot(datos1_decompo)
```



STL

A continuación se presenta la estimación de la tendencia via STL, vemos que esta sí estima el comportamiento de la tendencia al final de la serie, por lo tanto nos quedaremos con esta.

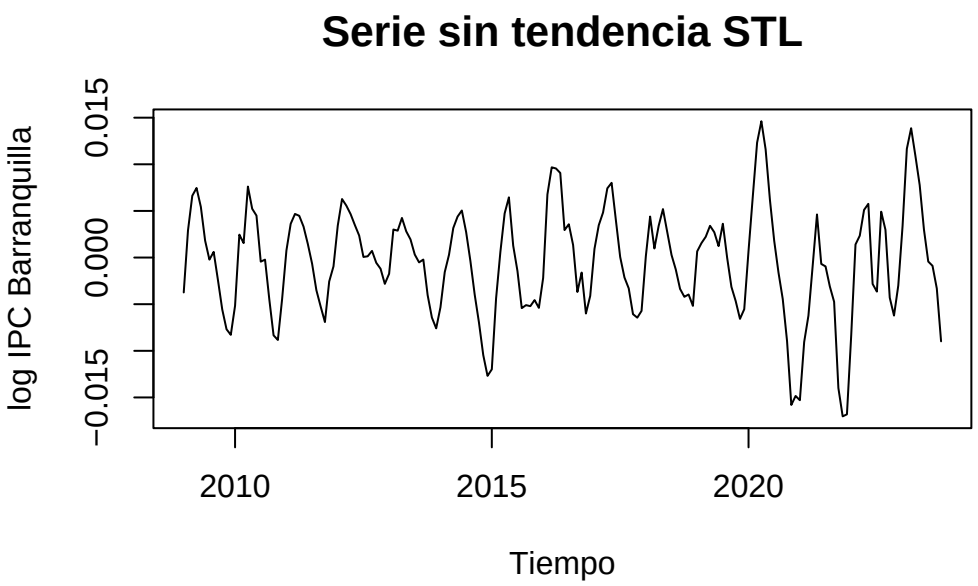
```
# Tendencia STL
tsibble_datos1<-as_tsibble(datos1)
tsibble_datos1 %>%
  model(
    STL(value ~ trend() +
          season(window = "periodic"),
        robust = TRUE)) %>%
  components() %>%
  autoplot()
```




```
modelo_stl<- tsibble_datos1 %>%
  model(
    STL(value ~ trend() +
          season(window = "periodic"),
        robust = TRUE)
  ) %>%
  components()
```

La siguiente gráfica muestra a la serie con varianza estabilizada y sin tendencia via STL.

```
# Guardar la serie sin tendencia STL
ElimiTenddatos1_STL<-datos1-modelo_stl$trend
plot(ElimiTenddatos1_STL,main="Serie sin tendencia STL",
     cex.main=1.3,
     xlab="Tiempo",
     ylab="log IPC Barranquilla",
     cex.lab=0.4)
```



Diferenciación

A continuación se presenta la serie diferenciada.

```
# Tendencia diferenciación
plot(diff(datos1), type="l", main="Primera Diferencia")
```

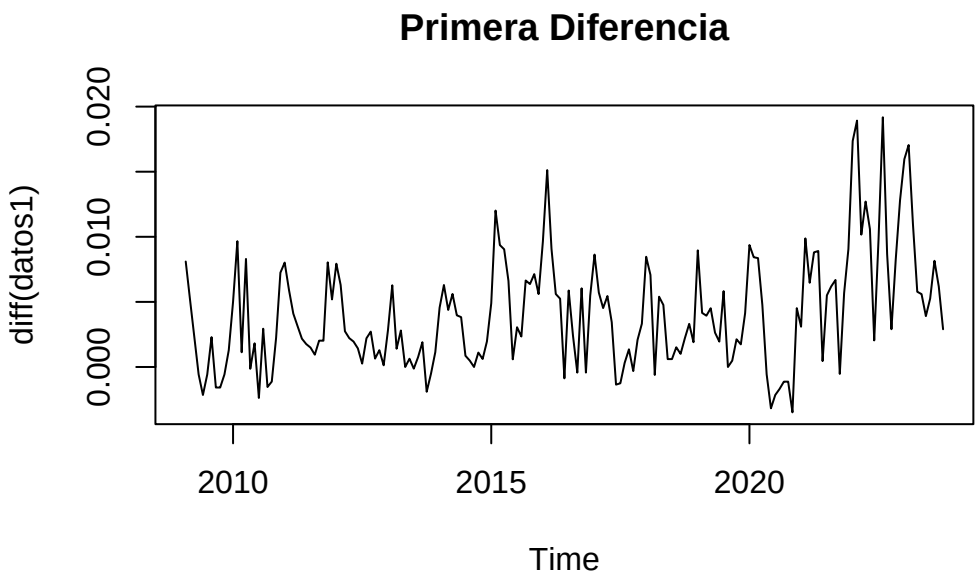
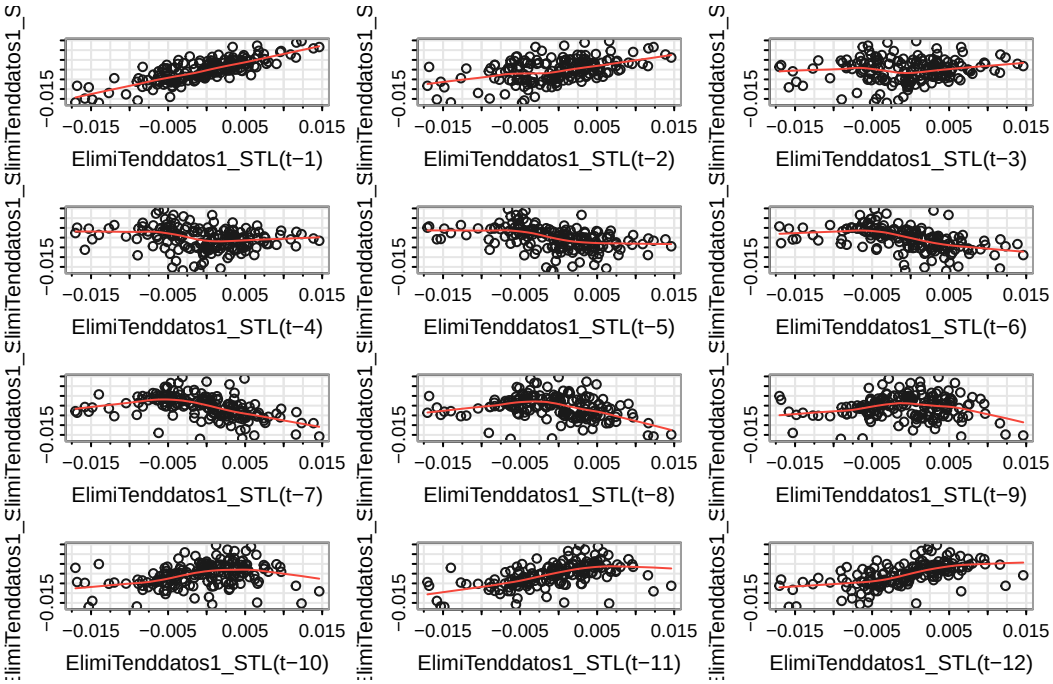


Gráfico de retardos

A continuación el gráfico de retardos muestra que se presenta una aparente correlación lineal entre el primer y posiblemente el segundo retardo.

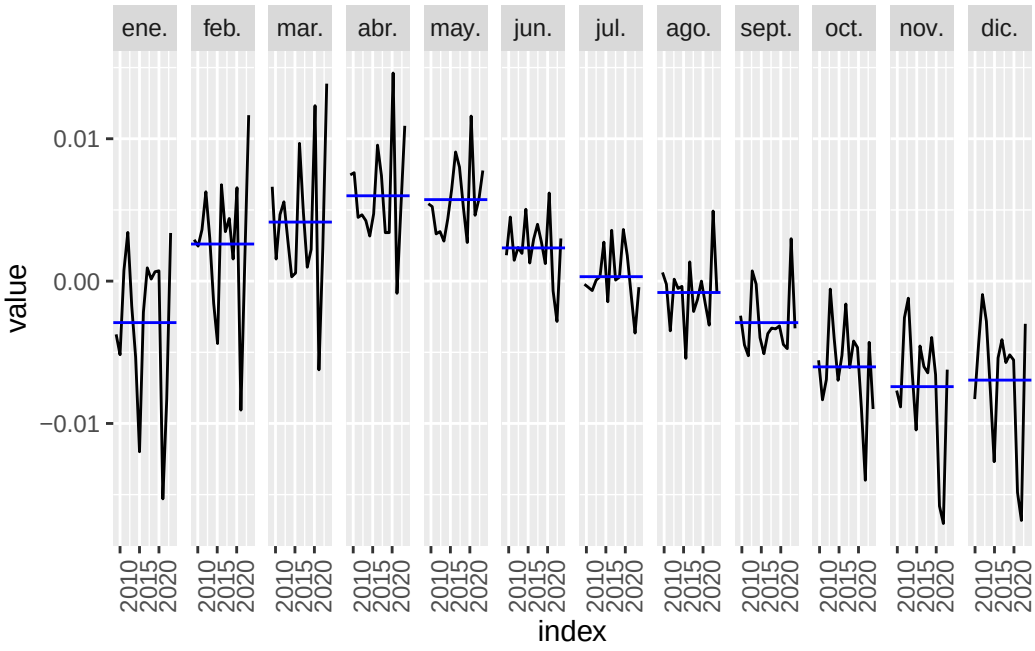
```
# Gráfico de retardos
par(mar = c(3,2,3,2))
astsa::lag1.plot(ElimiTenddatos1_STL, 12,corr=F)
```



Estimación de la estacionalidad

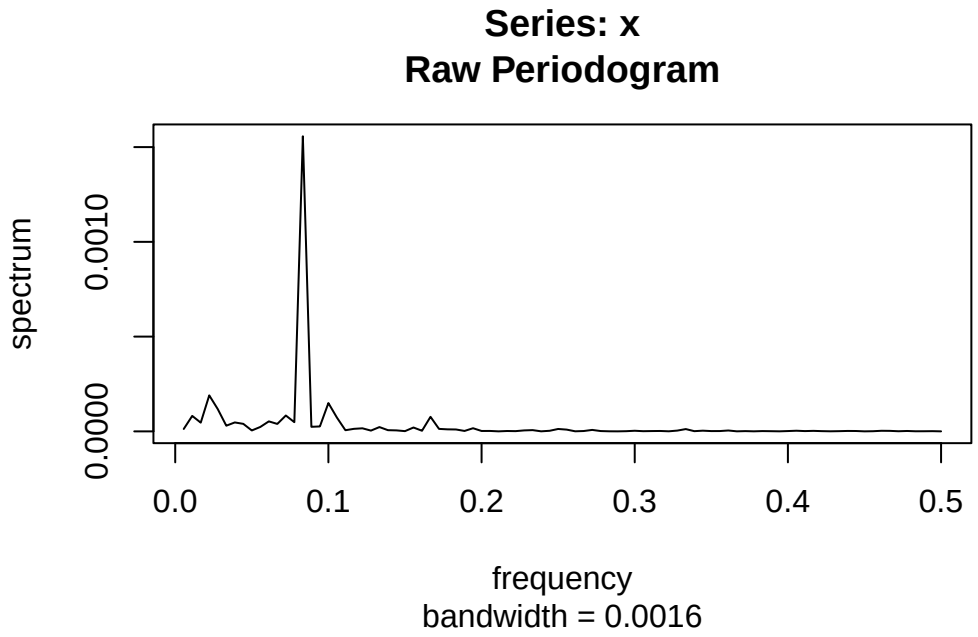
En el gráfico de subseries vemos que también la media de cada mes es distinta por lo que pensaríamos que aparentemente existe una estacionalidad de periodo 12.

```
datos1_tsbl_notend=as_tsibble(ElimiTenddatos1_STL)
datos1_tsbl_notend%>%gg_subseries(value)
```



El periodograma afirma que el periodo es de 12 meses.

```
# Periodograma
Periodgramadatos1_sintendencia=spectrum(as.numeric(ElimiTenddatos1_STL),log='no')
```




```
ubicaciondatos1=which.max(Periodgramadatos1_sintendencia$spec)
sprintf("El periodo correspondiente es aproximadamente: %s",1/Periodgramadatos1_sintendenc
```

```
[1] "El periodo correspondiente es aproximadamente: 12"
```

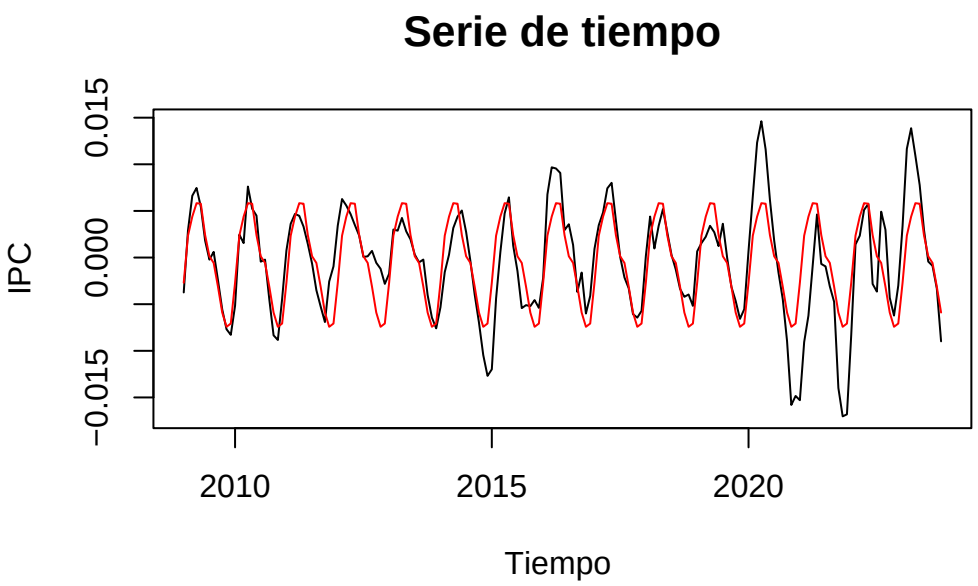
Estimamos la estacionalidad con dos métodos distintos: * Componentes de Fourier * Variables Dummy

Componentes de Fourier

En la gráfica se puede ver que las componentes de Fourier estiman "bien" la estacionalidad, a excepción de los cambios bruscos que pueda haber en la serie.

```
# Fourier
frec_ang=(2*pi/12)
datos1_copia=ElimiTenddatos1_STL

   #Fourier k=1
datos1_copia$sin = sin(c(1:178)*(1*frec_ang))
datos1_copia$cos = cos(c(1:178)*(1*frec_ang))
```

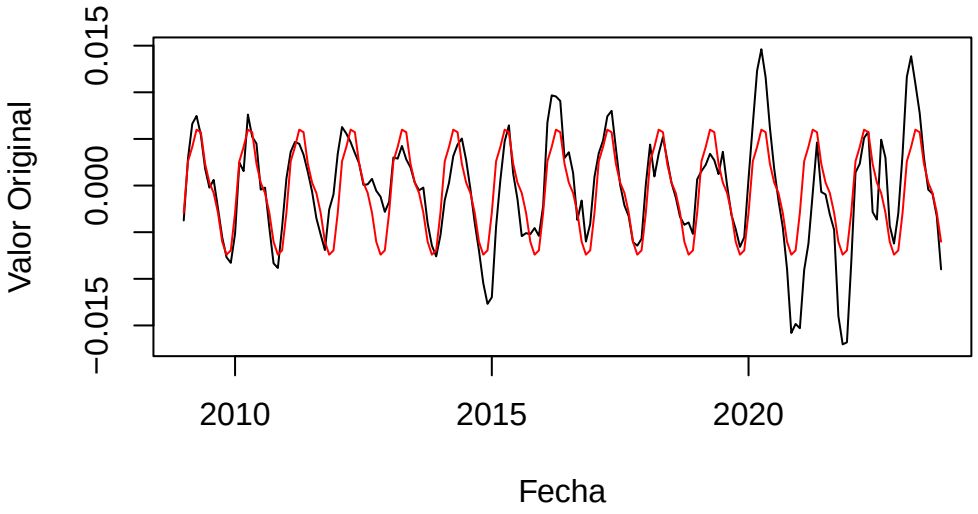
```
   #Fourier k=2
datos1_copia$sin2 = sin(c(1:178)*(2*frec_ang))
datos1_copia$cos2 = cos(c(1:178)*(2*frec_ang))

   #Fourier k=3
datos1_copia$sin3 = sin(c(1:178)*(3*frec_ang))
datos1_copia$cos3 = cos(c(1:178)*(3*frec_ang))

   #Fourier k=4
datos1_copia$sin4 = sin(c(1:178)*(4*frec_ang))
datos1_copia$cos4 = cos(c(1:178)*(4*frec_ang))

linmodel_ciclo_datos1<-lm(ElimiTenddatos1_STL~sin+cos+sin2+cos2+sin3+cos3+sin4+cos4,data=d

results_ciclo_datos1=linmodel_ciclo_datos1$fitted.values
results_ciclo_datos1<-as.data.frame(results_ciclo_datos1)
results_ciclo_ts<-ts(results_ciclo_datos1,start=c(2009,01),frequency=12)

plot(ElimiTenddatos1_STL, main="Serie de tiempo",
     cex.main=1.3,
     xlab="Tiempo ",
     ylab="IPC",
     cex.lab=0.4)
lines(results_ciclo_ts,col="red") # Estimación de la estacionalidad
```



Variables Dummy

Podemos ver que las variables Dummy estiman adecuadamente la estacionalidad adaptandose mejor a los cambios bruscos de la serie.

```
# Dummy
dummy <- seasonaldummy(ElimiTenddatos1_STL)
modelo <- lm(ElimiTenddatos1_STL ~ dummy)

   # Realiza predicciones con el modelo
nuevas_obs <- length(ElimiTenddatos1_STL)  # Número de observaciones en la serie de tiempo
predicciones <- predict(modelo, newdata = data.frame(dummy = dummy))

    # Crea un nuevo objeto de serie de tiempo con las predicciones
datos1_pred <- ts(predicciones, start = start(ElimiTenddatos1_STL), frequency = frequency(

    # Grafica los resultados
plot(ElimiTenddatos1_STL, col = "black", type = "l", xlab = "Fecha", ylab = "Valor Origina
lines(datos1_pred, col = "red")
```



Modelo SARIMA

Como vimos inicialmente en el análisis descriptivo, la serie presenta tanto tendencia como estacionalidad por lo que tiene sentido ajustar un modelo de la familia SARIMA. Adicionalmente se tienen que dividir los datos en entrenamiento y prueba unicamente.

```
# División de los datos
df_train4<- window(datos1, end = c(2022,03))
df_test4<- window(datos1, start = c(2022,02) + 1/12)
```

Se hace la prueba de Dicker Fuller para verificar la existencia de raíz unitaria, esto con el fin de determinar el orden de d. Dado que el $p - valor$ es grande se acepta la hipótesis nula, es decir hay presencia de raíz unitaria.

```
ar(datos1)
```

```
Call:
ar(x = datos1)

Coefficients:
```

```
     1
0.9797

Order selected 1  sigma^2 estimated as  0.001567
```

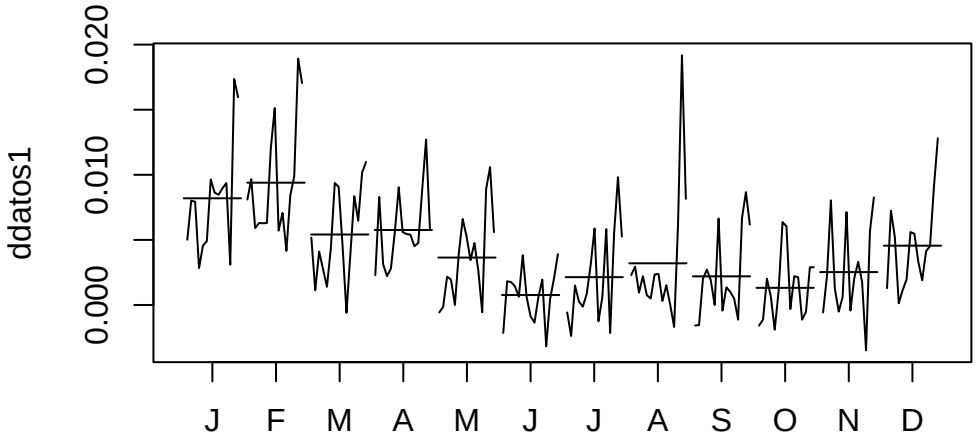
```
resultadodf_1<-adf.test(datos1,k=1)
resultadodf_1
```

```
    Augmented Dickey-Fuller Test

data:  datos1
Dickey-Fuller = -0.59687, Lag order = 1, p-value = 0.9765
alternative hypothesis: stationary
```

```
ddatos1<-diff(datos1) # Diferencia ordinaria
```

Para no eliminar la componente estacional la cual se evidenció en el análisis descriptivo, decidimos hacer solo una diferencia ordinaria. En el siguiente gráfico se presentan las subseries para la serie diferenciada, podemos notar que aparentemente hay una componente estacional.

```
monthplot(ddatos1)
```



Adicionalmente se sugiere hacer una diferencia estacional.

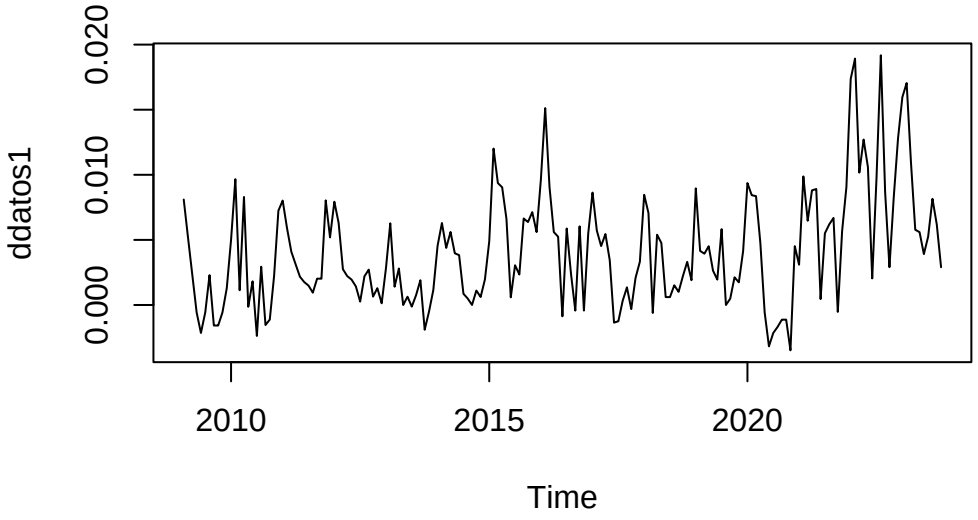
```
nsdiffs(ddatos1)
```

```
[1] 1
```

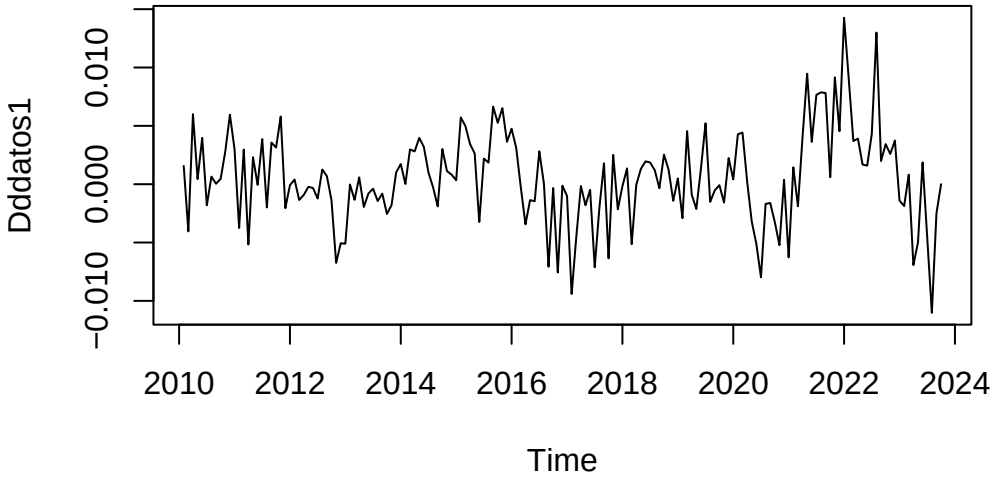
```
Dddatos1=diff(ddatos1,lag=12) ##Diferenciacion estacional
```

A continuación se muestra el gráfico de la serie diferenciada ordinalmente y la serie con diferencia ordinal y estacional.

```
plot(ddatos1)
```

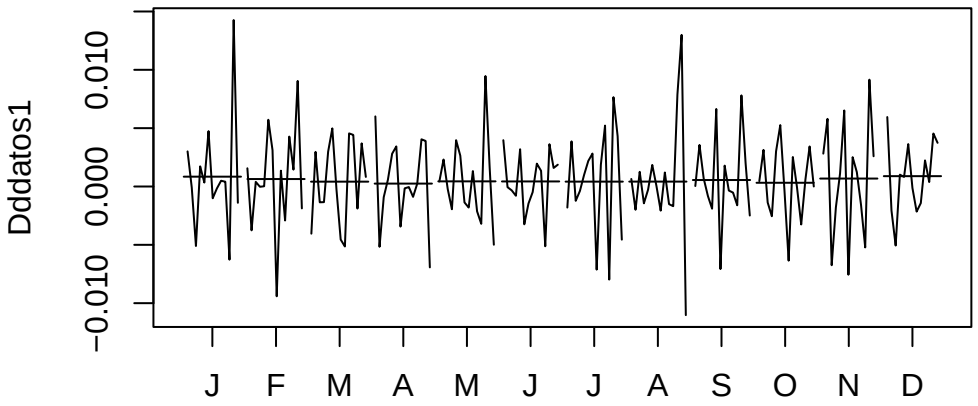



```
plot(Dddatos1)
```



En el gráfico de subseries se puede ver que ya las medias de todos los meses son iguales y en la salida se puede notar que ya no es necesario volver a diferenciar estacionalmente la serie.

```
monthplot(Dddatos1)
```

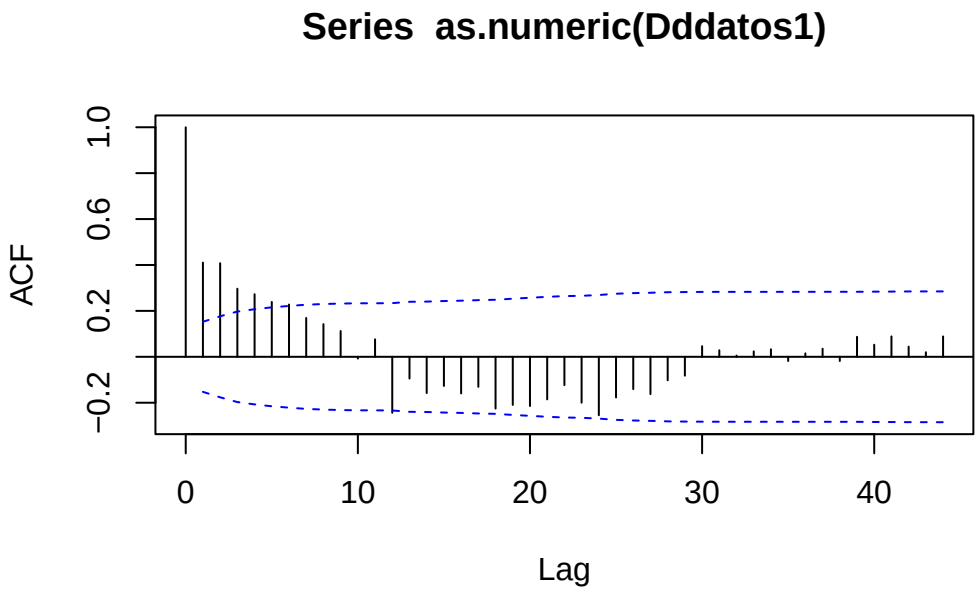



```
nsdiffs(Dddatos1)
```

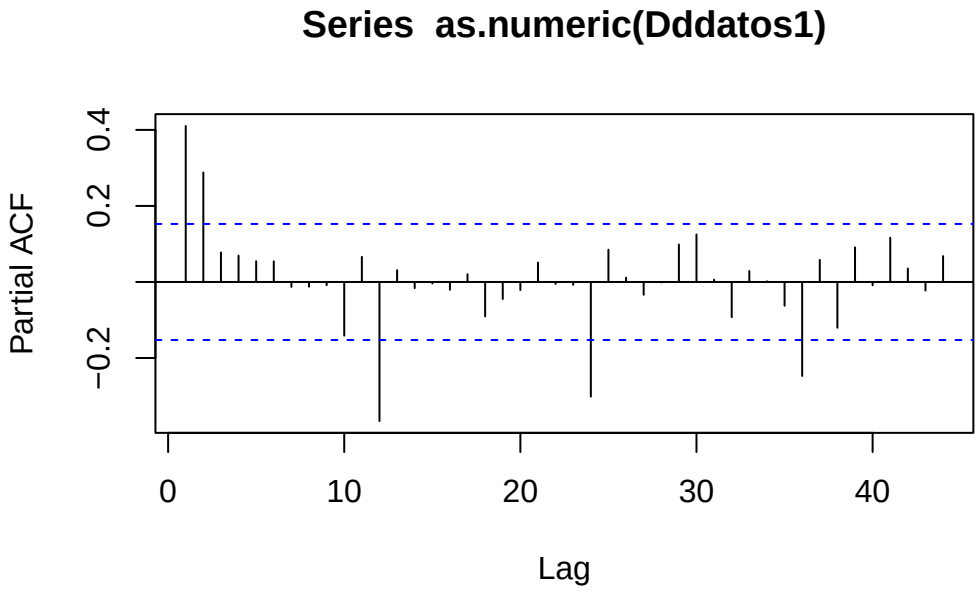
```
[1] 0
```

Como vimos previamente $d = 1$ y con el resultado anterior $D = 1$. Se procede a encontrar los ordenes p, q, P, Q. En el ACF se sugiere tomar ordenes $1, 2, 3, 4$ para q y $0, 1$ para Q. El PACF sugiere tomar 1 para el orden de p y $0, 1, 2$ para el orden de P.

```
acf(as.numeric(Dddatos1),lag.max = length(datos1)/4, ci.type='ma')
```




```
pacf(as.numeric(Dddatos1),lag.max = length(datos1)/4)
```



Procedemos a ajustar, con base en lo hallado anteriormente el modelo más completo.

```
modelo_SARIMA= Arima(df_train4, c(1, 1, 4),seasonal = list(order = c(2, 1, 1), period = 12
coeftest(modelo_SARIMA)
```

```
z test of coefficients:

      Estimate Std. Error z value  Pr(>|z|)
ar1  -0.889943   0.125036 -7.1175 1.099e-12 ***
ma1   1.241346   0.149582  8.2987 < 2.2e-16 ***
ma2   0.723496   0.133285  5.4282 5.693e-08 ***
ma3   0.463243   0.122555  3.7799 0.0001569 ***
ma4   0.055798   0.088478  0.6306 0.5282752
sar1  0.061487   0.124796  0.4927 0.6222248
sar2 -0.197184   0.121057 -1.6289 0.1033437
sma1 -0.816380   0.146471 -5.5737 2.494e-08 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

Se procede a refinar el modelo hasta que todos los coeficientes sean significativos.

```
modelo_SARIMA_ref = Arima(df_train4, c(1, 1, 4),seasonal = list(order = c(2, 1, 1), period
coeftest(modelo_SARIMA_ref)
```

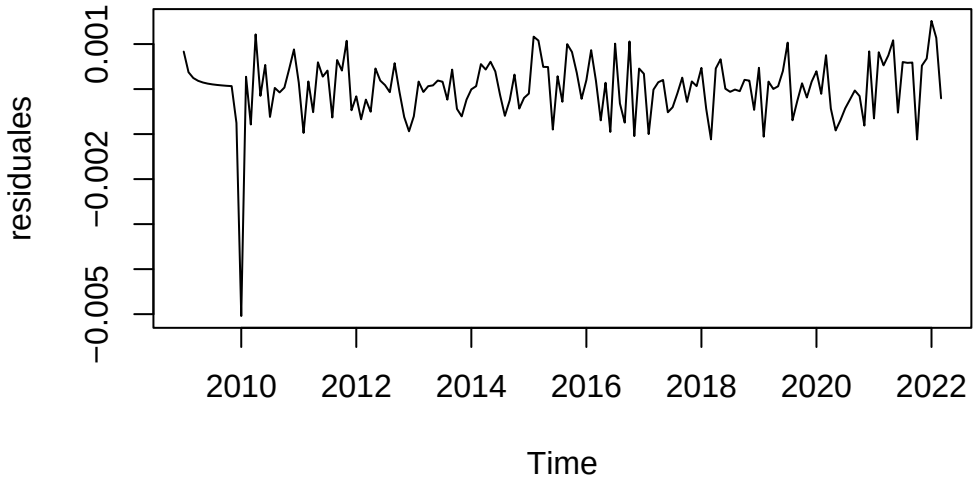
```
z test of coefficients:

      Estimate Std. Error z value  Pr(>|z|)
ar1   0.896452   0.083106 10.7868 < 2.2e-16 ***
ma1  -0.586218   0.132449 -4.4260 9.600e-06 ***
sma1 -0.905151   0.136533 -6.6296 3.367e-11 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

Validación de supuestos

Análisis de residuales

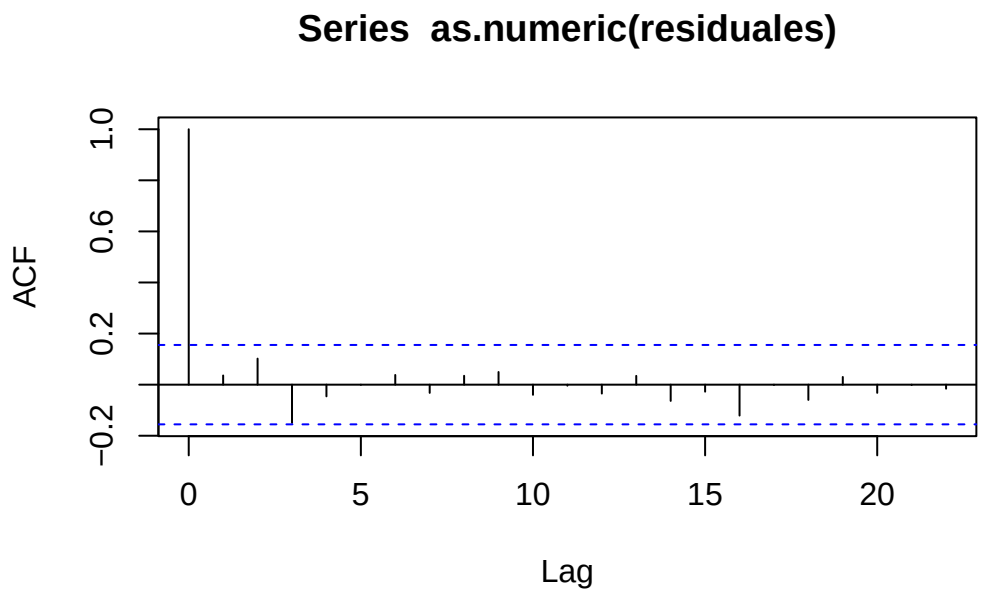
Del modelo refinado se obtienen los siguientes residuales.

```
residuales=modelo_SARIMA_ref$residuals
plot(residuales)
```

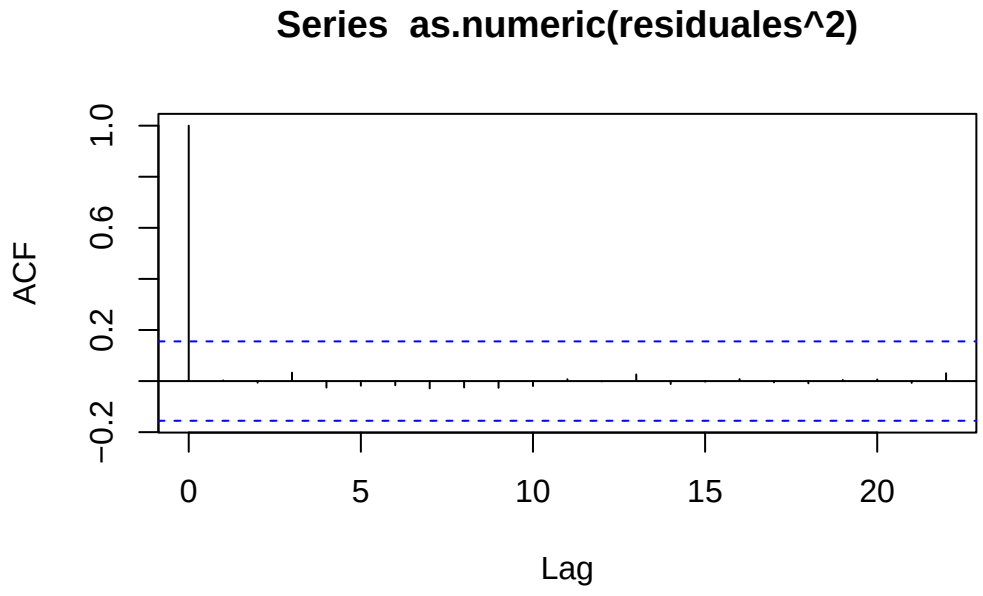


A continuación veremos las gráficas de ACF, ACF^2 y $PACF$ con el fin de observar si queda algo que el modelo no haya explicado. Se puede notar que no hay patrones o autocorrelaciones significativas.

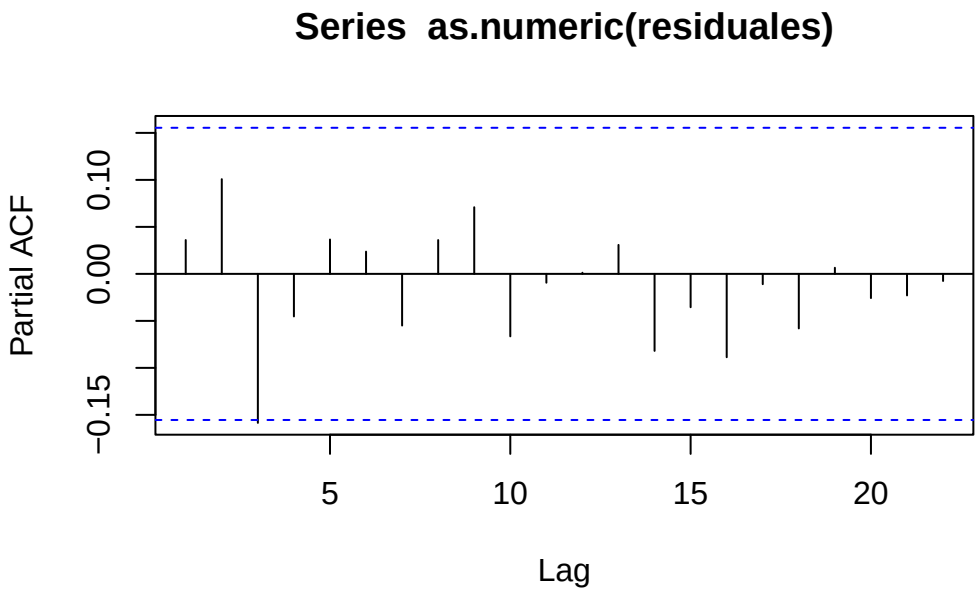
```
acf(as.numeric(residuales))
```




```
acf(as.numeric(residuales^2))
```



```r
pacf(as.numeric(residuales))
```



Test de normalidad

Para una significancia de 0.5, no hay normalidad en los datos.

```r
# Test de normalidad
tseries::jarque.bera.test(residuales)
```

```
    Jarque Bera Test

data:  residuales
X-squared = 2118.8, df = 2, p-value < 2.2e-16
```

Test de autocorrelación

El test de Ljung muestra que no hay autocorrelación entre los residuales.

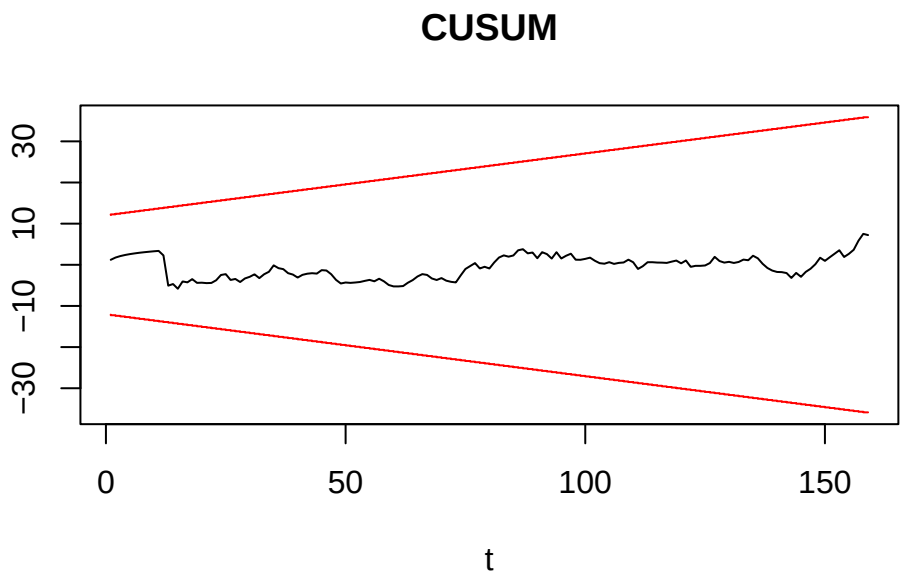
```r
# Test de autocorrelación
Box.test(residuales, lag =13 , type = "Ljung-Box", fitdf = 2)
```

```
    Box-Ljung test

data:  residuales
X-squared = 7.668, df = 11, p-value = 0.7427
```

Estadística CUSUM

Podemos ver que la estadística CUSUM no se sale de los límites, lo cual significa que los parámetros son estables.

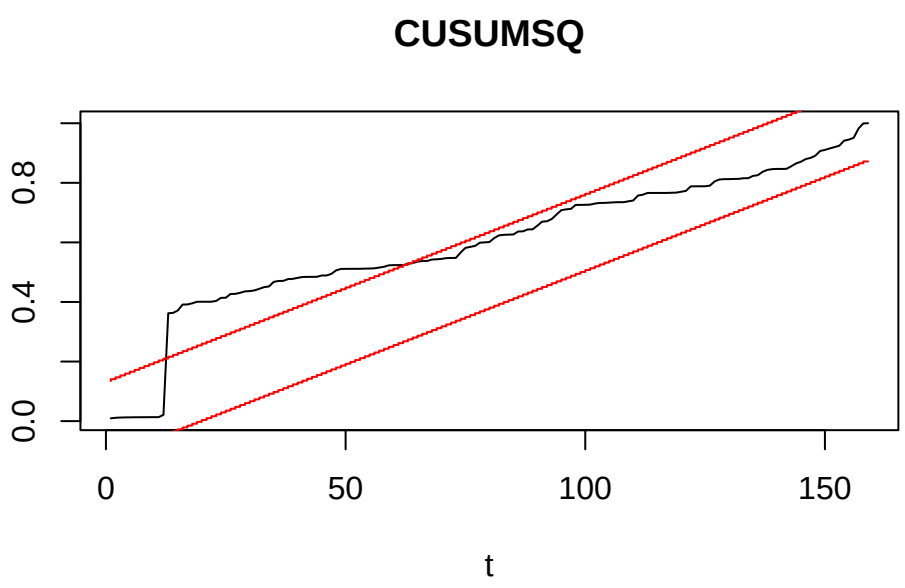
```r
res=residuales
cum=cumsum(res)/sd(res)
N=length(res)
cumq=cumsum(res^2)/sum(res^2)
Af=0.948
co=0.12819
LS=Af*sqrt(N)+2*Af*c(1:length(res))/sqrt(N)
LI=-LS
LQS=co+(1:length(res))/N
LQI=-co+(1:length(res))/N
plot(cum,type="l",ylim=c(min(LI),max(LS)),xlab="t",ylab="",main="CUSUM")
lines(LS,type="S",col="red")
lines(LI,type="S",col="red")
```



Estadística CUSUMSQ

Esta estadística se sale un poco de los límites, pero esto puede deberse a la presencia de datos atípicos.

```
plot(cumq,type="l",xlab="t",ylab="",main="CUSUMSQ")
lines(LQS,type="S",col="red")
lines(LQI,type="S",col="red")
```



Outliers

Hay 2 outliers, uno es aditivo y el otro de cambio de nivel.

```
# Outliers
resiO= residuals(modelo_SARIMA_ref)
coefO= coefs2poly(modelo_SARIMA_ref)
outliersSARIMA= locate.outliers(resiO,coefO,cval=4.5)
outliersSARIMA
```

```
  type ind      coefhat     tstat
1   AO  12  0.002079712  6.515310
3   LS  13 -0.004305411 -8.722707
```

RECM

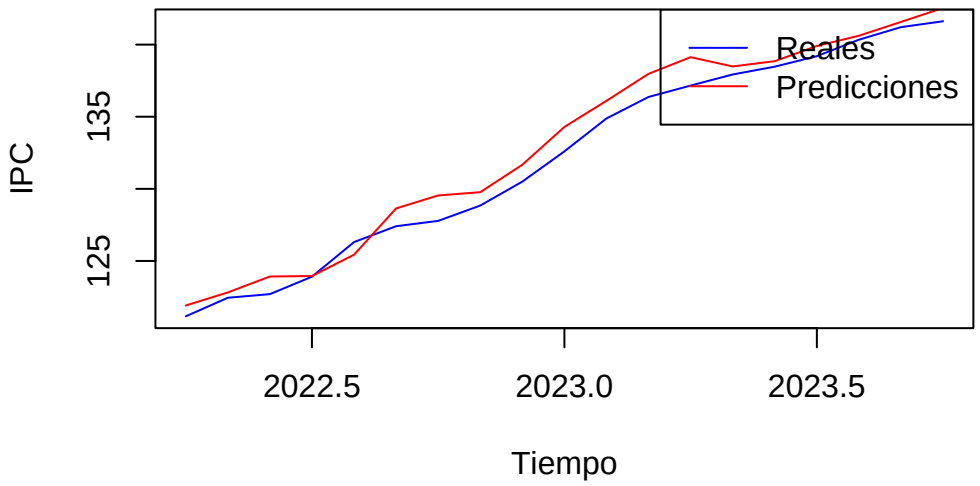
Para encontrar el RECM del modelo SARIMA utilizamos rolling. Se obtuvo un $RECM = 0.9300335$

```r
h=2 # Dos pasos adelante
ntest=length(df_test4)
verval=cbind(df_test4[1:ntest],c(df_test4[2:ntest],NA),c(df_test4[3:ntest],NA,NA))
fcmat=matrix(0,nrow=ntest,ncol=h)
for(i in 1:ntest){
  x=window(datos1,end=time(datos1)[159]+(i-1)/12)
  #print(length(x))
  refit=Arima(x, model=modelo_SARIMA_ref)
  fcmat[i,]=as.numeric(forecast::forecast(refit,h=h)$mean)
}
fcmat_menos_reales<-exp(verval[,2])-exp(fcmat)
ECM=apply(fcmat_menos_reales^2,MARGIN = 2,mean,na.rm=TRUE)
RECM=sqrt(ECM)
RECM[2]
```

```
[1] 0.9300335
```

Sobre los datos de prueba, así se ven las predicciones.

```r
verval_ts<-ts(exp(df_test4),start=time(datos1)[159]+1/12,frequency=12)
fchstepahe_ts<-ts(exp(fcmat[,2]),start=time(datos1)[159]+1/12,frequency=12)

plot(verval_ts, col = "blue", ylab = "IPC", xlab = "Tiempo")
lines(fchstepahe_ts, col = "red")
legend("topright", legend = c("Reales", "Predicciones"), col = c("blue", "red"), lty = 1)
```



Redes neuronales multicapa

Esta se trabajó en Python, puedes ver los resultados aquí

Pronóstico

Dado que el mejor modelo fue el de la familia SARIMA porque presentó menor RECM, haremos el pronóstico para Diciembre de 2023 con dicho modelo.

```
# Entrenar modelo con todos los datos
modelo_SARIMA_ref_todo = Arima(datos1, c(1, 1, 4),seasonal = list(order = c(2, 1, 1), peri
coeftest(modelo_SARIMA_ref_todo)
```

```
z test of coefficients:

      Estimate Std. Error z value  Pr(>|z|)
ar1   0.935057   0.042767 21.8638 < 2.2e-16 ***
ma1  -0.649851   0.087021 -7.4678 8.156e-14 ***
```

```
sma1 -0.824658   0.092987 -8.8685 < 2.2e-16 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```r
# Pronóstico
Pronosticos=forecast(modelo_SARIMA_ref_todo,h=2,level=0.95)
exp(Pronosticos$mean)[2]
```

```
[1] 143.5244
```