Proyecto 01

Modelos de Supervivencia y Series de Tiempo

2023-05-26

Suponga que usted se encuentra en diciembre 2018 y es el encargado de administrar un portafolio de inversión, usted debe elegir una acción para añadir al portafolio por lo que desea conocer un pronostico para el próximo año y medio para ver si es conveniente o no invertir $\$1,000,000$ de pesos en esta acción.

El primer paso para extraer información es usar Yahoo Finanzas, ingrese al anterior link para que pueda consultar la información histórica de una acción que usted prefiera (Como preferencia podría iniciar investigando el precio de una onza e indexarlo al portafolio, ya que es una serie fácil de trabajar). Yahoo finanzas es una de las principales páginas en la que usted podrá consultar información financiera, en ella además le permitirá ver gráficamente los datos y lo más importante descargar los datos en formato csv. Descargue los datos históricos del **1 de enero de 2001** al **30 Junio de 2020**, y en frecuencia indique periodicidad mensual, guarde estos datos en su carpeta de trabajo e impórtelos en R.

Con base a dichos valores realice.

1. Descargue e importe los datos en R y con ellos convierta el vector de precios a un objeto de series de tiempo y vea el comportamiento de sus datos, es decir calcule la media, moda, cuartiles, máximos, mínimos y varianza. Haga una gráfica de caja para ver visualmente estos resultados.

Nuestro interés está en ver la posibilidad de invertir en las acciones de *CEMEX* una empresa ligada al ámbito de la construcción que se autodenomina ser líder en la venta y producción de Cemento y otros materiales. Para saber si es una buena opción hacer está inversión revisaremos los datos históricos desde el *1 de enero de 2001* al *30 de junio de 2020*, usando algunos comandos y librerías de R extraeremos los datos desde Yahoo! Finanzas, tal como se muestra en el siguiente código:

```
getSymbols("CX", src = "yahoo", from = "2001-01-01", to = "2020-06-30",
           periodicity= "monthly")
base <- CX[,6] # para solo usar la columna de los precios de cierre ajustados
ts_base <- ts(base, start=2001, frequency = 12) # creamos el objeto de series de tiempo
```

Una vez que tenemos la base y el objeto series de tiempo comenzamos a hacer un análisis del comportamiento de los datos obteniendo algunos datos interesantes, que se pueden ver resumidos en la Tabla 1:

	Valores de la acción
Media	9.703
Moda*	7.463
Mínimo	2.120
Cuartil 0.25	6.322
Mediana	7.961
Cuartil 0.75	10.284
Máximo	27.697
Varianza	31.298

Note:
* = Es el valor de la locación estimada de la moda

Como se muestra en la Tabla 1 el valor promedio de la acción es de 9.703, además al graficar observamos que el mínimo se alcanza en *marzo de 2020* con valor al cierre ajustado de 2.12, que como se puede inferir se debió a la pandemia, ya que, al entrar en confinamiento la producción y venta de cemento sufrió una disminución, después de tocar su mínimo desde 2001, la acción se revalúa y empieza a subir de precio, observamos que su mejor precio fue en *mayo de 2007* con un valor de 27.697, luego tuvo fluctuaciones y pendientes negativas que coinciden con los años de la crisis de 2008, la gripe porcina 2009, la recesión de 2012, una caída en 2016, y después una caída alcanzando su mínimo hasta la pandemia de COVID-19.

Además a través de un gráfico de caja se puede ver el resumen de estas medidas, tal como se muestra en la Figura 1:

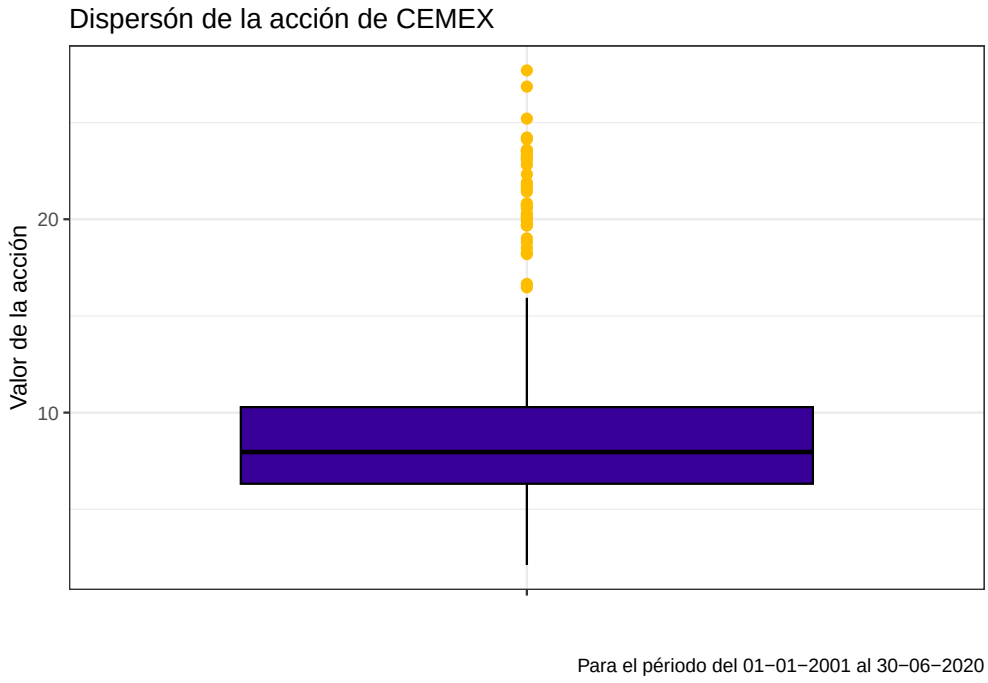


Figure 1: Boxplot de la acción de CEMEX

Graficamente, podemos observar que existe una gran cantidad de outliers, además hay 20 puntos de varianza con la media y el máximo, eso nos indica que esta muy lejos de el valor que llegó a tener la acción.

2. Realice una gráfica de la serie de tiempo.

Usando la libraría `ggplot()` se puede obtener la Figura 2: que representa la Serie de tiempo de la acción de Cemex:

3. Use la función `BoxCox.lambda(x, lower=0.5)` de la paquetería `forecast` para hallar el exponente λ, tal que minimice la variabilidad en

$$x'_t = x_t^{\lambda}$$

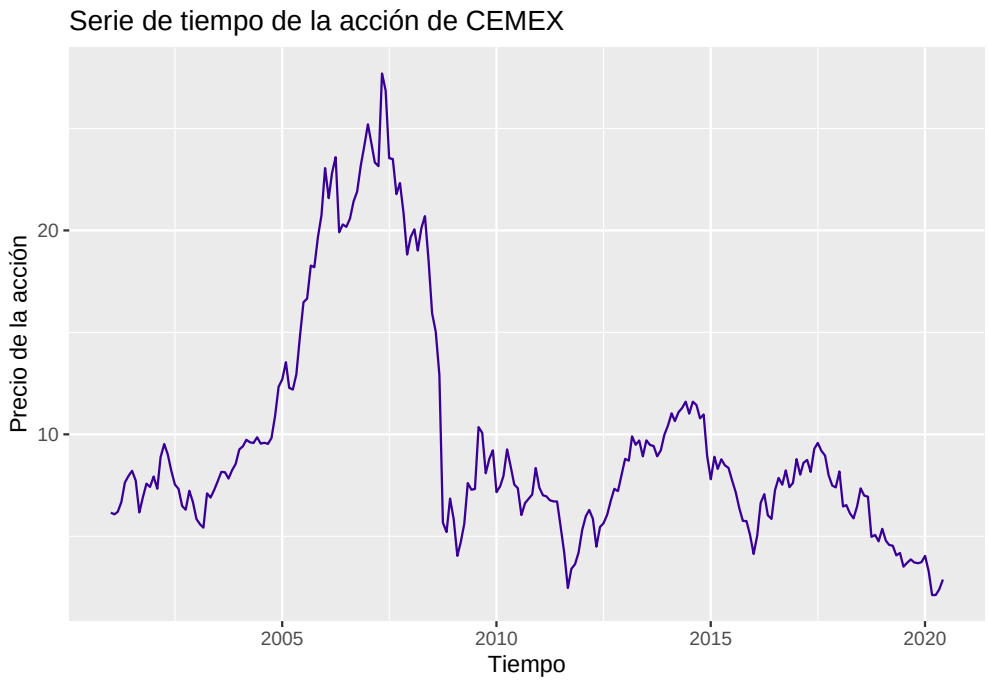


Figure 2: Serie de Tiempo de la acción de CEMEX

```
lambda <- BoxCox.lambda(ts_base, lower=0.5)
```

Usando dicho comando se puede obtener el valor de lambda que es: 0.595921

4. Una vez encontrado ese parámetro λ, realice la transformación logarítmica

$$x'_t = ln(x_t^{\lambda})$$

```
lambda_log <- BoxCox.lambda(log(ts_base), lower=0.5)
```

Con la transformación logaritmica el nuevo valor de lambda es: 1.9999416

5. Haga un análisis de esta nueva transformación, es decir, calcule la media, moda, cuartiles, máximos, mínimos y varianza. Haga una gráfica de caja para ver visualmente estos resultados.

Recalculando los datos con la transformación logaritmica se puede ver su resumen en la Tabla 2:

	Valores de la acción
Media	2.134
Moda*	2.041
Mínimo	0.751
Cuartil 0.25	1.844
Mediana	2.075
Cuartil 0.75	2.331

Máximo	3.321
Varianza	0.266

Note:
* = Es el valor de la locación estimada de la moda

Con los datos transformados vemos que en general se disminuye la variación de los datos, y se concentran de mejor manera.

```
log_base=log(ts_base)
```

Si se revisa el boxplot vemos que esta idea permanece, el cual puede ser observado en la Figura 3:

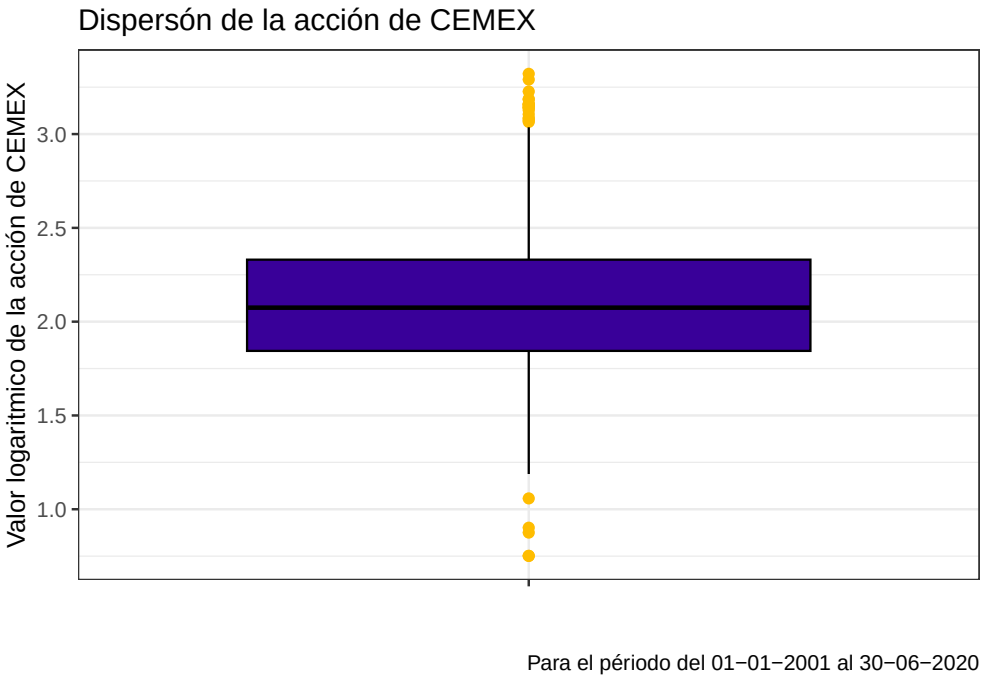


Figure 3: Boxplot del logaritmo de la acción de CEMEX

6. Grafique esta nueva serie de tiempo. ¿Observa las ventajas de realizar esta transformación?.

La gráfica de la nueva serie de tiempo se puede observar en la Figura 4:

De aquí vemos, que se vuelve mucho más estable, se disminuyen los grandes saltos en el tiempo, haciendola más estacionaria.

7. Realice una gráfica de descomposición aditiva para ver la tendencia, periodicidad y componente aleatorio asociado a los datos.

En la Figura 5, es posible observar como varía la tendencia, periodicidad y el componente aleatorio de la serie para el logaritmo de esta acción:

Aquí es claro notar, que la tendencia negativa de la acción y la periodicidad de la misma que es bastante constante.

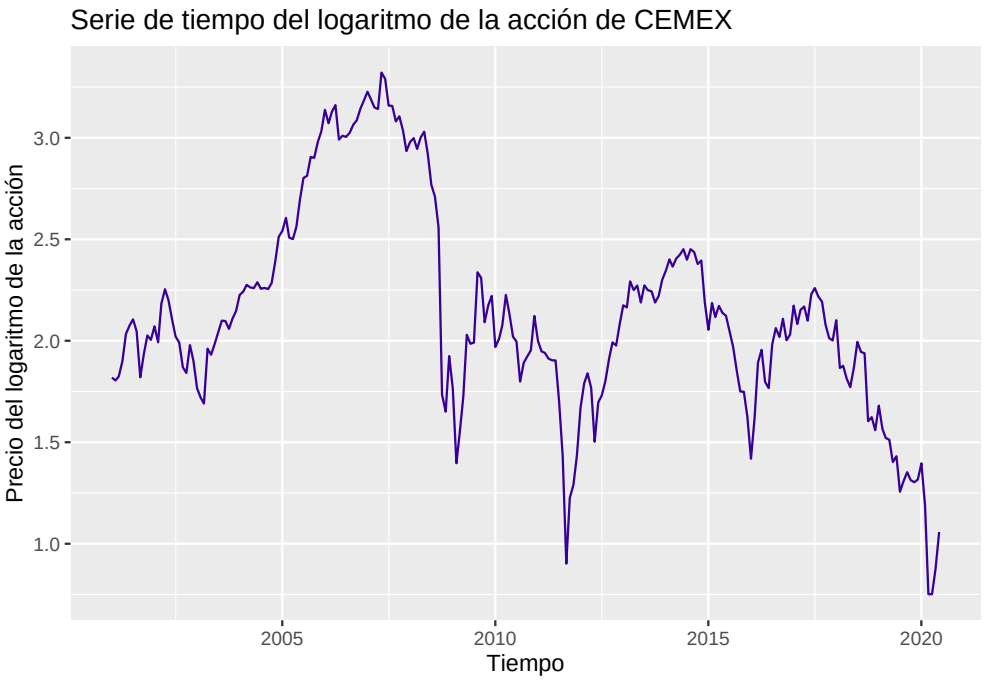


Figure 4: Serie de Tiempo del logaritmo de la acción de CEMEX

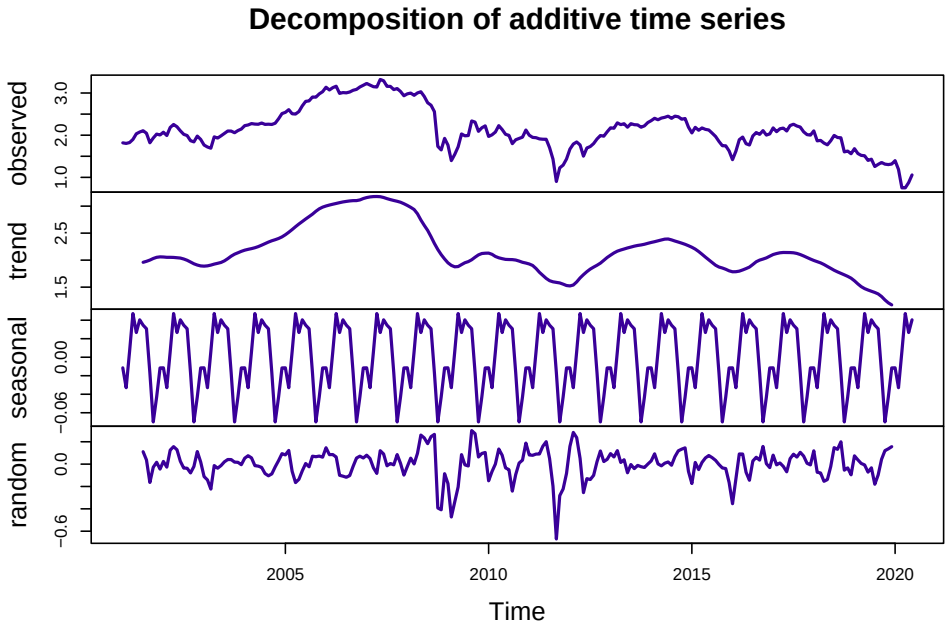


Figure 5: Descomposición aditiva del logaritmo de la acción de CEMEX

8. Realice una gráfica de descomposición multiplicativa para ver la tendencia, periodicidad y componente aleatorio asociado a los datos.

La descomposición multiplicativa para el logaritmo de la acción de Cemex se puede observar en la Figura 6:

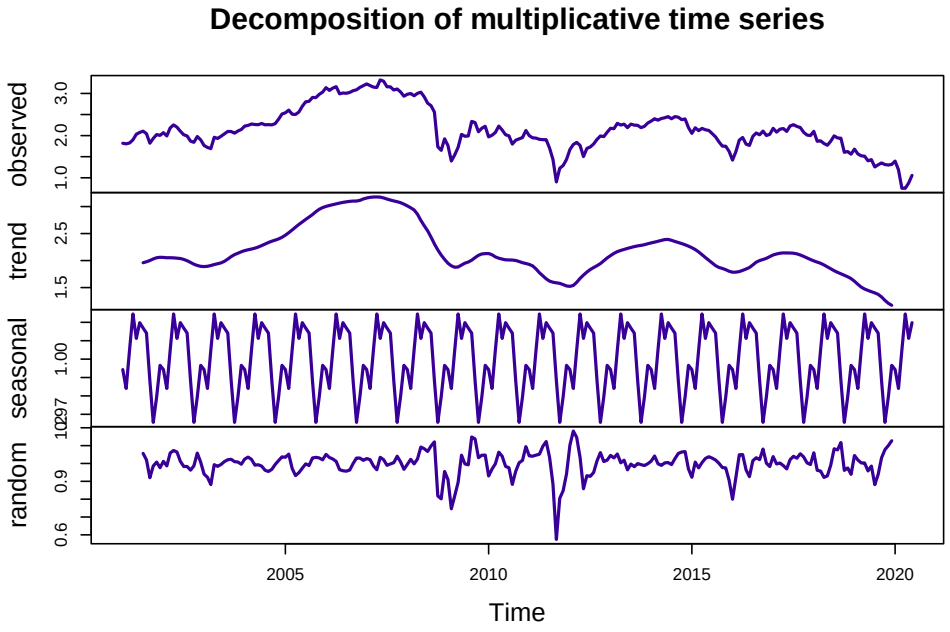


Figure 6: Descomposición multiplicativa del logaritmo de la acción de CEMEX

Para este caso, la descomposición multiplicativa coincide de forma considerable con la aditiva.

9. De estas dos descomposiciones ¿Cuál es la mejor para sus datos y por qué?

La mejor es la descomposicion multiplicativa en este caso, pues dado que la serie muestra un comportamiento no estacionario y aunque no es tan claro verlo en la original, al graficar el logaritmo de la serie se ve una tendencia claramente decreciente, la descomposicion multiplicativa permite modelar mejor la evolucion de la serie.

10. Realice una predicción de 18 meses hacia adelante usando el método de Holt-Winter aditivo. Grafique la predicción y guarde estas predicciones en una variable para posteriormente comparar los resultados.

Creando el objeto `hw_ts` crearemos la predicción a los 18 meses usando la base transformada, de aquí obtenemos la gráfica con las predicciones que se pueden observar en la Figura 7:

11. Con la serie transformada ¿Se puede asumir que la serie es estacionaria?

Para responder a esta pregunta revisaremos el gráfico `ggseasonal()` para ver la estacionariedad, el cual se puede apreciar en la Figura 8:

De la figura anterior, es posible observar que no es muy estacionaria la serie, por lo que sería bueno seguir transformandola.

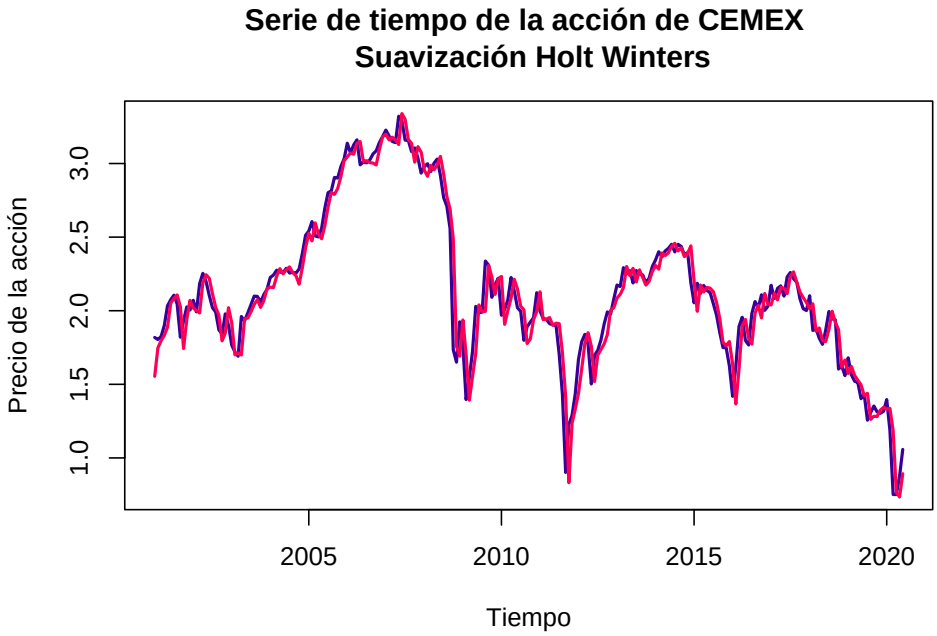


Figure 7: Método de Holt-Winter aditivo del logaritmo de la acción de CEMEX

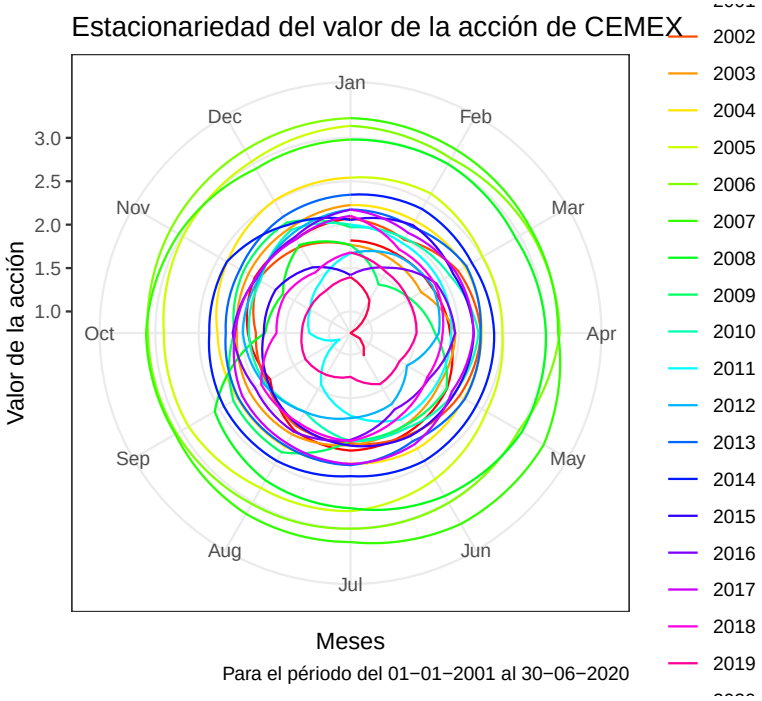


Figure 8: Logaritmo de la acción de CEMEX

12. Si concluyó que la serie no es estacionaría encuentre el orden de diferenciación que le permita pasar de una serie no estacionaria a una estacionaria.

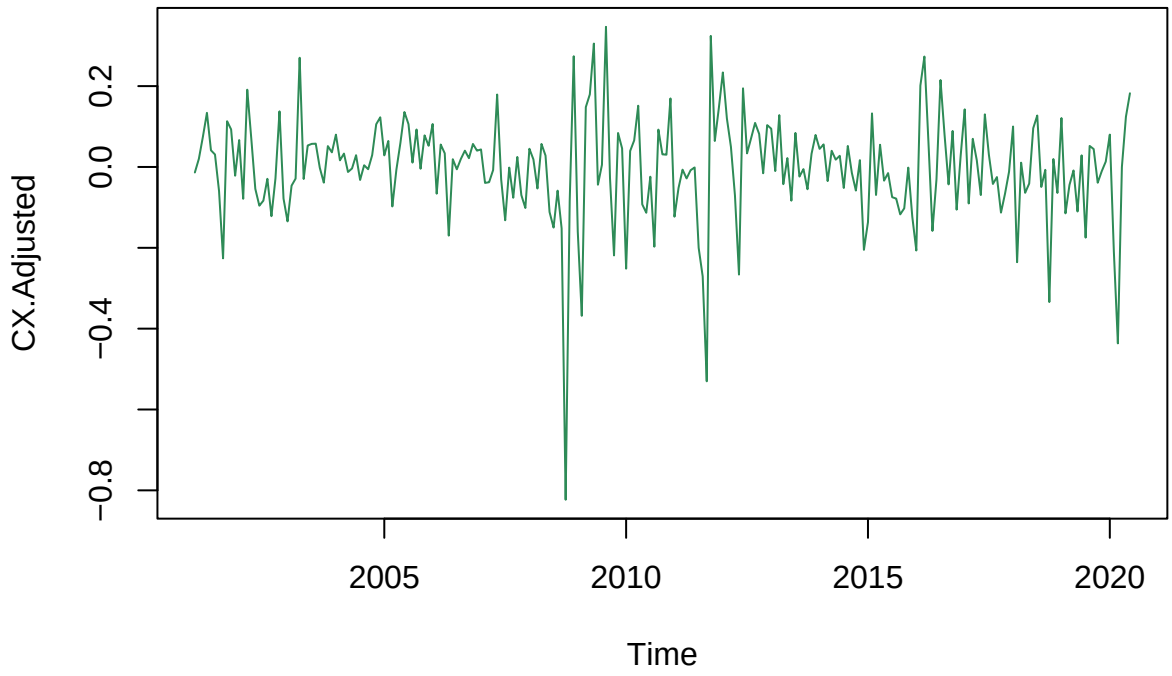
Primero utilizamos la función de la libreria forecast llamada ndiffs que nos da el número de diferenciaciones que debemos hacer paraq que la serie sea estacionaria

```
dorito=ndiffs(log(ts_base))
dorito
```

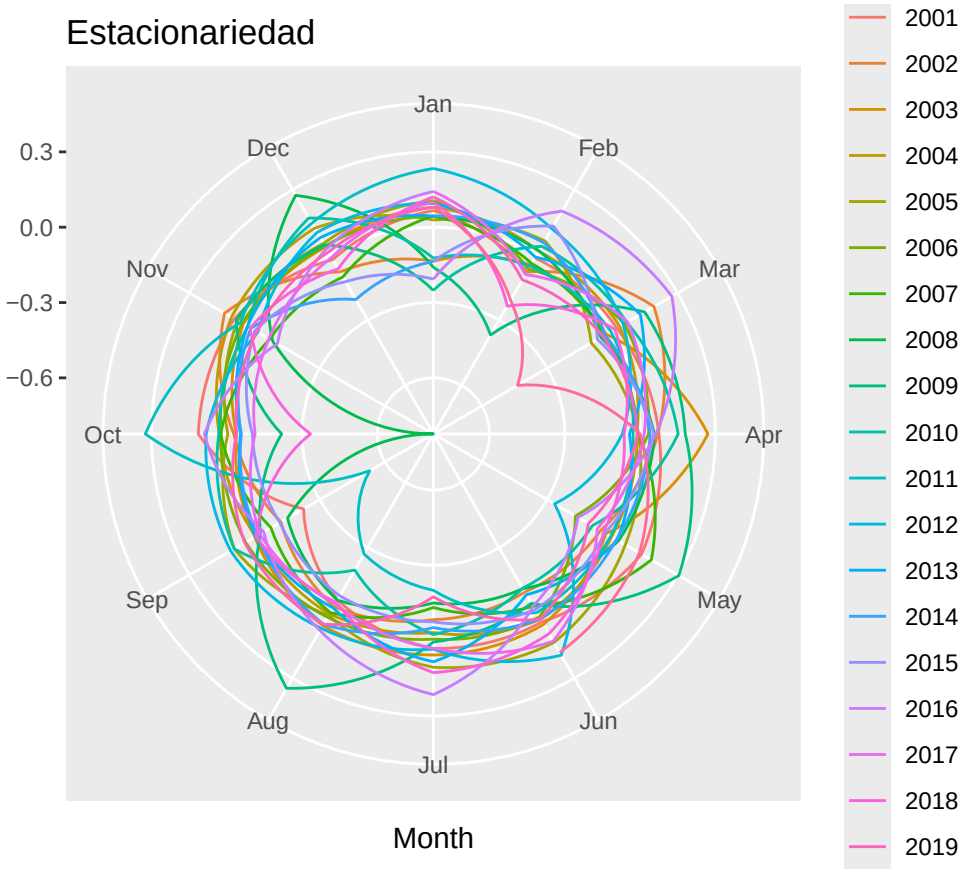
```
## [1] 1
```

Como podemos observar el número de diferenciaciones que debemos hacer es 1, el cual llamaremos dorito1

```
dorito1=diff(log_base, lag=1 )
plot(dorito1,col = "seagreen")
```

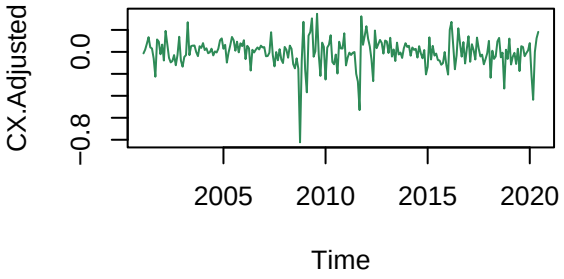


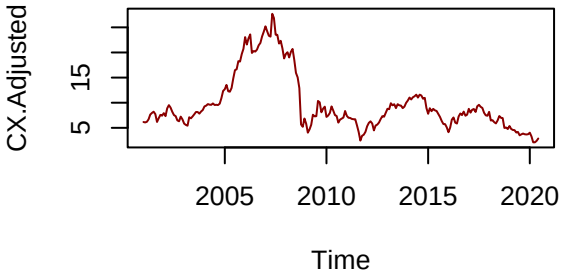

```
ggseasonplot(dorito1,polar=TRUE,main="Estacionariedad" )
```



Comparamos la diferenciación con la base original para ver los cambios, dado que el número de diferecniaciones es 1, tenemos que nuestro ARIMA(p,1,q)

```
par(mfrow=c(2,2))
plot(dorito1, type="l",col="seagreen")
plot(ts_base, type="l",col="darkred")
```





13. Compare estos resultados con la prueba Dickey-Fuller Test adf.test()

```
adf.test(dorito1)
```

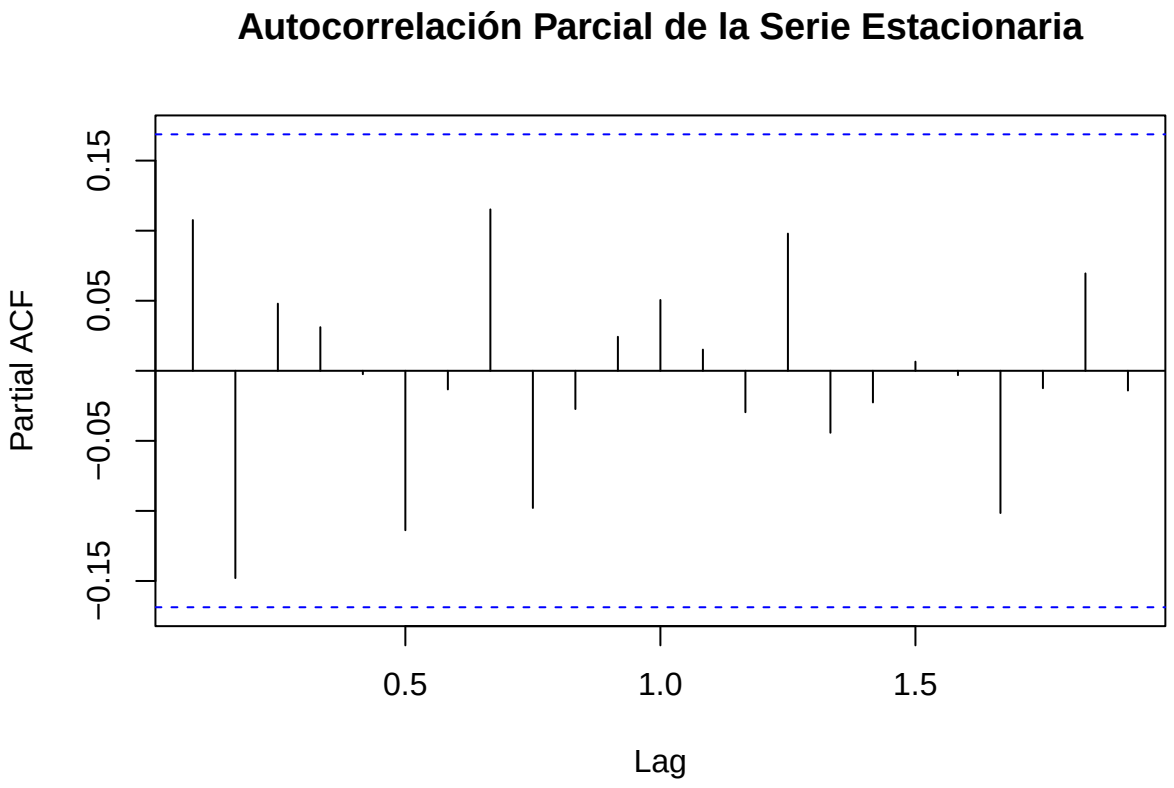
```
## Warning in adf.test(dorito1): p-value smaller than printed p-value
```

```
##
##  Augmented Dickey-Fuller Test
##
## data:  dorito1
## Dickey-Fuller = -6.2877, Lag order = 6, p-value = 0.01
## alternative hypothesis: stationary
```

Como podemos observar se rechaza la hipotesis nula, es decir, H_0= No es estacionaria vs H_1= Es estacionaria, por lo tanto se concluye que la base con la transformación dorito 1 es esatcionaria.

14. Con la serie diferenciada y con un 99% de confianza muestre el orden que debe ajustar a un AR(p).

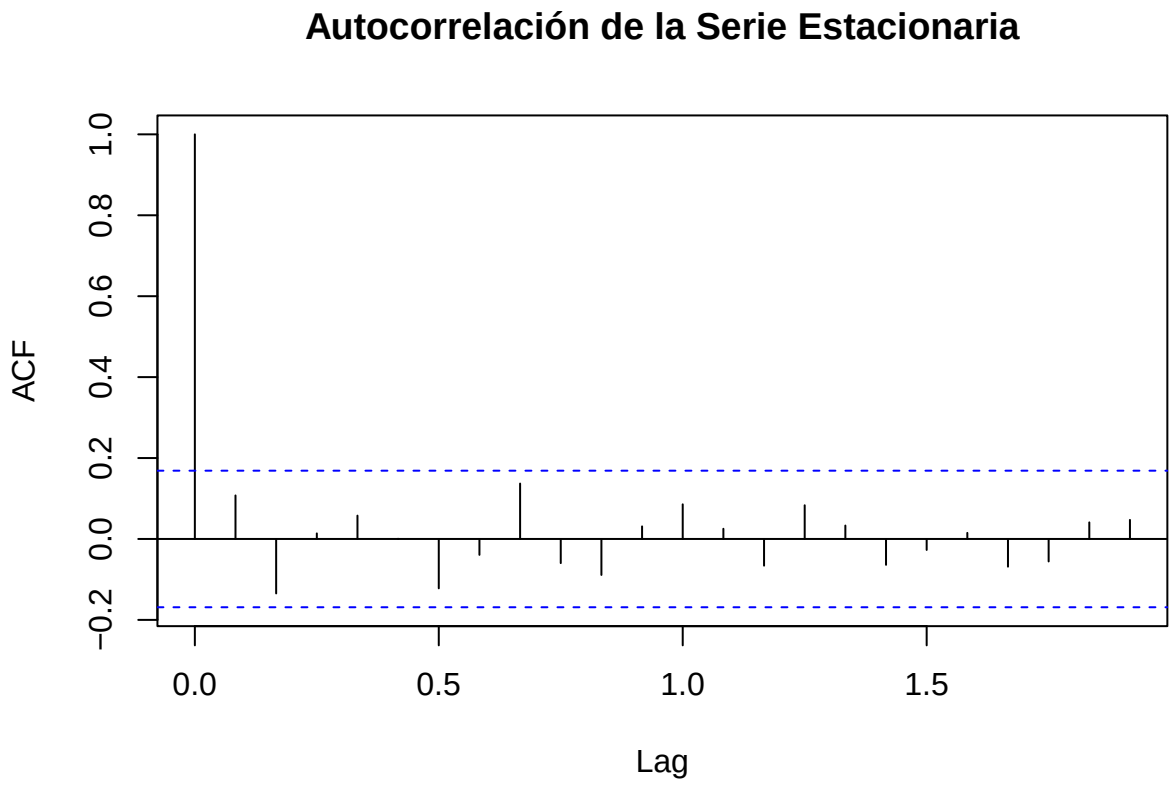
```
pacf(dorito1,ci = 0.99, main = "Autocorrelación Parcial de la Serie Estacionaria")
```



Como podemos observar, ninguna linea del autocorrelograma parcial se sale del margen al 99% de confianza, es decir el valor de $p = 0$, es decir AR(0).

15. Con la serie diferenciada y con un 99% de confianza muestre el orden que debe ajustar a un MA(q).

```
acf(dorito1,ci = 0.99, main = "Autocorrelación de la Serie Estacionaria")
```



Para el autocorrelograma al 99% de confianza notamos que ninguna linea se sale, es decir MA(0).

16. Escriba el modelo ARIMA, y si es posible pruebe dos modelos.

```
# Ya sabemos que el orden de diferenciacion es 1, por lo que el segundo parametro debe ser 1. Ahora vea

# Este modelo surge de observar 0 lineas que se salen en el PACF y en el ACF al 99% de confianza
modelo1=arima(log(ts_base), order=c(0,1,0), include.mean = FALSE)
modelo1
```

```
##
## Call:
## arima(x = log(ts_base), order = c(0, 1, 0), include.mean = FALSE)
##
##
## sigma^2 estimated as 0.01704:  log likelihood = 143.81,  aic = -285.63
```

```
# El modelo siguiendo el criterio pero al 95% de confianza
modelo2=arima(log(ts_base), order=c(1,1,1), include.mean = FALSE)
modelo2
```

```
##
## Call:
## arima(x = log(ts_base), order = c(1, 1, 1), include.mean = FALSE)
##
```

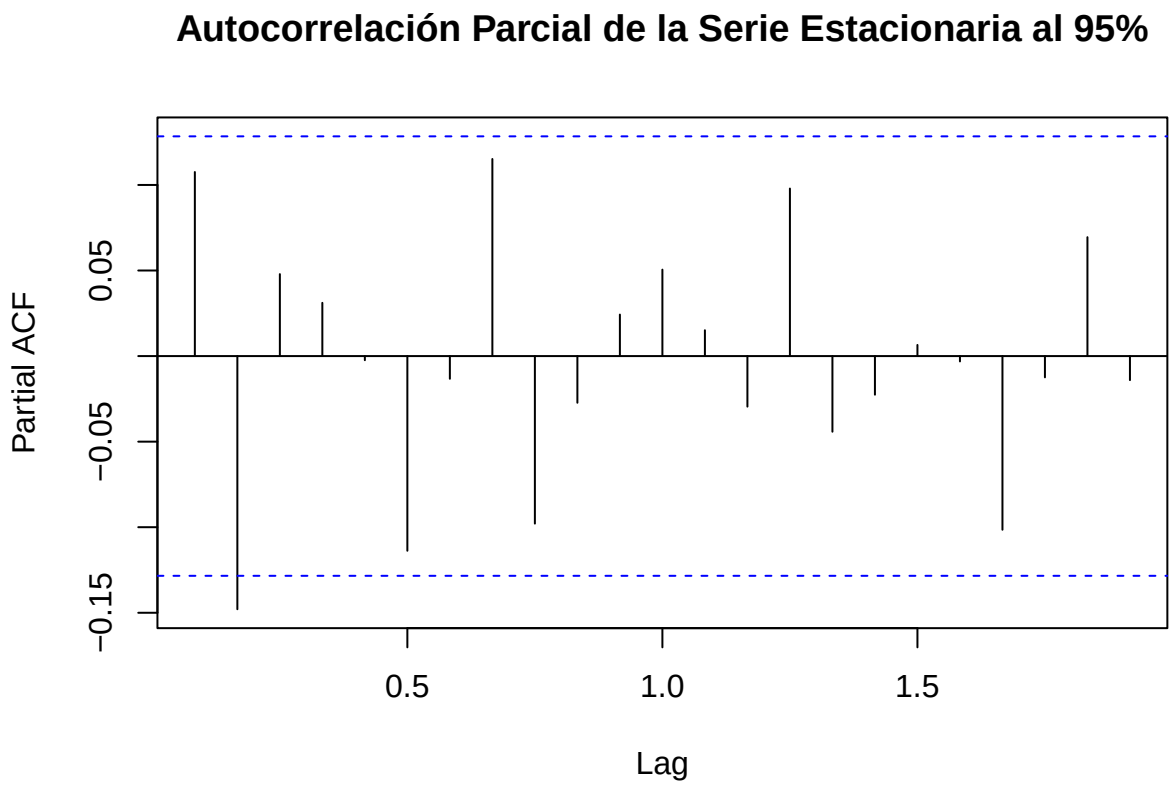
```
## Coefficients:
##           ar1    ma1
##       -0.4397  0.593
## s.e.   0.2126  0.187
##
## sigma^2 estimated as 0.01655:  log likelihood = 147.15,  aic = -288.31
```

```
# Este modelo surge de considerar que, al 95%, hay una linea que no sale pero "casi". Aqui la contamos
modelo3=arima(log(ts_base), order=c(1,1,2), include.mean = FALSE)
modelo3
```

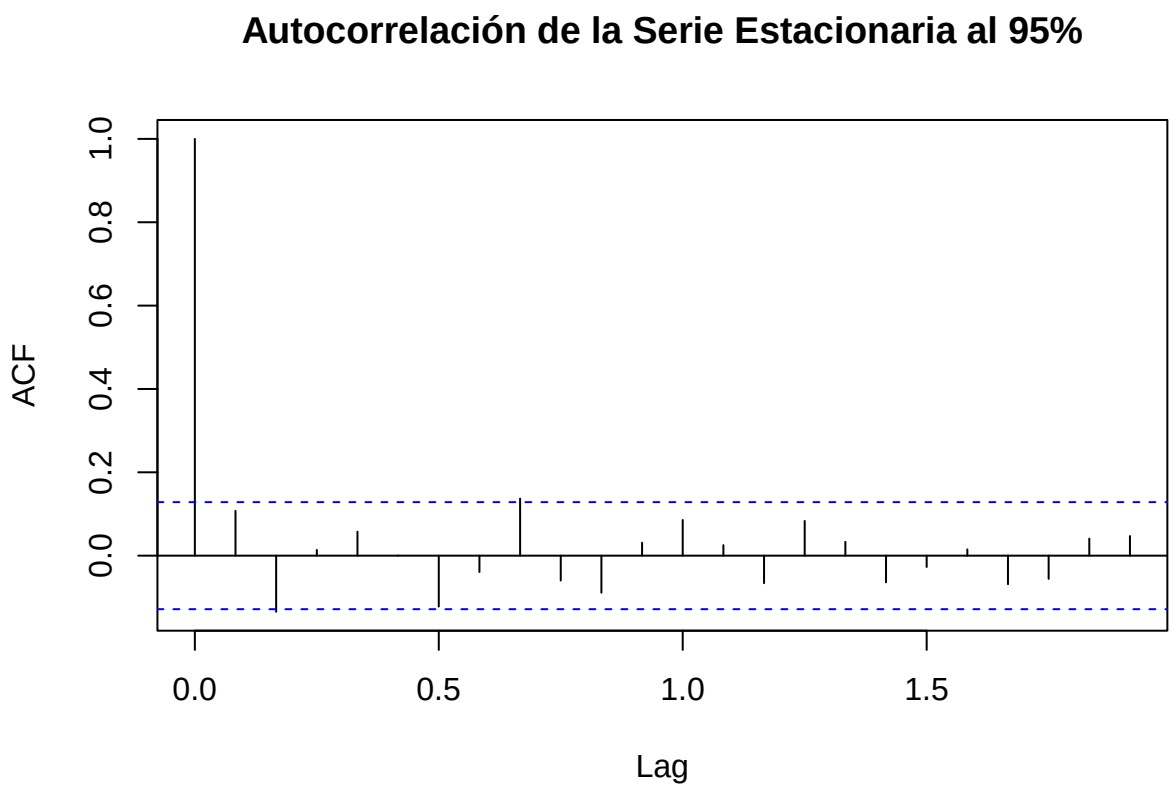
```
##
## Call:
## arima(x = log(ts_base), order = c(1, 1, 2), include.mean = FALSE)
##
## Coefficients:
##           ar1     ma1      ma2
##       -0.0348  0.1657  -0.1244
## s.e.   0.4140  0.4093   0.0893
##
## sigma^2 estimated as 0.01646:  log likelihood = 147.81,  aic = -287.63
```

```
# A continuacion, puede observarse ese PACF y ACF sin introducir un nivel de confianza
```

```
pacf(dorito1,ci = 0.95, main = "Autocorrelación Parcial de la Serie Estacionaria al 95%")
```



```r
acf(dorito1,ci = 0.95, main = "Autocorrelación de la Serie Estacionaria al 95%")
```

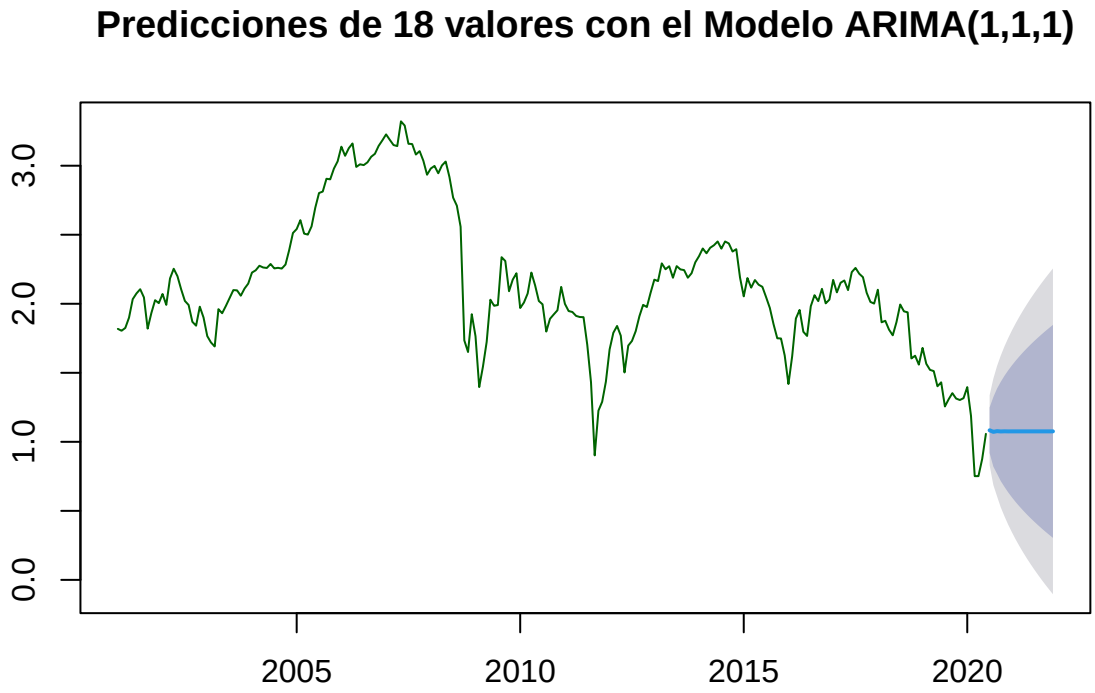


17. Con el mejor modelo realice un pronóstico 18 valores hacían adelante

Guiandonos por el AIC, es inmediato concluir que por mucho, el mejor modelo es el segundo o si no el tercero; es decir lo mas factible es que sea el ARIMA(1,1,1) y si no, seria el ARIMA(1,1,2). ya que los AIC de los 3 son:

Modelo	ARIMA	AIC
1	0,1,0	-285.63
2	1,1,1	-288.31
3	1,1,2	-287.63

```r
pronostico <- forecast::forecast(modelo2,18)
plot(pronostico,main="Predicciones de 18 valores con el Modelo ARIMA(1,1,1)",col="darkgreen")
```



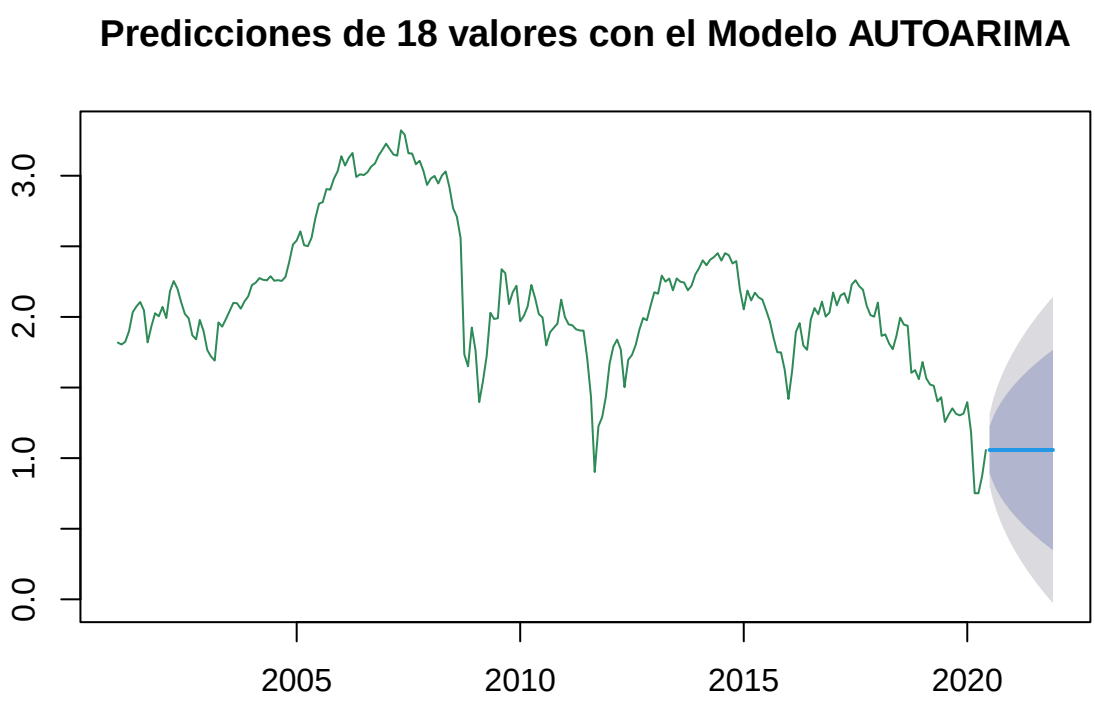
18. Ejecute la función auto.arima con los datos transformados (sin las diferenciaciones). ¿El modelo resultante le parece mejor que el que encontró?

```
(  autoarima<-auto.arima(log(ts_base))  )
```

```
## Series: log(ts_base)
## ARIMA(0,1,0)
##
## sigma^2 = 0.01704:  log likelihood = 143.81
## AIC=-285.63   AICc=-285.61   BIC=-282.17
```

la prediccion con este modelo se ve:

```
pronostico.autoarima <- forecast::forecast(autoarima,18)
plot(pronostico.autoarima,main="Predicciones de 18 valores con el Modelo AUTOARIMA",col="seagreen")
```



La prediccion se ve identica, sin embargo el auto.arima mejoro el AIC un poquito. El autoarima tiene un AIC de -282.17. Por lo que sigue solido el que el mejor modelo es o bien el ARIMA (1,1,1) o bien el ARIMA (1,1,2)

19. Con las predicciones más adecuadas realicé la transformación inversa para tener a los datos originales

$$x_t = exp(x_t')^{(1/lambda)}$$

```
x<-exp(as.data.frame(pronostico)[,1] )^(1/lambda)
x
```

```
##  [1] 6.162725 6.046053 6.097081 6.074590 6.084470 6.080124 6.082034 6.081194
##  [9] 6.081564 6.081401 6.081473 6.081441 6.081455 6.081449 6.081452 6.081450
## [17] 6.081451 6.081451
```

20. Consulte los datos reales del precio de la onza de oro de 2019 hasta junio 2020 y diga cual estimación fue la mejor de todas.

```
getSymbols("CX", src = "yahoo", from = "2019-01-01", to = "2020-06-30",
           periodicity= "monthly")
```

```
## [1] "CX"
```

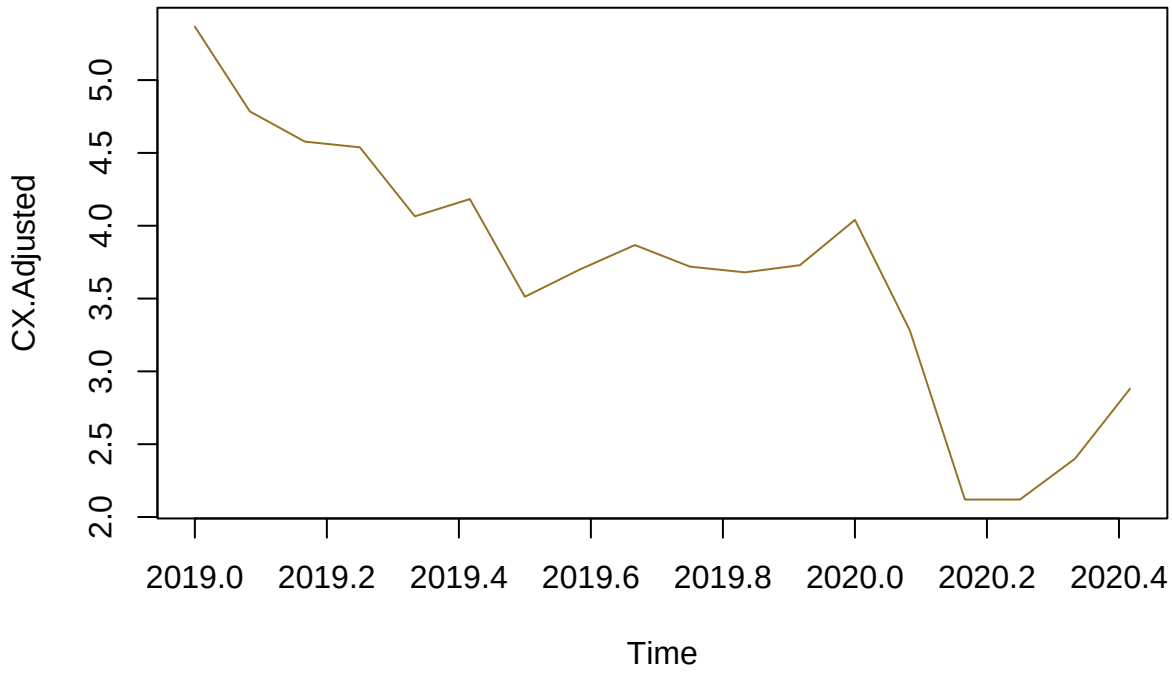
```r
base1920 <- CX[,6] # para solo usar la columna de los precios de cierre ajustados
ts_base_1920 <- ts(base1920, start=2019, frequency = 12) # creamos el objeto de series de tiempo

# Datos Reales
ts_base_1920
```

```
##           Jan      Feb      Mar      Apr      May      Jun      Jul      Aug
## 2019 5.366882 4.784812 4.577634 4.538172 4.064624 4.183011 3.512151 3.699597
## 2020 4.040000 3.280000 2.120000 2.120000 2.400000 2.880000
##           Sep      Oct      Nov      Dec
## 2019 3.867312 3.719328 3.679866 3.729194
## 2020
```

Echemosle un vistazo a esos precios reales

```r
plot(ts_base_1920,col="#977328")
```



Para decir cual fue la mejor estimacion de todas, no perdemos nada en comparar las 4 estimaciones con los datos reales. La prediccion con cada modelo es:

```r
# Modelo.a: AUTO.ARIMA
   pred.a <- forecast::forecast(autoarima,18)
   pronostico.a <- exp(as.data.frame(pred.a)[,1] )^(1/lambda)           # Transformacion para tener los
(  pronostico.a <-ts(pronostico.a, start=2019, frequency = 12)  )
```

```
##           Jan      Feb      Mar      Apr      May      Jun      Jul      Aug
## 2019 5.900584 5.900584 5.900584 5.900584 5.900584 5.900584 5.900584 5.900584
## 2020 5.900584 5.900584 5.900584 5.900584 5.900584 5.900584
##           Sep      Oct      Nov      Dec
## 2019 5.900584 5.900584 5.900584 5.900584
## 2020
```

```
# Modelo1: ARIMA(0,1,0)
   pred1 <- forecast::forecast(modelo1,18)
   pronostico1 <- exp(as.data.frame(pred1)[,1] )^(1/lambda)           # Transformacion para tener los d
(  pronostico1 <-ts(pronostico1, start=2019, frequency = 12)  )                        # Predicciones
```

```
##           Jan      Feb      Mar      Apr      May      Jun      Jul      Aug
## 2019 5.900584 5.900584 5.900584 5.900584 5.900584 5.900584 5.900584 5.900584
## 2020 5.900584 5.900584 5.900584 5.900584 5.900584 5.900584
##           Sep      Oct      Nov      Dec
## 2019 5.900584 5.900584 5.900584 5.900584
## 2020
```

```
# Modelo2: ARIMA(1,1,1)
   pred2 <- forecast::forecast(modelo2,18)
   pronostico2 <- exp(as.data.frame(pred2)[,1] )^(1/lambda)           # Transformacion para tener los d
(  pronostico2 <-ts(pronostico2, start=2019, frequency = 12)  )                        # Predicciones
```

```
##           Jan      Feb      Mar      Apr      May      Jun      Jul      Aug
## 2019 6.162725 6.046053 6.097081 6.074590 6.084470 6.080124 6.082034 6.081194
## 2020 6.081455 6.081449 6.081452 6.081450 6.081451 6.081451
##           Sep      Oct      Nov      Dec
## 2019 6.081564 6.081401 6.081473 6.081441
## 2020
```

```
# Modelo3: ARIMA(1,1,2)
   pred3 <- forecast::forecast(modelo3,18)
   pronostico3 <- exp(as.data.frame(pred3)[,1] )^(1/lambda)           # Transformacion para tener los d
(  pronostico3 <-ts(pronostico3, start=2019, frequency = 12)  )
```

```
##           Jan      Feb      Mar      Apr      May      Jun      Jul      Aug
## 2019 6.044494 5.819309 5.826997 5.826729 5.826738 5.826738 5.826738 5.826738
## 2020 5.826738 5.826738 5.826738 5.826738 5.826738 5.826738
##           Sep      Oct      Nov      Dec
## 2019 5.826738 5.826738 5.826738 5.826738
## 2020
```

En base a los datos que realmente ocurrieron, el mejor modelo fue el 3: el ARIMA(1,1,2)

21. ¿Le podría haber resultado conveniente invertir ese 1, 000, 000 de pesos en diciembre 2018?

No! Para nada hubiera sido conveniente invertir 1 000 000 pesos en diciembre 2018, en su lugar habria sido excelente elegir una venta en corto. Entre mas paciencia hubiesemos tenido, mayor habria sido la ganancia.

OBSERVESE que dicha estrategia, la habria respaldado claramente el modelo ARIMA(1,1,2) . Y evidentemente, no se habria equivocado