Ciencia de los datos

Análisis de pirámides poblacionales

López Cruz, Álvaro bo0121

Clustering

Para el clustering, vamos a utilizar los datos del fichero **pjangroup.csv** extraído de Eurostat, una agencia de la comisión europea. Contiene las pirámides poblacionales de 46 países europeos en los años 1991, 2001, 2011 y 2021. Entre los datos que contiene están la población total y desagregada por sexos y grupos de edad por rangos de 5 años.

(a) Carga los datos del fichero `pjangroup.csv` en un data frame.

```
population_df = as.data.frame(
read.csv("C:/repos/population_pyramid_analysis/pjangroup.csv",header=T,sep=";",d
ec=".") )
```

En nuestro caso, vamos a estudiar estos datos mediante un clustering, para los años 2001 y 2021 por separado.

(b) Selecciona todos los datos de 1991 o 2001, a tu elección, y quédate solo con las columnas correspondientes a país, sexo, grupo de edad y datos.

```
# Para elegir los datos de 2001
population_df = population_df[population_df$TIME_PERIOD == '2001',]

# Para elegir los datos de 2021
population_df = population_df[population_df$TIME_PERIOD == '2021',]
```

Para ambos casos, vamos a realizar un preprocesamiento de los datos, además de un escalado y una ponderación. Para ello,utilizaremos la librería Tidyverse, que nos permitirá obtener un dataframe en el que tenemos una fila por cada país y una columna por cada combinación de sexo/grupo de edad. La función **pivot_wider()** nos permite hacer exactamente esto, de forma que, conseguimos el resultado deseado con la siguiente línea de código, donde OBS_VALUE es el valor de población.

(c) Lee la sección “Tidy data” del libro “R for data science”, `https://r4ds.had.co.nz/tidy-data.html`.
Usa la función `pivot wider` de la librería `tidyverse` para convertir los datos en una estructura apropiada para realizar la clusterización: una fila por cada país y una columna por cada combinación sexo/grupo de edad.

```
population_df %>% pivot_wider(names_from = c("sex", "age"), values_from =
"OBS_VALUE")
```

Con esto, obtenemos un dataframe como el siguiente:

geo	F_TOTAL	M_TOTAL	T_TOTAL	F_Y00-05	F_Y05-09	F_Y10-14	F_Y15-19	F_Y2 24
AL	1535822	1527496	3063318	*NA*	*NA*	*NA*	*NA*	
AM	1672200	1543100	3215300	*NA*	*NA*	*NA*	*NA*	
AT	4139842	3881104	8020946	202154	230403	230299	237102	2
AZ	4126500	3954500	8081000	*NA*	*NA*	*NA*	*NA*	
BA	1932756	1856962	3789717	*NA*	*NA*	*NA*	*NA*	
BE	5245395	5018019	10263414	282540	298421	300911	297102	3
BG	4182045	3967423	8149468	166176	198717	252256	269408	3
BY	5302718	4687717	9990435	*NA*	*NA*	*NA*	*NA*	
CH	3684357	3519698	7204055	193501	204884	207446	202513	2

En este punto tenemos los datos bien estructurados, pero podemos observar valores **NA** que debemos eliminar. Una vez eliminados, tenemos un dataframe con una columna por país y una columna por grupo de edad sin valores nulos. Aún así, aún nos falta preprocesado por hacer, a continuación vamos a escalar y ponderar los datos para conseguir un clustering correcto.

(d) Si hay algún país en el que falten datos (`NA`), elimínalo.

```
population_df = population_df[complete.cases(population_df),]
```

Para esto, vamos a utilizar las columnas X_TOTAL que se ven en la imágen, donde X son cada uno de los sexos (Male, Female, Total). Lo que haremos será dividir cada uno de los valores de grupo de edad entre el valor total del sexo en cuestión. Por ejemplo dividiremos F_Y00-05 entre F_TOTAL para cada uno de los valores de cada país del dataframe, y así para todas las columnas. Después de este proceso, eliminamos las columnas X_TOTAL y geo ya que no queremos que influyan en el clustering, y tenemos los datos procesados y preparados para utilizar. Aprovecharemos la vectorización de R para hacer estos cálculos de la forma más óptima posible.

(e) Haz un clustering de los países usando los datos de población por sexo y grupo de edad. Puedes o no incluir el grupo “Total”. Considera cómo ponderar o escalar los datos. Usa el método elbow para determinar el valor más apropiado de k.

```
# Normalize data (21 age_groups, 3 sexes)
F_total = population_df$F_TOTAL
M_total = population_df$M_TOTAL
T_total = population_df$T_TOTAL

population_df[4:21] = population_df[4:21] / F_total
population_df[22:39] = population_df[22:39] / M_total
population_df[40:57] = population_df[40:57] / M_total
```

```r
# Clustering
kmdata_population = as.matrix(population_df)
kmdata = kmdata_population[,1:54]
wss = numeric(15)
for (k in 1:15) wss[k] <- sum(kmeans(kmdata, centers=k, nstart=25)$withinss)
# Pintamos la gráfica para poder ver dónde está el codo
plot(1:15, wss, type="b", xlab="Number of Clusters", ylab="Within Sum of
Squares")

km = kmeans(kmdata, 3, nstart = 25)
```

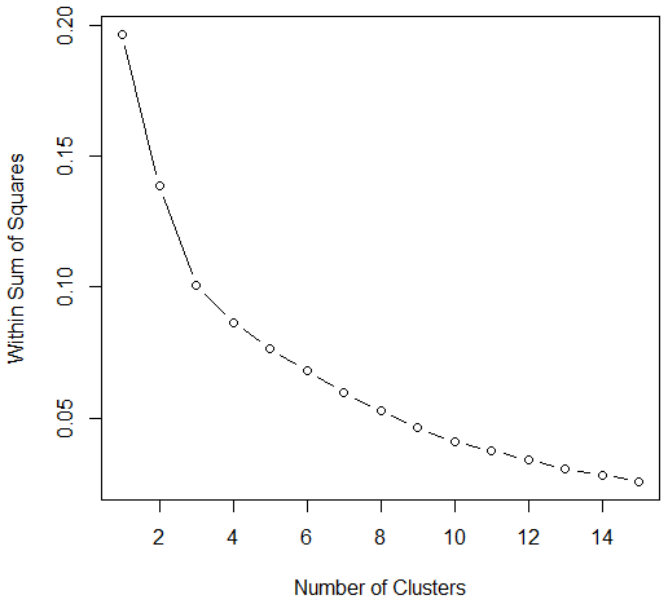
(f) Repite el proceso con los datos de 2011 o 2021, a tu elección.

(g) Estudia las características de los clústeres obtenidos: ¿Encuentras alguna característica común en los países de cada clúster? Busca cómo representar una pirámide poblacional y haz una gráfica para un país aleatorio de cada uno de los clústeres obtenidos (sólo para el año más moderno que hayas estudiado).

En este caso hemos decidido estudiarlo para 2001 y 2021, a continuación se explican las características de los clusters obtenidos y las representaciones de las pirámides poblacionales.

2001

Para el año 2001, antes de lanzar el clustering vamos a utilizar el método elbow para estimar el número de clusters óptimo. Obtenemos la siguiente gráfica:



Podemos ver que el codo se sitúa en 3, así que lanzamos el clustering con este número de clusters y los datos preprocesados anteriormente, tal y como se ve en el código anterior. El primer cluster que obtenemos es el siguiente:

Código de país	Nombre de país	Clúster
AT	Austria	1
BE	Belgium	1
CH	Switzerland	1
DE	Germany	1
DK	Denmark	1
FI	Finland	1
FR	France	1
IT	Italy	1
LI	Liechtenstein	1
LU	Luxembourg	1
NL	Netherlands	1
NO	Norway	1
SE	Sweden	1
UK	United Kingdoms	1

Podemos observar que en este clúster tenemos países de tamaño grande y mediano bastante desarrollados económicamente, entre ellos Francia, Alemania, Gran Bretaña o Alemania. Se puede ver también que España no está entre estos países pese a su parecido actual con algunos de ellos en la actualidad, posiblemente porque en 2001 aún sufría consecuencias de algunas guerras y problemas en su pirámide poblacional.

En el segundo clúster obtenemos:

Código de país	Nombre de país	Clúster
BG	Bulgaria	2
CZ	Czechia	2
EE	Estonia	2
EL	Greece	2

ES	Spain	2
HR	Croatia	2
HU	Hungary	2
LV	Latvia	2
PT	Portugal	2
RO	Romania	2
RS	Serbia	2
SI	Slovenia	2

En este clúster podemos observar que la gran característica común es su localización geográfica. La mayoría de estos países se encuentran en el este de Europa, es interesante observar que en el 2001 España y Portugal tenían más en común con Europa del este que con países como Alemania o Francia en cuanto a población.

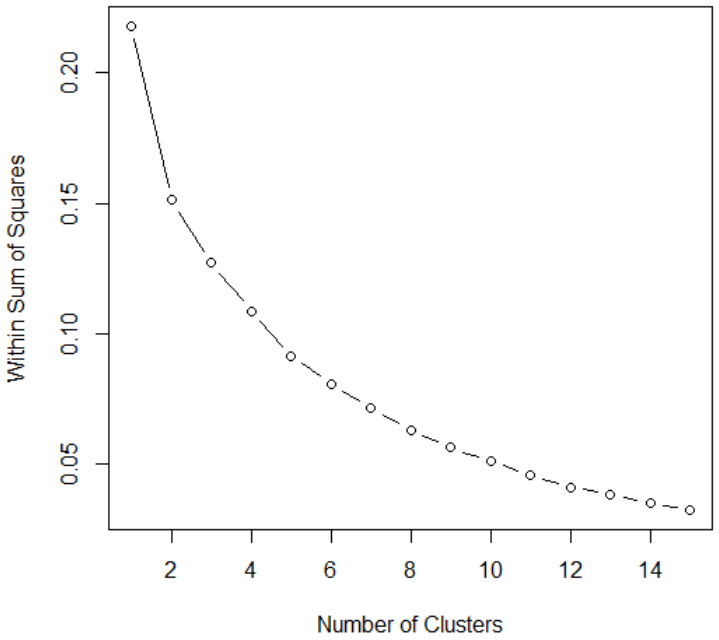
Por último, el tercer clúster:

Código de país	Nombre de país	Clúster
CY	Cyprus	3
IE	Ireland	3
IS	Iceland	3
LT	Lithuania	3
ME	Montenegro	3
MK	North Macedonia	3
MT	Malta	3
PL	Poland	3
SK	Slovakia	3

En el clúster número 3 podemos ver países más pequeños de Europa del Este e islas como Malta o Islandia. Parece que los países pequeños de esta zona y las pequeñas islas repartidas tienen similitud en cuanto a pirámide poblacional.

2021

Para 2021, de la misma forma vamos a utilizar el método del codo para determinar el número de clusters óptimo:



En esta ocasión no está tan claro dónde está el codo, pero vamos a elegir 3 de nuevo como el número de clusters, después de haber probado varios valores y ver que es el que mejor se ajustaba. Lanzamos el algoritmo kmeans y el resultado es el siguiente:

Para el cluster 1:

Código de país	Nombre de país	Clúster
AT	Austria	1
BE	Belgium	1
CH	Switzerland	1
DE	Germany	1
DK	Denmark	1
EL	Greece	1
ES	Spain	1
FI	Finland	1

FR	France	1
HR	Croatia	1
IT	Italy	1
LI	Liechtenstein	1
LT	Lithuania	1
NL	Netherlands	1
PT	Portugal	1
SE	Sweden	1

Podemos ver una gran diferencia de este clúster con el clúster número 1 del año 2001, países medianos y grandes bastante desarrollados. Es interesante ver cómo hay algunos países que en este lapso de tiempo se han desarrollado hasta entrar a este cluster, cuando en 2001 no estaban, por ejemplo España, Portugal o Grecia.

En el cluster número 2 tenemos:

Código de país	Nombre de país	Clúster
AL	Albania	2
CY	Cyprus	2
IE	Ireland	2
IS	Iceland	2
LU	Luxemburgo	2
ME	Montenegro	2
MK	North Macedonia	2
NO	Norway	2
TR	Turkey	2

Este clúster parece seguir la misma lógica que el cluster número 3 del 2001, estando compuesto en su mayoría por países más pequeños o poco habitados como islas. Se pueden ver algunas diferencias entre 2001 y 2021 en países específicos como por ejemplo Luxemburgo, que ahora tiene más parecido con este cluster que con países medianos y grandes más desarrollados.

Por último, en el tercer cluster:

Código de país	Nombre de país	Clúster
BG	Bulgaria	3
CZ	Czechia	3
EE	Estonia	3
HU	Hungary	3
LV	Latvia	3
MT	Malta	3
PL	Poland	3
RO	Romania	3
RS	Serbia	3
SI	Slovenia	3
SK	Slovakia	3
UA	Ukraine	3

De nuevo podemos observar una gran similitud de este clúster con el clúster número 2 de 2001, estando compuesto en su mayoría por países pequeños y medianos de Europa del este. Podemos observar que algunos países como España y Portugal han pasado a un cluster con países más desarrollados económicamente, y que otros países como Malta, que es una isla pequeña, ahora parece tener más parecido con este cluster, lo que indica que puede haberse desarrollado bastante en las últimas 2 décadas.

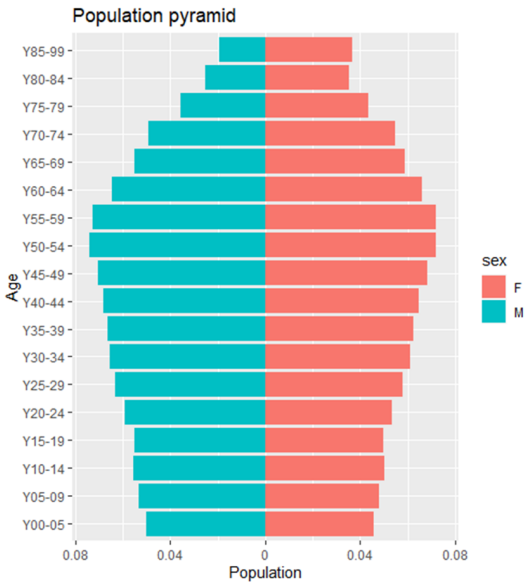
Pirámides poblacionales

Vamos a utilizar los datos obtenidos del clustering del 2021 para pintar las pirámides poblacionales de ciertos países para sacar conclusiones. Sin embargo, primero vamos a utilizar los datos de los centros de los clusters, que aunque no representan ningún país en concreto, pueden ser un buen dato para mostrar una pirámide poblacional que en media represente mejor a cada cluster.

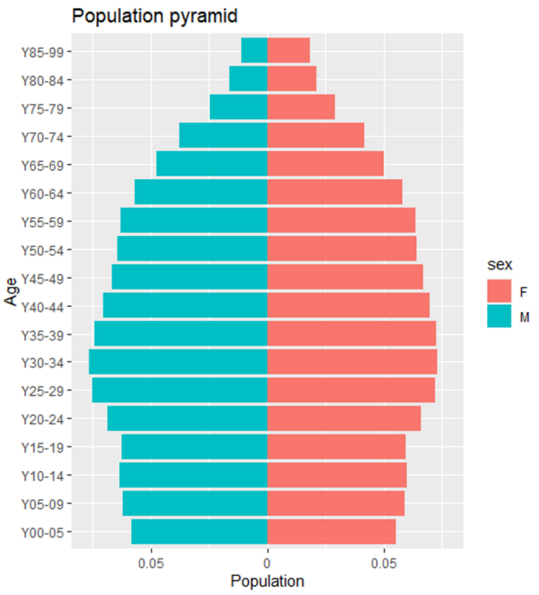
Por lo tanto, primero vamos a mostrar las gráficas de los centros de los clúster para ver la información general, y terminaremos el análisis del clustering con las gráficas de países específicos dentro de cada clúster.

Para pintar estas gráficas se han desarrollado 2 funciones, que permiten utilizarlas las veces que sea necesario:

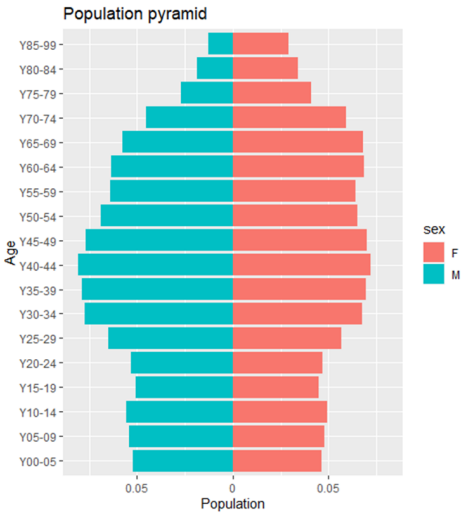
Cluster 1



Cluster 2



Cluster 3



Viendo el cluster 1, podemos observar que tiene forma de jarrón, el pico de número de habitantes está en el grupo de edad de los 50-60 años, y decae mientras la población sigue envejeciendo, pero sin demasiada pendiente. Esto puede significar que los países que forman el cluster tienen una buena calidad de vida y atención a la salud de los más mayores, aumentando su esperanza de vida. Estas conclusiones cuadran con los países que forman el cluster, ya que como hemos dicho son países medianos y grandes que están bien desarrollados económicamente.

En el cluster número 2, podemos ver una forma más de árbol, significando un porcentaje menor de niños respecto a adultos. También se puede observar que el pico es más notable

que en el cluster 1, teniendo de media menos población anciana. En general vemos una población distribuida más hacia la parte de abajo de la gráfica, significando menos población mayor.

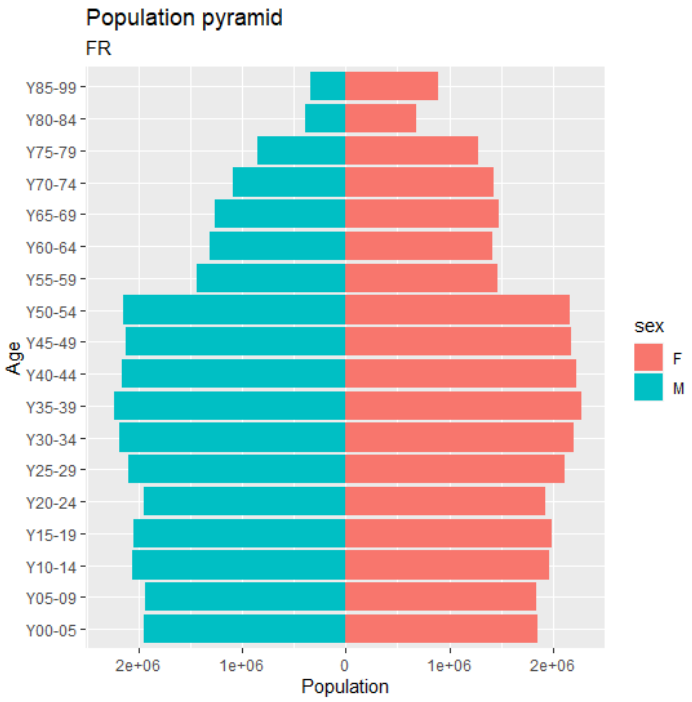
Finalmente, en el cluster 3 es en el que podemos observar más curvas en la gráfica. Podemos ver una diferencia notable entre el número de habitantes de edades tempranas y edades medias, por lo que podemos asumir una baja natalidad. De nuevo podemos observar un rápido decrecimiento de la población cuanto más avanza el grupo de edad, siendo mucho más notorio en este caso para hombres que para mujeres lo cual podría significar diferencias sociales en este aspecto.

Para pintar estas gráficas se han desarrollado 2 funciones, que permiten utilizarlas las veces que sea necesario:

```
# Population pyramids
# Function for plotting centers
plot_pyramid_df <- function(df) {

  g = ggplot(df, aes(x = age, fill = sex,
                     y = ifelse(test = sex == "M",
                                yes = -Population, no = Population))) +
       geom_bar(stat = "identity") +
       scale_y_continuous(labels = abs, limits = max(df$Population) * c(-1,1)) +
       coord_flip()
  g + labs(x = "Age", y = "Population") +labs(title = "Population pyramid")
}
# Function for plotting countries
plot_pyramid_country <- function(country) {
  df = initial_df[initial_df$geo==country,]
  df = df[df$age!='TOTAL',]
  df = df[df$sex!='T',]
  g = ggplot(df, aes(x = age, fill = sex,
                     y = ifelse(test = sex == "M",
                                yes = -OBS_VALUE, no = OBS_VALUE))) +
       geom_bar(stat = "identity") +
       scale_y_continuous(labels = abs, limits = max(df$OBS_VALUE) * c(-1,1)) +
       coord_flip()

  g + labs(x = "Age", y = "Population" +labs(title = "Population pyramid",
subtitle = country)
}
```

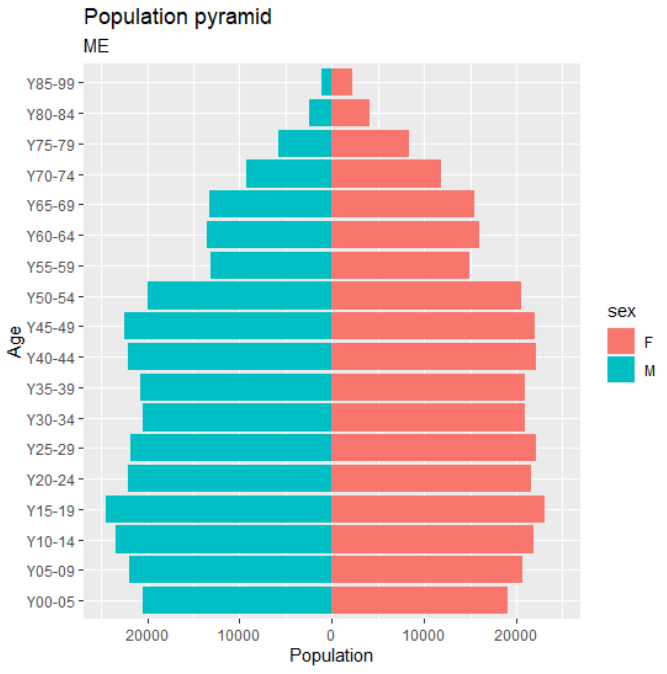
A continuación vamos a mostrar 3 países específicos de cada cluster.

Para el cluster 1, vamos a representar la pirámide poblacional de Francia:



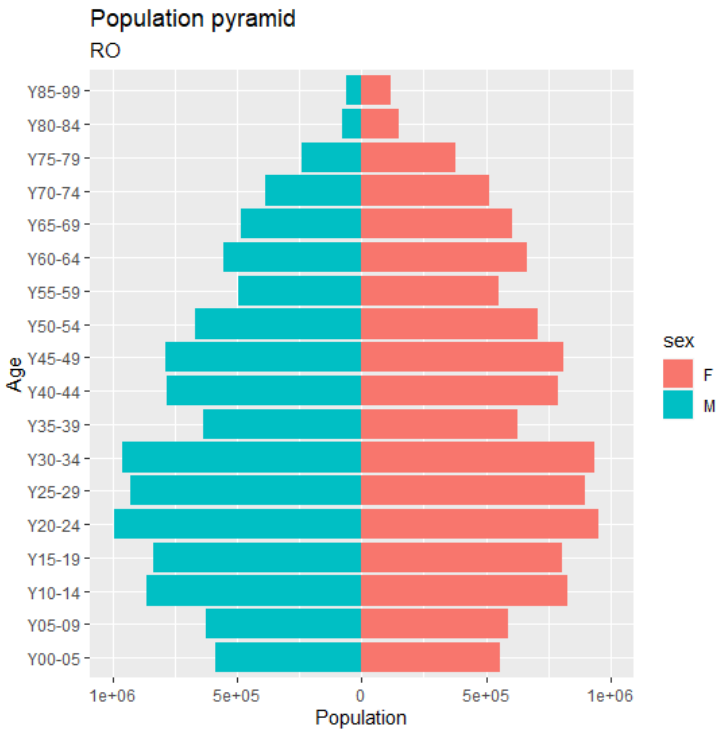
Podemos ver gran similitud con la pirámide del cluster 1, aunque vemos que este caso específico presenta una diferencia más notoria en cuanto a población anciana.

Para el cluster 2, vamos a representar a Montenegro:



De nuevo, vemos que sigue la estructura general de la representación de su clúster, aunque su caso específico tiene muy poca población anciana, y una gran diferencia con el resto de grupos de edad. Además podemos ver una natalidad un poco mayor que el resto de países de este segundo clúster. Finalmente, destacar que el número de habitantes global es mucho menor que en el resto de clusters, unos 20000 de máximo en determinados grupos de edad, mientras en otros clusters es del orden de millones.

Finalmente, para el clúster 3, vamos a representar a Rumanía:



Su distribución es similar a la de su clúster en general, aunque podemos ver una bajada repentina en el grupo de edad entre 35-39 años, que podría deberse a una posible migración de habitantes a otros países en busca de oportunidades. También es interesante el pico superior de la gráfica, donde la diferencia entre mujeres y hombres es de más del doble.

Conclusiones

En resumen, el clustering realizado separa los países en 3 clusters diferenciados:

- Clúster de países medianos y grandes, bien desarrollados económica y socialmente.
- Clúster de países situados al este de Europa, pequeños y medianos cuyo desarrollo económico es menor que el de países del clúster anterior.
- Clúster de países pequeños, como por ejemplo islas, donde la pirámide poblacional normalmente no encaja en ninguno de los otros 2 clusters.

Podemos ver algunas diferencias entre 2001 y 2021, sobre todo si miramos países específicos, pero en general el clustering que se realiza es muy similar.

El clustering es una técnica muy útil que nos ha permitido ver características de los países, relacionarlos entre ellos, y poder agruparlos en grupos de características similares de manera muy sencilla.

Series temporales

Vamos a estudiar datos relativos a Francia

Francia

(a) Carga los datos del fichero `prc hicp midx.csv` en un data frame.

```
df = as.data.frame(
read.csv("C:/repos/population_pyramid_analysis/prc_hicp_midx.csv",header=T))
```

(b) Escoge dos países, que pertenezcan a distintos clusters en la primera clasificación del apartado anterior.
(c) Comprueba que dichos países tienen, en ambos ficheros, una serie continua de datos consecutivos sin datos perdidos (`NA`) intermedios. De no ser así, escoge un país distinto.

```
df_fr = df[df$geo == "FR",]
```

Primero vamos a estudiar Francia.

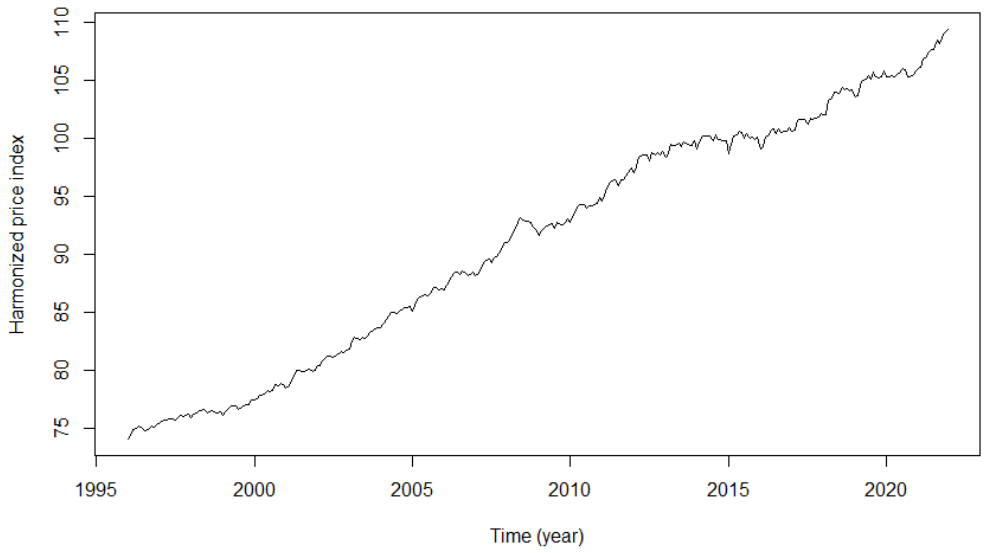
(d) Para cada uno de ellos, crea una serie temporal con los datos de índice armonizado de precios. Reserva los 3 últimos datos de cada serie para validar los modelos.

```
df_fr_train = df_fr[1:312,]
df_fr_test = df_fr[313:315,]

# Create a time series object from 2001 to 2021
hicp = ts(as.numeric(as.character(df_fr_train[,2])),start=1996, frequency = 12)
# Test set
hicp_test = ts(as.numeric(as.character(df_fr_test[,2])),start=1996+312/12,
frequency = 12)
```

(e) Ajusta a cada una de las dos series un modelo ARIMA posiblemente estacional (¿con qué periodo?). Razona los parámetros escogidos d, D, mediante la gráfica de la serie y sus diferencias, y las varianzas; p, q y P, Q, usando las gráficas de ACF y PACF). Opcionalmente, compara con el resultado con el de auto.arima.

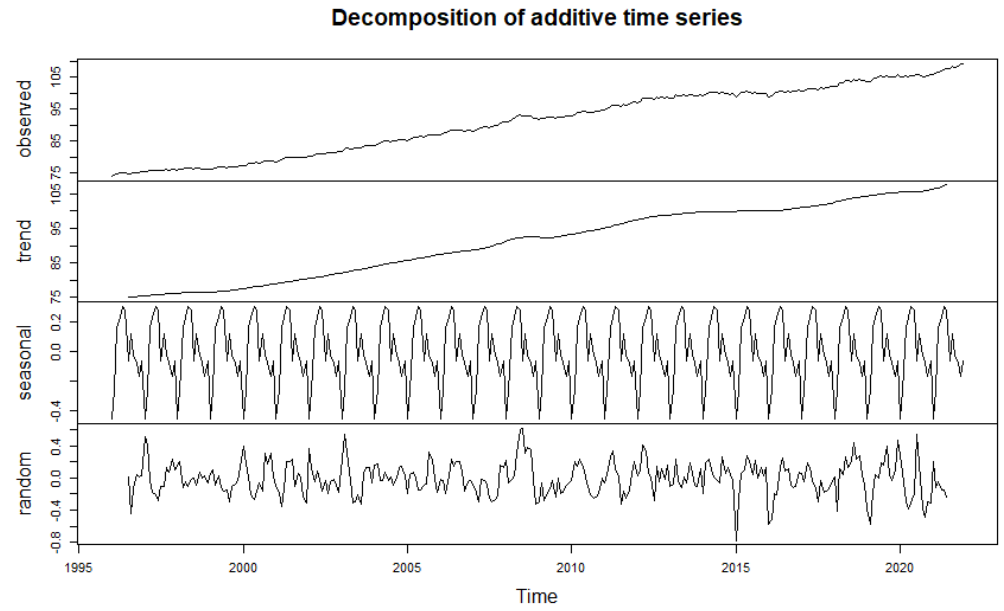
Primero vamos a realizar un estudio general de los datos:

```
plot(hicp, xlab = "Time (year)", ylab = "Harmonized price index")
```



Los picos a intervalos relativamente regulares nos pueden dar la idea de que esta serie podría ser estacionaria.

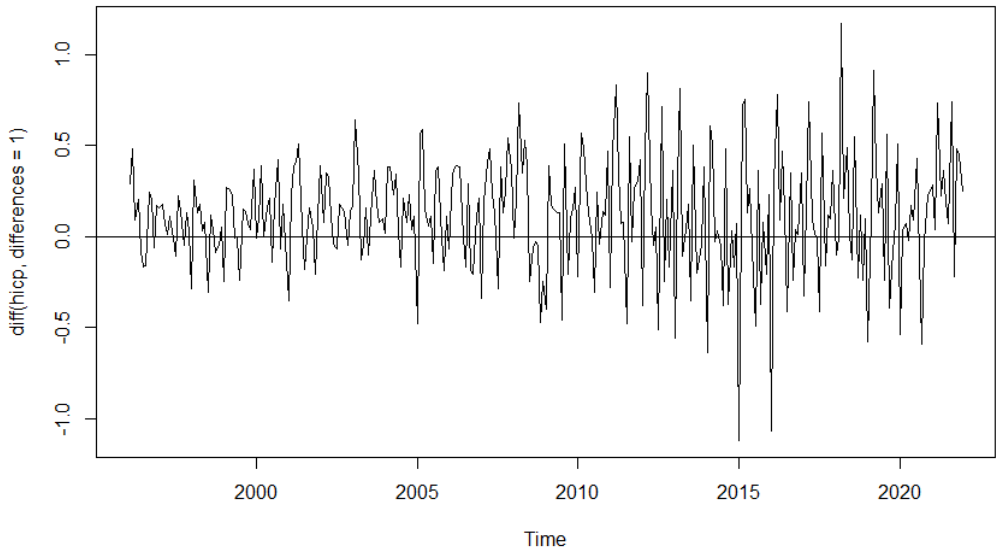
```
plot(decompose(hicp))
```



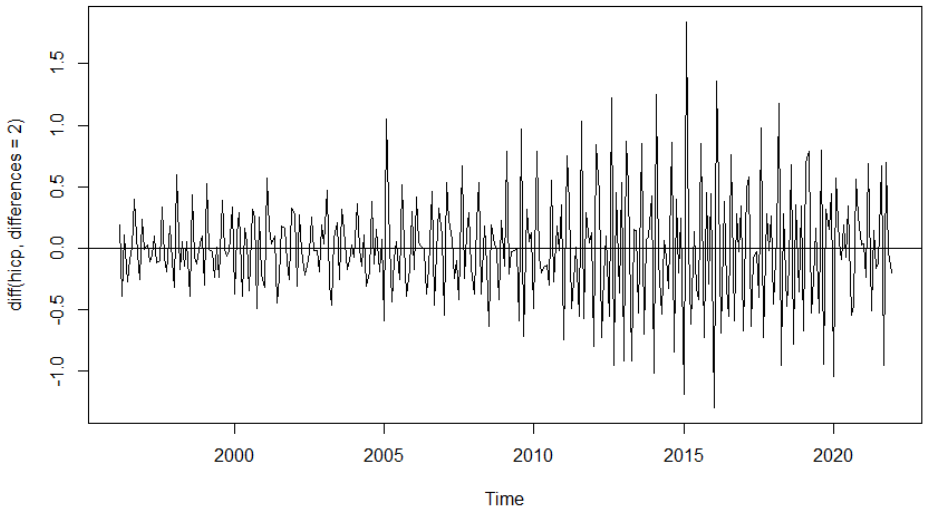
Vemos que los intervalos del ruido y de la parte estacionaria de la serie son comparables, una pista más que nos indica que la serie podría tener estacionalidad.

Ahora vamos a buscar los valores óptimos de d y D:

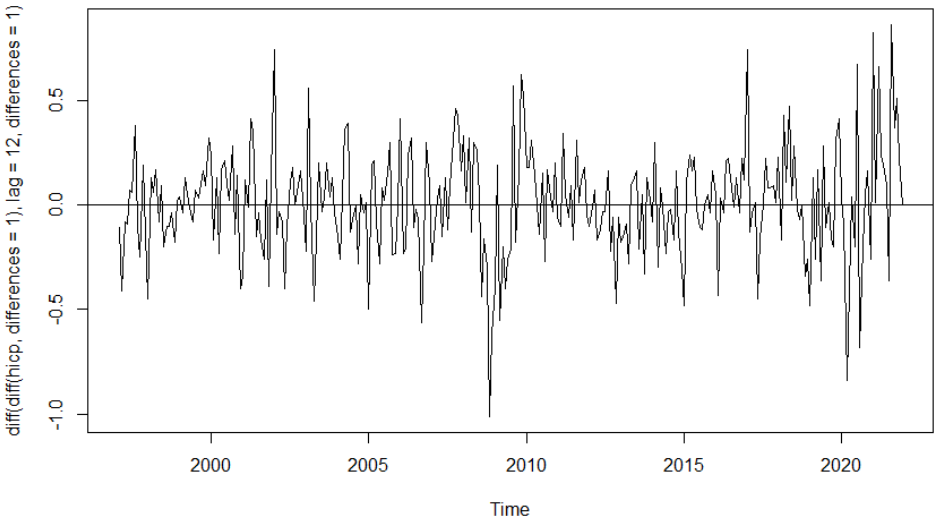
```r
# Get optimal d values
plot(diff(hicp, differences=1))
abline(a=0, b=0)
plot(diff(hicp, differences=2))
abline(a=0, b=0)
plot(diff(diff(hicp, differences=1), lag=12, differences=1))
abline(a=0, b=0)

var(diff(hicp))
var(diff(hicp, differences=2))
var(diff(diff(hicp, differences=1), lag=12, differences=1))
```



Varianza = 0.0957



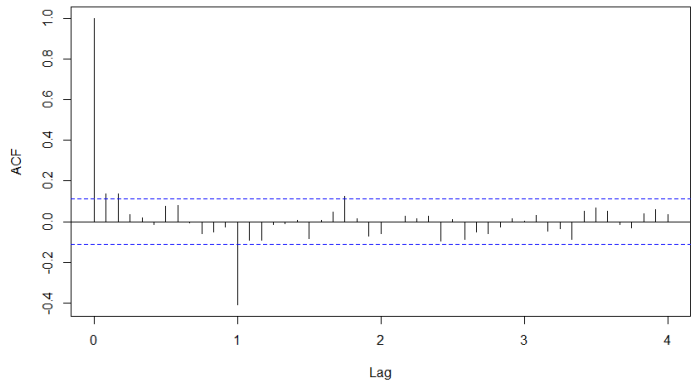
Varianza = 0.2089

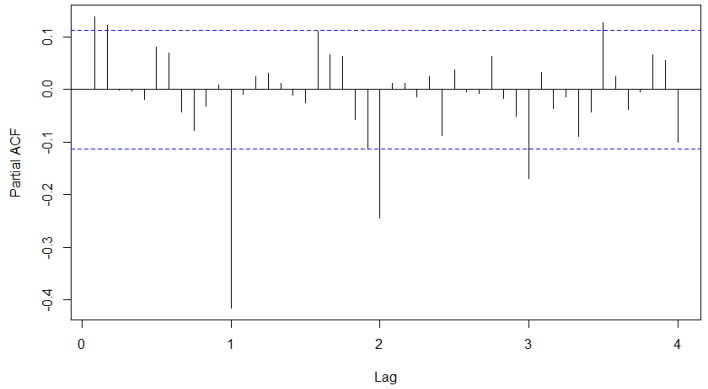


Varianza = 0.06731

Viendo las gráficas y sabiendo que probablemente haya estacionalidad, vamos a elegir d=1, D=1 para nuestro modelo.

Vamos a estudiar las gráficas de ACF y PACF:





Viendo la gráfica acf podemos ver un pico en 1, que desaparece y todos los demás valores están dentro del intervalo de confianza. El PACF lo confirma, ya que podemos ver ese pico en 1 que va decreciendo y termina dentro del intervalo de confianza. Por lo tanto podemos concluir que un buen modelo sería una media móvil (MA) de orden 1.

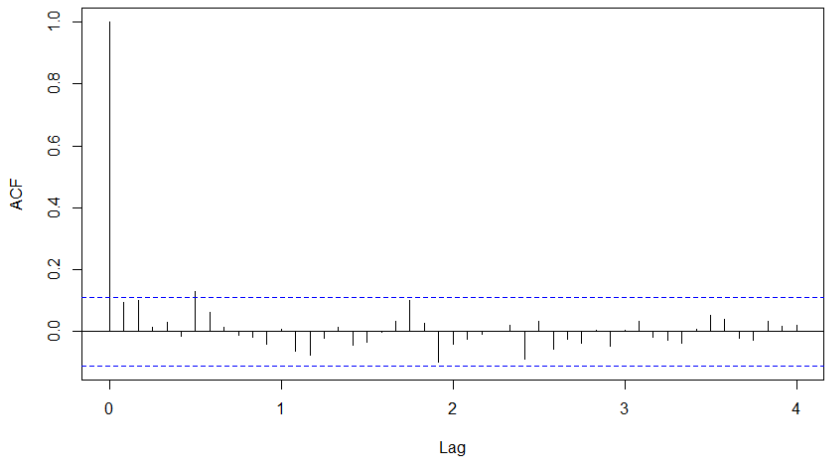
Sabiendo esto, nuestro primer modelo será dejar la componente estacional a 0, y meter un MA de orden 1 en la parte estacional, es decir, un modelo ARIMA (0,1,0)x(0,1,1) con s = 12.

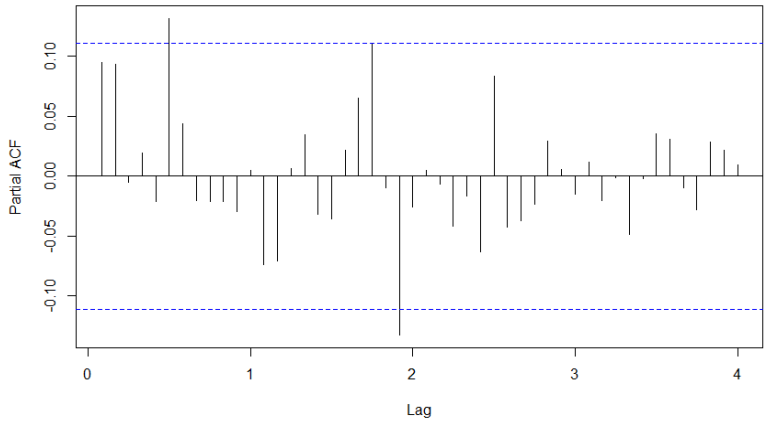
```
arima_1 <- arima (hicp,order=c(0,1,0),
                  seasonal = list(order=c(0,1,1),period=12))
arima_1

AIC(arima_1,k = log(length(hicp)))
```

El AIC que obtenemos es -53.03

Ahora vamos a estudiar los residuos para ver lo que pasa con la parte no estacional.

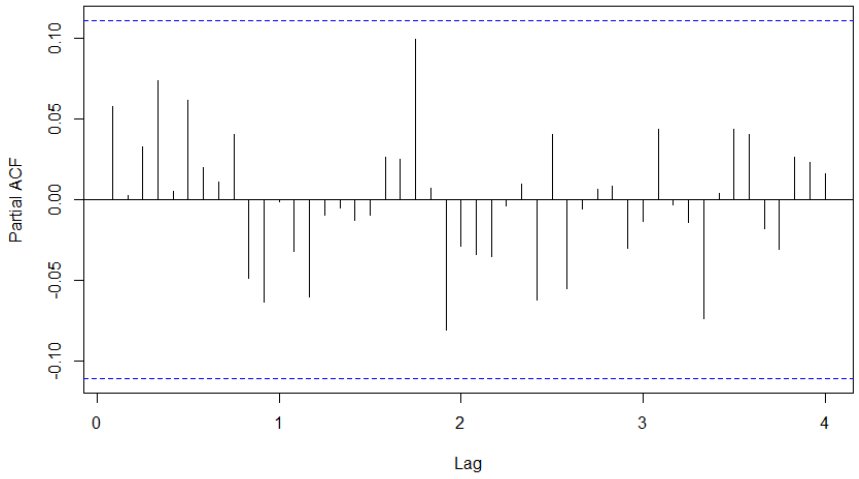
```
# examine ACF and PACF of the residuals
acf(arima_1$residuals, lag.max=48, main="")
pacf(arima_1$residuals, lag.max=48, main="")
```

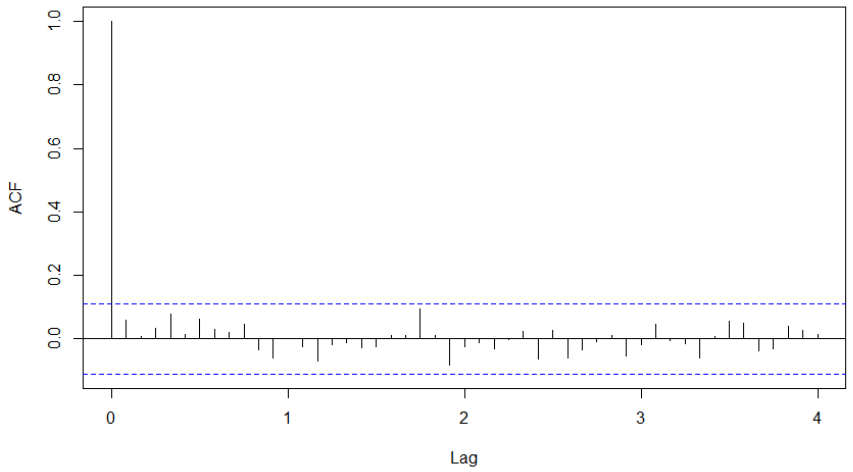




La gráfica ACF no nos da información muy relevante. Sin embargo, viendo la gráfica PACF podemos ver que tenemos picos por ejemplo hasta el 6. Vamos a intentar distribuir este 6 en la componente no estacionaria para ver como se comporta el modelo. Para ello, entrenamos entonces un modelo ARIMA (3,1,3)x(0,1,1). El AIC obtenido es -25.2, que es mejor que el primer modelo que entrenamos, por lo tanto seguiremos con este. Vamos a ver las gráficas ACF y PACF de este último modelo

```
arima_2 <- arima (hicp,order=c(3,1,3),
                  seasonal = list(order=c(0,1,1),period=12))

AIC(arima_2,k = log(length(hicp)))

# examine ACF and PACF of the residuals
acf(arima_2$residuals, lag.max=48, main="")
pacf(arima_2$residuals, lag.max=48, main="")
```

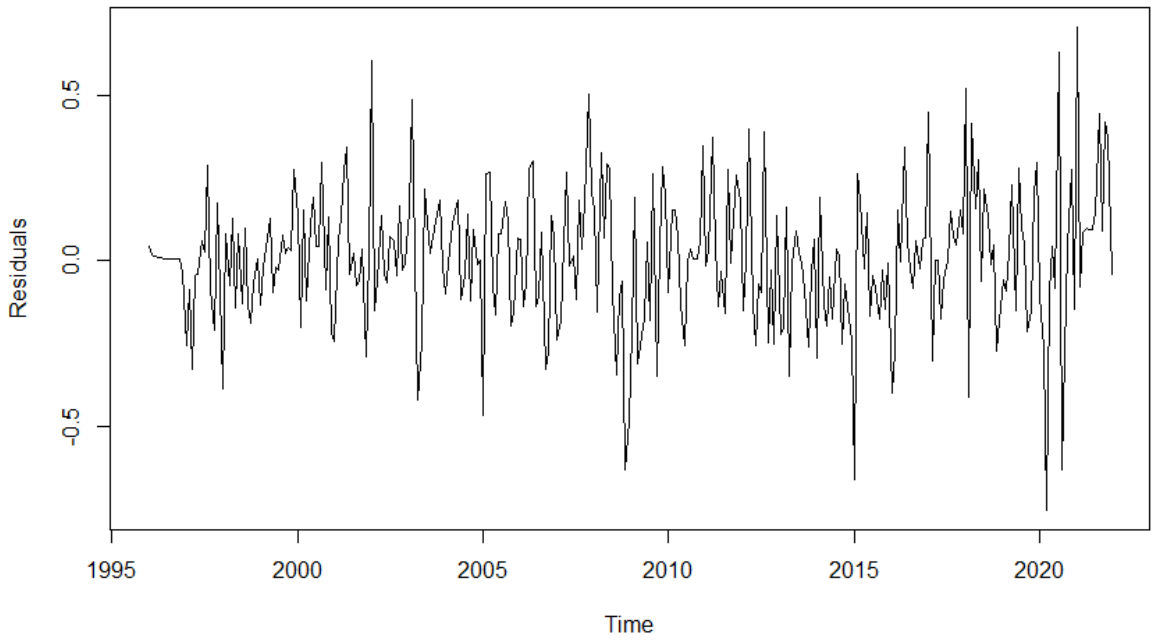




Vemos que las gráficas son mejores que con el primer modelo ARIMA, todos los valores están dentro del intervalo de confianza menos el primero de la gráfica ACF, que siempre debe ser 1.

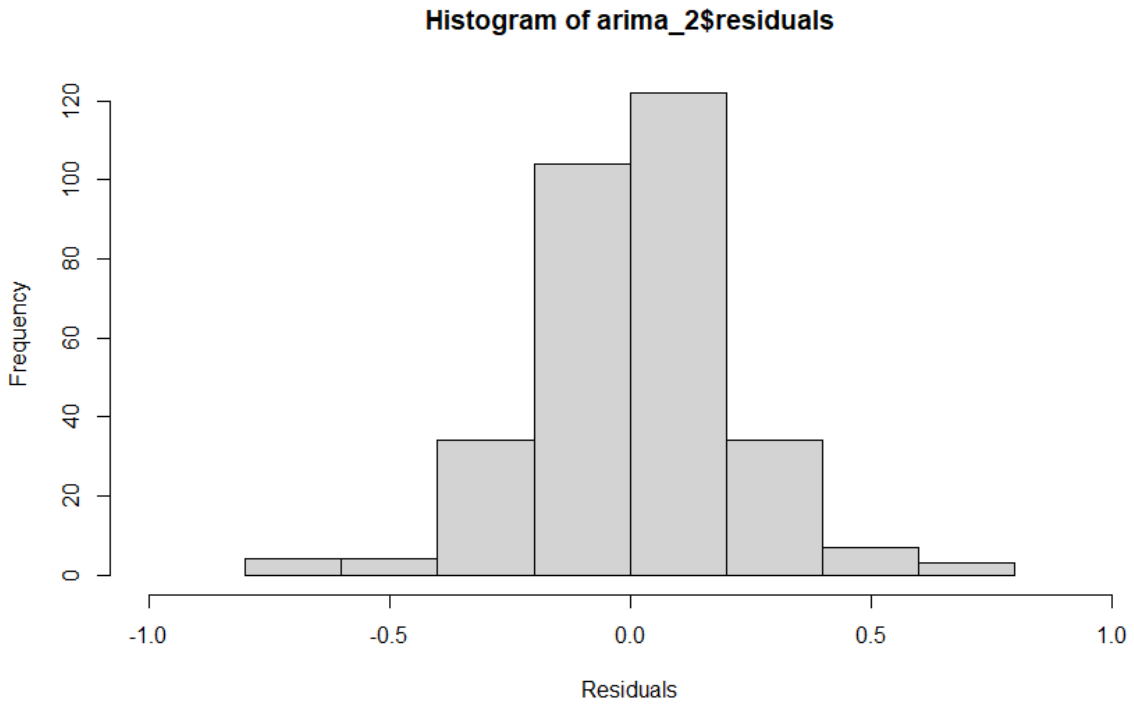
El siguiente paso es hacer pruebas relativas a la normalidad de los datos:

```
plot(arima_1$residuals, ylab = "Residuals")
abline(a=0, b=0)
```



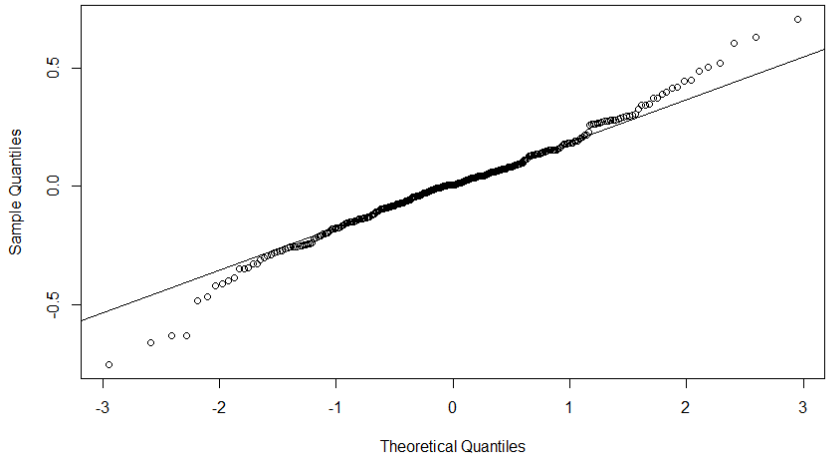
No podemos sacar muchas conclusiones de la gráfica de los residuos, vamos a continuar pintando un histograma, que debería adecuarse a una distribución normal:

```
hist(arima_2$residuals, xlab="Residuals", xlim=c(-1,1))
```



No podemos asegurar la normalidad de los datos, ya que la mayoría de los datos están muy cerca de cero, y en los extremos hay mucha menor cantidad, aunque no se aleja demasiado de una normal. Vamos a seguir con otra gráfica para apoyar:

```
qqnorm(arima_2$residuals, main="")
qqline(arima_2$residuals)
```



La gráfica cuartil-cuartil muestra una curvatura notable, aunque no se aleja demasiado de la recta esperada, lo cual indica que la distribución de los residuos no se asemeja del todo con una normal, aunque tampoco se aleja demasiado. Podemos esperar de este modelo predicciones que podrían no ser del todo buenas.

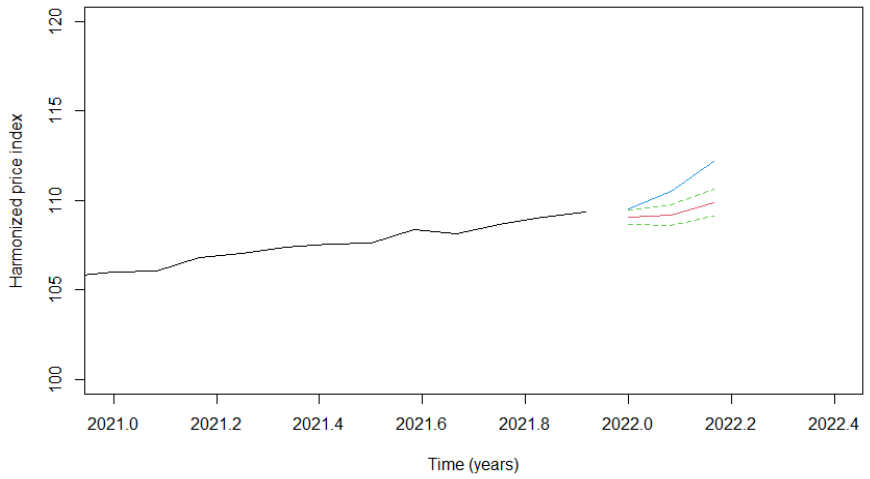
Si comparamos con el modelo generado por auto arima, nos propone un modelo arima (2,1,0)x(0,1,1). La parte estacional queda igual que la que habíamos estimado previamente, aunque distribuye los valores a la parte no estacionaria de manera distinta, consiguiendo un AIC algo menor. Estudiando la normalidad no se ve ninguna mejora, así que podemos quedarnos cualquiera de estos dos modelos, y funcionarán de forma similar.

(g) Utiliza los modelos para realizar la predicción en los siguientes 3 meses, representa gráficamente las predicciones, sus intervalos de confianza, y compara con los datos reales que has reservado como conjunto de test. Comenta los resultados.

Previamente hemos guardado un conjunto de test con los últimos tres meses, vamos a ver cómo se comporta el modelo al predecir:

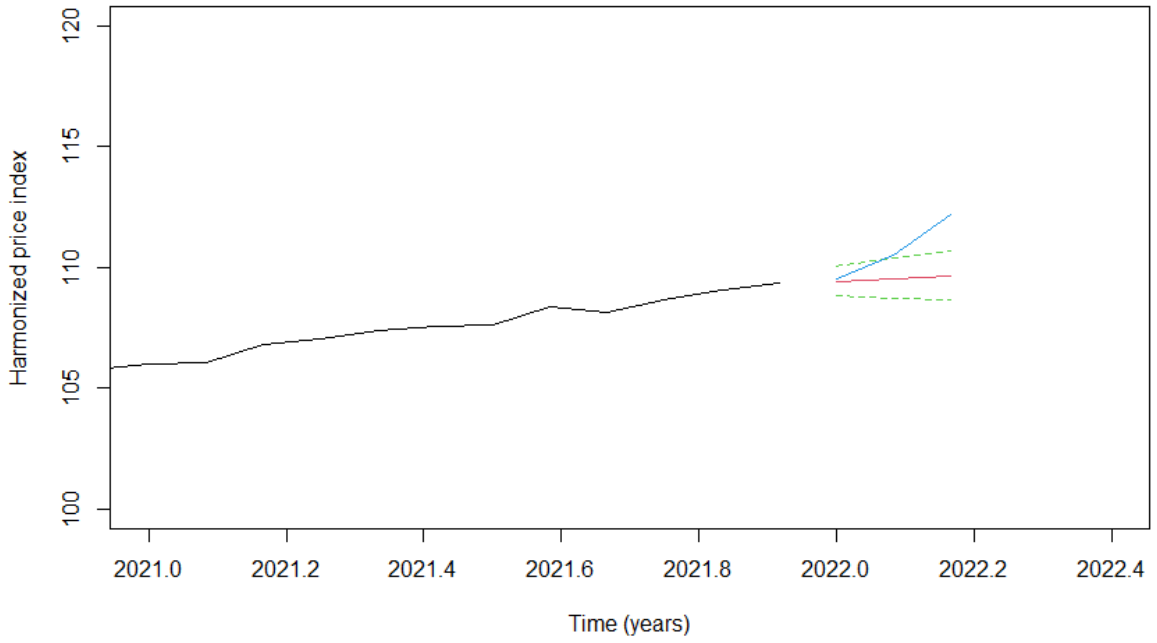
```
arima_2.predict <- predict(arima_2,n.ahead=3)
plot(hicp, xlim=c(2021,2022.4),
       xlab = "Time (years)",
       ylab = "Harmonized price index",
       ylim=c(70,120))
lines(arima_auto.predict$pred,col=2)
lines(arima_auto.predict$pred+1.96*arima_2.predict$se, col=3, lty=2)
lines(arima_2.predict$pred-1.96*arima_2.predict$se, col=3, lty=2)
lines(hicp_test,col=4)
```

Hemos acercado la gráfica para poder ver la predicción, ya que tenemos muchos datos en el conjunto de entrenamiento pero solo 3 en el de test. Obtenemos la siguiente gráfica



Comprobamos, como habíamos previsto, que los datos reales, los del conjunto de test, rápidamente quedan fuera del intervalo. Esto, además de por lo comentado anteriormente, podría ser por el bajo número de datos en el conjunto de test, que hace que el intervalo de confianza sea muy pequeño. Si se cogen unos meses más, aunque la predicción no es tampoco del todo buena (aunque tampoco mala), se queda dentro del intervalo de confianza.

Esto probablemente quiere decir que no existe estacionalidad en los datos, aunque habíamos visto claras tendencias al principio. Vamos a probar a quitar esta parte del modelo, y nos quedamos un ARIMA (2,1,1).

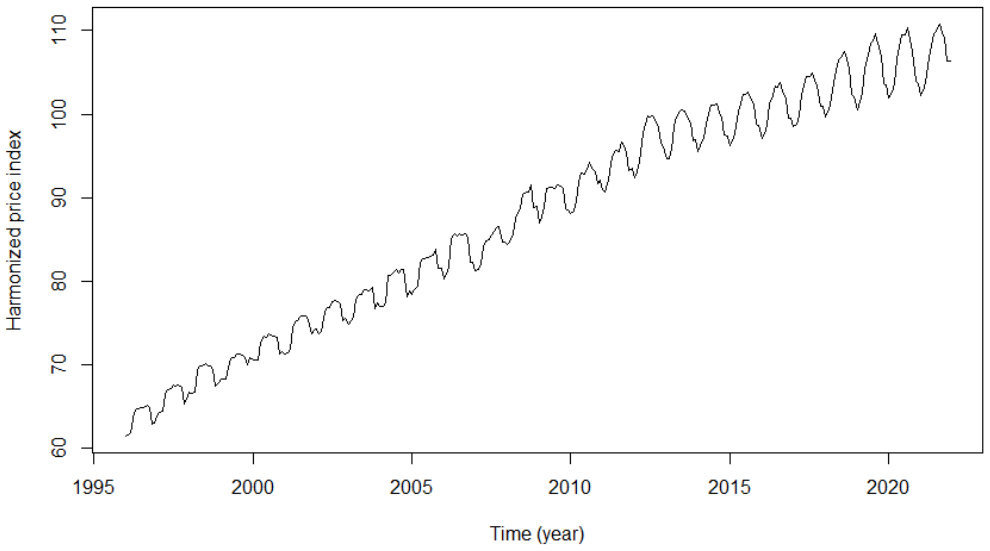


Vemos que realmente la predicción es muy similar, pero varía el intervalo de confianza.

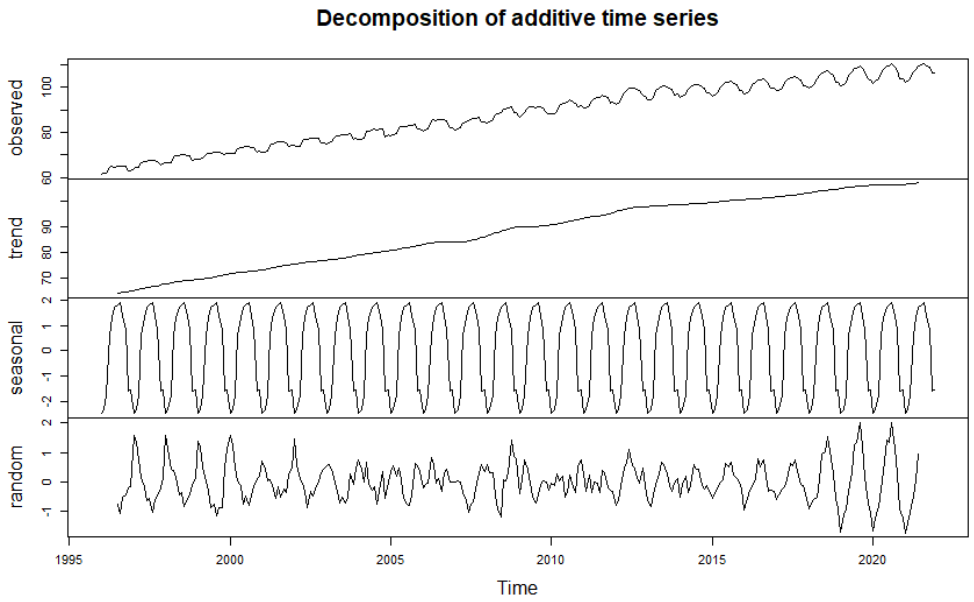
Malta

Vamos a realizar los mismos pasos, pero ahora estudiando la serie temporal relativa a Malta. Primero vamos a pintar los datos para hacernos una idea de como se ve esta serie temporal:

```
df_mt = df[df$geo == "MT",]
df_mt[1] = NULL

df_mt_train = df_mt[1:312,]
df_mt_test = df_mt[313:315,]


# Create a time series object from 2001 to 2021
hicp = ts(as.numeric(as.character(df_mt_train[,2])),start=1996, frequency = 12)
# Test set
hicp_test = ts(as.numeric(as.character(df_mt_test[,2])),start=1996+312/12,
frequency = 12)

#examine the time series
plot(hicp, xlab = "Time (year)", ylab = "Harmonized price index")
```



Podemos ver una estacionalidad mucho más clara que en el caso de Francia, pareciera que cada año el índice armonizado de precios tiene una clara bajada en los meses de invierno, aproximadamente desde octubre a marzo, dependiendo del año. Podría ser porque es un lugar más turístico, y en los meses en los que hay más gente, suben más los precios. Para asegurarnos, vamos a ver la descomposición:

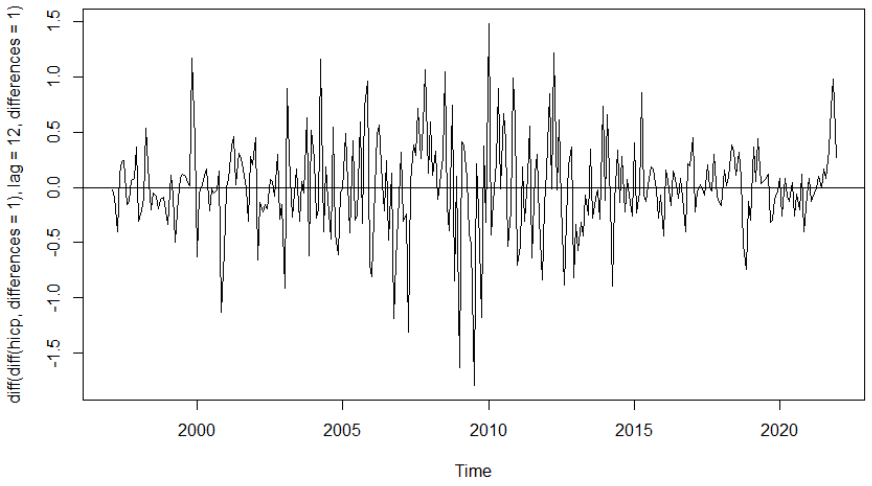


De nuevo se confirma esta tendencia, vemos que el rango del ruido y de la parte estacionaria de la serie son muy parecidos.

Vamos ahora a hallar los valores óptimos de d y D. Cabría esperar que consiguiéramos valores mejores añadiendo un valor en D, y no dejándola a 0, viendo la estacionalidad anterior.

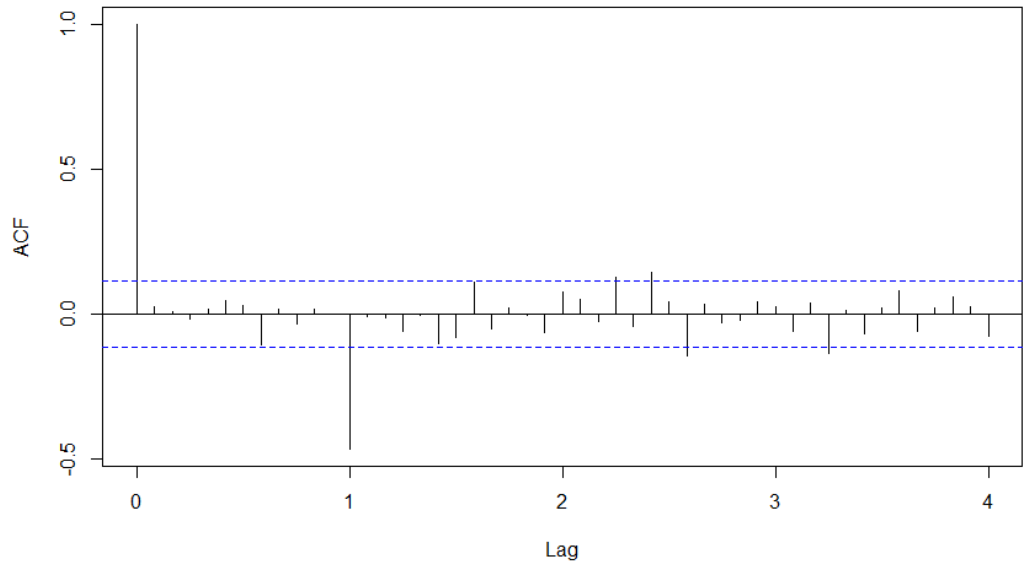
```
# Get optimal d values
plot(diff(hicp, differences=1))
abline(a=0, b=0)
plot(diff(hicp, differences=2))
abline(a=0, b=0)
plot(diff(diff(hicp, differences=1), lag=12, differences=1))
abline(a=0, b=0)

var(diff(hicp)) # 1.41
var(diff(hicp, differences=2)) # 1.93
var(diff(diff(hicp, differences=1), lag=12, differences=1)) # 0.19
```

En el código se ven comentados los valores obtenidos, y la gráfica de la serie en diferencias tomando d=1 y D=1, con s=12:

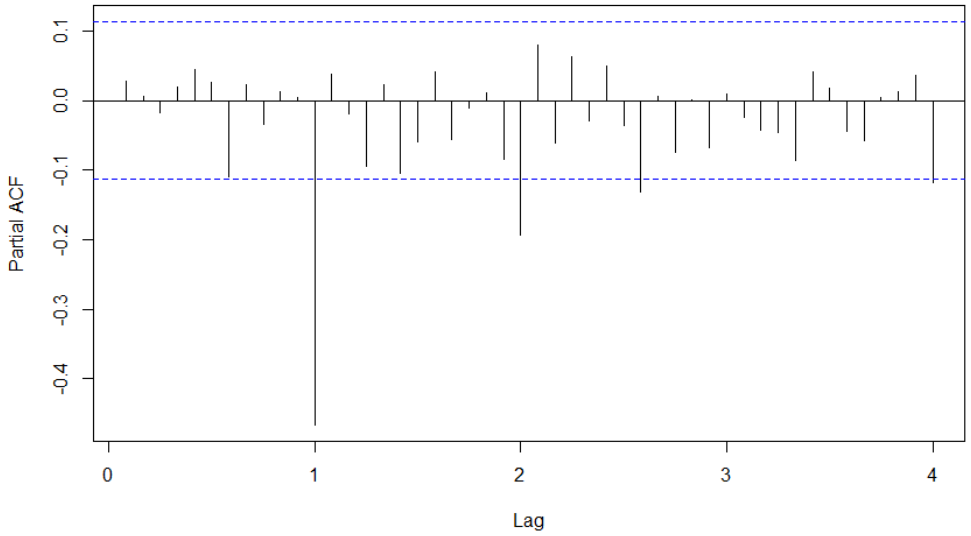


A continuación vamos a estudiar las gráficas de autocorrelación y autocorrelación parcial:

```
# Examine ACF and PACF of differenced series
acf(diff(diff(hicp,differences=1),lag=12,differences=1), lag.max=48, main="")
pacf(diff(diff(hicp,differences=1),lag=12,differences=1),  lag.max=48, main="")
```



Vemos un pico bastante notable en el 1, por lo que añadiremos un MA de orden 1.



La gráfica PACF muestra valores muy fuera del intervalo de confianza hasta el lag 2, por lo que añadiremos P=2, un modelo AR de orden 2.

Entrenamos entonces nuestro primer modelo ARIMA (0,1,0)x(2,1,1):

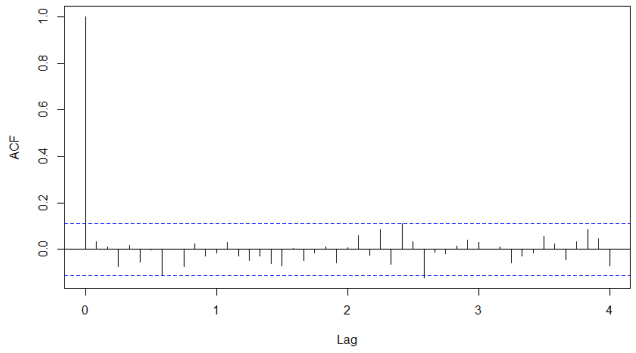
```
# fit a (0,1,0)x(2,1,1)
arima_1 <- arima (hicp,order=c(0,1,0), seasonal =
list(order=c(2,1,1),period=12))
```

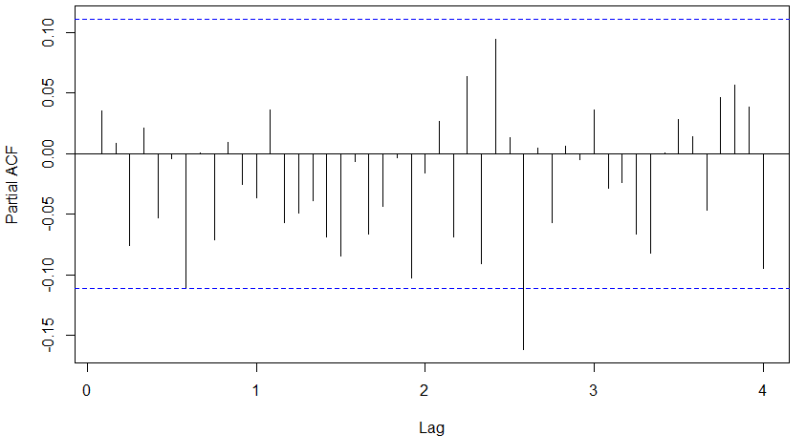
Obtenemos un AIC de 290.1:

```
AIC(arima_1,k = log(length(hicp)))
```

Examinamos el ruido:

```
# examine ACF and PACF of the residuals
acf(arima_1$residuals, lag.max=48, main="")
pacf(arima_1$residuals, lag.max=48, main="")
```



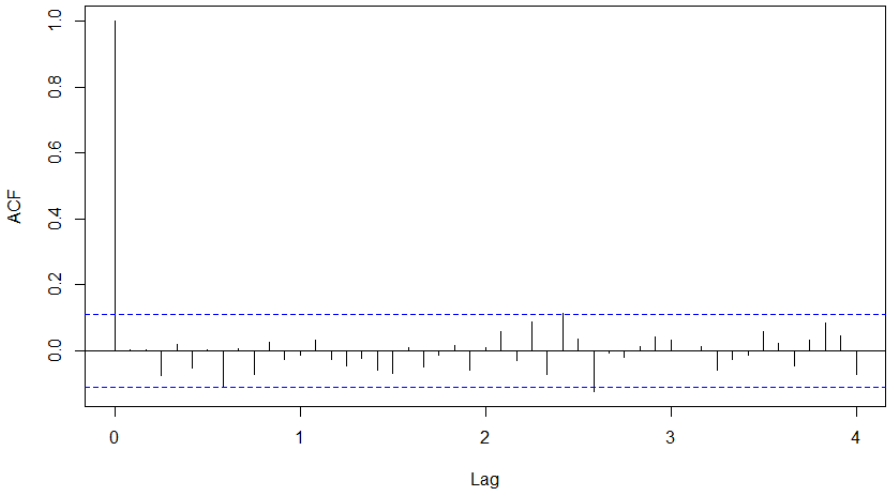


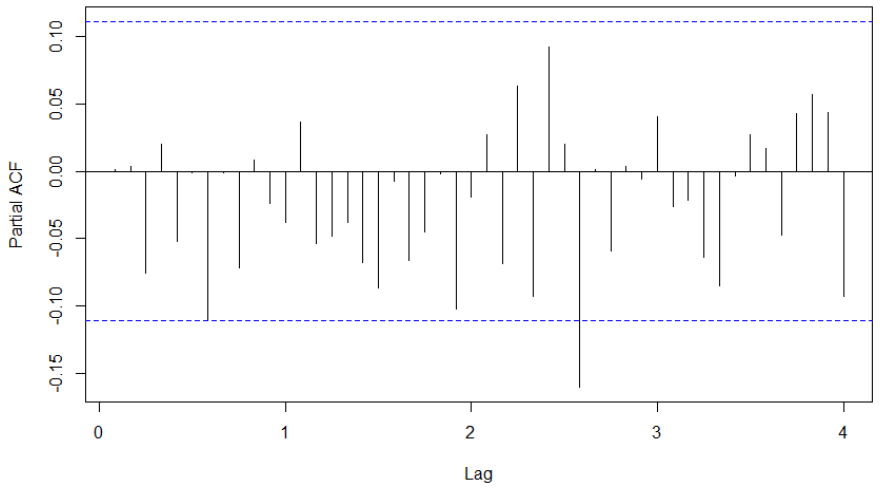
Vemos que en la gráfica ACF todos los valores están dentro del intervalo de confianza, salvo excepciones que están al límite. En la gráfica del PACF, sin embargo, podemos ver mejor ese mal punto en el lag 2, por lo que distribuiremos este coeficiente en la parte no estacionaria, entrenando un modelo ARIMA

```
arima_2 <- arima (hicp,order=c(2,1,0), seasonal =
list(order=c(2,1,1),period=12))
```

Obtenemos un AIC de 278.767, algo mejor que el primer modelo, así que utilizaremos este. Vamos a ver las gráficas de autocorrelación del ruido:

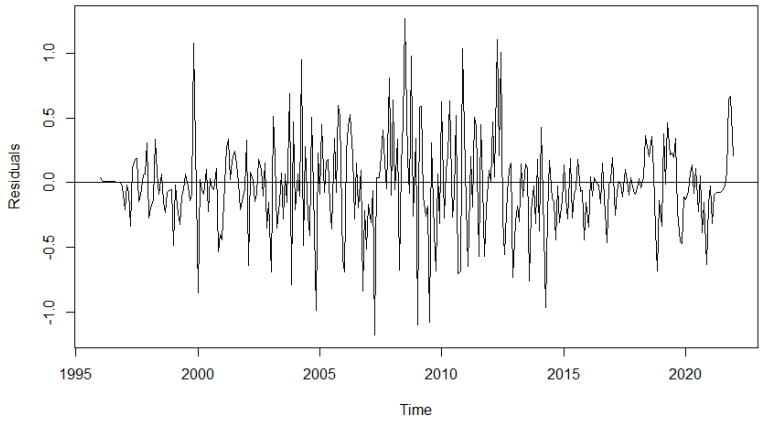
```
# examine ACF and PACF of the residuals
acf(arima_2$residuals, lag.max=48, main="")
pacf(arima_2$residuals, lag.max=48, main="")
```

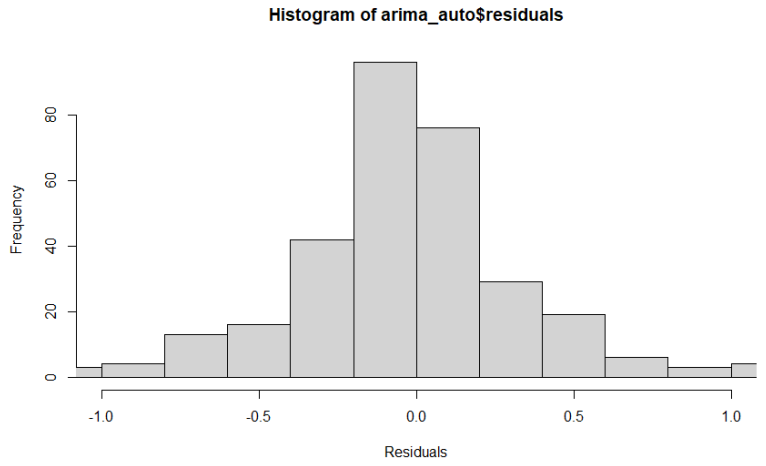




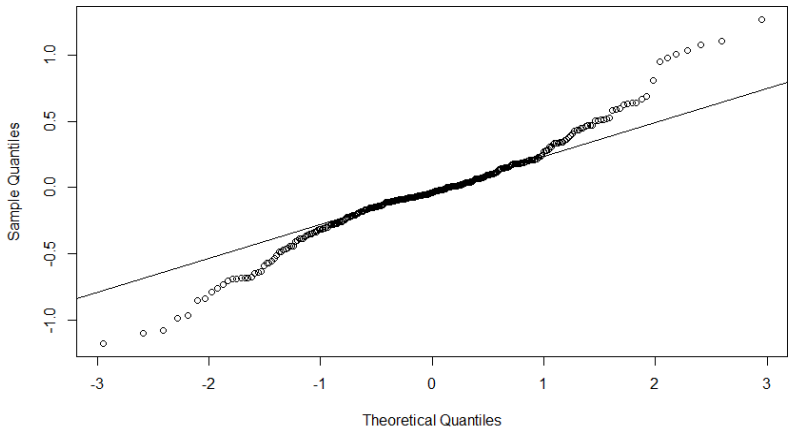
Sacamos gráficas muy similares al primer modelo, pero como mejora algo el AIC será el que vamos a elegir. Vamos a ver también si hay normalidad en los datos o no, lo que determinará si podemos hacer unas predicciones correctas:

```
# Normality and Constant Variance
plot(arima_2$residuals, ylab = "Residuals")
abline(a=0, b=0)
hist(arima_2$residuals, xlab="Residuals", xlim=c(-1,1))
qqnorm(arima_2$residuals, main="")
qqline(arima_2$residuals)
```





El histograma no da una información definitiva, podemos ver que tiene una estructura general de distribución normal, pero los extremos tienen una morfología demasiado curva. Para asegurarnos, vamos a pintar la gráfica cuartil-cuartil:



De nuevo tenemos una gráfica cuartil-cuartil que muestra una curvatura muy notable y distinta a la recta que esperamos, por lo que podemos decir con bastante seguridad que la distribución de los residuos se aleja de una normal. Esto es indicio de que los datos originales no se corresponden con una serie temporal estacionaria, aunque al principio podríamos pensar que sí, y que las predicciones podrían no ser del todo buenas.

Comparando con auto arima, un modelo muy similar:

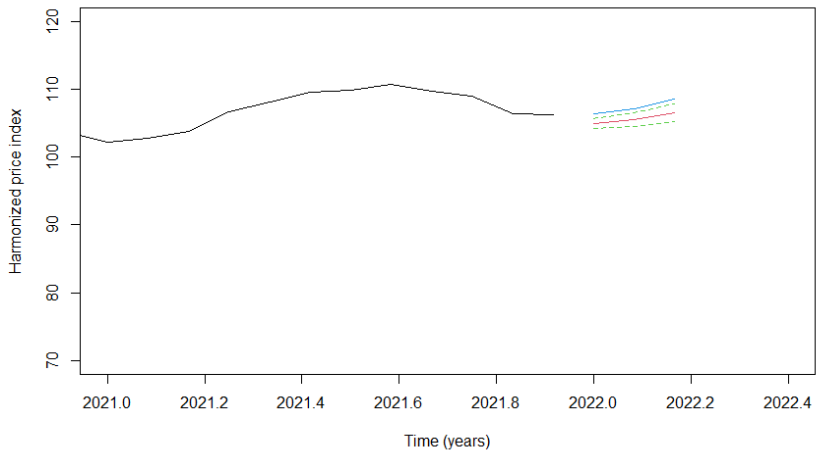
```
arima_auto =
auto.arima(hicp,d=1,D=1,max.order=8,trace=TRUE,approx=FALSE,allowdrift=FALSE,
stepwise=FALSE)
```

Un ARIMA(2,1,1)x(2,1,0). Los resultados son prácticamente idénticos, no justifica añadir un coeficiente para ninguna mejora.

Vamos a pintar las predicciones:

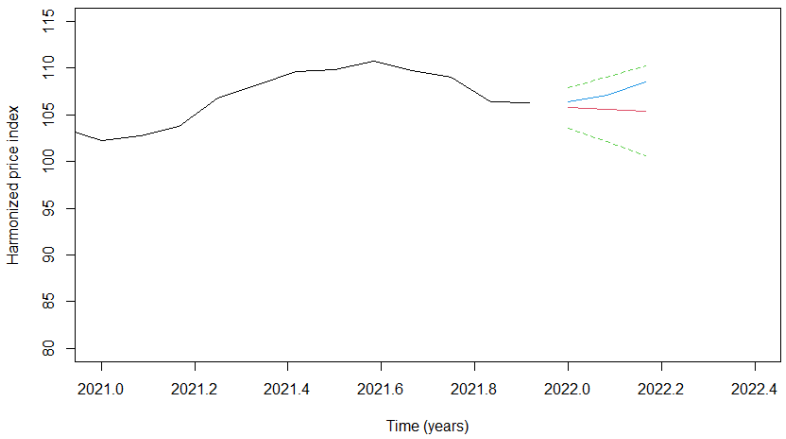
```
model = arima_2

model.predict <- predict(model,n.ahead=3)
plot(hicp, xlim=c(2021,2022.4),
       xlab = "Time (years)",
       ylab = "Harmonized price index",
       ylim=c(70,120))
lines(model.predict$pred,col=2)
lines(model.predict$pred+1.96*model.predict$se, col=3, lty=2)
lines(model.predict$pred-1.96*model.predict$se, col=3, lty=2)
lines(hicp_test,col=4)
```

Obtenemos la siguiente gráfica:



De igual forma que con Francia, parecía que estábamos ante una serie temporal estacionaria, pero ha resultado no ser así. Las predicciones, como se esperaba, son malas y se salen del intervalo de confianza, aunque este es muy pequeño dados los pocos datos

de test. Probamos a quitar la parte estacionaria, quedándonos con un ARIMA (2,1,0):



De igual forma que antes, vemos que realmente la predicción es muy similar, pero varía el intervalo de confianza.

Conclusiones

Las series temporales proporcionan datos muy útiles que permiten estudiar e incluso predecir comportamientos. Aunque la fuerza bruta no es un mal método, para conseguir buenos modelos y sacar conclusiones oportunas es necesario entender y analizar las gráficas para hallar buenos hiper parámetros. Aun así, datos como la inflación son prácticamente imposibles de predecir a día de hoy, ya que hay muchos factores que en la actualidad no pueden tenerse en cuenta para predecir con precisión.
Este es un claro ejemplo, ya que obtenemos modelos relativamente buenos, pero las predicciones después no lo son. También cabe mencionar la aparente estacionalidad de las series, que al final, tras un buen análisis, hemos visto que es inexistente.