Práctica 1

Manejo de Series Temporales con R

1 La clase `ts`

El paquete base tiene muchas funciones para analizar series temporales. Los objetos a los que se aplican tienen que ser de clase `ts`, que sólo maneja datos **equiespaciados**: uno dato por día, 1dato cada cuatro meses, etc... Si los datos no son equiespaciados, hay que usar objetos de clase `zoo`, que se crean con la librería del mismo nombre. Los objetos de clase `zoo` también pueden ser series de datos equiespaciados, de modo que un `zoo` es más general que un `ts`.

Comenzaremos explorando un conjunto de datos que ya es de clase `ts` y crearemos el gráfico que está en las transparencias del curso:

```
data("AirPassengers")
ts.plot(AirPassengers, col="darkblue",lwd=2)
class(AirPassengers)
AirPassengers
```

Veamos cómo convertir un vector en un objeto de clase `ts`:

En `https://homepage.univie.ac.at/robert.kunst/WINE.TSM` hay un archivo con las ventas mensuales (en litros) de vino tinto Australiano[1] en USA entre enero 1980 y octubre 1991. Leeremos el fichero y definiremos la serie, indicando la fecha de comienzo y la fecha final.

```
wine=read.table("https://homepage.univie.ac.at/robert.kunst/WINE.TSM")
head(wine)
wine=wine$V1
```

Definimos la fecha de comienzo (`R` tiene muchos formatos para fechas, buscarlas en el `help`)

```
start=as.Date("1980-01-01")

install.packages("zoo") # tiene una instruccion para usar solo meses
require(zoo)
start=as.yearmon(start)
start
end = as.yearmon(as.Date("1991-10-01"));end
winets=ts(wine,start=start,end=end,frequency=12)
winets  # ver que aspecto tiene el objeto
ts.plot(winets)
plot(decompose(winets))
```

[1] Ejemplo 1.1.1 del libro de Brockwell y Davis *Introduction to Time Series and Forecasting*

Este último gráfico corresponde a la descomposición de la serie en suma de una tendencia, una parte espacional y una de ruido puro. Veremos pronto en qué consiste ésto y cómo se hace sin recurrir a la función `decompose`.

La librería `ggplot2` tiene gráficos muy flexibles, por ejemplo, se puede partir la serie anterior en series anuales:

```
require(ggplot2);require(forecast)
ggseasonplot(winets) # grafico anual
ggseasonplot(window(winets,start=1990)) # elije sólo algunos
```

Y para series financieras, la librería `quantmod` resulta muy cómoda (porque, además, tiene funciones que bajan datos desde páginas como Yahoo!Finance, la Reserva Feredal de los Estados Unidos, y otras fuentes de datos financieros estándar.

```
getSymbols("AAPL") #  Apple stock  Yahoo Finance
dim(AAPL)
head(AAPL)
tail(AAPL)
chartSeries(AAPL)  #   Apple daily closing prices y  trading volume
addBBands(n=20,sd=2) #Bollinger bands (media móvil de tamaño n +/- 2 desviaciones
chartSeries(AAPL[,6]) # Columna 6 of the object "AAPL" de R.
chartSeries(AAPL[,6],theme="white")
```

2 La clase `zoo`

En el fichero `http://www.stat.colostate.edu/~estep/assets/uspopulation.txt` están los datos del ejemplo 1.1.5 del Brockwell y Davis, de la población en USA en intervalos de 10 años, entre 1790 y 1990.

Aunque los datos son equiespaciados (cada 10 años) construiremos un objeto de la clase `zoo` para ver cómo se hace.

```
usa=read.table("http://www.stat.colostate.edu/~estep/assets/uspopulation.txt")
usazoo=zoo(usa,order.by=seq(1790,1990,10))
class(usazoo)
plot(seq(1790,1990,10),usazoo,type="b", main="US Population", xlab="year", ylab="
#clase ts
usats=ts(usa$V1,start=1790,deltat=10)
ts.plot(usats)

# observar que ocurre si hacemos
usats[1:5] # perdemos la fecha
```

```
# para mantenerla, hay que usar la instrucci?n window
window(usats,1790,end=1830)
usats
```

3 Otras funciones para manipular series

Los objetos de clase `ts` y de clase `zoo` se pueden combinar de varias maneras.

Por ejemplo, `ts.union` junta las dos series, rellenando con `NA` los períodos en que una serie sea más larga que la otra; `intersect.ts` junta ambas series, pero sólo con los datos correspondientes al periodo común a ambas.

```
serie1=ts(1:20, freq=12, start=c(1981,3));serie1
serie2=ts(1:15, freq=12, start=c(1980,9));serie2
ts.union(serie1,serie2)
ts.intersect(serie1,serie2)
```

La función `lag.plot` representa la serie observada frente a una versión suya desfasada un número `lags` de unidades de tiempo, por lo que permite visualizar la "autodependencia" de la serie. Concretamente, `lag.plot(x,lags=k)` es equivalente a `plot(lag(x,k),x)`.

```
lag.plot(winets)
plot(winets,lag(winets),xlab="winets(t)",ylab="winets(t+1)")
lag.plot(winets,lags=4,layout=c(2,2))
```

Otra funcion útil para manipular series es `diff`. Pronto veremos qué es y para qué se usa.

4 Ajuste de la tendencia con `lm`

Para ajustar una función de la forma

$$m_t = \beta_0 + \beta_1^t + \beta_2 t^2$$

a los datos de población de USA, indexamos el tiempo para que $t = 1$ corresponda a 1790 y $t = 21$ corresponda a 1990, y hacemos la estimación de mínimos cuadrados, y obtenemos los coeficientes del ajuste.

```
usa=usa$V1
t=1:length(usa) ### el tiempo indexado
usa.lm=lm(usa~t+I(t^2))
summary(usa.lm)
beta=usa.lm$coefficients
plot(t,usa);lines(t,beta[1]+beta[2]*t+beta[3]*t^2,col="orchid2",lwd=2)
plot(t,usa.lm$residuals,main="Noise process (residuals)");abline(h=0)
```

Ejercicio: El conjunto de datos `LakeHuron` del paquete base de `R` corresopnde al ejemplo 1.3.5 del libro de Brockwell y Davis. Se trata de los nivels (en pies) del Lago Hurón (Canadá) entre los años 1875 y 19t2.

1. Importar los datos
2. Usar la instrucción `time()` para obtener el vector de tiempos de la serie.
3. Hacer un gráfico de la serie temporal.
4. Como el nivel del lago parece decrecer linealmente en el tiempo, ajustar un modelo de la forma $X_t = \beta_0 + \beta_1 t$, $t = 1, \ldots, 98$.
5. Agregar el modelo ajustado al gráfico anterior.
6. Hacer un gráfico de los residuos. Observar que en este gráfico no hay tendencia, pero se observa dependencia (¿en qué se nota?).

Finalmente, en `https://www.statmethods.net/advstats/timeseries.html` y `http://r-statistics.co/Time-Series-Analysis-With-R.html` hay tutoriales que pueden resultar útiles.