Anexo II

Código de la estructuración de los datos temporales

```
# Leemos los datos
sales <- read.csv("./monthly-car-sales-in-quebec-1960.csv")
names(sales)[2] <- "Car.Sales"

##############################################################################################

# Stats Package
library(stats) # Suele venir cargado por defecto

# Formato ts sin especificaciones temporales
ts.sales.1 <- ts(data = sales$Car.Sales)

# Formato ts con fecha de inicio manual y frecuencia
ts.sales.2 <- ts(data = sales$Car.Sales, start = c(1960,1), frequency = 12)

# Conversión de carácteres a fecha
as.Date("1960-01-01")
as.Date("1960/01/01")

# Plotting ts.sales y ts.sales.2 (diferencias en el eje x debido a la estructuración)
plot.ts(ts.sales.1, col = "blue")
plot.ts(ts.sales.2, col = "red")

# Seleccionando las observaciones correspondientes a 1963
window(ts.sales.2, start = c(1963,1), end = c(1963,12))

# Plotting del año 1963 con window
plot.ts(window(ts.sales.2, start = c(1963,1), end = c(1963,12), frequency = 12),
        col = "green")

# Uniendo dos series en un mismo objeto
ts.sales.3 <- ts.sales.2 + 10000
mult.sales <- cbind(ts.sales.2, ts.sales.3)
class(mult.sales) # mts
plot(mult.sales, plot.type = "single", col = c("blue", "red"))

##############################################################################################

# ZOO PACKAGE
install.packages("zoo") # instalar
library(zoo)

# Formato zoo con frecuencia mensual
zoo.sales.1 <- zoo(x = sales$Car.Sales, frequency = 12)

# Formato zoo con sequencia de fechas
dt <- seq.Date(from = as.Date("1960-01-01"), to = as.Date("1968-12-01"),
               by = "month")
zoo.sales.2 <- zoo(x = sales$Car.Sales, order.by = dt)

# Conversión de ts a zoo
ts.zoo.sales <- as.zoo(ts.sales.2)

# Generando fechas mensuales y cuatrimestrales
as.yearmon("1960-01-01")
as.yearqtr("2017-2")
```

```
# Seleccionando el índice y los datos de nuestro objeto zoo
index(zoo.sales.2)
coredata(zoo.sales.2)

# Filtrando por fechas
zoo.sales.2[seq.Date(from = as.Date("1963-01-01 "), to = as.Date("1963-12-01 "),
                     by = "month ")]

# Creamos una serie con NAs y la unimos a zoo.sales.2
zoo.nas <- zoo(x = c(rep(mean(coredata(zoo.sales.2)), 108)), order.by =
                 seq.Date(from = as.Date("1960-01-01 "), to = as.Date("1969-10-01 "),
                          by = "month "))
merge.zoo(zoo.sales.2, zoo.nas)

# Tratamiento de NAs
zoo.sales.nas <- zoo(x = c(rep(NA, 6), sales$Car.Sales, rep(NA, 6)), order.by =
                       seq.Date(from = as.Date("1959-06-01"),
                                to = as.Date("1969-05-01"), by = "month"))
zoo.sales.nas
na.aggregate(object = zoo.sales.nas, FUN = mean)
na.approx(object = zoo.sales.nas)
na.spline(object = zoo.sales.nas)
na.fill(object = zoo.sales.nas, fill = c(1,5,12))
na.locf(object = zoo.sales.nas)

# Introduciendo retardos en la serie
lag(x = zoo.sales.2, k = 12, na.pad = TRUE)
lag(x = zoo.sales.2, k = -12, na.pad = TRUE)

# Plotting del objeto zoo
plot(zoo.sales.2)
plot.zoo(zoo.sales.2)

# Plotting con ggplot2
install.packages("ggplot2")
library(ggplot2)

plot <- ggplot() +
  geom_line(aes(x = index(zoo.sales.2), y = coredata(zoo.sales.2)), color = "red") +
  ggtitle("Número mensual de ventas de coches en Quebec") + xlab("Tiempo") +
  ylab("Número de ventas")
plot

# Leyendo con read.csv.zoo
readzoo.sales <- read.csv.zoo(file = "./monthly-car-sales-in-quebec-1960.csv",
                              FUN = as.yearmon)
class(readzoo.sales)
readzoo.sales

# Aplicando funciones a nuestra serie
rollapply(data = readzoo.sales, width = 5, FUN = mean, align = "right")
rollapply(data = readzoo.sales, width = 10, FUN = sd, align = "right")
rollapply(data = readzoo.sales, width = 12, FUN = mean, align = "right", by = 12)

# Aplicando rollmean
rollmean(x = readzoo.sales, k = 12, align = "right")
rollapply(data = readzoo.sales, width = 12, FUN = mean, align = "right")

# Comparando tiempos entre rollmean y rollapply
```

```
data <- zoo(rnorm(100000, 200, 20))
rollmean(x = data, k = 12, align = "right") ==
  rollapply(data = data, width = 12, FUN = mean, align = "right") # TRUE
system.time(rollmean(x = data, k = 12, align = "right")) # Puede variar
system.time(rollapply(data = data, width = 12, FUN = mean, align = "right")) # Puede variar

# Generalizando tiempos de ejecución
obs <- c(100, 500, 1000, 5000, 10000, 20000, 30000, 50000, 70000, 100000,
         300000, 400000, 500000, 600000, 800000, 900000, 1000000)
df.apply <- data.frame(del = rep(1, length(obs)))
df.mean <- data.frame(del = rep(1, length(obs)))

for (t in 1:100) {
  timing_mean <- c()
  timing_apply <- c()

  for (i in obs) {
    data.sample <- zoo(rnorm(i, 200, 20))
    timing_mean <- c(timing_mean,
                     as.numeric(system.time(rollmean(x = data.sample, k = 12,
                                                     align = "right"))[3]))
    timing_apply <- c(timing_apply,
                      as.numeric(system.time(rollapply(data = data.sample, width = 12,
                                                       FUN = mean, align = "right"))[3]))
  }
  df.apply <- cbind.data.frame(df.apply, timing_apply)
  df.mean <- cbind.data.frame(df.mean, timing_mean)
}

df.mean <- df.mean[, -1]
df.apply <- df.apply[, -1]

timing_apply <- apply(df.apply, 1, mean)
timing_mean <- apply(df.mean, 1, mean)

timing.plot <- ggplot() +
  geom_line(aes(x = obs, y = timing_apply, colour = "timing_apply")) +
  geom_line(aes(x = obs, y = timing_mean, colour = "timing_mean")) +
  xlab("Número de observaciones") + ylab("Tiempo de ejecución medio\nen segundos") +
  scale_color_manual(name = "Método",
                     values = c("timing_apply" = "red", "timing_mean" = "blue")) +
  scale_color_manual(name = "Método",
                     values = c("timing_apply" = "red", "timing_mean" = "blue"),
                     labels = c("rollapply","rollmean"))
timing.plot

##############################################################################################

# XTS Package
install.packages("xts")
library(xts)

# Creando el objeto xts con yearmon
xts.sales <- xts(x = sales$Car.Sales, order.by = yearmon(1960 + seq(0, 107)/12))
plot(xts.sales)

# Conversión a xts desde ts
from.ts <- as.xts(ts(sales$Car.Sales, start = c(1960,1), frequency = 12))
plot(from.ts)
```

```
# Conversion a xts desde zoo
from.zoo <- as.xts(zoo(sales$Car.Sales,
                       order.by = yearmon(1960 + seq(0, 107)/12)))
plot(from.zoo)

# Conversion a xts desde timeSeries
install.packages("timeSeries")
library(timeSeries)
from.timeSeries <- as.xts(
  timeSeries(charvec = seq.Date(from = as.Date("1960-01-01"), to = as.Date("1968-12-01"),
                                by = "month"),
             data = sales$Car.Sales))
plot(from.timeSeries)

# Conversion a xts desde matrix
from.matrix <- as.xts(x = matrix(data = sales$Car.Sales, ncol = 1),
                      order.by = yearmon(1960 + seq(0, 107)/12))
plot(from.matrix)

# Conversion a xts desde data.frame
from.data.frame <- as.xts(x = data.frame(x = sales$Car.Sales),
                          order.by = yearmon(1960 + seq(0, 107)/12))
plot(from.data.frame)

# Subsetting de observaciones
xts.sales["1963-12"] # diciembre de 1963
xts.sales["1963"] # el año completo de 1963
xts.sales["/1963-7"] # todas las observaciones hasta julio de 1963
xts.sales["1963-7/"] # todas las observaciones a partir de julio de 1963
xts.sales["1962-7/1963-7"] # todas las observaciones comprendidas entre julio de 1962 y de 1963

# ejemplo de métodos de zoo que funcionan en xts
coredata(xts.sales)
index(xts.sales)
merge.zoo(xts.sales, xts.sales)
xts.sales.nas <- as.xts(zoo.sales.nas)
na.locf(xts.sales.nas)
rollmean(x = xts.sales, k = 12, align = "right")
lag(xts.sales, k = 12, na.pad = TRUE)

# subserie formada por las medias anuales de xts.sales
subserie.1 <- period.apply(x = xts.sales, INDEX = endpoints(xts.sales, on = "years"),
                           FUN = mean)
subserie.1[1] == mean(xts.sales["1960"]) # TRUE
```