**Práctica para la parte dlms. Curso INE**

**Introducción**

EN ESTA PARTE USAREMOS EL PAQUETE dlm DE R. SI NO LO TIENES INSTALADO ABRE TU R Y DA AL COMANDO

install.packages(“dlm”)

TE SOLICITARA EL REPOSITORIO DEL QUE QUIERES BAJAR EL PAQUETE dlm, P.EJ. Spain(Madrid).

UNA VEZ INSTALADO LO DEBES CARGAR CON

library(dlm)

PARA VER LOS PAQUETES QUE TENEIS CARGADOS

search()

PARA VER LOS PAQUETES QUE TENEIS INSTALADOS

library()

PARA VER LOS CONJUNTOS DE DATOS INSTALADOS

data()

**Ejemplo introductorio**

PARA EL EJEMPLO INTRODUCTORIO NO NECESITAMOS dlm. SIMULAMOS EN R OBSERVACIONES DEL MODELO DE NIVEL LOCAL. VEMOS EL EFECTO DE LAS VARIANZAS V y W.

x0<-100

yy<-matrix(0,1,30)

mu<-matrix(0,1,30)

mu[1]<-x0+rnorm(1,0,1)

yy[1]<-mu[1]+rnorm(1,0,1)

for (i in 2:30)

{mu[i]<-mu[i-1]+rnorm(1,0,1);

yy[i]<-mu[i]+rnorm(1,0,1)}

par(mfrow=c(2,2))

plot(mu[1,])

plot(yy[1,])

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

AHORA AUMENTAMOS LA RELACION V, W

x0<-100

mu[1]<-x0+rnorm(1,0,1)

yy[1]<-mu[1]+rnorm(1,0,2)

for (i in 2:30)

{mu[i]<-mu[i-1]+rnorm(1,0,1);

yy[i]<-mu[i]+rnorm(1,0,2)}

plot(mu[1,])

plot(yy[1,])

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

REALIZAMOS EL MISMO EJERCICIO CON UN MODELO DE CRECIMIENTO LINEAL

x0<-100

yy<-matrix(0,1,30)

mu<-matrix(0,1,30)

beta<-matrix(0,1,30)

mu[1]<-x0+rnorm(1,0,1)

yy[1]<-mu[1]+rnorm(1,0,1)

beta[1]<-1

for (i in 2:30)

{beta[i]<-beta[i-1]+rnorm(1,0,1);

mu[i]<-mu[i-1]+beta[i-1]+rnorm(1,0,1);

yy[i]<-mu[i]+rnorm(1,0,1)}

plot(yy[1,])

DISEÑA AHORA UN DLM CON CRECIMIENTO CUADRATICO y SIMULA DEL MISMO

**Uso de dlm**

PASAMOS YA A EMPLEAR dlm. RECUERDA QUE DEBES CARGARLO

library(dlm)

search()

AQUÍ DESCRIBIMOS LA SINTAXIS BASICA DE dlm. DEFINIMOS UN PRIMER DLM Y ESTUDIAMOS SUS CARACTERISTICAS

rw <- dlm(m0 = 0, C0 = 10, FF = 1, V = 1.4, GG = 1, W = 0.2)

unlist(rw)

ALGUNOS MODELOS HABITUALES SE DEFINEN CON FUNCIONES ESPECIALES.

rw <- dlmModPoly(order=1, C0 = 10, dV = 1.4, dW = 0.2)

rw

DEFINIMOS UN SEGUNDO DLM Y DESPLEGAMOS SUS CARACTERISTICAS

lg <- dlm(FF = matrix(c(1, 0), nr = 1),

V = 1.4,

GG = matrix(c(1, 0, 1, 1), nr = 2),

W = diag(c(0, 0.2)),

m0 = rep(0, 2),

C0 = 10 \* diag(2))

lg

is.dlm(lg)

CAMBIAMOS AHORA LA VARIANZA DE OBSERVACION A 0.8 Y LA DE SISTEMA A 0.5

V(lg) <- 0.8

W(lg)[2, 2] <- 0.5

V(lg)

W(lg)

lg

UN MODELO CON F’s, G’s CAMBIANTES (DE REGRESION DINAMICA)

help(dlm)

x <- rnorm(100) # covariates

dlr <- dlm(FF = matrix(c(1, 0), nr = 1),

V = 1.3,

GG = diag(2),

W = diag(c(0.4, 0.2)),

m0 = rep(0, 2), C0 = 10 \* diag(2),

JFF = matrix(c(0, 1), nr = 1),

X = x)

dlr

HACEMOS FILTRADO CON LOS DATOS DEL NILO. PRIMERO COMPRUEBA QUE LOS TIENES

data()

PARA QUE LOS VEAS

Nile

LOS EXPLORAMOS PRIMERO

plot(Nile)

acf(Nile)

CONSTRUIMOS, AJUSTAMOS Y PRESENTAMOS EL MODELO

NilePoly <- dlmModPoly(order = 1, dV = 15100, dW = 1468)

unlist(NilePoly)

FILTRAMOS CON LA FUNCION dlmFilter

NileFilt <- dlmFilter(Nile, NilePoly)

str(NileFilt, 1)

NileFilt

NileFilt$a

NileFilt$m

NileFilt$f

n <- length(Nile)

attach(NileFilt)

search()

dlmSvd2var(U.C[[n + 1]], D.C[n + 1, ])

AHORA USAMOS DOS MODELOS PARA VER EL IMPACTO DE LA RATIO W/V

plot(Nile)

plot(Nile, type='o', col = c("darkgrey"),

xlab = "", ylab = "Level")

PRIMER MODELO

mod1 <- dlmModPoly(order = 1, dV = 15100, dW = 755)

NileFilt1 <- dlmFilter(Nile, mod1)

lines(dropFirst(NileFilt1$m), lty = "longdash")

SEGUNDO MODELO

mod2 <- dlmModPoly(order = 1, dV = 15100, dW = 7550)

NileFilt2 <- dlmFilter(Nile, mod2)

lines(dropFirst(NileFilt2$m), lty = "dotdash")

AÑADIMOS LEYENDAS AL GRAFICO

leg <- c("data", paste("filtered, W/V =",

format(c(W(mod1) / V(mod1),

W(mod2) / V(mod2)))))

legend("bottomright", legend = leg,

col=c("darkgrey", "black", "black"),

lty = c("solid", "longdash", "dotdash"),

pch = c(1, NA, NA), bty = "n")

MOSTRAMOS PREDICCIONES CON LOS DATOS DEL NILO, CON LOS DOS MODELOS

QUE CONSIDERAMOS ANTES

PREPARAMOS LA INFO (DATOS y PREDICCIONES CON AMBOS MODELOS)

a <- window(cbind(Nile, NileFilt1$f, NileFilt2$f),

start = 1880, end = 1920)

a

HACEMOS LAS FIGURAS

plot(a[, 1], type = 'o', col = "darkgrey",

xlab = "", ylab = "Level")

lines(a[, 2], lty = "longdash")

lines(a[, 3], lty = "dotdash")

AÑADIMOS LAS LEYENDAS

leg <- c("data", paste("one-step-ahead forecast, W/V =",

format(c(W(mod1) / V(mod1),

W(mod2) / V(mod2)))))

legend("bottomleft", legend = leg,

col = c("darkgrey", "black", "black"),

lty = c("solid", "longdash", "dotdash"),

pch = c(1, NA, NA), bty = "n")

MOSTRAMOS ALGO MAS DEL IMPACTO DE LA RATIO W/V, QUE NO TIENE POR QUE SER CONSTANTE. P.EJ. EN INTERVENCIONES.

EL MODELO INICIAL

mod0 <- dlmModPoly(order = 1, dV = 15100, dW = 1468)

EN LAS OBSERVACIONES DEL 98 y 99 AUMENTAMOS W PARA UNA MAS RAPIDA ADAPTACION (MULTIPLICAMOS POR 12)

X <- ts(matrix(mod0$W, nc = 1, nr = length(Nile)),

start = start(Nile))

window(X, 1898, 1899) <- 12 \* mod0$W

EL MODELO MODIFICADO

modDam <- mod0

modDam$X <- X

modDam$JW <- matrix(1, 1, 1)

FILTRAMOS CON AMBOS MODELOS

damFilt <- dlmFilter(Nile, modDam)

mod0Filt <- dlmFilter(Nile, mod0)

PREPARAMOS LOS DATOS QUE NECESITAMOS: DATOS, PREDICCIONES CON AMBOS MODELOS

a <- window(cbind(Nile, mod0Filt$f, damFilt$f),

start = 1880, end = 1920)

plot(a[, 1], type = 'o', col = "darkgrey",

xlab = "", ylab = "Level")

lines(a[, 2], lty = "longdash")

lines(a[, 3], lty = "dotdash")

AÑADIMOS LEYENDAS

abline(v = 1898, lty = 2)

leg <- c("data", paste("one-step-ahead forecast -",

c("mod0", "modDam")))

legend("bottomleft", legend = leg,

col = c("darkgrey", "black", "black"),

lty = c("solid", "longdash", "dotdash"),

pch = c(1, NA, NA), bty = "n")

PREDICCIONES A 5 AÑOS CON EL MODELO NILEFILT1

HACEMOS PREDICCIONES, INCLUYENDO 3 TRAYECTORIAS

set.seed(1)

NileFore <- dlmForecast(NileFilt1, nAhead = 5, sampleNew = 3)

NileFore

DIBUJAMOS PRIMERO LA SERIE

plot(window(Nile, start = c(1871, 1)), type = 'o',

xlim = c(1871, 1976), ylim = c(500,1400),

xlab = "", ylab = "Nile flows")

LUEGO LAS PREDICCIONES Y LEYENDAS

names(NileFore)

attach(NileFore)

invisible(lapply(newObs, function(x)

lines(x, col = "darkgrey",

type = 'o', pch = 4)))

lines(f, type = 'o', lwd = 2, pch = 16)

abline(v = mean(c(time(f)[1], time(Nile)[length(Nile)])),

lty = "dashed")

detach()

APLICAMOS SUAVIZADO A LA SERIE DEL NILO

NileSmooth <- dlmSmooth(NileFilt)

str(NileSmooth, 1)

NileSmooth

NileSmooth$s

attach(NileSmooth)

drop(dlmSvd2var(U.S[[n + 1]], D.S[n + 1,]))

VALIDAMOS EL DLM NILEFILT1

mod1 <- dlmModPoly(order = 1, dV = 15100, dW = 755)

NileFilt1 <- dlmFilter(Nile, mod1)

EXTRAEMOS RESIDUOS

Nileresid1<-residuals(NileFilt1,sd=FALSE)

Nileresid1

EVALUAMOS RESIDUOS

qqnorm(Nileresid1)

qqline(Nileresid1)

acf(Nileresid1)

tsdiag(NileFilt1)

MODELO DE CRECIMIENTO LINEAL CON LOS DATOS DE INVERSION DE ESPAÑA 1960-2000. COLOCAIS EL FICHERO invest2.dat EN CIERTO DIRECTORIO

LEEMOS PRIMERO DATOS (OJO AL DIRECTORIO)

invSpain <- ts(read.table("C:/Users/DRIOS/Desktop/cursoine/dlms/invest2.txt", colClasses = "numeric")[,2],start = 1960)

invSpain

plot(invSpain)

acf(invSpain)

DEFINIMOS MODELO , FILTRAMOS Y PREDECIMOS

mod1 <- dlmModPoly(dV = 10, dW = c(102236, 321803))

mod1Filt <- dlmFilter(invSpain, mod1)

fut1 <- dlmForecast(mod1Filt, n=5)

DISTINTAS MEDIDAS DE BONDAD DE AJUSTE

mean(abs(mod1Filt$f - invSpain))

mean((mod1Filt$f - invSpain)^2)

mean(abs(mod1Filt$f - invSpain) / invSpain)

REPRESENTACIONES

par(mar=c(2,3,1,0) + 0.1, cex=0.7)

plot(invSpain, xlim=c(1960, 2005), ylim=c(2500, 22000),

xlab="", ylab="Investments", type='o', col = "darkgrey")

lines(dropFirst(mod1Filt$f), type='o', lty="431313", pch=3)

lines(fut1$f, lty="dashed")

legend("topleft", bty = "n",

legend=c("observed", "one-step-ahead forecast - model 1",

"5 years forecast - model 1"),

lty=c("solid", "431313", "dashed"), pch=c(1,3,-1),

col=c("darkgrey", rep("black", 4)), inset = 0.05)

DEFINICION GENERICA DE MODELOS EN dlm



EJEMPLO

mod1 <- dlmModTrig(s = 12, dV = 5.1118, dW = 0) +

dlmModPoly(1, dV = 0, dW = 81307e-3)

EJEMPLO DE ESTIMACION DE PARAMETROS MEDIANTE MLE

buildNile<-function(x) dlmModPoly(1,dV=x[1],dW=x[2])

buildNile

fitNile<-dlmMLE(Nile,parm=rep(100,2),build=buildNile,

lower=rep(1e-8,2),hessian=TRUE)

fitNile

fitNile$par