```
############# PROGRAM 1 ################################
# Author: M. Szyszkowicz
# Program 1: Fits individual C-RF shapes for the lagged concentration.
#########################################################
library(sme) # To have AIC (Akaike's IC)
library(gnm); library(splines); library(quantmod)
options(digits=6); options(na.action="na.exclude")
#Read the data: health and environmental
datSET <- read.table("EdmontonResp.csv",header=TRUE,sep=",")
#########################
# CREATE STRATA YR x MONTH x DOW
datSET$Y <- as.factor(datSET$year)
datSET$M <- as.factor(datSET$month)
datSET$W <- as.factor(datSET$dow)
##### Define hierarchical clusters.
datSET$Cluster <- as.factor(datSET$Y:datSET$M:datSET$W)
attach(datSET); nameF="RESLALL.txt"
#########################################################
# Define the function to minimize AIC
funLL <- function(param){
mu <- param[1]; tau <- param[2]; A <- param[3]
rtau= tau*diff(range(xs,na.rm=TRUE))
# Transformation T(Z)
# Other functions f(z): #XT<-sqrt(xs)/(1+exp((mu-xs)/rtau))
XT <- log(1+xs/A)/(1+ exp((mu-xs)/rtau))
#The used model:
modelG <- gnm(RESP ~ XT  + TNS + HNS,
data=datSET, family=poisson, eliminate=factor(Cluster))
# Retrieve the coefficients: Beta, SE, and p-value.
```

```
B=unname(summary(modelG)$coeff[1,1])
SE= unname(summary(modelG)$coeff[1,2])
P=unname(summary(modelG)$coeff[1,4])
RES=c(B,SE,mu,tau,extractAIC(modelG)[2]);
# The results sent to the file: collect all iterations
sink(file=nameF); print(RES); sink()
return(extractAIC(modelG)[2])
}##### Function description
##### Define Lag=M; air pollutant (xs)
##### Represent Temperature and relative humidity as ns (*,df=3)
M=1; xs = Lag(O3,M)
T=Lag(TEMP,M);  H=Lag(RHUM,M)
TNS=ns(T,df=3);  HNS=ns(H,df=3)
# Search for the best fit = find the optimal parameters mu, tau, and A
nlminb(c(100,0.1,150.0), funLL); #initial values, function funLL – returns AIC
#### Retrieve the last recorded line of the result (RES)
datX=read.table(nameF);  resT=(tail(datX,1))
B=(resT[1,2]) ; SE=resT[1,3]; mu=resT[1,4]; tau=resT[1,5]
########################################
### Calculate RR and 95%CI
Low=B-1.96*SE; Upp=B+1.96*SE; xp <- sort(O3)
rtau= tau*diff(range(xp,na.rm=TRUE))
# Another T(z): gg = sqrt(xp)/(1+exp((mu-xp)/rtau))
gg = log(1+xp/A)/(1+exp((mu-xp)/rtau))
yp=B*gg;  RR=exp(yp); Lyp= Low*gg;  Uyp= Upp*gg
LP = exp(Lyp);  UP= exp(Uyp)
##### Plot the C-RF for a specific lag
plot(xp,RR, ylim=c(min(RR,LP),max(RR,UP)),col="white")
lines(xp,RR, lwd=5,col="red"); abline (h=1)
```

```
lines(xp,LP, lwd=4,lty=3, col="blue"); lines(xp,UP, lwd=4,lty=3,col="blue")
############ The END of Program 1#################################
####&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&####
##**********************************************##
################## PROGRAM 2 ##########################
# Author: M. Szyszkowicz
# Program 2: Fits the global C-RF shape.
#########################################################
library(MASS);  library(minpack.lm)
#Read the data (Table 1)
dataIN <- read.table("ResLLogzA.csv",header=TRUE,sep=","); attach(datIN)
# Creat the X-axis; ozone from 0 to 200, Z=O3
xp=seq(0, 200, by=0.1);  x=xp; xa <- xp
N=7; N1=N+1; MIN=2; MAX=-2
#Prepare the data frame by lags 0-7
for (k in 1:N) {
if (k==1){
X <- xa;  TRR=as.vector(X)
TLR=TRR; TUR=TRR}  # for k=1
# The names in dataIN should be as used (TAU, etc.)
rtau= TAU[k]*diff(range(X,na.rm=TRUE))
logit=1/(1+exp((MU[k]-X)/rtau))
XT=log(X/AP[k]+1)*logit; y=BET[k]*XT
yl=(BET[k]-1.96*SEB[k])*XT; yu=(BET[k]+1.96*SEB[k])*XT
rr=exp(y);  rl=exp(yl);  ru=exp(yu)
TRR=cbind(TRR,rr);  TLR=cbind(TLR,rl);  TUR=cbind(TUR,ru)
# Find minimum and maximum values for plot
mi=min(rl)
if( mi<MIN) {MIN=mi}
mx=max(ru)
```

```
if (mx>MAX) {MAX=mx}
if (k==1)
{xa=x;ya=rr;yL=rl;yU=ru}
else{
xa=c(xa,x); ya=c(ya,rr); yL=c(yL,rl); yU=c(yU,ru) }
}# # Define frame with X, RR, RRlower, and RRupper (for each lag)
dfram=data.frame(xa,ya); dframe=dfram[order(xa),]
dframL=data.frame(xa,yL); dframLe=dframL[order(xa),]
dframU=data.frame(xa,yU); dframUe=dframU[order(xa),]
#################Define plot area (white) #############
plot(X,rr, ylim=c(MIN,MAX), col="white"); abline(h=1)
################# Fit a common C-RF for RRs ######################
#Start values:
r= diff(range(xp,na.rm=TRUE))
A=3.0; T=0.3; M=1; P= 0.1
fitM = nlsLM(dframe$ya~  ((dframe$xa+A)/A)^(T/(1+exp(-(dframe$xa-M)/(r*P)))),
start = list(A=A, T=T, M=M,P=P), data=dframe)
lines(dframe$xa,fitted(fitM), lwd=6,col="red")
# summary(fitM)
#########Fit a common C-RF for lower RRs ##########################
A=3.0; T=0.3; M=1; P= 0.1
fitL = nlsLM(dframLe$yL~  ((dframLe$xa+A)/A)^(T/(1+exp(-(dframLe$xa-M)/(r*P)))),
start = list( A=A,T=T, M=M,P=P), data=dframLe  )
lines(dframe$xa,fitted(fitL),  lwd=6,lty=3,col="blue" )
# summary(fitL)
####################Fit a common C-RF for upper RRs ###################
A=3.0;  T=0.3;  M=1;  P= 0.1
fitU = nlsLM(dframUe$yU~  ((dframUe$xa+A)/A)^(T/(1+exp(-(dframUe$xa-M)/(r*P)))),
start = list(A=A,T=T, M=M,P=P), data=dframUe  )
lines(dframe$xa,fitted(fitU),  lwd=6,lty=3,col="blue" )
```

```
# summary(fitU)
##### To see individual curves on a common plot
for (k in 1:N+1){
lines(X, TRR[,k], lwd=4, col="red") }
abline(h=1)
#########################################################################
for (k in 1:N+1){
lines(X, TLR[,k], lwd=3,lty=3,col="black") }
#########################################################################
for (k in 1:N+1){
lines(X, TUR[,k],lwd=3,lty=3, col="blue") }
```

################## **The END of Program 2** ########################