Supplementary Materials: Non–parametric generalized additive models as a tool for evaluating policy interventions.

R codes. Simulation analysis

```
# Libraries
library(prais)
library(nlme)
library(mgcv)

# Common parameters in simulations
n=100
simulations=500
percentage_int=0.1
n_firstint=(1-percentage_int)*n+1
n_int=n*percentage_int

# Variables
T=0:(n-1)
t_n=0:(n_int-1)
zero_preintervention=rep(0, n*(1-percentage_int))
one_intervention=rep(1,n*percentage_int)
I=c(zero_preintervention,one_intervention)
TI=c(zero_preintervention,t_n)

# Simulation models (choice one option)
# 1.- Linear model
Level_shift=5
model=30-0.08*T+Level_shift*I+0.02*TI
model_noint=30-0.08*T
Cumulative_effect=sum(model-model_noint)

# 2.- Quadratic model
Level_shift=-5
model=30-0.25*T+0.002*T^2+Level_shift*I+0.1*TI+0.005*TI^2
model_noint=30-0.25*T+0.002*T^2
Cumulative_effect=sum(model-model_noint)

# 3.- Polynomial model
Level_shift=-5
model=30+0.2235*T-0.008*T^2+0.00006*T^3+Level_shift*I+0.1*TI-0.05*TI^2+0.0001*TI^3
model_noint=30+0.2235*T-0.008*T^2+0.00006*T^3
Cumulative_effect=sum(model-model_noint)

# Plot int
plot(model, type="l", lwd="1.5", col="black", ylim=c(0,40), ylab="", xlab="")
lines(model_noint, col="black", lty=3)
```

```
abline(v=90, lwd=1, lty=2)

# Simulations with an AR(1) error
Y=model+arima.sim(list(order = c(1,0,0), ar = 0.3), n = n, sd = 0.5)
for(i in 1:(simulations-1)){
Y1=model+arima.sim(list(order = c(1,0,0), ar = 0.3), n = n, sd = 0.5)
Y=c(Y,Y1)}
Data=matrix(Y,ncol=simulations)

# Segmented linear regression model
Linear_Levelshift=0
Trend_coefficient=0
linear_CAeffect=0

for(i in 1:simulations){
depvar<-c(Data[,i])
df<-as.data.frame(cbind(depvar,T,I,TI))
linear_regression <- prais_winsten(depvar ~ T+I+TI, data = df)
Linear_Levelshift[i]=linear_regression$coefficients[3]
Trend_coefficient[i]=linear_regression$coefficients[4]
linear_CAeffect[i]=sum(Linear_Levelshift[i]+Trend_coefficient[i]*0:(n_int-1))}

# Level shift
mean(Linear_Levelshift)
sd(Linear_Levelshift)
MSE.Linear_Levelshift <-mean((Level_shift-Linear_Levelshift)^2)
MPE.Linear_Levelshift <- mean(abs((Level_shift-Linear_Levelshift)/Level_shift))
MSE.Linear_Levelshift
MPE.Linear_Levelshift

# Cumulative effect
mean(linear_CAeffect)
sd(linear_CAeffect)
MSE.linear_CAeffect <-mean((Cumulative_effect-linear_CAeffect)^2)
MPE.linear_CAeffect <- mean(abs((Cumulative_effect-linear_CAeffect)/Cumulative_effect))
MSE.linear_CAeffect
MPE.linear_CAeffect

# Plot for the last simulation
plot(depvar,type="l", col="blue", ylim=c(0,40))
lines(linear_regression$coefficients[1]+linear_regression$coefficients[2]*T,
 col="black", lty=3)
lines(linear_regression$fitted.values, col="black")
abline(v=90, lwd=1, lty=2)

# GAM
gam_Levelshift=0
gam_CAeffect=0

for(i in 1:simulations){
```

```
depvar<-c(Data[,i])
df<-as.data.frame(cbind(depvar,T, I, TI))
gam <- gamm(depvar ∼ s(T) + I + s(TI),
 data = df, correlation=corARMA(form = ∼ T, p=1), family = gaussian, method =
"REML")
 gam_Levelshift[i]=gam$gam$coefficients[2]
df.pre <-df
df.pre$I <-rep(0,n)
df.pre$TI <-rep(0,n)
gam_noint <- predict(gam$gam, newdata = df.pre, type="response", se=TRUE)
gam_CAeffect[i]=sum(gam$gam$fitted.values[91:100]-gam_noint$fit[91:100])}

# Level shift
mean(gam_Levelshift)
sd(gam_Levelshift)
MSE.gam_Levelshift <-mean((Level_shift-gam_Levelshift)^2)
MPE.gam_Levelshift <- mean(abs((Level_shift-gam_Levelshift)/Level_shift))
MSE.gam_Levelshift
MPE.gam_Levelshift

# Cumulative effect
mean(gam_CAeffect)
sd(gam_CAeffect)
MSE.gam_CAeffect <-mean((Cumulative_effect-gam_CAeffect)^2)
MPE.gam_CAeffect <- mean(abs((Cumulative_effect-gam_CAeffect)/Cumulative_effect))
MSE.gam_CAeffect
MPE.gam_CAeffect

# Plot for the last simulation
plot(depvar,type="l", col="blue", ylim=c(0,40))
lines(gam_noint$fit, col="black", lty=3)
lines(gam$gam$fitted.values, col="black")
abline(v=90, lwd=1, lty=2)
```

R codes. Illustration with real data

```
# Libraries
library(readxl)
library(see)
library(ggplot2)
library(gridExtra)
library(pastecs)
library(prais)
library(nlme)
library(mgcv)

# Data
Data <- read_excel("Realdata.xlsx",
 col_types = c("numeric",
 "numeric", "numeric", "numeric", "numeric",
 "numeric", "numeric"))
pre.intervention <- subset(Data, I==0)
post.intervention <- subset(Data, I==1)

# Descriptive statistics
stat.desc(pre.intervention$recipes)
stat.desc(post.intervention$recipes)

# Histogram
p1 <- ggplot(pre.intervention, aes(x = recipes)) + labs(x = "", y= "") +
 geom_histogram(binwidth = 0.07) +
 scale_x_continuous(breaks=seq(1.2, 2, 0.2), limits=c(1.2, 2)) +
 scale_y_continuous(breaks=seq(0, 20, 5), limits=c(0, 20)) +
 theme_modern()

p2 <- ggplot(post.intervention, aes(x = recipes)) + labs(x = "", y= "") +
 geom_histogram(binwidth = 0.07) +
 scale_x_continuous(breaks=seq(1.2, 2, 0.2), limits=c(1.2, 2)) +
 scale_y_continuous(breaks=seq(0, 20, 5), limits=c(0, 20)) +
 theme_modern()

plots(p1, p2, n_columns = 2)

# Shapiro–Wilk normality test
shapiro.test(pre.intervention$recipes)
shapiro.test(post.intervention$recipes)

## Segmented linear regression model
prais <- prais_winsten(recipes ~ T + I + TI + as.factor(month) + stockpiling, data
= Data)
summary(prais)
prais_escalon=prais$coefficients[3]
prais_trend=prais$coefficients[4]
prais_CAEffect=sum(prais$coefficients[3]+prais$coefficients[4]*0:41)
```

```
prais_CAEffect

# Segmented linear regression model plot

plot(Data$recipes,type="l",ylim=c(0,2),ylab= "",xlab= "Year",
 bty="l",xaxt="n", col="blue")
abline(v=103, lwd=1, lty=2)
axis(1,at=0:13*12,labels=F)
axis(1,at=0:12*12+6,tick=F,labels=2004:2016)
lines((1:144),prais$fitted.values,col="black")
lines((1:144),prais$fitted.values-prais$coefficient[3]*Data$I-prais$coefficient[4]*Data$TI,col="blac
 lty=3)

## GAM

dfgam_noint <-Data
dfgam_noint$I <-rep(0,144)
dfgam_noint$TI <-rep(0,144)

gam <- gamm(recipes ~ s(T) + I + s(TI)+s(month) + stockpiling, data=Data,
 correlation=corARMA(form = ~ T, p=1), family = gaussian, method = "REML")

summary(gam$gam)
xp.noint <- predict(gam$gam, newdata = dfgam_noint, type="response", se=TRUE)
xp.int <- predict(gam$gam, newdata = Data, type="response", se=TRUE)
gam_CAeffect=sum(xp.int$fit[103:144]-xp.noint$fit[103:144])
gam_CAeffect

# GAM check
par(mfrow=c(2,2))
gam.check(gam$gam)

# GAM plot
par(mfrow=c(1,1))
plot(Data$recipes,type="l",ylim=c(0,2),ylab="", xlab= "Year",
 bty="l",xaxt="n", col="blue")
abline(v=103, lwd=1, lty=2)
axis(1,at=0:13*12,labels=F)
axis(1,at=0:12*12+6,tick=F,labels=2004:2016)
lines((1:144),xp.int$fit,col="black")
lines((1:144),xp.noint$fit,col="black", lty=3)
```