The informational content of geographical indications

Jean-Sauveur AY[*]
INRA UMR CESAER

Replication Material Version 1.1 : July 30, 2019

Abstract

This file contents the Replication Material (RM) associated to the AAWE Working Paper No XXX entitled *The informational content of geographical indications*. Data, code and prediction materials are under the copyright license GNU GPL V3, which means that license notices must be preserved. Raw data are available from the INRA dataverse server https://data.inra.fr. Some R functions are reported in the Appendix to preserve the readability of codes in the main text. The most recent version of this document and a Shiny application about the econometric classification of vineyards in the *Côte d'Or* (Burgundy, France) are available from the remote repository https://github.com/jsay/geoInd.

Contents

[*] jsay@inra.fr, UMR CESAER, AgroSup, INRA, Univ. Bourgogne Franche-Comté, 26 bd Dr Petitjean, 21000 Dijon (FR).

1 Descriptive Statistics

1.1 Data shaping

The full detail of data construction is presented in a data paper available at https://github.com/jsay/geoInd/. The data paper also contains the dictionary of the variables used here. The result of these preliminary treatments can be directly downloaded from the INRA dataverse server at https://data.inra.fr.

The following R code allows to load the data once downloaded and located in the `/Inter/` folder at the root of the working directory of the R session. It loads a `SpatialPolygonDataFrame` object from the `sp` package that contains the characteristics of the vineyard plots under consideration (session information used for this article is reported at Section 8). It also reshapes some variables of particular interest:

- It reorders the *commune* levels along the North-South gradient
- It standardizes the variable about solar radiation
- It recodes the variable about exposition in 8 quadrants
- It projects the geographical coordinates inside the WGS84 system
- It selects the parcels with GIs and drop omitted values

```
library(sp) ; load("Inter/GeoRas.Rda")
Geo.Ras$LIBCOM <- factor(Geo.Ras$LIBCOM, levels=
             unique(Geo.Ras$LIBCOM[order(Geo.Ras$YCHF, decreasing= T)]))
Geo.Ras$RAYAT <- as.numeric(scale(Geo.Ras$SOLAR))
Geo.Ras$EXPO <- cut(Geo.Ras$ASPECT, breaks= c(-2, 1: 8* 45))
GR84 <- spTransform(Geo.Ras, CRS("+proj=longlat +ellps=WGS84"))
dd <- coordinates(GR84) ; Geo.Ras$X= dd[, 1] ; Geo.Ras$Y= dd[, 2]
dim(Reg.Ras <- subset(Geo.Ras, !is.na(AOClb) & !is.na(DEM) & !is.na(DESCR)
                      & !is.na(RUD) & !is.na(AOC36lab) & !is.na(REGION)))
```

```
[1] 59113    71
```

The resulting object is a `SpatialPolygonDataFrame` that contains 59 113 observations of vineyard plots with 72 variables without omitted values.

1.2 Geology and pedology

Another pre-regression treatment is the transformation of the geological and pedological variables into dummy variables in order to control sub-soil and soil characteristics of vineyards with fixed effects. A too small number of observation within a given fixed effect can be a problem for the precision and convergence of the estimation, hence we choose to include a fixed effects only for geological and pedological polygons with more than 1 000 vineyard plots. The details and robustness of this arbitrary choice are presented in the data paper mentioned above.

```r
Reg.Ras$NOTATION <- factor(Reg.Ras$NOTATION)
tmp <- table(Reg.Ras$NOTATION)< 1000
Reg.Ras$GEOL <- factor(
    ifelse(Reg.Ras$NOTATION %in% names(tmp[ tmp]), "0AREF",
           as.character(Reg.Ras$NOTATION)))
Reg.Ras$NOUC <- factor(Reg.Ras$NOUC)
tmp <- table(Reg.Ras$NOUC)< 1000
Reg.Ras$PEDO <- factor(
    ifelse(Reg.Ras$NOUC %in% names(tmp[tmp]), "0AREF",
           as.character(Reg.Ras$NOUC)))
apply(Reg.Ras@data[, c("GEOL", "PEDO")], 2, table)
```

```
$GEOL

0AREF     C     E    Fu    Fx    Fy    GP    j3   j3a   j3b   j4a
 5208 19014  1997  1060  2142  1460  8372  1288  2570  2539  1531
  j5a   j5b   j6a  p-IV
 3526  3928  3087  1391

$PEDO

0AREF    13    14    26    28    29    30    32    34    35    36
 3310  1553 17475  3718  8687  6241  4563  1802  1700  5255  1116
    5    69     8
 1051  1484  1158
```

The characteristics of sub-soils and soils are modeled with respectively 14 and 13 fixed effects. In each case, the reference modality coded `0AREF` is equal to 1 for all vineyards plots inside geological and pedological polygons without sufficient observations. Robustness checks have been made with other threshold values than 1 000 without this arbitrary choice changes the results.

1.3 Crossing GIs dimensions

The data are now ready for the econometric analysis. The GIs on the area of interest contains both an horizontal (*commune*) and a vertical (*hierarchical level*) dimension as detailed in the Working Paper. The balance of the two distributions can be assessed with the following Figure 3 (p.37) in the Working Paper.

```
library(lattice) ; library(RColorBrewer)
fig.dat <- aggregate(model.matrix(~0+ factor(Reg.Ras$AOC))*
                       Reg.Ras$AREA/ 1000, by= list(Reg.Ras$LIBCOM), sum)
names(fig.dat) <- c("LIBCOM", "BGOR", "BOUR", "VILL", "PCRU", "GCRU")
fig.dat$LIBCOM <- factor(fig.dat$LIBCOM, lev= rev(levels(fig.dat$LIBCOM)))
fig.crd <- t(apply(fig.dat[, -1], 1, function(t) cumsum(t)- t/2))
fig.lab <- round(t(apply(fig.dat[, -1], 1, function(t) t/ sum(t)))* 100)
my.pal  <- brewer.pal(n= 9, name = "BuPu")[ 2: 8]
barchart(LIBCOM~ BGOR+ BOUR+ VILL+ PCRU+ GCRU, xlim= c(-100, 10200),
         xlab="Vineyards delineated as Geographical Indications (hectare)",
         data= fig.dat, horiz= T, stack= T, col= my.pal, border= "black",
         par.settings= list(superpose.polygon= list(col= my.pal)),
         auto.key= list(space= "top", points= F, rectangles= T, columns= 5,
                        text=c("Coteaux b.", "Bourgogne",
                               "Village", "Premier cru", "Grand cru")),
         panel=function(x, y, ...) {
             panel.grid(h= 0, v = -11, col= "grey60")
             panel.barchart(x, y, ...)
             ltext(fig.crd, y, lab= ifelse(fig.lab> 0, fig.lab, ""))})
```

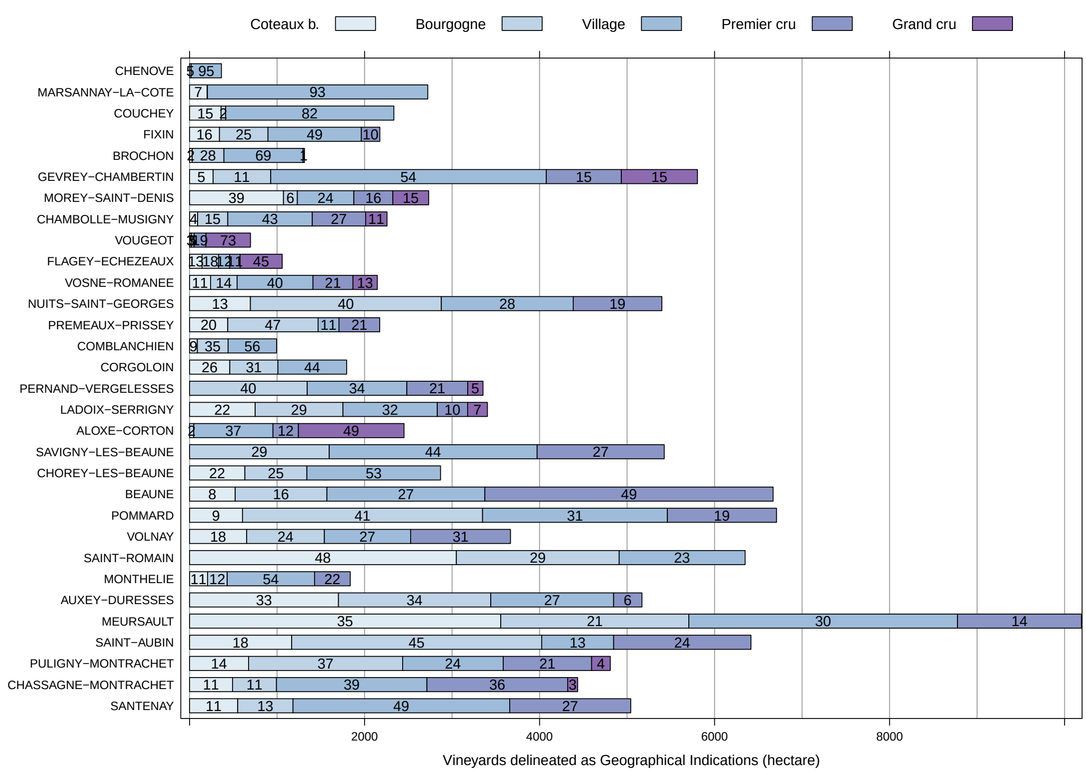


Figure 1: **Cross-distribution of current GI levels among communes**

We also use historical GI designation scheme from 1936, the year of creation of the French national institute in charge of geographical indications (INAO). At his time, the vertical dimension counted only three levels, whereas the horizontal dimension was identical. The balance of the distribution can be assessed by the following Figure.

```
library(lattice) ; library(RColorBrewer)
fig.old <- aggregate(model.matrix(~0+ factor(Reg.Ras$AOC36lvl))*
                     Reg.Ras$AREA/ 1000, by= list(Reg.Ras$LIBCOM), sum)
names(fig.old) <- c("LIBCOM", "BOUR", "VILL", "GCRU")
fig.old$LIBCOM <- factor(fig.old$LIBCOM, lev= rev(levels(fig.old$LIBCOM)))
old.crd <- t(apply(fig.old[, -1], 1, function(t) cumsum(t)- t/2))
old.lab <- round(t(apply(fig.old[, -1], 1, function(t) t/ sum(t)))* 100)
old.pal  <- brewer.pal(n= 9, name = "BuPu")[ c(2, 5, 8)]
barchart(LIBCOM~ BOUR+ VILL+ GCRU, xlim= c(-100, 10200),
         xlab="Vineyards delineated as 1936 GI (hectare)",
         data= fig.old, horiz= T, stack= T, col= old.pal, border= "black",
         par.settings= list(superpose.polygon= list(col= old.pal)),
         auto.key= list(space= "top", points= F, rectangles= T, columns= 3,
                        text=c("Bourgogne", "Village", "Grand cru")),
         panel=function(x, y, ...) {
             panel.grid(h= 0, v = -11, col= "grey60")
             panel.barchart(x, y, ...)
             ltext(old.crd, y, lab= ifelse(old.lab> 0, old.lab, ""))})
```

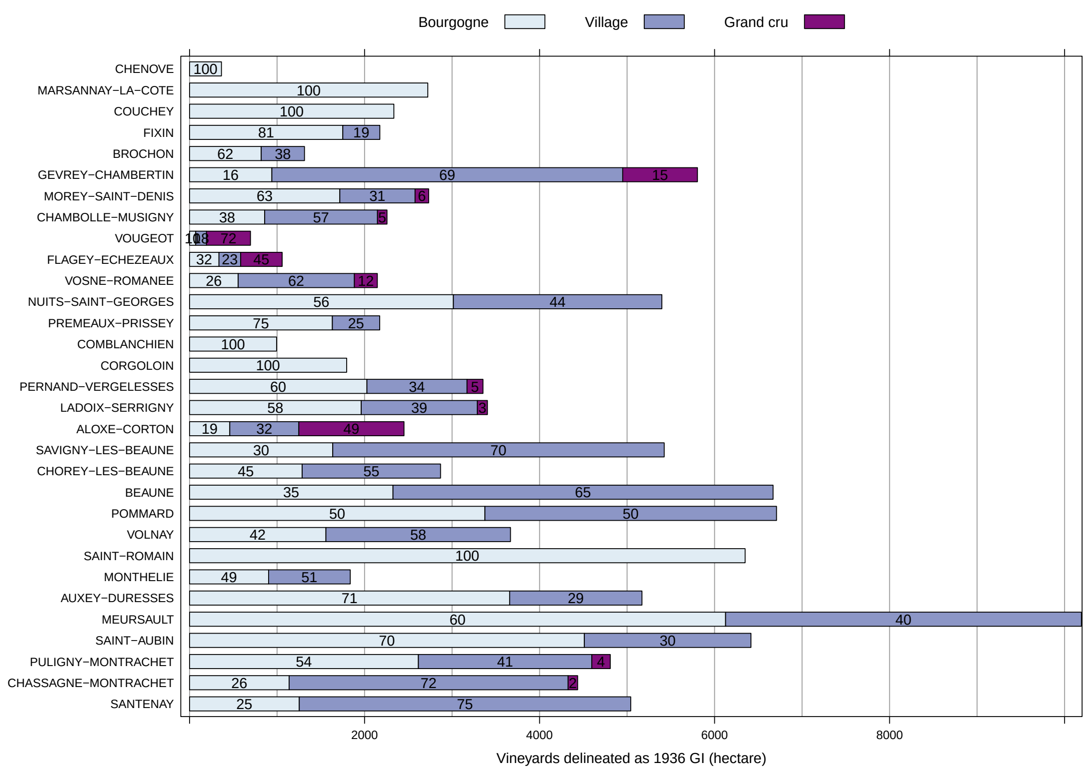


Figure 2: **Cross-distribution of 1936 GI levels among communes**

2 Models of GI designation

2.1 Parametric ordered logit models

We first estimate the benchmark parametric ordered logistic model `polm1` that corresponds to model (0) of Table 1 (p.22) in the Working Paper. Model `polm1a` is the auxiliary regression without *commune* fixed effects used to test the presence of omitted *terroir* effect as detailed in the Working Paper. Model `polm1b` is also an auxiliary regression without smoothing of spatial coordinates to compute the Fisher statistics associated to these terms in Table 1. We use for this the standard `polr` function from `MASS` package.

```
library(MASS)
polm1 <- polr(factor(AOC)~ 0+ LIBCOM+ EXPO+ GEOL+ PEDO
              + poly(DEM, 2)+ poly(SLOPE, 2)+ poly(RAYAT, 2)
              + poly(X, 3)* poly(Y, 3), data= Reg.Ras, Hess= TRUE)
polm1a <- polr(factor(AOC)~ 0+ EXPO+ GEOL+ PEDO
               + poly(DEM, 2)+ poly(SLOPE, 2)+ poly(RAYAT, 2)
               + poly(X, 3)* poly(Y, 3), data= Reg.Ras, Hess= TRUE)
polm1b <- polr(factor(AOC)~ 0+ LIBCOM+ EXPO+ GEOL+ PEDO
               + poly(DEM, 2)+ poly(SLOPE, 2)+ poly(RAYAT, 2)
              , data= Reg.Ras, Hess= TRUE)
```

```
Warning messages:
1: In polr(factor(AOC) ~ 0 + LIBCOM + EXPO + GEOL + PEDO + poly(DEM,  :
  une coordonnée à l'origine est nécessaire et assumée
2: In polr(factor(AOC) ~ 0 + LIBCOM + EXPO + GEOL + PEDO + poly(DEM,  :
  le plan ne semble pas de rang plein, des coefs seront ignorés
```

The warning messages come from the choice to drop the intercept in order to estimate a coefficient for each *commune* from the variable `LIBCOM`. This choice is made to compute more easily the ordinal superiority measures from fixed effects. This does not have any effect on the other estimated coefficients.

2.2 Ordered generalized additive models

We estimate the series of ordered generalized additive models (OGAMs) of GIs designations within a loop. Models (I) to (V) reported in Table 1 (p.22) of the Working Paper are only a subset of all models of the `gamod` object that can be downloaded directly from the INRA server, https://data.inra.fr/geoInd/gamod.Rda. Models with high complexities for the spatial effects (more than 600 edf) are long to run. They require about 8 hours each, with the full loop requires about 2 days to run with Intel Core i7-7820HQ CPU 2.90 GHz x 8 and 64 Go of RAM. We advise the reader to not run the full loop, but instead to select values of `listk` and estimate each model separately.

```
library(mgcv)
listk <- c(50, 100, 200, 300, 400, 500, 600, 700, 800, 900)
gamod <- vector("list", length(listk))
system.time(
for (i in 1: length(listk)){
    gamod[[ i]] <- gam(AOC~ 0+ LIBCOM+ EXPO+ GEOL+ PEDO
                        + s(DEM)+ s(SLOPE)+ s(RAYAT)+ s(X, Y, k= listk[ i])
                      , data= Reg.Ras, family= ocat(R= 5))
})
names(gamod) <- paste0("gam", listk)
save(gamod, file= "Inter/gamod.Rda")
```

```
utilisateur      système      écoulé
     113038          384      109562
```

The second loop below produces the `gammod` object that contains the auxiliary regressions to test the omitted *terroir* effects as presented in the Working Paper, Section 4.2. The reader is not expected to run the loop entirely but pick some value of `k` in `listk` between 0 and 1 000 to estimate each model individually.

```
gammod <- vector("list", length(listk))
system.time(
for (i in 1: length(listk)){
    gammod[[ i]] <- gam(AOC~ 0+ EXPO+ GEOL+ PEDO
                         + s(DEM)+ s(SLOPE)+ s(RAYAT)+ s(X, Y, k= listk[ i])
                       , data= Reg.Ras, family= ocat(R= 5))
})
names(gammod) <- paste0("gam", listk)
save(gammod, file= "Inter/gammod.Rda")
```

```
utilisateur      système      écoulé
     103037          262      102775
```

3 Diagnostics

3.1 Significance

We first reports the Chi-square statistics for the joint significance of the parametric ordered logit model `polm1` that corresponds to model (0) of Table 1 (p.22) in the Working Paper.

```
library(car)
res1a <- anova(polm1, polm1b)
(res1 <- Anova(polm1))
```

```
Le chargement a nécessité le package : carData

Analysis of Deviance Table (Type II tests)

Response: factor(AOC)
                       LR Chisq Df Pr(>Chisq)
LIBCOM                     9768 31     <2e-16 ***
EXPO                        743  7     <2e-16 ***
GEOL                       1716 14     <2e-16 ***
PEDO                       8811 13     <2e-16 ***
poly(DEM, 2)               4030  2     <2e-16 ***
poly(SLOPE, 2)              532  2     <2e-16 ***
poly(RAYAT, 2)             1885  2     <2e-16 ***
poly(X, 3)                 1933  3     <2e-16 ***
poly(Y, 3)                  178  3     <2e-16 ***
poly(X, 3):poly(Y, 3)      5257  9     <2e-16 ***
---
codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
Warning messages:
1: glm.fit: fitted probabilities numerically 0 or 1 occurred
2: glm.fit: fitted probabilities numerically 0 or 1 occurred
3: glm.fit: fitted probabilities numerically 0 or 1 occurred
4: glm.fit: fitted probabilities numerically 0 or 1 occurred
5: glm.fit: fitted probabilities numerically 0 or 1 occurred
```

Then, we compute the same Chi-square statistics for all the OGAMs with the function `resume`. They are also reported in Table 1 (p.22) in the Working Paper. Recall that the estimated models can be downloaded from https://data.inra.fr/geoInd/gamod.Rda.

```
load("Inter/gamod.Rda")
resume <- function(mod){
    tmp <- anova(mod)
    res <- c(as.vector(rbind(tmp$s.table[, 3], tmp$s.table[, 1])),
             as.vector(rbind(tmp$pTerms.tab[, 2], tmp$pTerms.tab[, 1])))
    names(res) <- c(as.vector(rbind(rownames(tmp$s.table), rep("", 4))),
                    as.vector(rbind(rownames(tmp$pTerms.tab), rep("", 2))))
    round(res, 1)
}
sapply(gamod[ 1: 5* 2], resume)
```

```
          gam100  gam300  gam500  gam700  gam900
s(DEM)    4123.2  1793.1  1189.9  1014.1   867.0
             8.9     8.9     8.9     8.8     8.8
s(SLOPE)   922.5   343.6   168.5   155.5   190.1
             8.3     8.2     8.3     8.2     7.7
s(RAYAT)  2091.3   981.6   797.7   646.5   531.0
             8.1     8.1     8.3     8.0     7.3
s(X,Y)   32524.2 59293.9 74154.2 78445.3 86597.1
            98.6   295.0   483.2   666.6   841.4
LIBCOM    3007.9  2295.2  2353.7  1721.6  1363.5
            31.0    31.0    31.0    31.0    31.0
EXPO        61.0    81.3   171.5   159.0   130.5
             7.0     7.0     7.0     7.0     7.0
GEOL       977.4   557.4   500.5   406.4   440.9
            14.0    14.0    14.0    14.0    14.0
PEDO      2447.2   713.1   450.4   408.6   387.9
            13.0    13.0    13.0    13.0    13.0
```

3.2 Goodness of fit

We report below the code used to compute the goodness-of-fit measures for model (0) reported in Table 1 (p.22): Pseudo-R^2, Akaike information criteria (AIC), and percent of good predictions.

```
psR2 <- function(x) 1- (logLik(x)/ logLik(update(x, . ~ + 1)))
round(c(psR2= psR2(polm1), AIC= AIC(polm1)/ 1000,
        Pcgp= sum(diag(table(predict(polm1),
                             Reg.Ras$AOC)))/nrow(Reg.Ras)), 2)
```

```
psR2    AIC   Pcgp
0.37 104.15   0.64
```

And the same goodness of fit measures for OGAMs.

```
library(mgcv)
pcgp <- function(x){
    sum(diag(table(cut(x$line, c(-Inf, x$family$getTheta(TRUE), Inf)),
                   x$model[, 1])))/ nrow(x$model)* 100
}
rbind(psR2= sapply(gamod[ 1: 5* 2], psR2),
      AIC=  sapply(gamod[ 1: 5* 2], AIC)/ 1000,
      Pcgp= sapply(gamod[ 1: 5* 2], pcgp))
```

```
Le chargement a nécessité le package : nlme
This is mgcv 1.8-28. For overview type 'help("mgcv-package")'.

      gam100 gam300 gam500  gam700  gam900
psR2  0.5323  0.631  0.684  0.7248  0.7565
AIC  77.2170 61.397 53.088 46.7579 41.9259
Pcgp 74.8600 80.387 84.376 87.2566 89.4778
```

3.3 Omitted variable bias

As indicated in the Working Paper (Appendix A.1), we evaluate the potential omitted *terroir* variables through the joint significance of *commune* fixed effects on the residuals from auxiliary regressions without such fixed effects. Code below allows to compute the bootstrapped Fisher statistics with 100 replications from parametric ordered logistic model. The absence of correlated effects is strongly rejected. Note that we use the `sure` package to compute the surrogate residuals from this parametric model.

```
library(lmtest) ; library(sandwich) ; library(sure)
wal1 <- rep(NA, times= nsim <- 100)
for (i in 1: nsim){
    tmp <- surrogate(polm1a)- polm1a$lp
    wal1[ i] <- waldtest(lm(tmp~ Reg.Ras$LIBCOM), .~ 1, vcov= vcovHC)$F[ 2]
}
quantile(wal1, c(.05, .5, .95))
```

```
   5%    50%    95%
151.3 155.9 160.6
```

Note that the values obtained are not exactly equal to those reported in the Working Paper because of the bootstrap procedure.

The `sure` package does not allow to compute surrogate residuals for `gam` models from the `mgcv` package. Because this framework is also consistent for OGAMs, we write the function `sureOGAM` presented and tested in Appendix A.3 to adapt the framework. This function is also available in the file of custom function `./myFcts.R` that is sourced in the following code. Hence, we compute the bootstrapped F-statistics for the full set of OGAM belows. The estimation of auxiliary models is presented above, they can be directly downloaded from https://data.inra.fr.

```
load("Inter/gammod.Rda") ; library(ggplot2) ; source("myFcts.R")
omitVar <- function(mod, var, nsim= 100){
    usq <- rep(NA, nsim)
    for(i in 1: nsim) {
        RES <- sureOGAM(mod)
        tmp <- lm(I(RES- mod$linear.pred)~ factor(var))
        usq[ i] <- waldtest(tmp, . ~ 1, vcov= vcovHC)$F[ 2]
    }
    usq
}
wal2 <- sapply(gammod, function(x) omitVar(x, Reg.Ras$LIBCOM, nsim= 100))
apply(wal2[, -1], 2, function(x) quantile(x, c(.05, .5, .95)))
```

```
    gam100 gam200 gam300 gam400 gam500 gam600 gam700 gam800 gam900
5%   15.22  5.724  4.983  4.033  3.522  2.787  2.032  1.699  1.361
50%  16.86  6.504  5.658  4.690  4.056  3.373  2.439  2.132  1.722
95%  18.35  7.429  6.536  5.487  4.916  4.024  3.195  2.827  2.203
```

Again, the values are not exactly the same. Note that the critical value at 0.01% for the F-distribution in this case is 2.3, as can be assessed from `qf(.9999, 31, Inf)`.

The following plot resumes the specification diagnostics and shows the relevance of OGAMs to control for omitted spatial effects. It corresponds to Figure 7 (p.42) in the Working Paper.

```
library(lattice)
pltdat <- stack(data.frame(logit= wal1, wal2))
Fstat <- round(qf(.9999, 31, Inf), 2)
bwplot(log(values)~ ind, data= pltdat, type=c("l","g"), horizontal= FALSE,
       xlab= 'Model of GI designation',
       ylab= 'Bootstraped F-statistics (log scale)',
       par.settings = list(box.rectangle=list(col='black'),
                           plot.symbol  = list(pch='.', cex = 0.1)),
       scales=list(y= list(at= log((1: 15)^2), lab= (1: 15)^2)),
       panel = function(..., box.ratio) {
           panel.grid(h= 0, v = -11)
           panel.abline(h= log((1: 15)^2), col= "grey80")
           panel.violin(..., col = "lightblue",
                        varwidth = FALSE, box.ratio = box.ratio)
           panel.bwplot(..., col='black',
                        cex=0.8, pch='|', fill='gray', box.ratio = .1)
           panel.abline(h= log(Fstat), col= "red", lty= 2, cex= 1.5)
           panel.text(2, log(Fstat)+ .1,
                      paste0("F= ", Fstat, " : critical value at .01%"))})
```

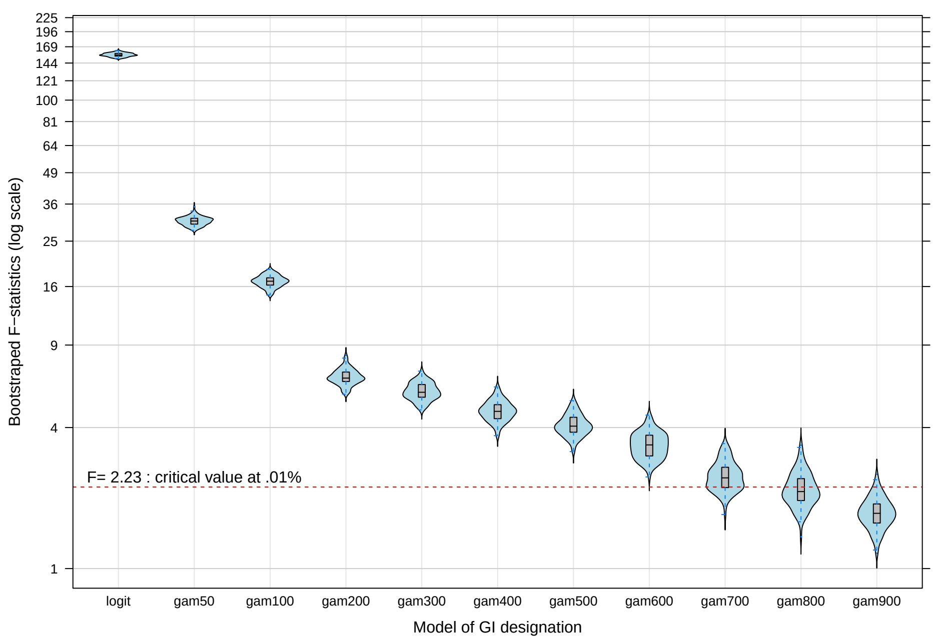


Figure 3: **F-statistics for the diagnostic of correlated residual effects**

3.4 Specification

The estimation of surrogate residuals from the full models can be used to test the specification of the effects of explanatory variables. The Figures from the code below are not reported in this document are they are too detailed.

```
library(sure) ; library(gridExtra)
var <- c("DEM", "SLOPE", "RAYAT", "EXPO", "LIBCOM", "X", "Y")
plots <- lapply(var, function(.x)
    autoplot(polm1, what= "covariate", x= Reg.Ras@data[, .x], xlab= .x))
do.call(grid.arrange, c(list(autoplot(polm1, what= "qq")), plots))
restmp <- sureOGAM(gamod$gam900)- gamod$gam900$line
plot(qlogis(1: nrow(Reg.Ras)/ nrow(Reg.Ras), scale= 1), sort(restmp))
abline(0, 1)
pltSURE <- function(resid, xvar, lab){
    plot(xvar, resid, xlab= lab, main= paste("Surrogate Analysis", lab))
    abline(h= 0, col= "red", lty= 3, lwd= 2)
    lines(smooth.spline(resid ~ xvar), lwd= 3, col= "blue")
}
par(mfrow= c(3, 3)) ; for (i in var) pltSURE(restmp, Reg.Ras@data[, i], i)
```

4 Marginal effects

4.1 Parametric ordered logit

The marginal effects from parametric model polm1 can be directly plotted with the package effect. The following plots corresponds to the dotted lines in Figure 5 (p.40) in the Appendix of the Working Paper.

```
library(effects)
plot(predictorEffects(polm1, ~ DEM+ SLOPE+ RAYAT+ EXPO, latent= TRUE,
                      xlevels=list(DEM= 200: 500,
                                   SLOPE= 0: 400/ 10, RAYAT= -60: 30/ 10)))
```

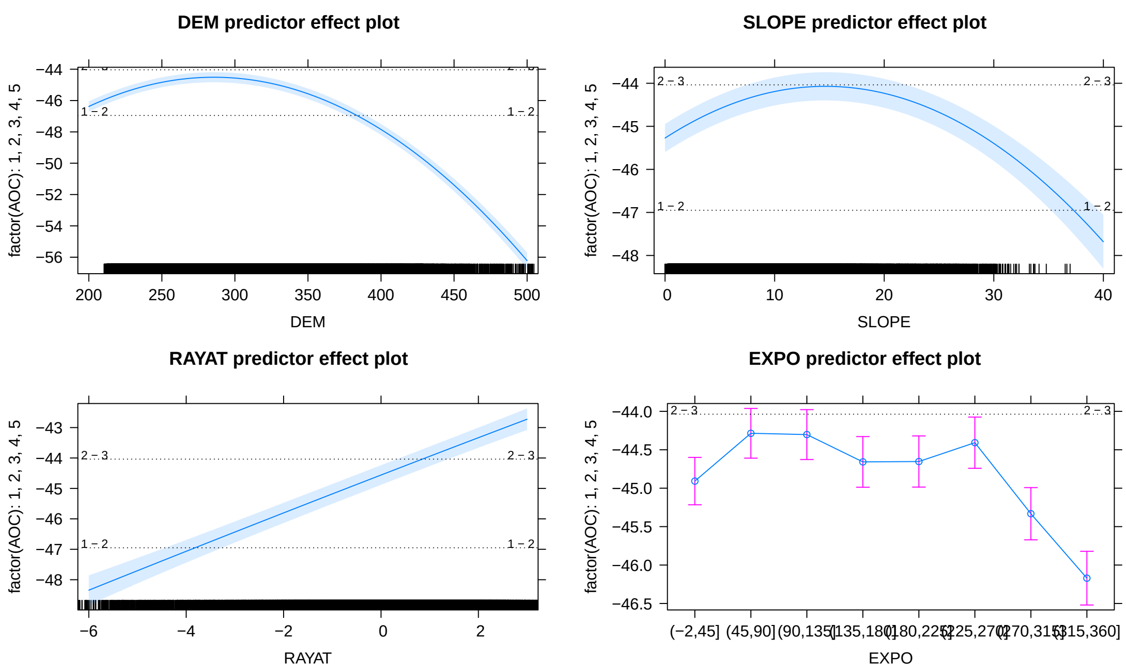


Figure 4: **Marginal effects of topographic variables from ordered logit**

4.2 Ordered generalized additive

The same effect plots can be drawn for the OGAMs models. We report below the effects from the OGAM `gam900` which corresponds to a maximum effective degrees of freedom of 900. For all models of `gamod`, we obtain the gray curves of Figure 5 (p.40) in the Appendix of the Working Paper.

```
plot(gamod[[ 10]], pages= 1, scale= 0)
```

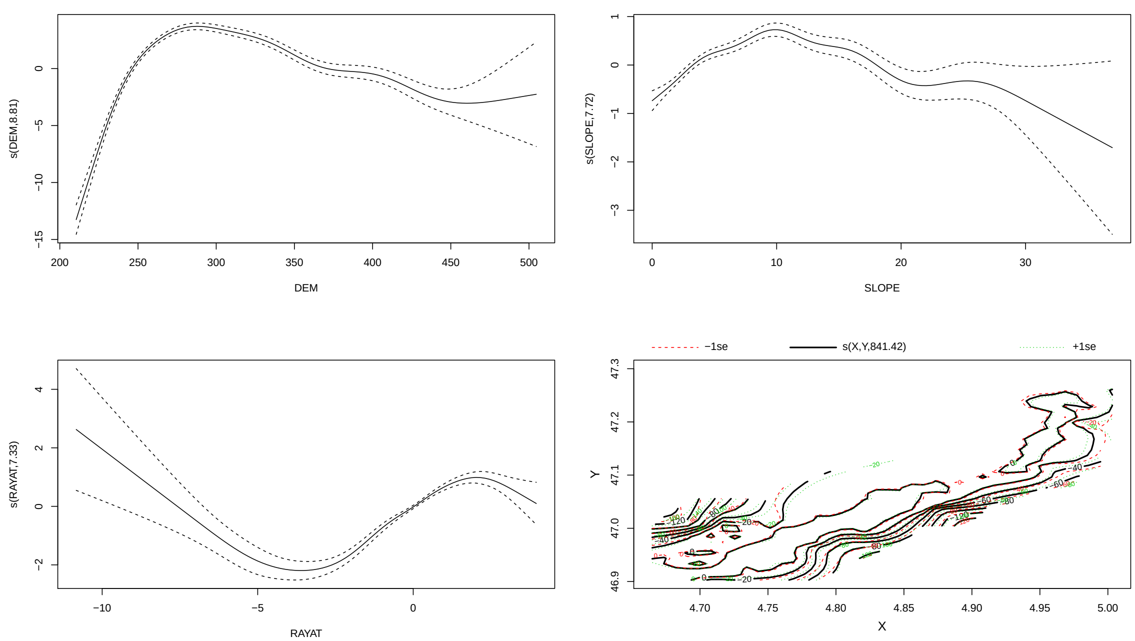


Figure 5: **Marginal effects of topographic variables from OGAM with edf= 900**

4.3 Ordinal superiority figure

From the equation (4) of the Working Paper (p.14), we can compute ordinal superiority measures for each *communes* relatively to the average. The code below reproduces the Figure 2 (p. 23) of the Working Paper. Note that we drop the isolated Northern *communes* of *Chenôve*, *Marsannay-la-Côte* and *Couchey* which do not have comparable neighbors. The effect of the proximity to Dijon is too high for these *communes* and they present high ordinal superiority measures without high rated vineyards.

```r
library(latticeExtra)
plogi <- function(x) exp(x/ sqrt(2))/ (1+ exp(x/ sqrt(2)))
xx <- data.frame(sapply(gamod, function(x)
    2* plogi(I(x$coeff[ 4: 31]- mean(x$coeff[ 4: 31])))- 1))
foo_key <- list(x = .35, y = .95, corner = c(1, 1),
            text = list(c("Côte de Beaune", "Côte de Nuits")),
            rectangle = list(col = c("chartreuse", "tomato")))
ww <- data.frame(xx,
                 LIBCOM= substr(names(gamod[[1]]$coef[ 4: 31]), 7, 30),
                 REGION= c(rep("tomato", 12), rep("chartreuse", 16)),
                 MIN= apply(xx[ 8: 10], 1, min),
                 MAX= apply(xx[ 8: 10], 1, max),
                 MEAN= apply(xx[ 8: 10], 1, mean))
segplot(reorder(factor(LIBCOM), MEAN)~ MIN+ MAX, length= 5, draw.bands= T,
        data= ww[order(ww$MEAN), ], center= MEAN, type= "o",
        key= foo_key, col= as.character(ww$REGION[order(ww$MEAN)]),
        unit = "mm", axis = axis.grid, col.symbol= "black", cex= 1,
        xlab= "Min, Mean and Max of Ordinal Superiorty Measures")
```

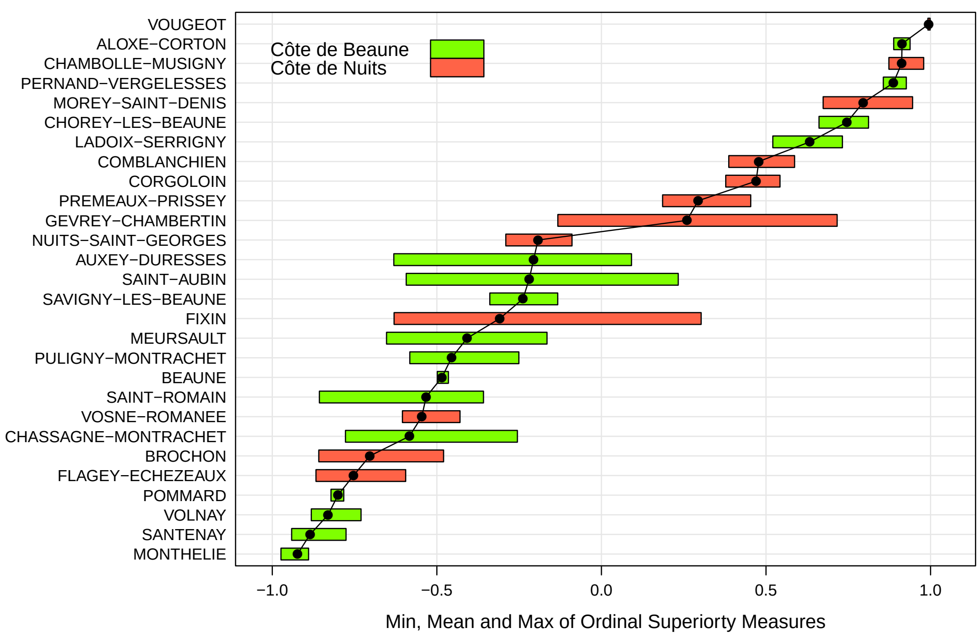


Figure 6: **Ordinal superiorty measures for the *communes* of the region**

4.4 Correlation between *Communes*

Below the code to produce the Figure 8 in Appendix p.42 of the Working Paper. It shows the correlation between the average vertical GI score and the mean ordinal superiority measures estimated from OGAMs with high effective degrees of freedom.

```
library(plyr) ; library(ggrepel)
yy <- ddply(Reg.Ras@data, .(LIBCOM),
            function(x) weighted.mean(x$AOC, x$AREA))
zz <- merge(ww, yy, by= "LIBCOM")
m <- lm(V1~ MEAN, data= zz)
a <- signif(coef(m)[1], digits = 2)
b <- signif(coef(m)[2], digits = 2)
c <- signif(summary(m)$r.sq, digits = 2)
textlab <- paste("y = ", a, " + ", b, " x ", ", R2 = ", c, sep= "")
ggplot(zz, aes(MEAN, V1, label= LIBCOM)) +
    geom_smooth(method= lm, aes(MEAN, V1))+
    geom_text_repel(point.padding = NA) +
    annotate("text", x= -.75, y= 4, label= textlab, size= 4, parse= F)+
    xlab("Ordinal superiority measure") +
    ylab("Average GI grade (between 0 and 5)")
```

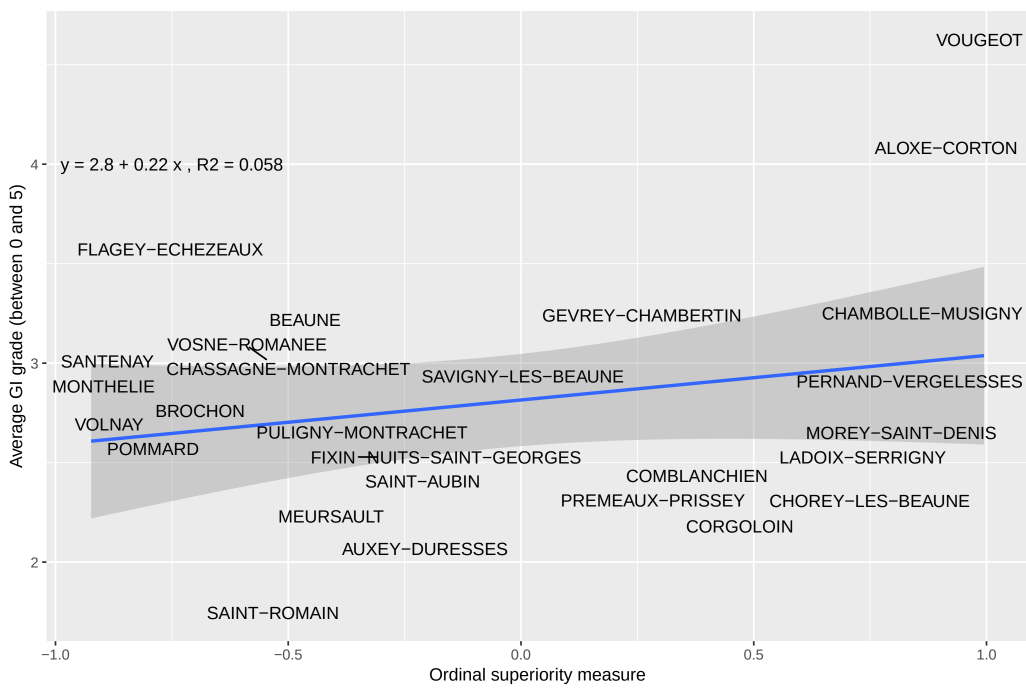


Figure 7: **Correlation between ranking and ordinal superiority**

5 Informational content

5.1 Decomposition tables

We proceed now to the decomposition of variance of the latent quality index from the GI designations. The mathematical formula and codes used in the decomposition are presented and tested in Appendix A.4. These functions are also available in the file of custom function ./myFcts.R that can be directly sourced. The following codes perform the decomposition for the subset of models reported in Table 2 (p.25) of the Working Paper. The predictions of the latent quality index in the first rows need some time to run, the decomposition that follow are computed rapidly.

```
ddtt <- data.frame(AOC= Reg.Ras$AOC, LIBCOM= Reg.Ras$LIBCOM,
                   sapply(gamod[ 1: 5* 2], function(x)
                       rowSums(predict(x, type= 'terms')[, -1])))
dcmp <- sapply(names(ddtt[, 3: 7]), function(x)
    c("Total Signal"= var(ddtt[, x]), "Total Noise"= pi^2/ 3,
      jointSignal(ddtt, x),                      jointNoise(ddtt, x),
      vertiSignal(ddtt, x), vertiResid(ddtt, x), vertiNoise(ddtt, x),
      horizSignal(ddtt, x), horizResid(ddtt, x), horizNoise(ddtt, x)))
round(t(apply(dcmp, 1, function(x) x/ (pi^2/ 3+ dcmp[1, ])* 100)), 1)
```

```
                    gam100 gam300 gam500 gam700 gam900
Total Signal          85.3   94.5   96.0   97.3   97.5
Total Noise           14.7    5.5    4.0    2.7    2.5
Joint Signal          69.7   70.1   76.7   75.2   78.6
Joint Noise           15.6   24.3   19.3   22.2   18.9
Vertical Signal       54.1   48.8   51.7   56.2   65.2
Vertical Residual     15.7   21.4   25.0   18.9   13.4
Vertical Noise        31.3   45.7   44.4   41.1   32.3
Horizontal Signal     18.3   16.6   25.6   22.6   23.8
Horizontal Residual   51.4   53.6   51.1   52.6   54.8
Horizontal Noise      67.0   77.9   70.5   74.7   73.7
```

6 Models for GIs of 1936

6.1 Descriptive statistics

We turn now to the detail of the analysis with past 1936 GIs. We make the same analysis than for actual GIs, first with some descriptive statistics.

```
Reg.Old <- subset(Reg.Ras, !Reg.Ras$LIBCOM %in%
                           c("CHENOVE", "MARSANNAY-LA-COTE", "COUCHEY",
                             "COMBLANCHIEN","CORGOLOIN", "SAINT-ROMAIN"))
Reg.Old$LIBCOM <- factor(Reg.Old$LIBCOM)
Reg.Old$AOCo <- as.numeric(ifelse(Reg.Old$AOC36lvl== "0", 1,
                           ifelse(Reg.Old$AOC36lvl== "3", 2, 3)))
table(Reg.Old$AOCo, Reg.Old$AOC)
```

```
       1     2     3     4     5
1   7124 11452  5111   575    39
2      5   536 15175  8101   261
3      0     1    13     3  1604
```

6.2 Estimation

We estimate both the parametric and generalized additive models we the following codes. Because of the long computation times, the reader would prefer to fit the models individually.

```
library(MASS)
polm2 <- polr(factor(AOCo)~ 0+ LIBCOM+ EXPO+ GEOL+ PEDO
              + poly(DEM, 2)+ poly(SLOPE, 2)+ poly(RAYAT, 2)
              + poly(X, 3)* poly(Y, 3), data= Reg.Old, Hess= T)
polm2a <- polr(factor(AOCo)~ 0+ EXPO+ GEOL+ PEDO
               + poly(DEM, 2)+ poly(SLOPE, 2)+ poly(RAYAT, 2)
               + poly(X, 3)* poly(Y, 3), data= Reg.Old, Hess= T)
polm2b <- polr(factor(AOCo)~ 0+ LIBCOM+ EXPO+ GEOL+ PEDO
               + poly(DEM, 2)+ poly(SLOPE, 2)+ poly(RAYAT, 2)
             , data= Reg.Old, Hess= T)

library(mgcv)
listk <- c(25, 50, 75, 100, 125, 150, 200, 250)
gamold <- vector("list", length(listk))
system.time(
    for (i in 1: length(listk)){
        gamold[[ i]] <- gam(AOCo~ 0+ LIBCOM+ EXPO+ GEOL+ PEDO
                            + s(DEM)+ s(SLOPE)+ s(RAYAT)
                            + s(X, Y, k= listk[ i])
                          , data= Reg.Old, family= ocat(R= 3))
    }
)
names(gamold) <- paste0("gam", listk)
save(gamold, file= "Inter/gamold.Rda")

gammold <- vector("list", length(listk))
system.time(
for (i in 1: length(listk)){
    gammold[[ i]] <- gam(AOCo~ 0+ EXPO+ GEOL+ PEDO
                         + s(DEM)+ s(SLOPE)+ s(RAYAT)
                         + s(X, Y, k= listk[ i])
                       , data= Reg.Old, family= ocat(R= 3))
})
names(gammold) <- paste0("gam", listk)
save(gammold, file= "Inter/gammold.Rda")
```

```
utilisateur     système      écoulé
    20454.2       309.5     20766.0
utilisateur     système      écoulé
    28307.5       462.8     28772.0
```

6.3 Significance

We first assess the joint significance of variables in all OGAMs of GIs designation. The following results are reported in Table 5 in Appendix p.43 of the Working Paper.

```r
load("Inter/gamold.Rda")
res2a <- anova(polm2, polm2b)
res2 <- Anova(polm2)
sapply(gamold[ 1: 7], resume)
```

```
           gam25  gam50  gam100  gam125  gam150  gam200  gam250
s(DEM)    1503.8 1196.2   197.7   219.6   144.8   265.0   253.0
             8.6    8.8     7.6     8.4     8.2     8.7     7.4
s(SLOPE)   534.2  478.1   466.5   332.8   297.1   190.4   169.1
             8.7    8.8     8.7     8.8     8.7     8.8     7.5
s(RAYAT)   339.4  208.8   139.4   150.2    99.2    87.7   142.8
             8.3    8.0     1.1     8.0     8.1     7.4     7.4
s(X,Y)    4789.1 6760.0 14558.9 15981.2 17285.3 18979.3 20905.7
            23.9   48.7    98.0   122.4   147.1   194.3   235.3
LIBCOM    5828.9 3720.9  2639.2  2378.3  2177.2  1831.7  1264.7
            25.0   25.0    25.0    25.0    25.0    25.0    25.0
EXPO       258.0  177.5   131.9   101.2    58.5    43.0    64.0
             7.0    7.0     7.0     7.0     7.0     7.0     7.0
GEOL      1018.5 1047.0   692.1   772.8   710.2   585.8   509.3
            14.0   14.0    14.0    14.0    14.0    14.0    14.0
PEDO      3335.3 2820.6   898.8   660.3   599.4   537.0   539.3
            12.0   12.0    12.0    12.0    12.0    12.0    12.0
```

6.4 Goodness of fit

Goodness of fit measures from the same Table 5 in Appendix p.43 of the Working Paper.

```r
round(c(McFaddenR2= psR2(polm2), AIC= AIC(polm2)/ 1000,
        Pcgp= sum(diag(table(predict(polm2), Reg.Old$AOCo)))/ nrow(Reg.Old)), 2)
rbind(Pcgp= sapply(gamold[1: 7 ], pcgp),
      AIC=  sapply(gamold[1: 7 ], AIC)/ 1000,
      psR2= sapply(gamold[1: 7 ], psR2))
```

```
McFaddenR2         AIC        Pcgp
      0.45       45.22        0.82
        gam25   gam50  gam100 gam125  gam150  gam200  gam250
Pcgp 82.8820 83.7580 87.8840 88.606 89.8400 91.3480 92.2060
AIC  43.9251 41.2140 31.8196 30.039 28.0878 25.1203 23.1212
psR2  0.4629  0.4968  0.6132  0.636  0.6606  0.6982  0.7236
```

6.5 Omitted variable

Bootstrapped statistics for omitted variables, reported in Table 5 in Appendix p.43 of the Working Paper.

```
library(lmtest) ; library(sandwich) ; library(sure) ; library(ggplot2)
wal3 <- rep(NA, nsim <- 100)
for (i in 1: nsim){
    tmp <- surrogate(polm2a)- polm2a$lp
    wal3[ i] <- waldtest(lm(tmp~ Reg.Old$LIBCOM), . ~ 1, vcov= vcovHC)$F[ 2]
}
load("Inter/gammold.Rda") ; library(ggplot2) ; source("myFcts.R")
wal4 <- sapply(gammold, function(x) omitVar(x, Reg.Old$LIBCOM, nsim))
wold <- data.frame(logit= wal3, wal4)
apply(wold, 2, function(x) quantile(x, c(.05, .5, .95)))
```

```
    logit gam25  gam50 gam100 gam125 gam150 gam200 gam250
5%  88.18 24.15  8.776  3.903  3.577  2.025  1.700  1.207
50% 92.78 26.62 10.171  4.802  4.372  2.740  2.308  1.748
95% 97.08 28.57 11.472  5.594  5.647  3.638  3.452  2.583
```

Now the same violon plot as for current GIs, not reported the Working Paper but mentioned at p.24.

```
library(lattice)
poldat <- stack(wold)
Fstat <- round(qf(.9999, 31, Inf), 2)
bwplot(log(values)~ ind, data= poldat, type=c("l","g"), horizontal= FALSE,
       xlab= 'Model of GI designation',
       ylab= 'Bootstraped F-statistics (log scale)',
       par.settings = list(box.rectangle=list(col='black'),
                           plot.symbol  = list(pch='.', cex = 0.1)),
       scales=list(y= list(at= log((1: 15)^2), lab= (1: 15)^2)),
       panel = function(..., box.ratio) {
           panel.grid(h= 0, v = -11)
           panel.abline(h= log((1: 15)^2), col= "grey80")
           panel.violin(..., col = "lightblue",
                        varwidth = FALSE, box.ratio = box.ratio)
           panel.bwplot(..., col='black',
                        cex=0.8, pch='|', fill='gray', box.ratio = .1)
           panel.abline(h= log(Fstat), col= "red", lty= 2, cex= 1.5)
           panel.text(2, log(Fstat)+ .1,
                      paste0("F= ", Fstat, " : critical value at .01%"))})
```

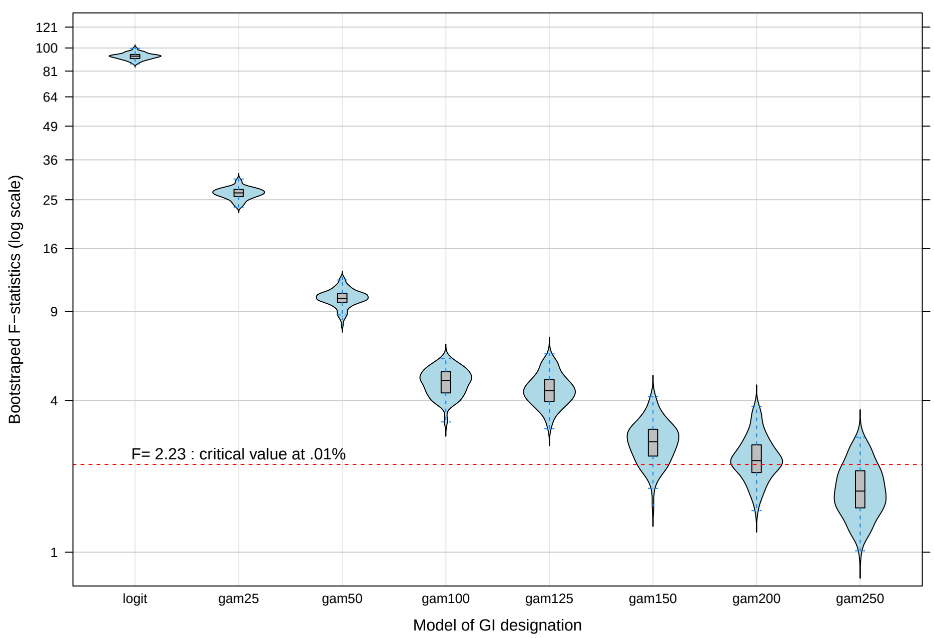


Figure 8: **F-statistics for the omitted *terroir* effects in 1936 GIs**

6.6 Specification

The use of surrogate residuals to test the specification process for models of 1936 GI designations. As before, the results are not reported because the resulting file is too big.

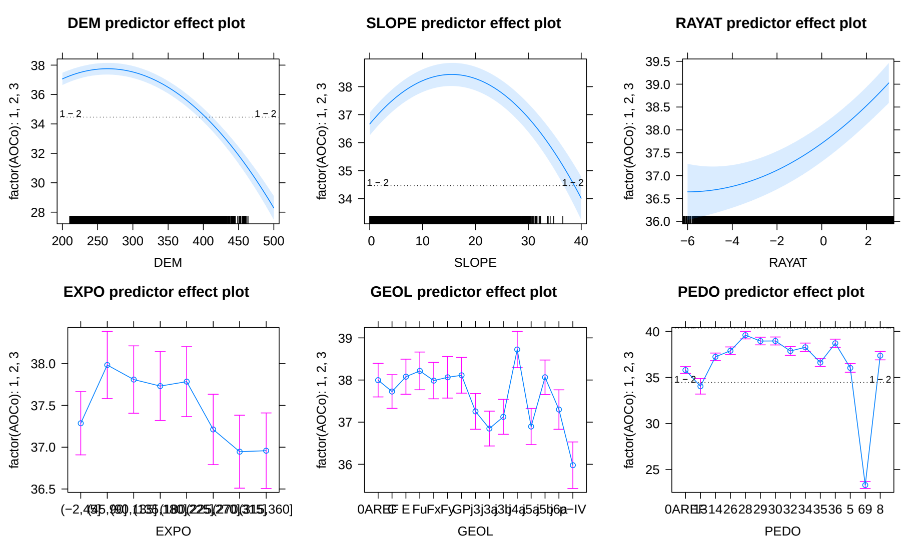
```r
library(sure) ; library(ggplot2) ; library(gridExtra)
var <- c("DEM", "SLOPE", "RAYAT", "EXPO", "LIBCOM", "X", "Y")
plots <- lapply(var, function(.x)
    autoplot(polm2, what= "covariate", x= Reg.Old@data[, .x], xlab= .x))
do.call(grid.arrange, c(list(autoplot(polm2, what= "qq")), plots))

restmp <- sureOGAM(gamold$gam150)- gamold$gam150$line
plot(qlogis(1: nrow(Reg.Old)/ nrow(Reg.Old), scale= 1), sort(restmp))
abline(0, 1)
var <- c("DEM", "SLOPE", "RAYAT", "EXPO", "LIBCOM", "X", "Y")
par(mfrow= c(3, 3)) ; for (i in var) pltSURE(restmp, Reg.Old@data[, i], i)
```

6.7 Marginal effects

Marginal effect can be assessed as for current GIs, the code belows can be used on the models from the `gamold` object to produce Figure 9 in the Appendix p.44 in the Working Paper.

```r
library(effects)
plot(predictorEffects(polm2, ~ DEM+ SLOPE+ RAYAT+ EXPO+ GEOL+ PEDO,
                      latent= TRUE,
                      xlevels=list(DEM= 200: 500,
                                   SLOPE= 0: 400/ 10, RAYAT= -60: 30/ 10)))
# plot(gamold$gam125, pages= 1, scale= 0)
```



6.8 Ordinal superiority measures

Ordinal superiority for the GIs of 1936, that corresponds to Figure 11 in the Appendix p. 46 of the Working Paper.

```
plogi <- function(x) exp(x/ sqrt(2))/ (1+ exp(x/ sqrt(2)))
load("Inter/gamold.Rda")
xxx <- data.frame(sapply(gamold, function(x)
    2* plogi(I(x$coef[ 1: 25]- mean(x$coef[ 1: 25])))- 1))
www <- data.frame(xxx,
                  LIBCOM= substr(names(gamold[[ 1]]$coef[ 1: 25]), 7, 30),
                  REGION= c(rep("tomato", 10), rep("chartreuse", 15)),
                  MIN= apply(xxx[ 11: 12], 1, min),
                  MAX= apply(xxx[ 11: 12], 1, max),
                  MEAN= apply(xxx[ 11: 12], 1, mean))
segplot(reorder(factor(LIBCOM), MEAN)~ MIN+ MAX, length= 5, draw.bands= T,
        data= www[order(www$MEAN), ], center= MEAN, type= "o",
        key= foo_key, col= as.character(www$REGION[order(www$MEAN)]),
        unit = "mm", axis = axis.grid, col.symbol= "black", cex= 1,
        xlab= "Min, Mean and Max of Ordinal Superiorty Measures")
```

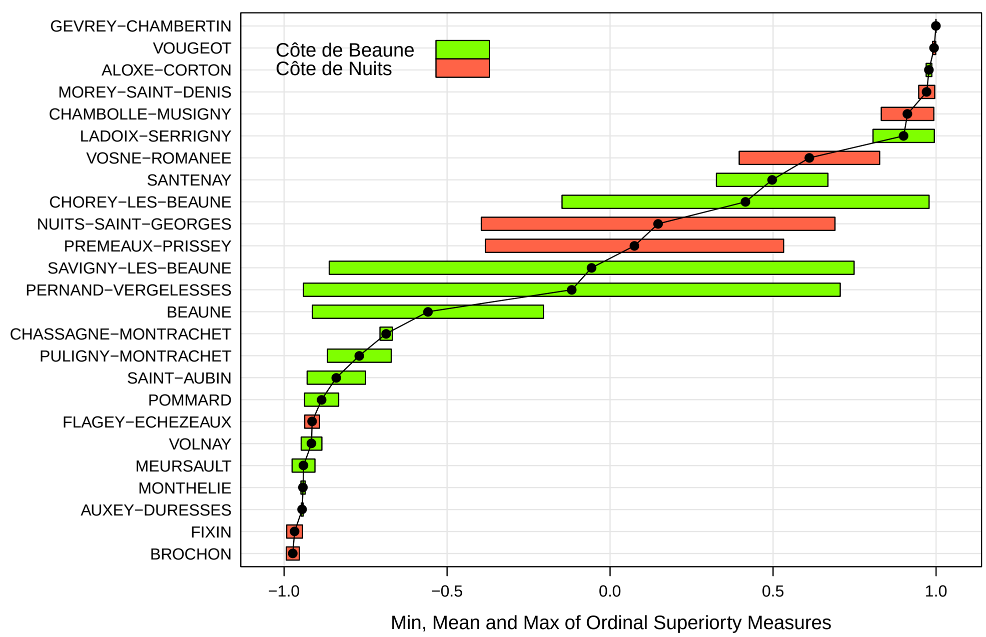


Figure 9: **Ordinal superiorty measures for the 1936 GIs**

6.9 Temporal variations

An additional Figure 3 (p.26) of the Working Paper.

```
zzz <- merge(ww, www, by= "LIBCOM")
segplot(reorder(factor(LIBCOM), MEAN.x)~ MEAN.y+ MEAN.x, data= zzz,
        segments.fun = panel.arrows, length = 5, unit = "mm",
        key= foo_key, col= as.character(zzz$REGION.x),
        draw.bands= F, axis = axis.grid, lwd= 3,
        xlab= "Variation of ordinal superiority measure from 1936 to now")
```

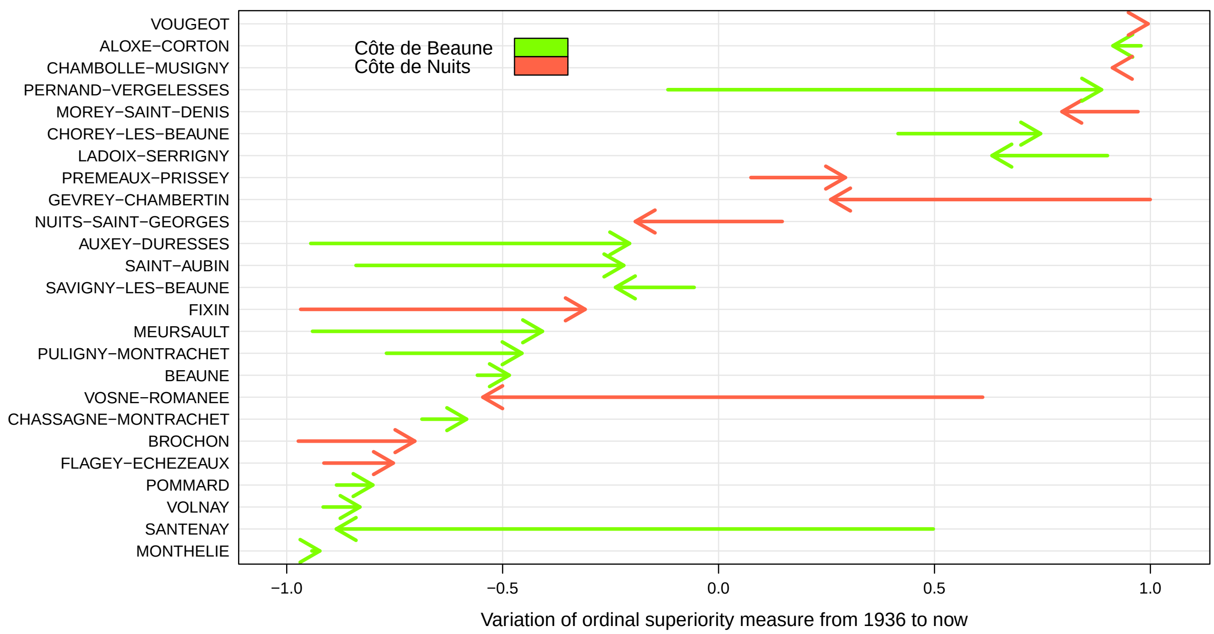


Figure 10: **Evolution of superiority measures between 1936 and now**

6.10 Decomposition table

The code below compute the decomposition table for GIs of 1936, unreported.

```
ddoo <- data.frame(AOCo= Reg.Old$AOCo, LIBCOM= Reg.Old$LIBCOM,
                   sapply(gamold[ 1: 7], function(x)
                       rowSums(predict(x, type= 'terms')[, -1])))
dcop <- sapply(names(ddoo[, 3: 9]), function(x)
    c("Total Signal"= var(ddoo[, x]), "Total Noise"= pi^2/ 3,
      jointSignal(ddoo, x, "AOCo"), jointNoise(ddoo, x, "AOCo"),
      vertiSignal(ddoo, x, "AOCo"), vertiResid(ddoo, x, "AOCo"),
      vertiNoise(ddoo,  x, "AOCo"),
      horizSignal(ddoo, x, "AOCo"), horizResid(ddoo, x, "AOCo"),
      horizNoise(ddoo, x,  "AOCo")))
round(t(apply(dcop, 1, function(x) x/ (pi^2/ 3+ dcop[1, ])* 100)), 1)
```

	gam25	gam50	gam100	gam125	gam150	gam200	gam250
Total Signal	95.9	98.3	97.2	97.4	100.0	99.1	99.6
Total Noise	4.1	1.7	2.8	2.6	0.0	0.9	0.4
Joint Signal	90.8	95.0	72.2	56.1	98.5	59.2	84.4
Joint Noise	5.1	3.4	25.0	41.4	1.5	39.9	15.2
Vertical Signal	2.4	1.3	19.7	16.8	3.1	20.3	13.0
Vertical Residual	88.4	93.7	52.6	39.2	95.4	38.9	71.4
Vertical Noise	93.5	97.1	77.5	80.6	96.9	78.8	86.6
Horizontal Signal	86.0	92.0	54.5	31.7	97.8	39.7	74.8
Horizontal Residual	4.8	3.0	17.7	24.4	0.7	19.5	9.6
Horizontal Noise	9.9	6.4	42.7	65.8	2.1	59.4	24.8

7 Alternative GI designations

7.1 Change latent vineyard quality

We conclude this work with the simulations of alternative GIs designations schemes. Below are scenarios S0 to S3 where the counterfactual GI designations are computed according to (we note $\hat{q}_i^{gam} = B(X_i)^\top \hat{\beta}^{gam}$):

$$
\begin{aligned}
y_i^{S0} &= \sum_{j=0}^{5} j \cdot \mathbb{1}[\hat{\alpha}_{j_i-1} + \hat{\mu}_{c_i} \geqslant \hat{q}_i^{gam} + \hat{\xi}_i^{sur} \geqslant \hat{\alpha}_{j_i} + \hat{\mu}_{c_i}] \\
y_i^{S1} &= \sum_{j=0}^{5} j \cdot \mathbb{1}[\hat{\alpha}_{j_i-1} + \hat{\mu}_{c_i} \geqslant \hat{q}_i^{gam} \geqslant \hat{\alpha}_{j_i} + \hat{\mu}_{c_i}] \\
y_i^{S2} &= \sum_{j=0}^{5} j \cdot \mathbb{1}[\hat{\alpha}_{j_i-1} \geqslant \hat{q}_i^{gam} + \hat{\xi}_i^{sur} \geqslant \hat{\alpha}_{j_i}] \\
y_i^{S3} &= \sum_{j=0}^{5} j \cdot \mathbb{1}[\hat{\alpha}_{j_i-1} \geqslant \hat{q}_i^{gam} \geqslant \hat{\alpha}_{j_i}]
\end{aligned}
$$

```
prdd <- predict(gamod$gam900, type= 'terms')
thsld <- c(-Inf, gamod$gam900$family$getTheta(TRUE), Inf)
ltt0 <- rowSums(prdd)- (sureOGAM(gamod$gam900)- gamod$gam900$line)
ltt1 <- rowSums(prdd)
ltt2 <- mean(prdd[, 1])+ rowSums(prdd[, -1])-
    (sureOGAM(gamod$gam900)- gamod$gam900$line)
ltt3 <- mean(prdd[, 1])+ rowSums(prdd[, -1])
Simu <- data.frame(Reg.Ras, ltt= rowSums(prdd[, -1]),
                   OLD= Reg.Ras$AOC36lvl, S0= cut(ltt0, thsld),
                   SI= cut(ltt1, thsld), SII= cut(ltt2, thsld),
                   SIII= cut(ltt3, thsld))
table(Simu$AOC, Simu$S0) ; table(Simu$AOC, Simu$SI)
table(Simu$AOC, Simu$SII) ; table(Simu$AOC, Simu$SIII)
```

```
   (-Inf,-1] (-1,5.34] (5.34,14] (14,21] (21, Inf]
 1      7847      1510       269      40         9
 2      1688      9476      2126     158        98
 3       146      2360     20652    2005       146
 4        25       117      2160    5956       421
 5         0         1        84     455      1364

   (-Inf,-1] (-1,5.34] (5.34,14] (14,21] (21, Inf]
 1      8592      1021        62       0         0
 2       562     11787      1147      50         0
 3         7       929     23528     834        11
 4         0         9      1089    7446       135
 5         0         0         1     363      1540

   (-Inf,-1] (-1,5.34] (5.34,14] (14,21] (21, Inf]
 1      7580      1770       280      34        11
 2      2150      7655      3482     150       109
 3       409      5038     16162    3521       179
 4        28       127      2039    5389      1096
 5         0         8       185     611      1100

   (-Inf,-1] (-1,5.34] (5.34,14] (14,21] (21, Inf]
 1      8197      1403        73       2         0
 2      1624      8961      2875      85         1
 3       111      4655     17666    2873         4
 4         0        24      1631    6229       795
 5         0         0        83     636      1185
```

7.2 Add a vertical level in GIs

Here we simulate counterfactual GIs designations from scenarios S4, S5, and S6. In each case, we use the GIs from S0 and add a vertical level by computing the mean of the thresholds.

```
thrldBOUR <- mean(ltt0[Reg.Ras$AOC== 2])
thrldVILL <- mean(ltt0[Reg.Ras$AOC== 3])
thrldPCRU <- mean(ltt0[Reg.Ras$AOC== 4])
Simv <- data.frame(Simu,
       SIV= ifelse(Reg.Ras$AOC< 2, Reg.Ras$AOC,
            ifelse(Reg.Ras$AOC== 2 & ltt0< thrldBOUR, 2,
            ifelse(Reg.Ras$AOC== 2 & ltt0>= thrldBOUR, 3, Reg.Ras$AOC+1))),
       SV = ifelse(Reg.Ras$AOC< 3, Reg.Ras$AOC,
            ifelse(Reg.Ras$AOC== 3 & ltt0< thrldVILL, 3,
            ifelse(Reg.Ras$AOC== 3 & ltt0>= thrldVILL, 4, Reg.Ras$AOC+1))),
       SVI= ifelse(Reg.Ras$AOC< 4, Reg.Ras$AOC,
            ifelse(Reg.Ras$AOC== 4 & ltt0< thrldPCRU, 4,
            ifelse(Reg.Ras$AOC== 4 & ltt0>= thrldPCRU, 5, Reg.Ras$AOC+1))))
table(Simv$AOC, Simv$SIV) ; table(Simv$AOC, Simv$SV) ; table(Simv$AOC, Simv$SVI)
```

```
       1     2     3     4     5     6
1   9675     0     0     0     0     0
2      0  7400  6146     0     0     0
3      0     0     0 25309     0     0
4      0     0     0     0  8679     0
5      0     0     0     0     0  1904

       1     2     3     4     5     6
1   9675     0     0     0     0     0
2      0 13546     0     0     0     0
3      0     0 12792 12517     0     0
4      0     0     0     0  8679     0
5      0     0     0     0     0  1904

       1     2     3     4     5     6
1   9675     0     0     0     0     0
2      0 13546     0     0     0     0
3      0     0 25309     0     0     0
4      0     0     0  4072  4607     0
5      0     0     0     0     0  1904
```

7.3 Decomposition table

And we conclude with the decomposition that are reported Table 3 (p.27) of the Working Paper.

```
decf <- sapply(names(Simv[, 76: 83]), function(x)
    c("Total Signal"= var(Simv[, "ltt"]), "Total Noise"= pi^2/ 3,
      jointSignal(Simv, "ltt", vt= x), jointNoise(Simv, "ltt", vt= x),
      vertiSignal(Simv, "ltt", vt= x), vertiResid(Simv, "ltt", vt= x),
      vertiNoise(Simv,  "ltt", vt= x),
      horizSignal(Simv, "ltt", vt= x), horizResid(Simv, "ltt", vt= x),
      horizNoise(Simv,  "ltt", vt= x)))
round(t(apply(decf, 1, function(x) x/ (pi^2/ 3+ decf[1, ])* 100)), 1)
```

```
                     OLD   S0   SI  SII SIII  SIV   SV  SVI
Total Signal        97.5 97.5 97.5 97.5 97.5 97.5 97.5 97.5
Total Noise          2.5  2.5  2.5  2.5  2.5  2.5  2.5  2.5
Joint Signal        48.0 78.2 81.0 79.5 81.5 79.0 79.5 78.9
Joint Noise         49.5 19.3 16.5 18.1 16.0 18.5 18.0 18.6
Vertical Signal     34.4 64.6 68.2 69.7 72.6 65.6 66.1 65.5
Vertical Residual   13.6 13.6 12.8  9.7  8.9 13.4 13.4 13.4
Vertical Noise      63.1 32.9 29.3 27.8 24.9 31.9 31.4 32.0
Horizontal Signal   23.8 23.8 23.8 23.8 23.8 23.8 23.8 23.8
Horizontal Residual 24.2 54.4 57.2 55.7 57.7 55.2 55.7 55.1
Horizontal Noise    73.7 73.7 73.7 73.7 73.7 73.7 73.7 73.7
```

8 Session information

```
sessionInfo()
```

```
R version 3.6.0 (2019-04-26)
Platform: x86_64-pc-linux-gnu (64-bit)
Running under: Ubuntu 18.04.2 LTS
Matrix products: default
BLAS:   /usr/lib/x86_64-linux-gnu/blas/libblas.so.3.7.1
LAPACK: /usr/lib/x86_64-linux-gnu/lapack/liblapack.so.3.7.1
locale:
 [1] LC_CTYPE=fr_FR.UTF-8       LC_NUMERIC=C
 [3] LC_TIME=fr_FR.UTF-8        LC_COLLATE=fr_FR.UTF-8
 [5] LC_MONETARY=fr_FR.UTF-8    LC_MESSAGES=fr_FR.UTF-8
 [7] LC_PAPER=fr_FR.UTF-8       LC_NAME=C
 [9] LC_ADDRESS=C               LC_TELEPHONE=C
[11] LC_MEASUREMENT=fr_FR.UTF-8 LC_IDENTIFICATION=C
attached base packages:
[1] stats     graphics  grDevices utils     datasets  methods
[7] base
other attached packages:
 [1] latticeExtra_0.6-28 effects_4.0-3       gridExtra_2.3
 [4] ggplot2_3.1.0       sure_0.2.0          sandwich_2.5-0
 [7] lmtest_0.9-36       zoo_1.8-4           mgcv_1.8-28
[10] nlme_3.1-140        car_3.0-2           carData_3.0-1
[13] MASS_7.3-51.1       RColorBrewer_1.1-2  lattice_0.20-38
[16] sp_1.3-1
loaded via a namespace (and not attached):
 [1] tidyselect_0.2.5  purrr_0.3.2       splines_3.6.0
 [4] haven_1.1.2       colorspace_1.3-2  survival_2.43-3
 [7] rlang_0.3.4       nloptr_1.0.4      pillar_1.3.0
[10] foreign_0.8-71    glue_1.3.0        withr_2.1.2
[13] readxl_1.1.0      bindrcpp_0.2.2    plyr_1.8.4
[16] bindr_0.1.1       munsell_0.5.0     gtable_0.2.0
[19] cellranger_1.1.0  zip_1.0.0         labeling_0.3
[22] rio_0.5.10        forcats_0.3.0     curl_3.2
[25] Rcpp_1.0.0        scales_1.0.0      abind_1.4-5
[28] lme4_1.1-18-1     hms_0.4.2         openxlsx_4.1.0
[31] dplyr_0.7.8       survey_3.33-2     grid_3.6.0
[34] rgdal_1.3-6       tools_3.6.0       magrittr_1.5
[37] lazyeval_0.2.1    tibble_1.4.2      crayon_1.3.4
[40] pkgconfig_2.0.2   Matrix_1.2-17     data.table_1.11.4
[43] minqa_1.2.4       assertthat_0.2.1  R6_2.4.0
[46] nnet_7.3-12       compiler_3.6.0
```

A Custom R functions

A.1 Translation of geology

```
trans_geol <- data.frame(
    GEOL= Reg.Ras$GEOL[!duplicated(Reg.Ras$GEOL)],
    GEOf= c(
        "Calcaires massifs de \"Comblanchien\" (Bathonien sup.)",
        "Marnes et calcaires divers (Callovien inférieur)",
        "Marnes et calcaires argileux (Oxfordien moyen)",
        "Eboulis ordonnés cryoclastiques et colluvions diverses",
        "Oolithe ferrugineuse (Oxfordien moyen-sup)",
        "Calcaires hydrauliques de Molesmes et Noiron (Oxfordien sup.)",
        "Colluvions diverses",
        "Dépôts argilo-limoneux, sables et graviers du Villafranchien",
        "Calcaires de Tonnerre, Oisellemont et calcaires á Astartes",
        "Eboulis et glissements de terrains",
        "Calcaires grenus bicolores (Bathonien terminal)",
        "Terrasse argilo-limoneuse de Saint-Usage",
        "Formation de Saint-Cosme (marnes fluvio-lacustres varvées)",
        "Alluvions anciennes indifférenciées, argilo-limoneuses",
        "Calcaires bioclastiques, graveleux, á oolithes (Bathonien inf.)"
    ),
    GEOe= c(
        "Massive limestones from \"Comblanchien\" (upper Bathonian)",
        "Various marls and limestones (lower Callovian)",
        "Marls and argillaceous limestones (middle Oxfordian)",
        "Ordered cryoclastic scree and various colluviums",
        "Ferruginous Oolite (middle-upper Oxfordian)",
        "Hydraulic limestones of Molesmes and Noiron (upper Oxfordian)",
        "Various colluviums",
        "Clay-silt deposits, sand and gravel from Villafranchien",
        "Limestones of Thunder, Oisellemont and limestones in Astartes",
        "Screes and landslides",
        "Two-tone gray limestones (terminal Bathonian)",
        "Clay-silty terrace of Saint-Usage",
        "Formation of Saint-Cosme (varnished fluvio-lacustrine marls)",
        "Undifferentiated ancient alluvium, clay-silty",
        "Bioclastic limestones, gravelly, with oolites (lower Bathonian)")
)
```

A.2 Translation of pedology

```
trans_pedo <- data.frame(
    PEDO= Reg.Ras$PEDO[!duplicated(Reg.Ras$PEDO)],
    PEDf= c(
        "Vignoble de la Côte de de Beaune",
        "Cônes de déjection du pied de Côte",
        "Côteaux viticoles des Hautes Côtes de Nuits",
        "Courtes pentes marneuses des plateaux plio-pléistocène",
        "Piedmont de la côte viticole",
        "Versants pentus des Hautes Côtes de Beaune",
        "Sommets des collines des Hautes Côtes de Beaune",
        "Alluvions récentes calcaires des vallées (Ouche, Tille, Meuzin)",
        "Pentes liasiques du Haut-Auxois",
        "Basses terrasses gravelo-caillouteuses des plaines alluviales",
        "Basses terrasses argileuses des plaines alluviales",
        "Terrasse argilo-limoneuse de Saint-Usage",
        "Vignoble de la Côte de Nuits",
        "Rebord oriental des plateaux calcaires dominant la Côte viticole"
    ),
    PEDe= c(
        "Vineyard of the Côte de Beaune",
        "Coot footing cones",
        "Wine hills of Hautes Côtes de Nuits",
        "Oxfordian limestone-marly trays of the Hautes Côtes",
        "Short marly slopes of Plio-Pleistocene plateaus",
        "Piedmont of the vineyard of the Côte",
        "Sloping slopes of the Hautes Côtes de Beaune",
        "Summits of the hills of the Hautes Côtes de Beaune",
        "Recent alluvial limestone valleys (Ouche, Tille, Meuzin)",
        "Liastic slopes of Haut-Auxois",
        "Gravelo-stony low terraces of alluvial plains",
        "Low clay terraces of alluvial plains",
        "Vineyard of the Côte de Nuits",
        "Eastern edge of the limestone plateaus overlooking the Côte"
    )
)
```

A.3 Surrogate Residuals

The R package sure allows to simulate the surrogate residuals from a large panel of ordered parametric models https://koalaverse.github.io/sure/index.html. Actually, it is not possible to compute the residuals for semiparametric ordered generalized additive model fitted with the package mgcv. Here, we first define the truncLogis function for the simulation of random draws from a truncated logistic distribution with a vector of inputs (locations and thresholds) as the package truncdist is only designed for a given value of location and thresholds. Then, we code the function surePOLR inspired from the sure package which simulate surrogate residuals from polr models from the MASS package. This will be used to check the validity of used functions.

```
truncLogis <- function(n, spec, a = -Inf, b = Inf, ...) {
    require(truncdist)
    p <- runif(n, min = 0, max = 1)
    G <- get(paste("p", spec, sep = ""), mode = "function")
    Gin <- get(paste("q", spec, sep = ""), mode = "function")
    G.a <- G(a, ...)
    G.b <- G(b, ...)
    pmin(pmax(a, Gin(G(a, ...) + p * (G(b, ...) - G(a, ...)), ...)), b)
}

surePOLR <- function(mod, newd= NULL){
    if (mod$method!= "logistic") stop("Logistic required")
    gg <- as.numeric(mod$zeta)
    if (is.null(newd)){
        g1 <- as.integer(model.response(model.frame(mod)))
        g6 <- mod$lp
    } else {
        g1 <- as.integer(newd[, "AOCc"])
        g6 <- gg[ 1]-qlogis(predict(mod, newdata= newd, type= 'probs')[, 1])
    }
    nn <- length(g1)
    suls <- sapply(g1, switch,
                   "1"= c(-Inf  , gg[ 1]), "2"= c(gg[ 1], gg[ 2]),
                   "3"= c(gg[ 2], gg[ 3]), "4"= c(gg[ 3], gg[ 4]),
                   "5"= c(gg[ 4], Inf   ))
    sls <- data.frame(unlist(t(suls)))
    truncLogis(nn, spec= "logis", a= sls[, 1], b= sls[, 2],
               location= g6, scale= 1)
}
```

```
sure1 <- surrogate(polm1)+ polm1$zeta[ 1]
sure2 <- resids(polm1)
polr1 <- surePOLR(polm1) ; polr2 <- surePOLR(polm1)- polm1$lp
```

The custom function surePOLR allows to compute the same surrogate value and surrogate residuals than the functions surrogate and resids from the sure package.

Now we use the same structure to simulate the surrogate residuals for the OGAM through the function sureOGAM. Again, the function is tested for a random OGAM.

```
sureOGAM <- function(mod, newd= NULL){
```

```
    if (is.null(newd)){
        g1 <- as.integer(mod$y)
        g6 <- mod$linear.predictors
    } else {
        g1 <- as.integer(newd[, names(mod$model[ 1])])
        g6 <- predict(mod, newdata= newd)
    }
    nn <- length(g1)
    gt <- data.frame(rep(NA, nn), rep(NA, nn))
    gg <- c(mod$family$getTheta(TRUE), Inf)
    kk <- c(- Inf, gg[ 1])
    for (j in 2: length(unique(g1))) kk <- rbind(kk, c(gg[ j- 1], gg[ j]))
    gt <- data.frame(t(sapply(g1, function(x) kk[x, ])))
    truncLogis(nn, spec= "logis", a= gt[, 1], b= gt[, 2], location= g6)
}
```

```
library(mgcv)
fit.ogam <- gam(AOC~ poly(DEM, 2)+ poly(SLOPE, 2)
                + poly(RAYAT, 2)+ poly(ASPECT, 2)+ poly(PERMEA, 2)
              , family= ocat(R= 5), data= Reg.Ras)
ogam1 <- sureOGAM(fit.ogam)
ogam2 <- sureOGAM(fit.ogam)- fit.ogam$linear.pred

par(mfrow= c(3, 2))
plot(sure1, polr1)
abline(h= fit.polr$zeta, v= fit.polr$zeta, lty= 2, col= "blue")
abline(0, 1, col= "orange")
plot(sure2, polr2)
abline(0, 1, col= "orange")

plot(polr1, ogam1- mean(ogam1))
abline(h= fit.ogam$family$getTheta(TRUE)- mean(ogam1),
       v= fit.polr$zeta, lty= 2, col= "blue")
abline(0, 1, col= "orange")
plot(polr2, ogam2)
abline(0, 1, col= "orange")

plot(sure1, ogam1- mean(ogam1))
abline(h= fit.ogam$family$getTheta(TRUE)- mean(ogam1),
       v= fit.polr$zeta, lty= 2, col= "blue")
abline(0, 1, col= "orange")
plot(sure2, ogam2)
abline(0, 1, col= "orange")
```

A.4 Decomposition terms

For each terms of the decomposition presented in the main text, we code a different function as reported below. First note the vector of predicted latent quality index $\hat{q}_i = B(X_i)^\top\hat{\beta}$. With N_y, N_c and $N_{y,c}$ the numbers of vineyard plots respectively in rank y, in *commune* c and both in rank y and *commune* c, we define:

$$\begin{aligned}
\overline{q}_{y_i} &= \frac{1}{N_{y_i}} \sum\nolimits_{\ell=1}^{N} \mathbb{1}[y_\ell = y_i] \cdot \hat{q}_\ell \\
\overline{q}_{c_i} &= \frac{1}{N_{c_i}} \sum\nolimits_{\ell=1}^{N} \mathbb{1}[c_\ell = c_i] \cdot \hat{q}_\ell \\
\overline{q}_{y_i,c_i} &= \frac{1}{N_{y_i,c_i}} \sum\nolimits_{\ell=1}^{N} \mathbb{1}[(y_\ell, c_\ell) = (y_i, c_i)] \cdot \hat{q}_\ell
\end{aligned}$$

The **joint signal** terms is the variance of the expected quality conditionally on vertical and horizontal dummies:

$$\mathbb{V}\{\,\mathbb{E}[q(X^*) \mid y, c]\,\} = \frac{1}{N} \sum\nolimits_{i=1}^{N} [\overline{q}_{y_i,c_i} - \overline{q}]^2 \tag{1}$$

```
jointSignal <- function(dat, lt, vt= "AOC", hz= "LIBCOM"){
    jS <- rep(0, nrow(dat))
    for (i in unique(dat[, vt])){
        for (j in unique(dat[, hz])){
            tmp <- dat[, vt]== i & dat[, hz]== j
            jS[ tmp] <- mean(dat[tmp, lt])
        }
    }
    c("Joint Signal"= var(jS))
}
```

The **joint noise** terms is the expectation of the variance quality conditionally on vertical and horizontal dummies:

$$\mathbb{E}\{\,\mathbb{V}[q(X^*) \mid y, c]\,\} = \sum\nolimits_{y=1}^{J} \sum\nolimits_{c=1}^{C} \left[\tfrac{N_{y,c}}{N} \sum\nolimits_{i=1}^{N} \mathbb{1}[(y_i, c_i) = (y, c)] \cdot (\hat{q}_i - \overline{q}_{y_i,c_i})^2\right] \tag{2}$$

```
jointNoise <- function(dat, lt, vt= "AOC", hz= "LIBCOM"){
    jN <- 0
    for (i in unique(dat[, vt])){
        for (j in unique(dat[, hz])){
            tmp <- dat[, vt]== i & dat[, hz]== j
            if (sum(tmp)> 1) jN <- jN+ var(dat[ tmp, lt])* mean(tmp)
        }
    }
    c("Joint Noise"= jN)
}
```

The **vertical signal** terms is the variance of the expectation quality conditionally on vertical GI dummies:

$$\mathbb{V}\{\,\mathbb{E}[q(X^*)\mid y]\,\} = \frac{1}{N}\sum\nolimits_{i=1}^{N}[\overline{q}_{y_i}-\overline{q}]^2 \tag{3}$$

```r
vertiSignal <- function(dat, lt, vt= "AOC", hz= "LIBCOM"){
    vS <- rep(0, nrow(dat))
    for (i in unique(dat[, vt])){
        vS[ dat[, vt]== i] <- mean(dat[dat[, vt]== i, lt])
    }
    c("Vertical Signal"= var(vS))
}
```

The **vertical residual** terms is the expectation of the conditional on horizontal variance of the expectation quality conditionally on vertical GI dummies:

$$\mathbb{E}\{\,\mathbb{V}[\mathbb{E}(q(X^*)\mid y,c)\mid c]\,\} = \sum\nolimits_{y=1}^{J}\left[\frac{N_y}{N}\sum_{i=1}^{N}(\overline{q}_{y_i}-\overline{q})^2\right] \tag{4}$$

```r
vertiResid <- function(dat, lt, vt= "AOC", hz= "LIBCOM"){
    sig <- rep(0, nrow(dat)) ; vR <- 0
    for (i in unique(dat[, vt])){
        for (j in unique(dat[, hz])){
            tmp <- dat[, vt]== i & dat[, hz]== j
            sig[ tmp] <- mean(dat[ tmp, lt])
        }
    }
    for (i in unique(dat[, vt])){
        vR <- vR+ var(sig[dat[, vt]== i])* mean(dat[, vt]== i)
    }
    c("Vertical Residual"= vR)
}
```

The **vertical Noise** terms is the expectation of the variance of the quality conditionally on vertical GI dummies:

$$\mathbb{E}\{\,\mathbb{V}[q(X^*)\mid y]\,\} = \sum\nolimits_{y=1}^{J}\left[\frac{N_y}{N}\sum\nolimits_{i=1}^{N}\mathbb{1}[y_i=y]\cdot(\hat{q}_i-\overline{q}_{y_i})^2\right] \tag{5}$$

```r
vertiNoise <- function(dat, lt, vt= "AOC", hz= "LIBCOM"){
    vN <- 0
    for (i in unique(dat[, vt])){
        vN <- vN+ var(dat[dat[, vt]== i, lt])* mean(dat[, vt]== i)
    }
    c("Vertical Noise"= vN)
}
```

The **horizontal signal** terms is the variance of the expectation quality conditionally on horizontal GI dummies:

$$\mathbb{V}\{\,\mathbb{E}[q(X^*) \mid c]\,\} = \frac{1}{N}\sum\nolimits_{i=1}^{N}\left[\overline{q}_{c_i} - \overline{q}\right]^2 \tag{6}$$

```
horizSignal <- function(dat, lt, vt= "AOC", hz= "LIBCOM"){
    hS <- rep(0, nrow(dat))
    for (j in unique(dat[, hz])){
        hS[ dat[, hz]== j] <- mean(dat[dat[, hz]== j, lt])
    }
    c("Horizontal Signal"= var(hS))
}
```

The **horizontal residual** terms is the expectation of the conditional on vertical variance of the expectation quality conditionally on horizontal GI dummies:

$$\mathbb{E}\{\,\mathbb{V}[\mathbb{E}(q(X^*) \mid y, c) \mid y]\,\} = \sum\nolimits_{c=1}^{C}\left[\frac{N_c}{N}\sum_{i=1}^{N}(\overline{q}_{c_i} - \overline{q})^2\right] \tag{7}$$

```
horizResid <- function(dat, lt, vt= "AOC", hz= "LIBCOM"){
    sig <- rep(0, nrow(dat)) ; hR <- 0
    for (i in unique(dat[, vt])){
        for (j in unique(dat[, hz])){
            tmp <- dat[, vt]== i & dat[, hz]== j
            sig[ tmp] <- mean(dat[ tmp, lt])
        }
    }
    for (j in unique(dat[, hz])){
        hR <- hR+ var(sig[dat[, hz]== j])* mean(dat[, hz]== j)
    }
    c("Horizontal Residual"= hR)
}
```

The **horizontal Noise** terms is the expectation of the variance of the quality conditionally on horizontal GI dummies:

$$\mathbb{E}\{\,\mathbb{V}[q(X^*) \mid c]\,\} = \sum\nolimits_{c=1}^{C}\left[\frac{N_c}{N}\sum\nolimits_{i=1}^{N}\mathbb{1}[c_i = c]\cdot(\hat{q}_i - \overline{q}_{c_i})^2\right] \tag{8}$$

```
horizNoise <- function(dat, lt, vt= "AOC", hz= "LIBCOM"){
    hN <- 0
    for (j in unique(dat[, hz])){
        hN <- hN+ (var(dat[dat[, hz]== j, lt])* mean(dat[, hz]== j))
    }
    c("Horizontal Noise"= hN)
}
```