Spatial CODA

Supplemental material

Thi Huong An Nguyen, Anne Ruiz-Gazen, Christine Thomas-Agnan, Thibault Laurent

Contents

We provide the data and the **R** code used in the article "Spatial CODA" so that readers may reproduce all the figures, tables and statistics presented in the article with the **R** software.

If you use this code, please cite:

Nguyen T.H.A, Ruiz-Gazen, A., Thomas-Agnan C. and T. Laurent (2019). Spatial CODA. *WP*.

1 Prerequisites

Required packages:

```
install.packages(c("compositions", "mvnfast", "quantmod", "plot3D", "sp"))
```

Loading packages:

```
require("classInt") # discretize numeric variable
require("compositions") # compositional data
require("ggplot2") # ggplot functions
require("mvnfast") # multivariate Student distribution
require("quantmod")  # import financial data
require("plot3D") # plot distribution in 3D
require("RColorBrewer") # palette colors with R
require("rgdal") # import spatial data
require("sp") # spatial data
require("spdep") # spatial econometric modelling
```

Information about the current R session :

```
sessionInfo()
```

```
## R version 3.5.3 (2019-03-11)
## Platform: x86_64-pc-linux-gnu (64-bit)
```

```
## Running under: Ubuntu 16.04.6 LTS
##
## Matrix products: default
## BLAS: /usr/lib/openblas-base/libblas.so.3
## LAPACK: /usr/lib/libopenblasp-r0.2.18.so
##
## 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   base
##
## other attached packages:
##  [1] spdep_0.8-1         spData_0.3.0        Matrix_1.2-15
##  [4] rgdal_1.3-6         sp_1.3-1            RColorBrewer_1.1-2
##  [7] plot3D_1.1.1        quantmod_0.4-13     TTR_0.23-4
## [10] xts_0.11-2          zoo_1.8-4           mvnfast_0.2.5
## [13] ggplot2_3.1.0       compositions_1.40-2 bayesm_3.1-1
## [16] energy_1.7-5        robustbase_0.93-3   tensorA_0.36.1
## [19] classInt_0.3-1
##
## loaded via a namespace (and not attached):
##  [1] gtools_3.8.1      tidyselect_0.2.5  xfun_0.5
##  [4] purrr_0.2.5       splines_3.5.3     lattice_0.20-38
##  [7] expm_0.999-3      colorspace_1.4-0  htmltools_0.3.6
## [10] yaml_2.2.0        rlang_0.3.1       e1071_1.7-0
## [13] pillar_1.3.1      glue_1.3.0        withr_2.1.2
## [16] plyr_1.8.4        stringr_1.3.1     munsell_0.5.0
## [19] gtable_0.2.0      coda_0.19-2       evaluate_0.12
## [22] misc3d_0.8-4      knitr_1.21        curl_3.3
## [25] class_7.3-15      DEoptimR_1.0-8    Rcpp_1.0.0
## [28] scales_1.0.0      gdata_2.18.0      deldir_0.1-16
## [31] digest_0.6.18     gmodels_2.18.1    stringi_1.2.4
## [34] dplyr_0.8.0.1     grid_3.5.3        LearnBayes_2.15.1
## [37] tools_3.5.3       magrittr_1.5      lazyeval_0.2.1
## [40] tibble_2.0.1      crayon_1.3.4      pkgconfig_2.0.2
## [43] MASS_7.3-51.1     assertthat_0.2.0  rmarkdown_1.11
## [46] R6_2.3.0          boot_1.3-20       nlme_3.1-137
## [49] compiler_3.5.3
```

2 Simulation study

This section demonstrates how to obtain the results presented in the section 3 of the article. We first present our functions which can be adapted to another framework different from our simulation process.

2.1 Simulation of spatial multivariate Y

The function *simu_spatial_multi_y()* simulates a multivariate Y of the form $Y = Y\Gamma + WYR + X\beta + \epsilon$ where ϵ follows either a multivariate Gaussian (**method_simulate = "N"**), or the Independent multivariate Student (**method_simulate = "IT"**) distributions.

Input arguments are :

- **X**, the matrix of explanatory variables of size $n \times K$,
- **beta_true**, the β matrix of size $K \times L$:

$$\begin{pmatrix} \beta_{11} & \dots & \beta_{1L} \\ \beta_{21} & \dots & \beta_{2L} \\ \vdots & & \vdots \\ \beta_{K1} & \dots & \beta_{KL} \end{pmatrix}$$

- **method_simulate**, the method of simulation (a character among "N", "IT"),
- **Sigma**, the matrix of size $L \times L$,
- **GAMMA**, the matrix of size $L \times L$,
- **RHO**, the matrix of size $L \times L$,
- **W**, the matrix of size $n \times n$,
- **nu**, for Student distribution only.

The function returns a matrix of size $n \times L$. To load the function:

```
source("./R/simu_spatial_multi_y.R")
```

2.2 Examples

2.2.1 Preparation of the data

Import the Midi-Pyrénées communes boundaries into **R** which was used in Goulard et al. (2017):

```
mapMAP <- readOGR(dsn = "contours", layer = "ADTCAN_region")
```

We convert the type of the identification units into numeric values:

```
mapMAP@data$CODE <- as.numeric(as.character(mapMAP@data$CODE))
```

The number of observations equals to n:

```
n <- nrow(mapMAP)
```

We consider one spatial weight matrix W, based on the 10-nearest neighbours and row-normalized. W is relatively sparse (96.5% of null values).

```
coords <- coordinates(mapMAP)
W1.listw <- nb2listw(knn2nb(knearneigh(coords, 10)),
                     style = "W")
W_simu <- listw2mat(W1.listw)
```

2.2.2 Simulation of a multivariate SAR process

2.2.2.1 Example when $L = 2$

We plan to simulate a multivariate Y of size $L = 2$:

```
L_simu <- 2
```

```
L_simu <- 1
```

We simulate the explanatory variables:

```
set.seed(1234)
x1 <- rnorm(n, 15, 3)
x2 <- rbinom(n, 100, 0.45)
x3 <- log(round(runif(n, 1, n),0))
x_simu <- cbind(rep(1, n), x1, x2, x3)
p_simu <- ncol(x_simu)
```

We fiw some parameters of simulations:

$$\beta = \left(\begin{array}{cc} 15 & 20 \\ 2 & -3 \\ 1 & 2 \\ -1 & 3 \end{array} \right), \Sigma = \left(\begin{array}{cc} 2 & 0 \\ 0 & 3 \end{array} \right)$$

```
beta_true <- matrix(c(15, 2, 1, -1, 20, -3, -2, 3), byrow = F,
                    nrow = p_simu, ncol = L_simu)
```

```
Sigma <- matrix(c(2, 0, 0, 3),
                nrow = L_simu, ncol = L_simu)
```

Now, we vary some parameters.

2.2.3 Model simulation 1

- $$R = \left(\begin{array}{cc} 0.5 & 0 \\ 0 & 0.3 \end{array} \right), \Gamma = \left(\begin{array}{cc} 0 & 0 \\ 0 & 0 \end{array} \right)$$

```
RHO <- matrix(c(0.5, 0, 0, 0.3),
              nrow = L_simu, ncol = L_simu)
GAMMA <- matrix(c(0, 0, 0, 0),
                nrow = L_simu, ncol = L_simu)
```

We simulate the process:

```
set.seed(1)
y_N_mod_1 <- simu_spatial_multi_y(X = x_simu, beta_true = beta_true,
                            method_simulate = "N",
                            Sigma = Sigma,
                            GAMMA = GAMMA,
                            RHO = RHO,
                            W = W_simu)
mapMAP@data[, c("y_N_mod_1_1", "y_N_mod_1_2")] <- y_N_mod_1
```

2.2.4 Model simulation 2

- $$R = \begin{pmatrix} 0.5 & 0.2 \\ 0.15 & 0.3 \end{pmatrix}, \Gamma = \begin{pmatrix} 0 & 0 \\ 0 & 0 \end{pmatrix}$$

```
RHO <- matrix(c(0.5, 0.2, 0.15, 0.3),
              nrow = L_simu, ncol = L_simu)
GAMMA <- matrix(c(0, 0, 0, 0),
                nrow = L_simu, ncol = L_simu)
```

We simulate the process:

```
set.seed(1)
y_N_mod_2 <- simu_spatial_multi_y(X = x_simu, beta_true = beta_true,
                            method_simulate = "N",
                            Sigma = Sigma,
                            GAMMA = GAMMA,
                            RHO = RHO,
                            W = W_simu)
mapMAP@data[, c("y_N_mod_2_1", "y_N_mod_2_2")] <- y_N_mod_2
```

2.2.5 Model simulation 3

- $$R = \begin{pmatrix} 0.5 & 0.2 \\ 0.15 & 0.3 \end{pmatrix}, \Gamma = \begin{pmatrix} 0 & 0.1 \\ 0.2 & 0 \end{pmatrix}$$

```
RHO <- matrix(c(0.5, 0.2, 0.15, 0.3),
              nrow = L_simu, ncol = L_simu)
GAMMA <- matrix(c(0, 0.2, 0.4, 0),
                nrow = L_simu, ncol = L_simu)
```

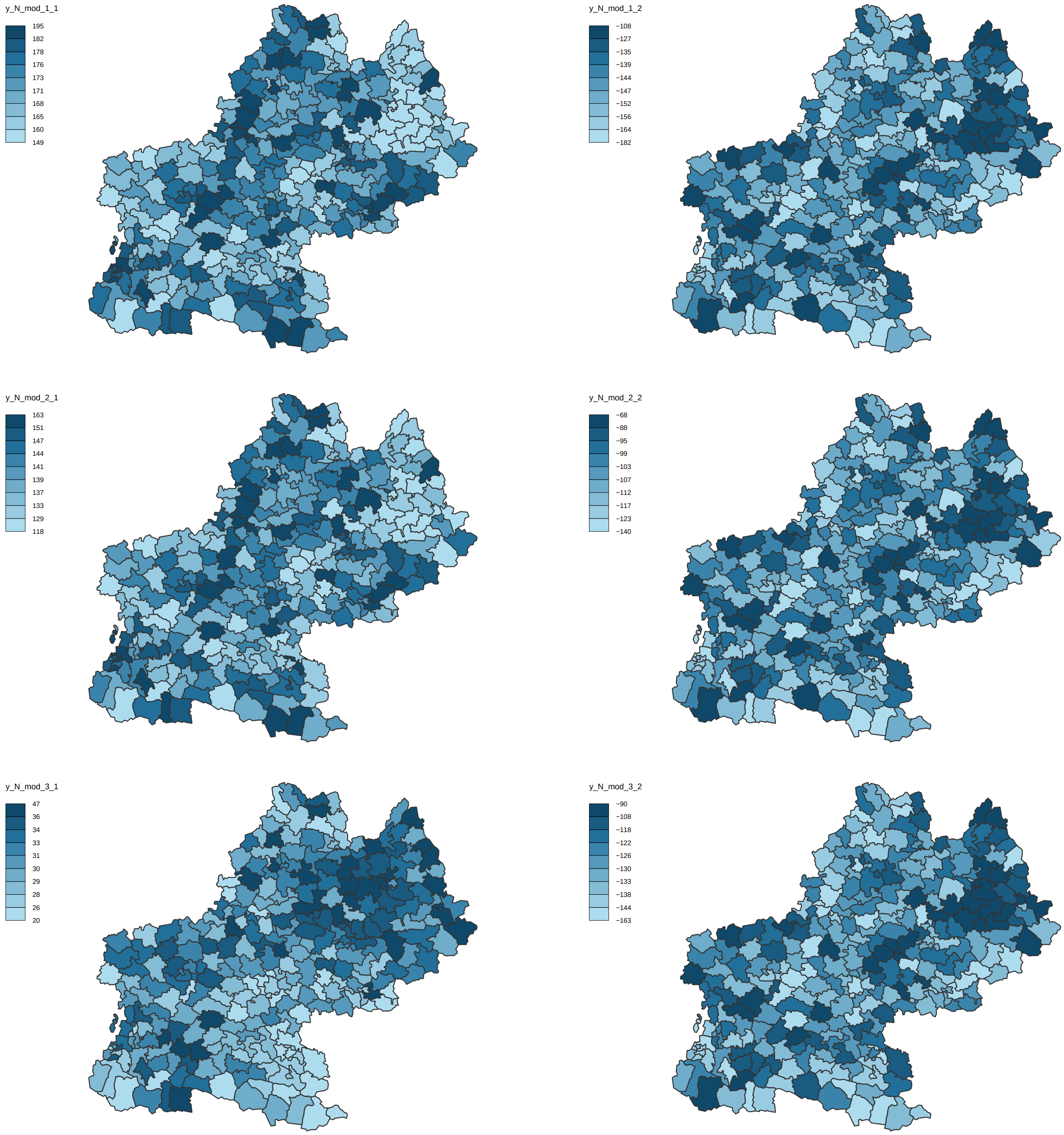
We simulate the process:

```
set.seed(1)
y_N_mod_3 <- simu_spatial_multi_y(X = x_simu, beta_true = beta_true,
                            method_simulate = "N",
                            Sigma = Sigma,
                            GAMMA = GAMMA,
                            RHO = RHO,
                            W = W_simu)
mapMAP@data[, c("y_N_mod_3_1", "y_N_mod_3_2")] <- y_N_mod_3
```

We plot the two component of Y on the map:

```
library("cartography")
op <- par(mfrow = c(3, 2), oma = c(0, 0, 0, 0), mar = c(0, 0, 1, 0))
choroLayer(spdf = mapMAP, var = "y_N_mod_1_1", legend.pos = "topleft",
           method = "quantile")
choroLayer(spdf = mapMAP, var = "y_N_mod_1_2", legend.pos = "topleft",
           method = "quantile")
choroLayer(spdf = mapMAP, var = "y_N_mod_2_1", legend.pos = "topleft",
           method = "quantile")
choroLayer(spdf = mapMAP, var = "y_N_mod_2_2", legend.pos = "topleft",
           method = "quantile")
choroLayer(spdf = mapMAP, var = "y_N_mod_3_1", legend.pos = "topleft",
```

```r
           method = "quantile")
choroLayer(spdf = mapMAP, var = "y_N_mod_3_2", legend.pos = "topleft",
           method = "quantile")
```

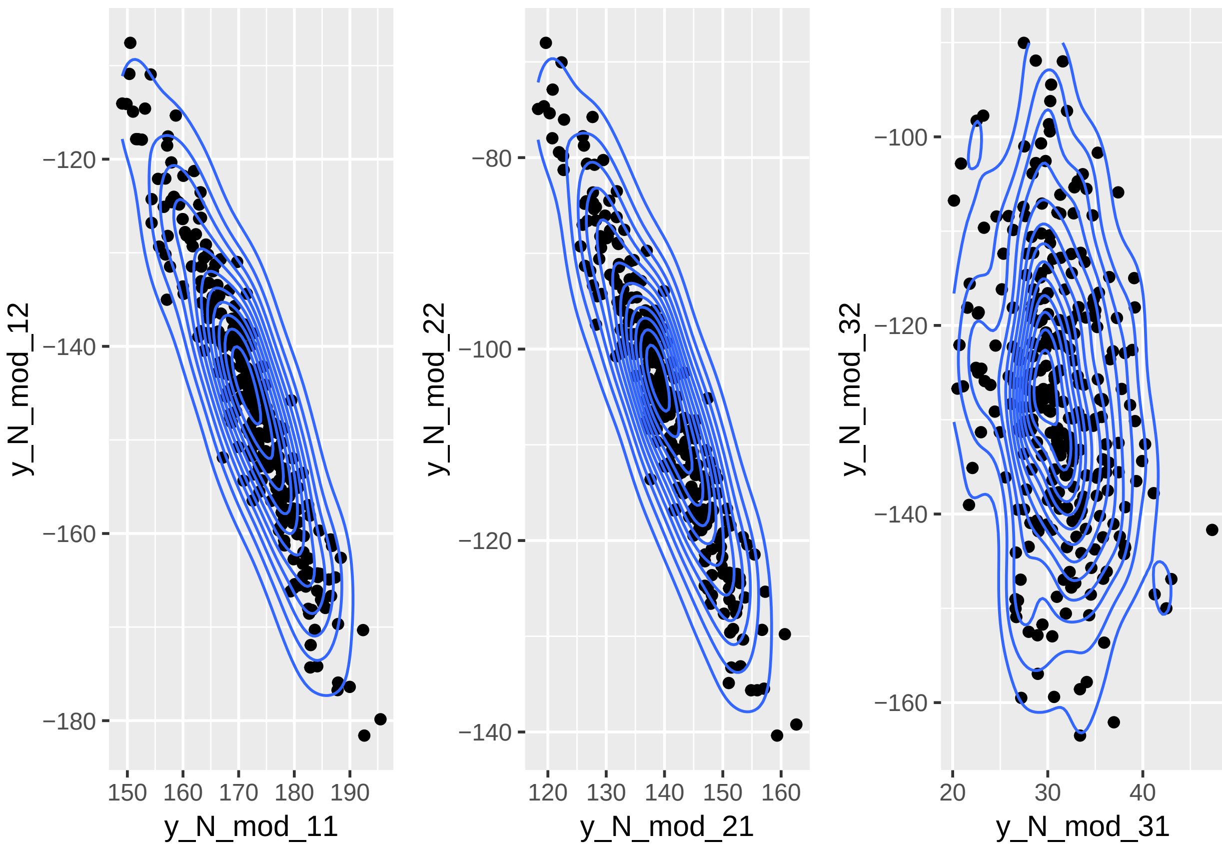



```r
par(op)
```

We also plot the joint distribution:

```r
y_N_df <- data.frame(y_N_mod_11 = y_N_mod_1[, 1],
                     y_N_mod_12 = y_N_mod_1[, 2],
                     y_N_mod_21 = y_N_mod_2[, 1],
                     y_N_mod_22 = y_N_mod_2[, 2],
```

```
                     y_N_mod_31 = y_N_mod_3[, 1],
                     y_N_mod_32 = y_N_mod_3[, 2],
                     x_simu)
plot1 <- ggplot(y_N_df, aes(x = y_N_mod_11, y = y_N_mod_12)) +
  geom_point() + geom_density_2d()
plot2 <- ggplot(y_N_df, aes(x = y_N_mod_21, y = y_N_mod_22)) +
  geom_point() + geom_density_2d()
plot3 <- ggplot(y_N_df, aes(x = y_N_mod_31, y = y_N_mod_32)) +
  geom_point() + geom_density_2d()
gridExtra::grid.arrange(plot1, plot2, plot3, nrow = 1, ncol = 3)
```



3 Estimation

3.1 Estimation of the parameters by a 2SLS method

The function *estimate_spatial_multi_N()* estimates the coefficients associated to the multivariate Gaussian SAR model. The algorithm is based on Kelejian and Prucha (1998).

Input arguments are :

- $\mathbf{Y}$, a matrix of size $n \times L$
- $\mathbf{X}$, a matrix of explanatory variables of size $n \times K$,
- $\mathbf{W}$, a spatial weight matrix of size $n \times n$,
- **GAMMA_esti**, a boolean which indicates if we estimate or nor the parameter associated to Γ.

The function returns a list with :

- the estimate of the β parameters
- the estimate of the Γ matrix
- the estimate of the R matrix
- the estimate of the Σ matrix

To load the function:

```
source("./R/estimate_spatial_multi_N.R")
source("./R/estimate_spatial_multi_gen_N.R")
```

3.1.1 Examples:

3.1.2 Model simulation 1

- $$\beta = \begin{pmatrix} 15 & 20 \\ 2 & -3 \\ 1 & 2 \\ -1 & 3 \end{pmatrix}, \Sigma = \begin{pmatrix} 2 & 0 \\ 0 & 3 \end{pmatrix}, R = \begin{pmatrix} 0.5 & 0 \\ 0 & 0.3 \end{pmatrix}, \Gamma = \begin{pmatrix} 0 & 0 \\ 0 & 0 \end{pmatrix}$$

```
(res_multi_N <- estimate_spatial_multi_gen_N(Y = y_N_mod_1, X = x_simu,
  W = W_simu,
  ind_beta = matrix(c(T, T, T, T, T, T, T, T, T, T, T, T), 4, 3),
  ind_RHO = matrix(c(T, F, F, T), 2, 2),
  ind_GAMMA = matrix(c(F, F, F, F), 2, 2)))
```

```
## $res_beta
##            [,1]      [,2]
## [1,] 10.6127684 14.583187
## [2,]  2.0483889 -2.979599
## [3,]  0.9841032 -1.991509
## [4,] -0.9333073  3.028197
##
## $GAMMA
##      [,1] [,2]
## [1,]    0    0
## [2,]    0    0
##
## $RHO
##           [,1]      [,2]
## [1,] 0.5237904 0.0000000
## [2,] 0.0000000 0.2680817
##
## $SIGMA
##             [,1]        [,2]
## [1,]  2.08942439 -0.06579664
## [2,] -0.06579664  2.78314431
```

Which is equivalent to :

```
(s2sls_lm_1 <- stsls(y_N_mod_11 ~ x1 + x2 + x3, data = y_N_df, listw = W1.listw))
```

```
##
```

```
## Call:
## stsls(formula = y_N_mod_11 ~ x1 + x2 + x3, data = y_N_df, listw = W1.listw)
##
## Coefficients:
##         Rho (Intercept)          x1          x2          x3
##   0.5237904  10.6127684   2.0483889   0.9841032  -0.9333073
```

```
(s2sls_lm_2 <- stsls(y_N_mod_12 ~ x1 + x2 + x3, data = y_N_df, listw = W1.listw))
```

```
##
## Call:
## stsls(formula = y_N_mod_12 ~ x1 + x2 + x3, data = y_N_df, listw = W1.listw)
##
## Coefficients:
##         Rho (Intercept)          x1          x2          x3
##   0.2680817  14.5831871  -2.9795985  -1.9915092   3.0281966
```

3.1.3 Model simulation 2

- $$\beta = \begin{pmatrix} 15 & 20 \\ 2 & -3 \\ 1 & 2 \\ -1 & 3 \end{pmatrix}, \Sigma = \begin{pmatrix} 2 & 0 \\ 0 & 3 \end{pmatrix}, R = \begin{pmatrix} 0.5 & 0.2 \\ 0.15 & 0.3 \end{pmatrix}, \Gamma = \begin{pmatrix} 0 & 0 \\ 0 & 0 \end{pmatrix}$$

```
(res_multi_N <- estimate_spatial_multi_gen_N(Y = y_N_mod_2, X = x_simu,
  W = W_simu,
  ind_beta = matrix(c(T, T, T, T, T, T, T, T, T, T, T, T), 4, 3),
  ind_RHO = matrix(c(T, T, T, T), 2, 2),
  ind_GAMMA = matrix(c(F, F, F, F), 2, 2)))
```

```
## $res_beta
##            [,1]      [,2]
## [1,]  9.2098179 19.562617
## [2,]  2.0474016 -2.971131
## [3,]  0.9855854 -1.992524
## [4,] -0.9337311  3.021586
##
## $GAMMA
##      [,1] [,2]
## [1,]    0    0
## [2,]    0    0
##
## $RHO
##           [,1]      [,2]
## [1,] 0.5513781 0.1666536
## [2,] 0.1404236 0.2245505
##
## $SIGMA
##             [,1]        [,2]
## [1,]  2.08466362 -0.05936801
## [2,] -0.05936801  2.78774972
```

3.1.4 Model simulation 3

- $$\beta = \begin{pmatrix} 15 & 20 \\ 2 & -3 \\ 1 & 2 \\ -1 & 3 \end{pmatrix}, \Sigma = \begin{pmatrix} 2 & 0 \\ 0 & 3 \end{pmatrix}, R = \begin{pmatrix} 0.5 & 0.2 \\ 0.15 & 0.3 \end{pmatrix}, \Gamma = \begin{pmatrix} 0 & 0.1 \\ 0.2 & 0 \end{pmatrix}$$

```
(res_multi_N <- estimate_spatial_multi_gen_N(Y = y_N_mod_3, X = x_simu,
  W = W_simu,
  ind_beta = matrix(c(T, T, T, T, T, T, T, T, T, T, T, T), 4, 3),
  ind_RHO = matrix(c(T, T, T, T), 2, 2),
  ind_GAMMA = matrix(c(F, T, T, F), 2, 2)))
```

```
## $res_beta
##            [,1]      [,2]
## [1,]  0.7177273  4.941842
## [2,]  3.4154366 -3.594369
## [3,]  1.9401087 -2.130969
## [4,] -2.4477609  2.839205
##
## $GAMMA
##           [,1]     [,2]
## [1,] 0.0000000 0.887644
## [2,] 0.8553006 0.000000
##
## $RHO
##            [,1]        [,2]
## [1,]  0.3868011 -0.02928875
## [2,] -0.2673855  0.10518122
##
## $SIGMA
##           [,1]      [,2]
## [1,]  2.517224 -2.457917
## [2,] -2.457917  2.560851
```

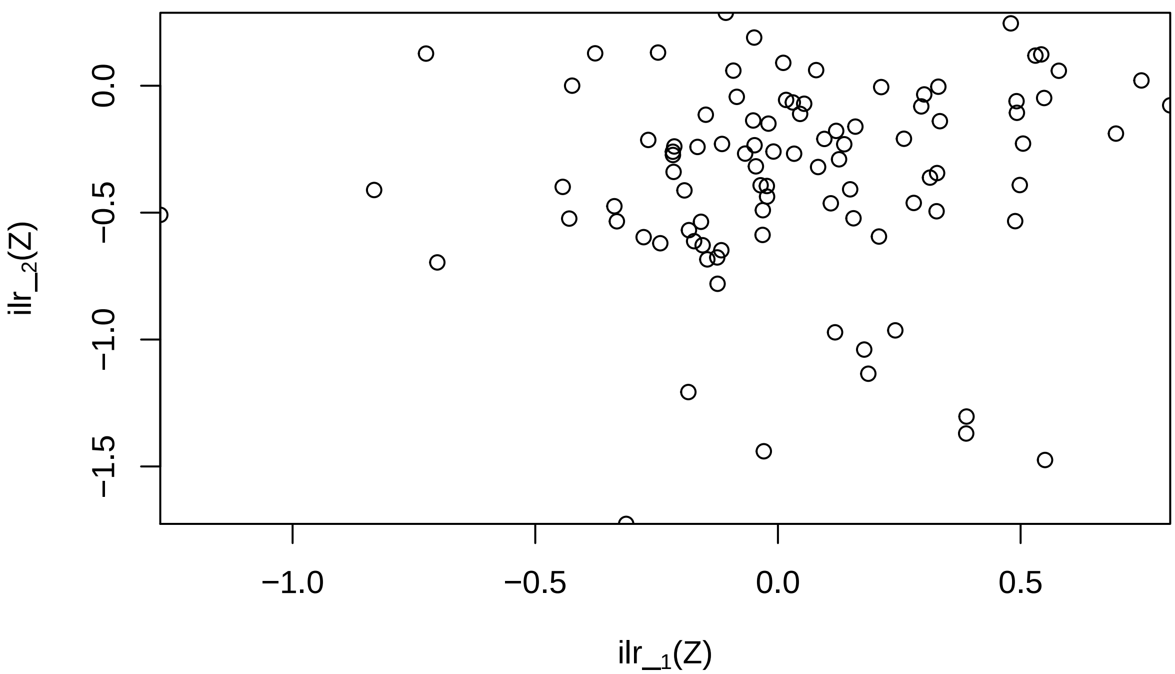
4 Application to the real data

We first load the data:

```
source("R/preparation_base_ilr.R")
```

Then, we plot the data.

```
Ye <- as(y_ilr, "matrix")
plot(Ye[,1], Ye[,2], xlab = expression(paste("ilr","_"[1],'(Z)')),
     ylab = expression(paste("ilr","_"[2],'(Z)')),
     xaxs = "i", yaxs = "i")
```



We prepare the explanatory variables:

```
Xe <- as(cbind(1, x2_df[, c("age3_ilr1", "age3_ilr2",
                             "unemp_rate", "income_rate")]),
         "matrix")
```

4.1 Multivariate Gaussian model

We estimate first a multivariate gaussian model by using the *lm()* function.

```
res_N <- lm(Ye ~ Xe - 1)
```

Then, we look the spatial distribution of the residuals. For this, we first compute a spatial weight matrix based on the 4-nearest neighbours.

```
coords_fr <- coordinates(dep.2015.spdf)
W_listw <- nb2listw(knn2nb(knearneigh(coords_fr, 4)),
                    style = "W")
W_dep <- listw2mat(W_listw)
```

We test the spatial autocorrelation in the residuals component by component:

```
moran.mc(residuals(res_N)[, 1], listw = W_listw, nsim = 1000)
```

```
##
##  Monte-Carlo simulation of Moran I
##
## data:  residuals(res_N)[, 1]
## weights: W_listw
## number of simulations + 1: 1001
##
## statistic = 0.25723, observed rank = 1001, p-value = 0.000999
## alternative hypothesis: greater
```

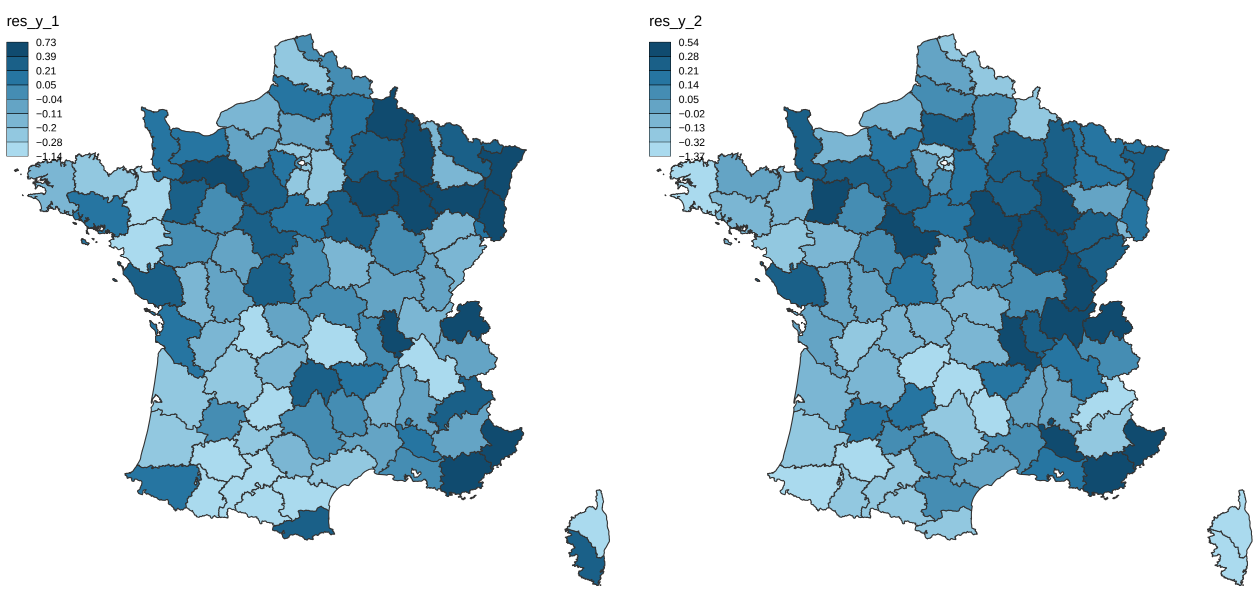
```r
moran.mc(residuals(res_N)[, 2], listw = W_listw, nsim = 1000)
```

```
##
##  Monte-Carlo simulation of Moran I
##
## data:  residuals(res_N)[, 2]
## weights: W_listw
## number of simulations + 1: 1001
##
## statistic = 0.24837, observed rank = 1001, p-value = 0.000999
## alternative hypothesis: greater
```

We plot the residuals:

```r
dep.2015.spdf@data[, c("res_y_1", "res_y_2")] <- residuals(res_N)
```

```r
library("cartography")
op <- par(mfrow = c(1, 2), oma = c(0, 0, 0, 0), mar = c(0, 0, 1, 0))
choroLayer(spdf = dep.2015.spdf, var = "res_y_1", legend.pos = "topleft",
           method = "quantile", legend.values.rnd = 2)
choroLayer(spdf = dep.2015.spdf, var = "res_y_2", legend.pos = "topleft",
           method = "quantile", legend.values.rnd = 2)
```




```r
par(op)
```

4.2 Multivariate Gaussian SAR model

We estimate a multivariate gaussian SAR model by using the *lm()* function.

```r
(res_sar_N <- estimate_spatial_multi_N(Ye, Xe, W_dep, GAMMA_esti = F))
```

```
## $res_beta
##              [,1]       [,2]
## [1,] -0.1969978 -3.1077680
## [2,]  0.2719789  0.8525314
## [3,]  0.2965863 -0.2870640
```

```
## [4,] -4.5082459 13.2323330
## [5,]  1.7306086  2.0687909
##
## $GAMMA
##      [,1] [,2]
## [1,]    0    0
## [2,]    0    0
##
## $RHO
##           [,1]      [,2]
## [1,] 0.8895999 0.5202213
## [2,] 0.5246872 0.6698683
##
## $SIGMA
##            [,1]       [,2]
## [1,] 0.08637361 0.01996587
## [2,] 0.01996587 0.08599642
```

5 References

- Goulard M., Laurent T. and Thomas-Agnan C. (2017). About predictions in spatial autoregressive models: optimal and almost optimal strategies, Spatial Economic Analysis, 12:2-3, 304-325, DOI: 10.1080/17421772.2017.1300679
- Nguyen T.H.A, Ruiz-Gazen, A., Thomas-Agnan C. and T. Laurent (2019). Multivariate Student versus Multivariate Gaussian Regression Models with Application to Finance. Journal of Risk and Financial Management, 12(1), 28.