A simultaneous spatial autoregressive model for compositional data (Supplementary material)

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

Last update: 2020-10-19

Contents

We provide the data and the **R** code used in the article "A simultaneous spatial autoregressive model for compositional data" so that readers may reproduce all the figures, tables and statistics presented in the article with the **R** software. The pdf version of the supplementary variable is available here.

If you use this code, please cite:

Nguyen T.H.A, Thomas-Agnan C., Laurent T. and A. Ruiz-Gazen (2020). A simultaneous spatial autoregressive model for compositional data. *Spatial Economic Analysis* (to appear).

1 Prerequisites

Required packages:

```
install.packages(c("classInt", "compositions", "dplyr", "ggplot2", "Matrix",
                   "isoband", "rgdal", "sp", "sf", "spdep"))
```

Loading packages:

```
require("classInt") # discretize numeric variable
require("compositions") # compositional data
require("dplyr") # dplyr data
require("ggplot2") # ggplot functions
```

```r
require("isoband") # Joint distribution
require("Matrix") # sparse matrix
require("rgdal") # import spatial data
require("sp") # spatial data
require("sf") # spatial data V2
require("spdep") # spatial econometric modelling
```

Information about the current R session :

```r
sessionInfo()
```

```
## R version 4.0.3 (2020-10-10)
## Platform: x86_64-pc-linux-gnu (64-bit)
## Running under: Ubuntu 18.04.5 LTS
##
## Matrix products: default
## BLAS:   /usr/lib/x86_64-linux-gnu/openblas/libblas.so.3
## LAPACK: /usr/lib/x86_64-linux-gnu/libopenblasp-r0.2.20.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_1.1-5        spData_0.3.8       sf_0.9-6           rgdal_1.5-17
##  [5] sp_1.4-4           Matrix_1.2-18      isoband_0.2.2      ggplot2_3.3.2
##  [9] dplyr_1.0.2        compositions_2.0-0 classInt_0.4-3
##
## loaded via a namespace (and not attached):
##  [1] gtools_3.8.2       tidyselect_1.1.0   xfun_0.18          purrr_0.3.4
##  [5] splines_4.0.3      lattice_0.20-41    expm_0.999-5       colorspace_1.4-1
##  [9] vctrs_0.3.4        generics_0.0.2     htmltools_0.5.0    yaml_2.2.1
## [13] rlang_0.4.8        e1071_1.7-3        pillar_1.4.6       glue_1.4.2
## [17] withr_2.3.0        DBI_1.1.0          lifecycle_0.2.0    robustbase_0.93-6
## [21] stringr_1.4.0      munsell_0.5.0      gtable_0.3.0       raster_3.3-13
## [25] codetools_0.2-16   coda_0.19-4        evaluate_0.14      knitr_1.30
## [29] class_7.3-17       DEoptimR_1.0-8     Rcpp_1.0.5         KernSmooth_2.23-17
## [33] scales_1.1.1       gdata_2.18.0       deldir_0.1-29      tensorA_0.36.1
## [37] digest_0.6.25      gmodels_2.18.1     stringi_1.5.3      grid_4.0.3
## [41] LearnBayes_2.15.1  tools_4.0.3        magrittr_1.5       tibble_3.0.3
## [45] crayon_1.3.4       pkgconfig_2.0.3    MASS_7.3-53        ellipsis_0.3.1
## [49] bayesm_3.1-4       rmarkdown_2.4      R6_2.4.1           boot_1.3-25
## [53] nlme_3.1-149       units_0.6-7        compiler_4.0.3
```

2 Simulation study

This section demonstrates how to obtain the results presented in Section 4 of the article. We first present our functions which can be adapted to situations different from our simulation process.

2.1 Simulation of spatial multivariate Y^*

The function *simu_spatial_multi_y()* simulates a $n \times M$ matrix Y^* of the form $Y^* = Y^*\Gamma^* + X\beta^* + WY^*R^* + \epsilon^*$ where ϵ^* follows either a multivariate Gaussian $N(0_M, \Sigma^*)$ (**method_simulate = "N"**), or the Independent multivariate Student (**method_simulate = "IT"**) distributions. It can be found here.

```
simu_spatial_multi_y(X, beta_true, method_simulate = "N",
   Sigma, RHO, W, GAMMA = NULL, nu = NULL)
```

Input arguments are:

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

$$\begin{pmatrix} \beta_{0,1} & \dots & \beta_{0,L} \\ \beta_{1,1} & \dots & \beta_{1,L} \\ \vdots & & \vdots \\ \beta_{K-1,1} & \dots & \beta_{K-1,M} \end{pmatrix}$$

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

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

```
source(url("http://www.thibault.laurent.free.fr/code/spatial_coda/R/simu_spatial_multi_y.R"))
```

2.2 Examples

2.2.1 Data preparation

We first import the Midi-Pyrénées communes boundaries into **R** (data used in Goulard et al. (2017)):

```
URL <- "http://www.thibault.laurent.free.fr/code/GLT/"
fil <- "contours.zip"
if (!file.exists(fil)) download.file(paste(URL, fil, sep = ""), fil)
unzip(fil)
```

```
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 is equal 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 $M = 2$

In this article, we simulate a multivariate Y^* of size $M = 2$:

```
M_simu <- 2
```

We simulate the explanatory variables:

```
set.seed(1234)
x1 <- rnorm(n, 0, 9)
x2 <- rnorm(n, 0, 6)
x3 <- rnorm(n, 0, 9)
x_simu <- cbind(rep(1, n), x1, x2, x3)
p_simu <- ncol(x_simu)
```

We set the β^* parameters:

$$\beta^* = \begin{pmatrix} 3 & -3 \\ 2 & -3 \\ 1 & -2 \\ -1 & 3 \end{pmatrix}$$

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

We define different covariance and spatial autocorrelation matrices:

$$\Sigma_{\bar{d}}^* = \begin{pmatrix} 0.7 & 0.09 \\ 0.09 & 0.1 \end{pmatrix}$$

$$\Sigma_d^* = \begin{pmatrix} 0.7 & 0 \\ 0 & 0.1 \end{pmatrix}$$

$$R_{\bar{d}}^* = \begin{pmatrix} 0.5 & 0.6 \\ 0.4 & 0.3 \end{pmatrix}$$

$$R_d^* = \begin{pmatrix} 0.5 & 0 \\ 0 & 0.3 \end{pmatrix}$$

```
sigma_bar_d <- matrix(c(0.7, 0.09, 0.09, 0.1),
              nrow = M_simu, ncol = M_simu)
sigma_d <- matrix(c(0.7, 0, 0, 0.1),
              nrow = M_simu, ncol = M_simu)
```

```
RHO_bar_d <- matrix(c(0.5, 0.4, 0.6, 0.3),
                   nrow = M_simu, ncol = M_simu)
RHO_d <- matrix(c(0.5, 0, 0, 0.3),
                   nrow = M_simu, ncol = M_simu)
```

Now, we run simulations with the different parameters choices.

2.2.3 Model simulation 1

- $\Sigma_{\bar{d}}^*$ and $R_{\bar{d}}^*$

We simulate the process:

```
set.seed(1)
y_N_mod_1 <- simu_spatial_multi_y(X = x_simu, beta_true = beta_true,
                                  Sigma = sigma_bar_d,
                                  RHO = RHO_bar_d,
                                  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

- $\Sigma_{\bar{d}}^*$ and R_d^*

We simulate the process:

```
set.seed(1)
y_N_mod_2 <- simu_spatial_multi_y(X = x_simu, beta_true = beta_true,
                                  Sigma = sigma_bar_d,
                                  RHO = RHO_d,
                                  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

- Σ_d^* and $R_{\bar{d}}^*$

We simulate the process:

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

2.2.6 Model simulation 4

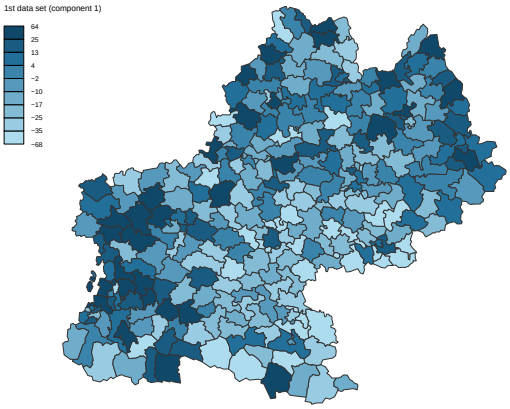
- Σ_d^* and R_d^*

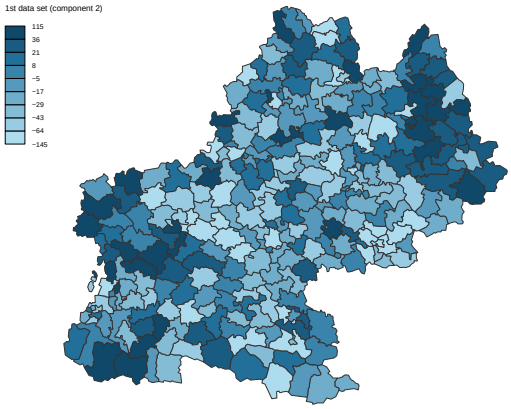
We simulate the process:

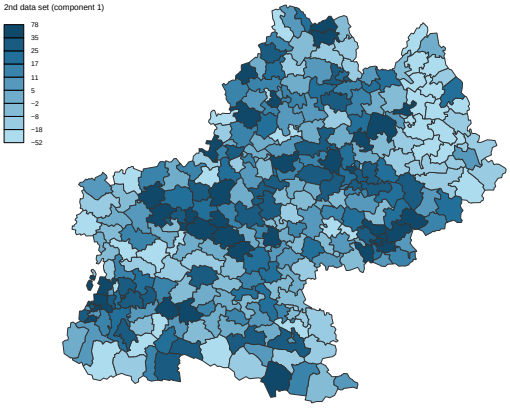
```
set.seed(1)
y_N_mod_4 <- simu_spatial_multi_y(X = x_simu, beta_true = beta_true,
                                  Sigma = sigma_d,
                                  RHO = RHO_d,
                                  W = W_simu)
mapMAP@data[, c("y_N_mod_4_1", "y_N_mod_4_2")] <- y_N_mod_4
```

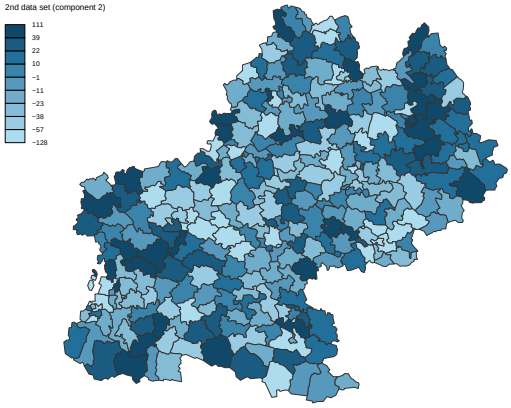
We plot the two coordinates of Y^* on the map:

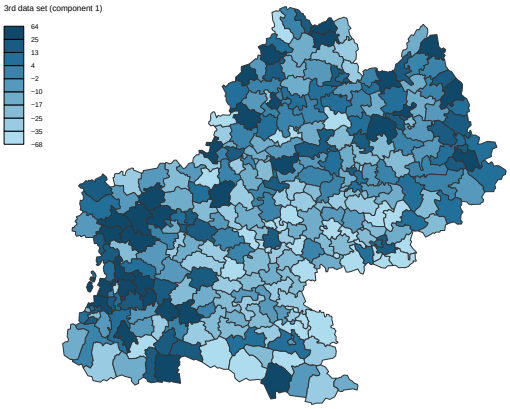
```
library("cartography")
op <- par(mfrow = c(4, 2), oma = c(0, 0, 0, 0), mar = c(0, 0, 1, 0))
choroLayer(spdf = mapMAP, var = "y_N_mod_1_1", legend.pos = "topleft",
           method = "quantile", legend.title.txt = "1st data set (component 1)")
choroLayer(spdf = mapMAP, var = "y_N_mod_1_2", legend.pos = "topleft",
           method = "quantile", legend.title.txt = "1st data set (component 2)")
choroLayer(spdf = mapMAP, var = "y_N_mod_2_1", legend.pos = "topleft",
           method = "quantile", legend.title.txt = "2nd data set (component 1)")
choroLayer(spdf = mapMAP, var = "y_N_mod_2_2", legend.pos = "topleft",
           method = "quantile", legend.title.txt = "2nd data set (component 2)")
choroLayer(spdf = mapMAP, var = "y_N_mod_3_1", legend.pos = "topleft",
           method = "quantile", legend.title.txt = "3rd data set (component 1)")
choroLayer(spdf = mapMAP, var = "y_N_mod_3_2", legend.pos = "topleft",
           method = "quantile", legend.title.txt = "3rd data set (component 2)")
choroLayer(spdf = mapMAP, var = "y_N_mod_4_1", legend.pos = "topleft",
           method = "quantile", legend.title.txt = "4th data set (component 1)")
choroLayer(spdf = mapMAP, var = "y_N_mod_4_2", legend.pos = "topleft",
           method = "quantile", legend.title.txt = "4th data set (component 2)")
```

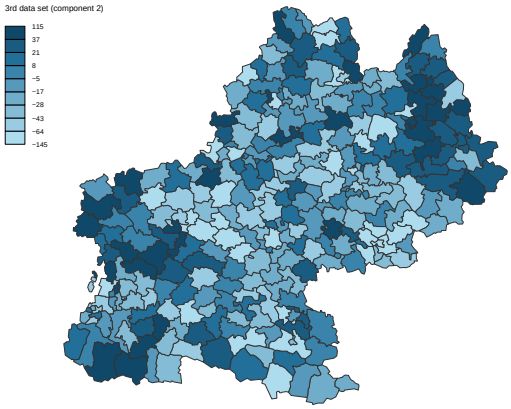

1st data set (component 1)
64
25
13
4
−2
−10
−17
−25
−35
−68

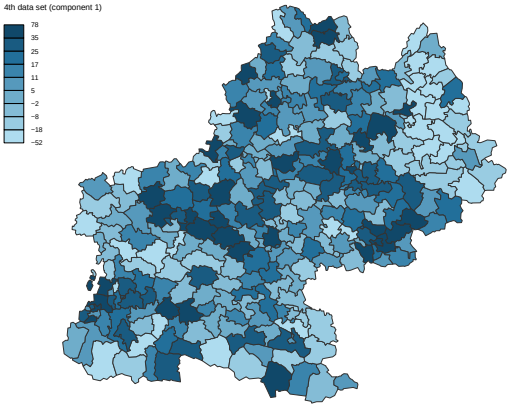

1st data set (component 2)
115
36
21
8
−5
−17
−29
−43
−64
−145

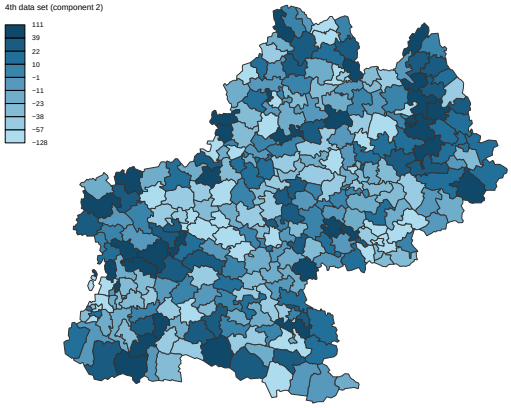

2nd data set (component 1)
78
35
25
17
11
5
−2
−8
−18
−52


2nd data set (component 2)
111
39
22
10
−1
−11
−23
−38
−57
−128


3rd data set (component 1)
64
25
13
4
−2
−10
−17
−25
−35
−68


3rd data set (component 2)
115
37
21
8
−5
−17
−28
−43
−64
−145

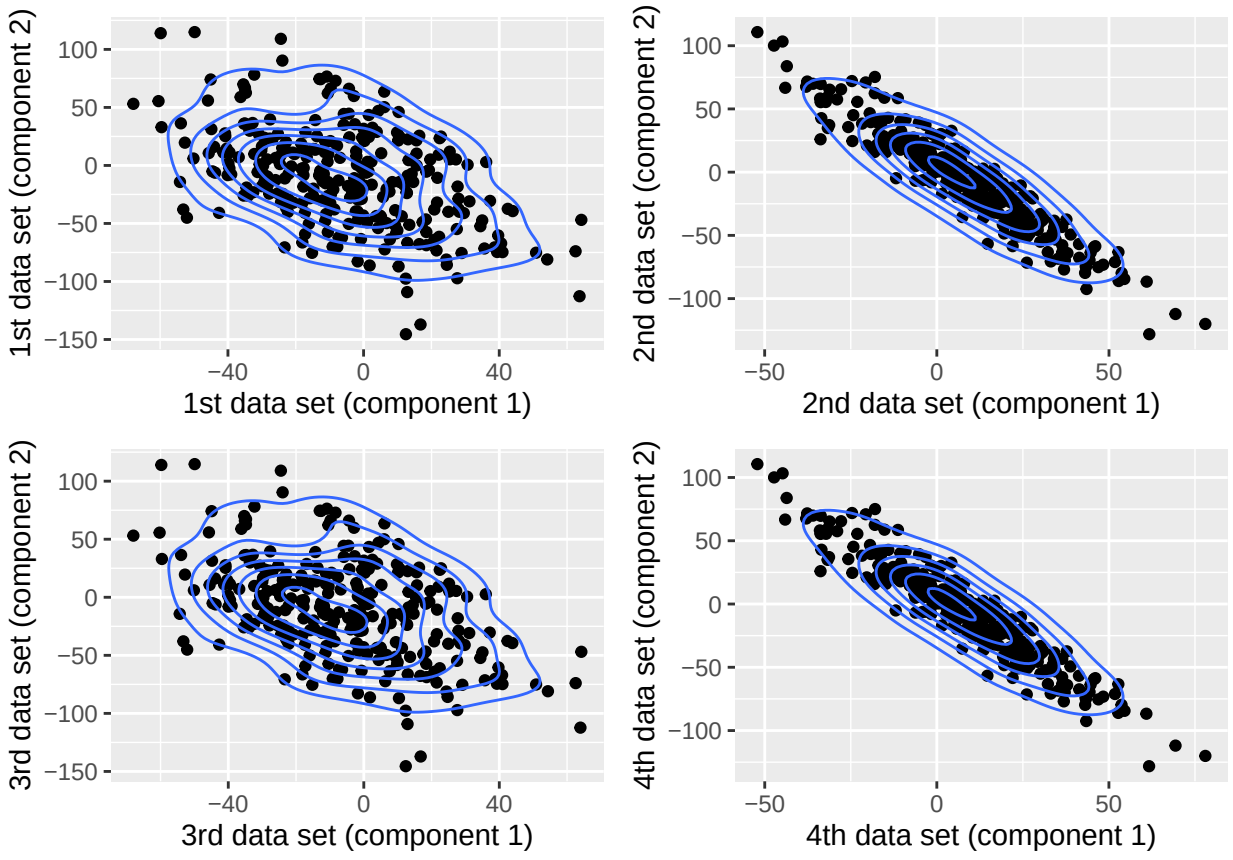

4th data set (component 1)
78
35
25
17
11
5
−2
−8
−18
−52


4th data set (component 2)
111
39
22
10
−1
−11
−23
−38
−57
−128

```
par(op)
```

We also plot the joint distribution:

```
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],
                     y_N_mod_41 = y_N_mod_4[, 1],
                     y_N_mod_42 = y_N_mod_4[, 2],
                     x_simu)
plot1 <- ggplot(y_N_df, aes(x = y_N_mod_11, y = y_N_mod_12)) +
  geom_point() + geom_density_2d() +
  labs(x = "1st data set (component 1)", y = "1st data set (component 2)")
plot2 <- ggplot(y_N_df, aes(x = y_N_mod_21, y = y_N_mod_22)) +
  geom_point() + geom_density_2d() +
  labs(x = "2nd data set (component 1)", y = "2nd data set (component 2)")
plot3 <- ggplot(y_N_df, aes(x = y_N_mod_31, y = y_N_mod_32)) +
  geom_point() + geom_density_2d()  +
  labs(x = "3rd data set (component 1)", y = "3rd data set (component 2)")
plot4 <- ggplot(y_N_df, aes(x = y_N_mod_41, y = y_N_mod_42)) +
  geom_point() + geom_density_2d()  +
  labs(x = "4th data set (component 1)", y = "4th data set (component 2)")
gridExtra::grid.arrange(plot1, plot2, plot3, plot4, nrow = 2, ncol = 2)
```



3 Multivariate LAG regression model estimation

The function *estimate_spatial_multi_gen_N()* estimates the parameters associated to the multivariate Gaussian SAR model. The algorithm is based on Kelejian and Prucha (1998). The codes can be found here.

```
estimate_spatial_multi_gen_N (Y, X, W, method = "s2sls",
     ind_beta = matrix(T, ncol(X), ncol(Y)),
     ind_RHO = matrix(T, ncol(Y), ncol(Y)),
     ind_GAMMA = matrix(F, ncol(Y), ncol(Y)))
```

Input arguments are :

- **Y**, a matrix of size $n \times M$,
- **X**, a matrix of explanatory variables of size $n \times K$,
- **W**, a spatial weight matrix of size $n \times n$,
- **ind_beta**, the boolean matrix of size $K \times M$ which indicates coordinate by coordinate the variables to be included in the models,
- **ind_RHO**, the boolean matrix of size $M \times M$, which indicates coordinate by coordinate the spatial autocorrelation parameters to be included in the model,
- **ind_GAMMA**, the boolean matrix of size $M \times M$, which indicates coordinate by coordinate the endogenous variables to be included in the model. By default, there are none.

The function returns a list with:

- the estimate of the β^* parameters
- the estimate of the R^* matrix
- the estimate of the Γ^* matrix
- the estimate of the Σ^* matrix
- the standard deviation of the $\hat{\beta}^*$ parameters
- the standard deviation of the $\hat{R}^*$ matrix
- the standard deviation of the $\hat{\Gamma}^*$ matrix

To load the function:

```
source(url("http://www.thibault.laurent.free.fr/code/spatial_coda/R/estimate_spatial_multi_gen_N.R"))
```

3.1 Examples:

3.1.1 Model simulation 1

```
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, T, T, T), 2, 2),
  ind_GAMMA = matrix(c(F, F, F, F), 2, 2))
```

```
## $est_beta
##            [,1]      [,2]
## [1,]  3.0073831 -2.999979
## [2,]  1.9957032 -2.996228
```

```
## [3,]  0.9920166 -1.997774
## [4,] -0.9922775  3.005340
##
## $est_RHO
##           [,1]      [,2]
## [1,] 0.4987330 0.6032345
## [2,] 0.4012719 0.2991003
##
## $est_GAMMA
##      [,1] [,2]
## [1,]    0    0
## [2,]    0    0
##
## $est_SIGMA
##            [,1]       [,2]
## [1,] 0.65536417 0.08808715
## [2,] 0.08808715 0.10437862
```

3.1.2 Model simulation 2

```
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, F, F, T), 2, 2),
  ind_GAMMA = matrix(c(F, F, F, F), 2, 2))
```

```
## $est_beta
##            [,1]      [,2]
## [1,]  3.0139088 -3.003506
## [2,]  1.9963291 -2.996475
## [3,]  0.9921161 -1.997923
## [4,] -0.9920515  3.005698
##
## $est_RHO
##           [,1]      [,2]
## [1,] 0.4958463 0.0000000
## [2,] 0.0000000 0.2991613
##
## $est_GAMMA
##      [,1] [,2]
## [1,]    0    0
## [2,]    0    0
##
## $est_SIGMA
##            [,1]       [,2]
## [1,] 0.65667365 0.08799486
## [2,] 0.08799486 0.10430220
```

3.1.3 Model simulation 3

```
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, F, F, F), 2, 2))
```

```
## $est_beta
##            [,1]      [,2]
## [1,]  3.0072329 -3.001141
## [2,]  1.9963298 -2.995324
## [3,]  0.9924955 -1.996391
## [4,] -0.9916633  3.004466
##
## $est_RHO
##           [,1]      [,2]
## [1,] 0.4989441 0.6030423
## [2,] 0.4015495 0.2985400
##
## $est_GAMMA
##      [,1] [,2]
## [1,]    0    0
## [2,]    0    0
##
## $est_SIGMA
##            [,1]       [,2]
## [1,] 0.65669411 0.00554497
## [2,] 0.00554497 0.10448315
```

3.1.4 Model simulation 4

In the case of model simulation 4 :

```
(res_multi_N <- estimate_spatial_multi_gen_N(Y = y_N_mod_4, X = x_simu,
  W = W_simu,
  ind_beta = matrix(c(T, T, T, T, T, T, T, T, T, T, T, T), 4, 2),
  ind_RHO = matrix(c(T, F, F, T), 2, 2),
  ind_GAMMA = matrix(c(F, F, F, F), 2, 2)))
```

```
## $est_beta
##            [,1]      [,2]
## [1,]  3.0122703 -3.004682
## [2,]  1.9968877 -2.995638
## [3,]  0.9925667 -1.996556
## [4,] -0.9913932  3.004855
##
## $est_RHO
##           [,1]      [,2]
## [1,] 0.4961402 0.0000000
## [2,] 0.0000000 0.2985633
##
## $est_GAMMA
##      [,1] [,2]
## [1,]    0    0
## [2,]    0    0
##
## $est_SIGMA
```

```
##             [,1]        [,2]
## [1,] 0.657811749 0.005389536
## [2,] 0.005389536 0.104489734
```

we find exactly the same results that if we were using the *stsls()* function coordinate by coordinate:

```
s2sls_lm_1 <- spatialreg::stsls(y_N_mod_41 ~ x1 + x2 + x3, data = y_N_df, listw = W1.listw)
summary(s2sls_lm_1)
```

```
##
## Call:spatialreg::stsls(formula = y_N_mod_41 ~ x1 + x2 + x3, data = y_N_df,
##     listw = W1.listw)
##
## Residuals:
##       Min        1Q    Median        3Q       Max
## -2.191742 -0.515151  0.016653  0.520885  2.454429
##
## Coefficients:
##               Estimate Std. Error  t value  Pr(>|t|)
## Rho          0.4961402  0.0044199  112.252 < 2.2e-16
## (Intercept)  3.0122703  0.0617277   48.799 < 2.2e-16
## x1           1.9968877  0.0054378  367.220 < 2.2e-16
## x2           0.9925667  0.0077145  128.663 < 2.2e-16
## x3          -0.9913932  0.0057700 -171.819 < 2.2e-16
##
## Residual variance (sigma squared): 0.66018, (sigma: 0.81251)
```

```
s2sls_lm_2 <- spatialreg::stsls(y_N_mod_42 ~ x1 + x2 + x3, data = y_N_df, listw = W1.listw)
summary(s2sls_lm_2)
```

```
##
## Call:spatialreg::stsls(formula = y_N_mod_42 ~ x1 + x2 + x3, data = y_N_df,
##     listw = W1.listw)
##
## Residuals:
##       Min        1Q    Median        3Q       Max
## -1.209143 -0.195419  0.018163  0.203198  0.942949
##
## Coefficients:
##               Estimate Std. Error  t value  Pr(>|t|)
## Rho          0.2985633  0.0012171   245.31 < 2.2e-16
## (Intercept) -3.0046819  0.0216517  -138.77 < 2.2e-16
## x1          -2.9956382  0.0021498 -1393.43 < 2.2e-16
## x2          -1.9965556  0.0030744  -649.41 < 2.2e-16
## x3           3.0048546  0.0023050  1303.64 < 2.2e-16
##
## Residual variance (sigma squared): 0.10487, (sigma: 0.32383)
```

4 Simulation process

To obtain Table 1 of the article, we repeat $N = 1000$ times the process of simulation/estimation and we compute the RRMSE. The codes used for this step are available here.

The RRMSE:

col_1	col_2	col_3	col_4	col_5	col_6	col_7
1.92	1.91	2.13	2.13	2.15	2.15	2.15
0.28	0.27	0.29	0.29	0.29	0.29	0.29
0.82	0.81	0.77	0.77	0.83	0.83	0.83
0.62	0.63	0.59	0.59	0.60	0.60	0.60
0.75	0.70	0.72	0.72	0.72	0.72	0.72
0.07	0.07	0.07	0.07	0.07	0.07	0.07
0.15	0.15	0.15	0.15	0.15	0.15	0.15
0.08	0.08	0.08	0.08	0.08	0.08	0.08
1.04	1.07	0.90	0.89	0.92	0.92	0.92
0.51	0.50	NaN	NaN	NaN	NaN	NaN
0.52	0.52	NaN	NaN	NaN	NaN	NaN
0.41	0.39	0.39	0.38	0.40	0.40	0.40
8.29	8.45	8.39	8.39	8.80	8.80	8.87
18.01	Inf	18.15	18.16	Inf	Inf	NaN
18.01	Inf	18.15	18.16	Inf	Inf	NaN
8.31	8.54	8.21	8.21	8.49	8.49	8.54

We also present the results for the biais which were not presented in the article :

col_1	col_2	col_3	col_4	col_5	col_6	col_7
0.00	0	0.00	0.00	0	0	0.00
0.00	0	0.00	0.00	0	0	0.00
0.00	0	0.00	0.00	0	0	0.00
0.00	0	0.00	0.00	0	0	0.00
0.00	0	0.00	0.00	0	0	0.00
0.00	0	0.00	0.00	0	0	0.00
0.00	0	0.00	0.00	0	0	0.00
0.00	0	0.00	0.00	0	0	0.00
0.00	0	0.00	0.00	0	0	0.00
0.00	0	0.00	0.00	0	0	0.00
0.00	0	0.00	0.00	0	0	0.00
0.00	0	0.00	0.00	0	0	0.00
-0.01	0	-0.01	-0.01	0	0	-0.01
0.00	0	0.00	0.00	0	0	0.00
0.00	0	0.00	0.00	0	0	0.00
0.00	0	0.00	0.00	0	0	0.00

4.1 Another simulation process

4.1.1 R^* is null

We have also computed the RRMSE when the matrix R^* is null (it corresponds to the non-spatial model). In that case the RRMSE with $\Sigma^*_{\bar{d}}$ and Σ^*_d are :

col_0a	col_0b
1.59	1.67
0.28	0.28
0.78	0.77
0.58	0.57

col_0a	col_0b
0.63	0.63
0.07	0.07
0.15	0.15
0.08	0.08
NaN	NaN
NaN	NaN
NaN	NaN
NaN	NaN
8.61	8.32
19.65	NaN
19.65	NaN
8.53	8.24

4.1.2 Different values for Σ

We now do the same simulation process changing the values of covariance matrices:

- $$\Sigma^*_{\bar{d}} = \begin{pmatrix} 2 & 0.5 \\ 0.5 & 1 \end{pmatrix}$$

  and

$$\Sigma^*_{d} = \begin{pmatrix} 2 & 0 \\ 0 & 1 \end{pmatrix}$$

We obtain the following RRMSE:

col_1	col_2	col_3	col_4	col_5	col_6	col_7
1.92	1.91	2.13	2.13	2.15	2.15	2.15
0.28	0.27	0.29	0.29	0.29	0.29	0.29
0.82	0.81	0.77	0.77	0.83	0.83	0.83
0.62	0.63	0.59	0.59	0.60	0.60	0.60
0.75	0.70	0.72	0.72	0.72	0.72	0.72
0.07	0.07	0.07	0.07	0.07	0.07	0.07
0.15	0.15	0.15	0.15	0.15	0.15	0.15
0.08	0.08	0.08	0.08	0.08	0.08	0.08
1.04	1.07	0.90	0.89	0.92	0.92	0.92
0.51	0.50	NaN	NaN	NaN	NaN	NaN
0.52	0.52	NaN	NaN	NaN	NaN	NaN
0.41	0.39	0.39	0.38	0.40	0.40	0.40
8.29	8.45	8.39	8.39	8.80	8.80	8.87
18.01	Inf	18.15	18.16	Inf	Inf	NaN
18.01	Inf	18.15	18.16	Inf	Inf	NaN
8.31	8.54	8.21	8.21	8.49	8.49	8.54

- $$\Sigma^*_{\bar{d}} = \begin{pmatrix} 25 & 20 \\ 20 & 16 \end{pmatrix}$$

  and

$$\Sigma_d^* = \begin{pmatrix} 25 & 0 \\ 0 & 16 \end{pmatrix}$$

We obtain the following RRMSE:

col_1	col_2	col_3	col_4	col_5	col_6	col_7
11.28445	11.45046	12.68398	9.78145	12.90210	12.90048	12.90847
1.69430	1.62567	1.71222	1.69745	1.71109	1.71091	1.71079
4.90474	4.85234	4.59479	4.59348	4.94530	4.94530	4.94608
3.70887	3.75422	3.52312	3.50427	3.55387	3.55372	3.55324
9.02756	9.17667	8.90996	7.82370	9.05209	9.05364	9.04896
0.90363	0.94658	0.90729	0.90539	0.87545	0.87551	0.87499
1.96189	1.88109	1.83741	1.83739	1.88847	1.88847	1.88851
0.98903	0.97562	0.94080	0.93449	0.96180	0.96176	0.96192
6.22147	6.40166	5.36365	0.15180	5.52144	5.51432	5.44316
6.22147	6.47730	NaN	NaN	NaN	NaN	NaN
3.14496	3.11430	NaN	NaN	NaN	NaN	NaN
5.03194	4.91818	4.89307	0.13813	5.03184	5.03203	5.00911
8.47455	8.44630	8.40381	8.35466	8.78792	8.78770	8.85663
8.47455	Inf	8.39609	8.35428	Inf	Inf	NaN
8.47455	Inf	8.39609	8.35428	Inf	Inf	NaN
8.47455	8.55906	8.38205	8.35389	8.49635	8.49643	8.54123

5 Application to a real dataset

We first load the data which are also used in Nguyen et al. (2019). The data (resp. codes) can be found here (resp. here).

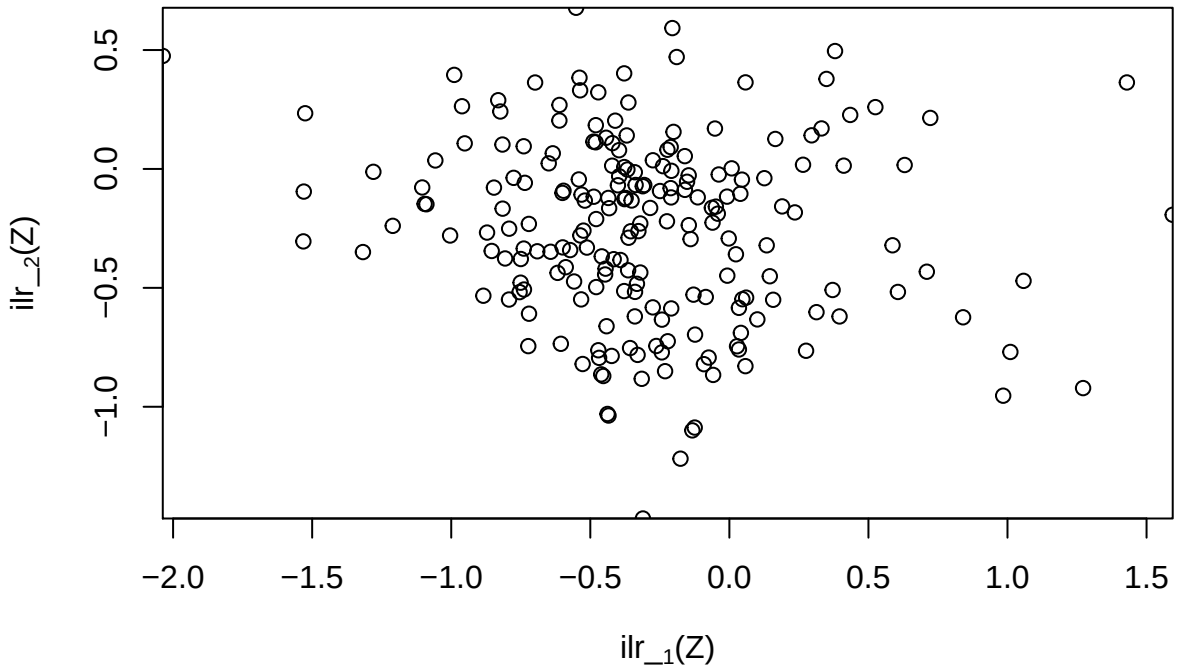
```
load(url("http://www.thibault.laurent.free.fr/code/spatial_coda/R/data_cantons.RData"))
```

```
source(url("http://www.thibault.laurent.free.fr/code/spatial_coda/R/preparation_base.R"))
```

```
##   missForest iteration 1 in progress...done!
##   missForest iteration 2 in progress...done!
##   missForest iteration 3 in progress...done!
```

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("diplome3_ilr1", "diplome3_ilr2",
                            "employ_ilr1", "employ_ilr2", "employ_ilr3",
                            "employ_ilr4",
                            "age3_ilr1", "age3_ilr2",
                            "unemp_rate", "income_rate", "voters")]),
         "matrix")
ne <- nrow(Xe)
k <- ncol(Xe)
```

5.1 Linear model

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

```
res_N_comp_1 <- lm(Ye[, 1] ~ Xe - 1)
res_N_comp_2 <- lm(Ye[, 2] ~ Xe - 1)
```

```
stargazer::stargazer(res_N_comp_1,
                     res_N_comp_2,
                     coef = list(res_N_comp_1$coefficients, res_N_comp_2$coefficients),
                     p = list(coef(summary(res_N_comp_1))[, 4], coef(summary(res_N_comp_2))[, 4]),
                     digits = 3,
                     type = "html")
```

Dependent variable:

Ye[, 1]

Ye[, 2]

(1)

(2)

Xe1

-4.045***

-4.576***

(1.159)

(0.576)

Xediplome3_ilr1

-1.262**

-0.292

(0.504)

(0.250)

Xediplome3_ilr2

-0.037

-0.969***

(0.607)

(0.302)

Xeemploy_ilr1

-0.185

-0.102

(0.139)

(0.069)

Xeemploy_ilr2

0.498***

-0.020

(0.163)

(0.081)

Xeemploy_ilr3

-0.208*

0.004

(0.113)

(0.056)

Xeemploy_ilr4

0.212***

0.016

(0.059)

(0.029)

Xeage3_ilr1

-1.153***

1.022***

(0.378)

(0.188)

Xeage3_ilr2

0.485

-1.272***

(0.313)

(0.155)

Xeunemp_rate

0.215

8.964***

(2.368)

(1.177)

Xeincome_rate

4.379***

1.276***

(0.899)

(0.447)

Xevoters

0.039

0.223***

(0.090)

(0.045)

Observations

207

207

R2

0.434

0.772

Adjusted R2

0.399

0.758

Residual Std. Error (df = 195)

0.455

0.226

F Statistic (df = 12; 195)

12.474***

55.087***

Note:

p<0.1; ***p<0.05;*** p<0.01

which is equivalent to:

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

Adjusted R^2 is equal to 40% in the first component and 75.8% in the second component.

To obtain the results of the linear model presented in Table 3:

```
s_res_N_comp_1 <- summary(res_N_comp_1)
s_res_N_comp_2 <- summary(res_N_comp_2)
res_lm <- cbind(paste0(round(s_res_N_comp_1$coefficients[, 1], 2),
             "(", round(s_res_N_comp_1$coefficients[, 2], 2), ")",
             ifelse(s_res_N_comp_1$coefficients[, 4] < 0.001, "***",
       ifelse(s_res_N_comp_1$coefficients[, 4] < 0.01 &
                s_res_N_comp_1$coefficients[, 4] > 0.001, "**",
              ifelse(s_res_N_comp_1$coefficients[, 4] < 0.05 &
                       s_res_N_comp_1$coefficients[, 4] > 0.01, "*", ""))),

      paste0(round(s_res_N_comp_2$coefficients[, 1], 2),
             "(", round(s_res_N_comp_2$coefficients[, 2], 2), ")",
             ifelse(s_res_N_comp_2$coefficients[, 4] < 0.001, "***",
       ifelse(s_res_N_comp_2$coefficients[, 4] < 0.01 &
                s_res_N_comp_2$coefficients[, 4] > 0.001, "**",
              ifelse(s_res_N_comp_2$coefficients[, 4] < 0.05 &
                       s_res_N_comp_2$coefficients[, 4] > 0.01, "*", ""))))
res_lm
```

```
##       [,1]             [,2]
##  [1,] "-4.04(1.16)***" "-4.58(0.58)***"
##  [2,] "-1.26(0.5)*"    "-0.29(0.25)"
##  [3,] "-0.04(0.61)"    "-0.97(0.3)**"
##  [4,] "-0.19(0.14)"    "-0.1(0.07)"
##  [5,] "0.5(0.16)**"    "-0.02(0.08)"
##  [6,] "-0.21(0.11)"    "0(0.06)"
##  [7,] "0.21(0.06)***"  "0.02(0.03)"
##  [8,] "-1.15(0.38)**"  "1.02(0.19)***"
##  [9,] "0.48(0.31)"     "-1.27(0.16)***"
## [10,] "0.21(2.37)"     "8.96(1.18)***"
## [11,] "4.38(0.9)***"   "1.28(0.45)**"
## [12,] "0.04(0.09)"     "0.22(0.04)***"
```

To compute the Σ matrix:

```
res <- cbind(residuals(res_N_comp_1),
             residuals(res_N_comp_2))
t(res) %*% res / (ne - k)
```

```
##             [,1]        [,2]
## [1,]  0.20683418 -0.01688136
## [2,] -0.01688136  0.05112472
```

Then, we examine the spatial distribution of the residuals. For this, we first compute a spatial weight matrix based on the 5-nearest neighbours.

```
coords_fr <- st_coordinates(st_centroid(contours_occitanie_no0))
W_listw <- nb2listw(knn2nb(knearneigh(coords_fr[, 1:2], 5)),
                    style = "W")
W_dep <- listw2mat(W_listw)
```

We test the spatial autocorrelation of the residuals ilr by ilr:

```
lm.morantest(res_N_comp_1, listw = W_listw)
```

```
##
##  Global Moran I for regression residuals
##
## data:
## model: lm(formula = Ye[, 1] ~ Xe - 1)
## weights: W_listw
##
## Moran I statistic standard deviate = 7.9998, p-value = 6.232e-16
## alternative hypothesis: greater
## sample estimates:
## Observed Moran I      Expectation         Variance
##      0.282178519     -0.021936516      0.001445172
```

```
lm.morantest(res_N_comp_2, listw = W_listw)
```

```
##
##  Global Moran I for regression residuals
##
## data:
## model: lm(formula = Ye[, 2] ~ Xe - 1)
## weights: W_listw
##
## Moran I statistic standard deviate = 5.2721, p-value = 6.743e-08
## alternative hypothesis: greater
## sample estimates:
## Observed Moran I      Expectation         Variance
##      0.178485632     -0.021936516      0.001445172
```

We apply the LM test for each ilr:

```
summary(lm.LMtests(res_N_comp_1,  W_listw, test = "all"))
```

```
##  Lagrange multiplier diagnostics for spatial dependence
## data:
## model: lm(formula = Ye[, 1] ~ Xe - 1)
## weights: W_listw
##
##        statistic parameter   p.value
## LMerr  48.271640         1 3.711e-12 ***
## LMlag  55.527723         1 9.215e-14 ***
## RLMerr  0.044439         1  0.833040
## RLMlag  7.300522         1  0.006893 **
## SARMA  55.572162         2 8.563e-13 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

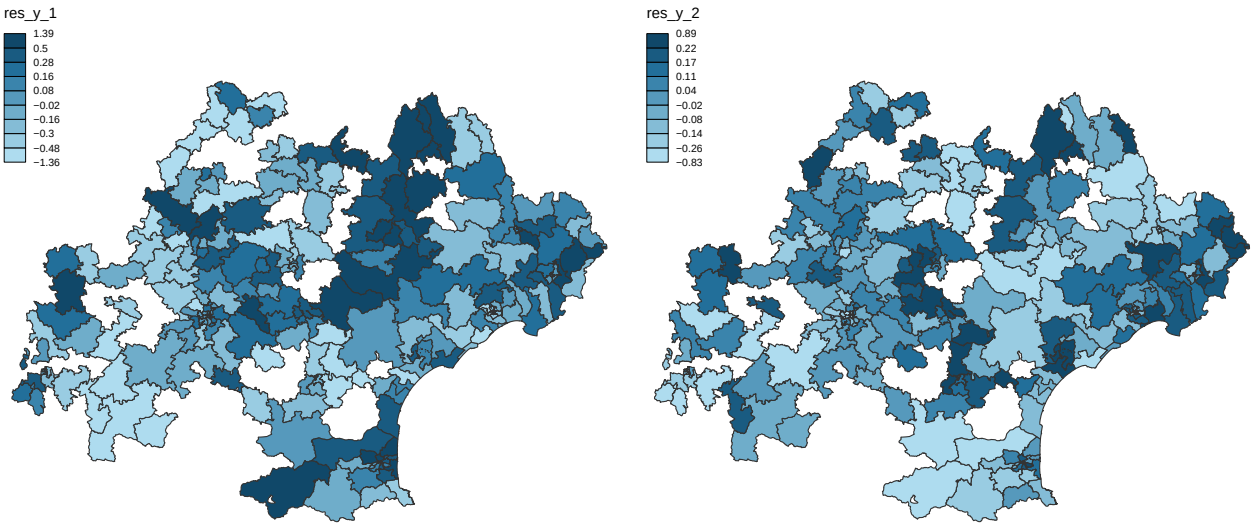
```
summary(lm.LMtests(res_N_comp_2,  W_listw, test = "all"))
```

```
##  Lagrange multiplier diagnostics for spatial dependence
## data:
## model: lm(formula = Ye[, 2] ~ Xe - 1)
## weights: W_listw
##
##         statistic parameter   p.value
## LMerr  19.3130415         1 1.109e-05 ***
## LMlag  38.6406889         1 5.095e-10 ***
## RLMerr  0.0095178         1    0.9223
## RLMlag 19.3371652         1 1.096e-05 ***
## SARMA  38.6502067         2 4.048e-09 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

We plot the residuals:

```
contours_occitanie_no0[, c("res_y_1", "res_y_2")] <- residuals(res_N)
```

```
library("cartography")
op <- par(mfrow = c(1, 2), oma = c(0, 0, 0, 0), mar = c(0, 0, 1, 0))
choroLayer(contours_occitanie_no0, var = "res_y_1", legend.pos = "topleft",
           method = "quantile", legend.values.rnd = 2)
choroLayer(contours_occitanie_no0, var = "res_y_2", legend.pos = "topleft",
           method = "quantile", legend.values.rnd = 2)
```




```
par(op)
```

5.2 Univariate Gaussian LAG model

We fit a spatial regression model of the type LAG by using *stsls()* and *lagsarlm()* to compare the results when considering component by component:

```
s2sls_1 <- spatialreg::stsls(Ye[, 1] ~ Xe[, -1], listw = W_listw, W2X = T)
summary(s2sls_1)
```

```
##
```

```
## Call:spatialreg::stsls(formula = Ye[, 1] ~ Xe[, -1], listw = W_listw,
##     W2X = T)
##
## Residuals:
##       Min        1Q    Median        3Q       Max
## -1.311612 -0.225274  0.014653  0.241455  1.158600
##
## Coefficients:
##                         Estimate Std. Error t value  Pr(>|t|)
## Rho                     0.689022   0.154420  4.4620 8.120e-06
## (Intercept)            -3.496957   1.011420 -3.4575 0.0005453
## Xe[, -1]diplome3_ilr1 -0.618114   0.459604 -1.3449 0.1786629
## Xe[, -1]diplome3_ilr2 -0.050972   0.525571 -0.0970 0.9227392
## Xe[, -1]employ_ilr1   -0.104411   0.121944 -0.8562 0.3918740
## Xe[, -1]employ_ilr2    0.321793   0.146389  2.1982 0.0279348
## Xe[, -1]employ_ilr3   -0.204722   0.097687 -2.0957 0.0361099
## Xe[, -1]employ_ilr4    0.137338   0.053880  2.5490 0.0108047
## Xe[, -1]age3_ilr1     -0.669202   0.345050 -1.9394 0.0524483
## Xe[, -1]age3_ilr2      0.479057   0.270786  1.7691 0.0768716
## Xe[, -1]unemp_rate     1.186916   2.063358  0.5752 0.5651321
## Xe[, -1]income_rate    3.474142   0.805150  4.3149 1.597e-05
## Xe[, -1]voters         0.095356   0.078963  1.2076 0.2271959
##
## Residual variance (sigma squared): 0.15527, (sigma: 0.39404)
```

```
ml_1 <- spatialreg::lagsarlm(Ye[, 1] ~ Xe[, -1], listw = W_listw)
summary(ml_1)
```

```
##
## Call:spatialreg::lagsarlm(formula = Ye[, 1] ~ Xe[, -1], listw = W_listw)
##
## Residuals:
##        Min         1Q     Median         3Q        Max
## -1.3045994 -0.2377183  0.0059494  0.2284552  1.2051824
##
## Type: lag
## Coefficients: (asymptotic standard errors)
##                         Estimate Std. Error z value  Pr(>|z|)
## (Intercept)            -3.605047   0.984501 -3.6618 0.0002504
## Xe[, -1]diplome3_ilr1 -0.745125   0.428545 -1.7387 0.0820820
## Xe[, -1]diplome3_ilr2 -0.048308   0.515256 -0.0938 0.9253033
## Xe[, -1]employ_ilr1   -0.120390   0.118208 -1.0185 0.3084611
## Xe[, -1]employ_ilr2    0.356508   0.138794  2.5686 0.0102105
## Xe[, -1]employ_ilr3   -0.205348   0.095769 -2.1442 0.0320166
## Xe[, -1]employ_ilr4    0.151979   0.050666  2.9996 0.0027031
## Xe[, -1]age3_ilr1     -0.764667   0.324370 -2.3574 0.0184038
## Xe[, -1]age3_ilr2      0.480177   0.266167  1.8040 0.0712244
## Xe[, -1]unemp_rate     0.995144   2.016276  0.4936 0.6216202
## Xe[, -1]income_rate    3.652582   0.766772  4.7636 1.902e-06
## Xe[, -1]voters         0.084246   0.076406  1.1026 0.2701918
##
## Rho: 0.55312, LR test value: 43.555, p-value: 4.1213e-11
## Asymptotic standard error: 0.067289
##     z-value: 8.22, p-value: 2.2204e-16
## Wald statistic: 67.568, p-value: 2.2204e-16
```

```
##
## Log likelihood: -102.6624 for lag model
## ML residual variance (sigma squared): 0.14909, (sigma: 0.38612)
## Number of observations: 207
## Number of parameters estimated: 14
## AIC: 233.32, (AIC for lm: 274.88)
## LM test for residual autocorrelation
## test value: 0.0073803, p-value: 0.93154
```

```
s2sls_2 <- spatialreg::stsls(Ye[, 2] ~ Xe[, -1], listw = W_listw, W2X = T)
summary(s2sls_2)
```

```
##
## Call:spatialreg::stsls(formula = Ye[, 2] ~ Xe[, -1], listw = W_listw,
##     W2X = T)
##
## Residuals:
##       Min        1Q    Median        3Q       Max
## -0.776875 -0.122633 -0.002534  0.120726  0.655848
##
## Coefficients:
##                         Estimate Std. Error t value  Pr(>|t|)
## Rho                     0.633125   0.090516  6.9946 2.660e-12
## (Intercept)            -2.467083   0.597089 -4.1319 3.599e-05
## Xe[, -1]diplome3_ilr1 -0.461125   0.225298 -2.0467 0.0406843
## Xe[, -1]diplome3_ilr2 -0.580405   0.275459 -2.1070 0.0351136
## Xe[, -1]employ_ilr1   -0.116459   0.061932 -1.8804 0.0600512
## Xe[, -1]employ_ilr2    0.037127   0.072833  0.5098 0.6102242
## Xe[, -1]employ_ilr3    0.078128   0.051251  1.5244 0.1274013
## Xe[, -1]employ_ilr4   -0.023983   0.026932 -0.8905 0.3731975
## Xe[, -1]age3_ilr1      0.316707   0.196064  1.6153 0.1062403
## Xe[, -1]age3_ilr2     -0.640628   0.165709 -3.8660 0.0001106
## Xe[, -1]unemp_rate     1.853182   1.463898  1.2659 0.2055406
## Xe[, -1]income_rate    0.700395   0.408353  1.7152 0.0863140
## Xe[, -1]voters         0.145280   0.041517  3.4993 0.0004664
##
## Residual variance (sigma squared): 0.040914, (sigma: 0.20227)
```

```
ml_2 <- spatialreg::lagsarlm(Ye[, 2] ~ Xe[, -1], listw = W_listw)
summary(ml_2)
```

```
##
## Call:spatialreg::lagsarlm(formula = Ye[, 2] ~ Xe[, -1], listw = W_listw)
##
## Residuals:
##       Min        1Q    Median        3Q       Max
## -0.792960 -0.118818 -0.010506  0.135172  0.669926
##
## Type: lag
## Coefficients: (asymptotic standard errors)
##                         Estimate Std. Error z value  Pr(>|z|)
## (Intercept)            -3.061754   0.547120 -5.5961 2.192e-08
## Xe[, -1]diplome3_ilr1 -0.413420   0.217667 -1.8993 0.0575214
## Xe[, -1]diplome3_ilr2 -0.690084   0.262855 -2.6253 0.0086562
## Xe[, -1]employ_ilr1   -0.112484   0.060160 -1.8698 0.0615164
```

```
## Xe[, -1]employ_ilr2    0.020892   0.070393  0.2968 0.7666207
## Xe[, -1]employ_ilr3    0.057241   0.048903  1.1705 0.2418004
## Xe[, -1]employ_ilr4   -0.012604   0.025740 -0.4897 0.6243727
## Xe[, -1]age3_ilr1      0.515567   0.171405  3.0079 0.0026306
## Xe[, -1]age3_ilr2     -0.818532   0.143939 -5.6867 1.296e-08
## Xe[, -1]unemp_rate     3.858148   1.144207  3.3719 0.0007465
## Xe[, -1]income_rate    0.862557   0.390528  2.2087 0.0271960
## Xe[, -1]voters         0.167118   0.040253  4.1517 3.300e-05
##
## Rho: 0.45462, LR test value: 38.239, p-value: 6.2598e-10
## Asymptotic standard error: 0.063518
##     z-value: 7.1574, p-value: 8.2245e-13
## Wald statistic: 51.228, p-value: 8.2234e-13
##
## Log likelihood: 39.33595 for lag model
## ML residual variance (sigma squared): 0.038619, (sigma: 0.19652)
## Number of observations: 207
## Number of parameters estimated: 14
## AIC: -50.672, (AIC for lm: -14.433)
## LM test for residual autocorrelation
## test value: 0.00091347, p-value: 0.97589
```

5.3 Multivariate Gaussian LAG model

We estimate a multivariate gaussian LAG model by using the *estimate_spatial_multi_gen_N()* function.

```
W_listw <- nb2listw(knn2nb(knearneigh(coords_fr[, 1:2], 5)),
                    style = "W")
W_dep <- listw2mat(W_listw)
(res_sar_N <- estimate_spatial_multi_gen_N(Y = Ye, X = Xe, W = W_dep,
                  method = "s2sls",
                  ind_RHO = matrix(c(T, T, T, T), 2, 2), compute_sd = T)
)
```

```
## $est_beta
##               [,1]        [,2]
##  [1,] -2.95195812 -2.46697013
##  [2,] -0.68340377 -0.46066267
##  [3,]  0.05313783 -0.58046292
##  [4,] -0.11070865 -0.11640142
##  [5,]  0.34265939  0.03699924
##  [6,] -0.18507538  0.07812100
##  [7,]  0.12890759 -0.02402895
##  [8,] -0.87237423  0.31712636
##  [9,]  0.64741980 -0.64071009
## [10,] -0.73899601  1.85473474
## [11,]  3.34920597  0.69984590
## [12,]  0.07294522  0.14532839
##
## $est_RHO
##               [,1]      [,2]
## [1,] 0.6674169444 0.1687522
## [2,] 0.0004725084 0.6330463
##
```

```
## $est_GAMMA
##      [,1] [,2]
## [1,]    0    0
## [2,]    0    0
##
## $est_SIGMA
##             [,1]        [,2]
## [1,]  0.15179208 -0.02792752
## [2,] -0.02792752  0.04069942
##
## $sd_beta
##             [,1]       [,2]
##  [1,] 1.15067519 0.59582993
##  [2,] 0.45952030 0.23794373
##  [3,] 0.53091039 0.27491016
##  [4,] 0.12075093 0.06252592
##  [5,] 0.14637362 0.07579357
##  [6,] 0.09874328 0.05113016
##  [7,] 0.05399630 0.02795977
##  [8,] 0.40177055 0.20804039
##  [9,] 0.32031605 0.16586252
## [10,] 2.86493590 1.48348949
## [11,] 0.80671300 0.41772322
## [12,] 0.08150726 0.04220519
##
## $sd_RHO
##            [,1]       [,2]
## [1,] 0.15434114 0.17624167
## [2,] 0.07991922 0.09125951
##
## $sd_GAMMA
##      [,1] [,2]
## [1,]    0    0
## [2,]    0    0
```

Finally, we compute the p-values:

```
# for the beta
n_test_beta <- round(2 * (1 - pnorm(abs(res_sar_N$est_beta / res_sar_N$sd_beta))), 2)
sign_beta <- ifelse(n_test_beta <= 0.001, "***",
       ifelse(n_test_beta <= 0.01 & n_test_beta > 0.001, "**",
              ifelse(n_test_beta <= 0.05 & n_test_beta > 0.01, "*", "")))
# for the rho
n_test_rho <- round(2 * (1 - pnorm(abs(res_sar_N$est_RHO / res_sar_N$sd_RHO))), 2)
sign_RHO <- ifelse(n_test_rho <= 0.001, "***",
       ifelse(n_test_rho <= 0.01 & n_test_rho > 0.001, "**",
              ifelse(n_test_rho <= 0.05 & n_test_rho > 0.01, "*", "")))
```

To obtain the results of the Multivariate Gaussian LAG model presented in Table 3:

```
res_sar <- cbind(paste0(round(c(res_sar_N$est_beta[, 1], res_sar_N$est_RHO[1, ]), 2),
             "(", round(c(res_sar_N$sd_beta[, 1], res_sar_N$sd_RHO[1, ]), 2), ")",
             c(sign_beta[, 1], sign_RHO[1, ])),

      paste0(round(c(res_sar_N$est_beta[, 2], res_sar_N$est_RHO[2, ]), 2),
             "(", round(c(res_sar_N$sd_beta[, 2], res_sar_N$sd_RHO[2, ]), 2), ")",
```

```
            c(sign_beta[, 2], sign_RHO[2, ])))
```

Finally, we obtain Table 3:

```
res_lm <- rbind(res_lm, matrix(NA, 2, 2))
cbind(res_lm, res_sar)
```

```
##       [,1]             [,2]             [,3]            [,4]
##  [1,] "-4.04(1.16)***" "-4.58(0.58)***" "-2.95(1.15)**" "-2.47(0.6)***"
##  [2,] "-1.26(0.5)*"    "-0.29(0.25)"    "-0.68(0.46)"   "-0.46(0.24)*"
##  [3,] "-0.04(0.61)"    "-0.97(0.3)**"   "0.05(0.53)"    "-0.58(0.27)*"
##  [4,] "-0.19(0.14)"    "-0.1(0.07)"     "-0.11(0.12)"   "-0.12(0.06)"
##  [5,] "0.5(0.16)**"    "-0.02(0.08)"    "0.34(0.15)*"   "0.04(0.08)"
##  [6,] "-0.21(0.11)"    "0(0.06)"        "-0.19(0.1)"    "0.08(0.05)"
##  [7,] "0.21(0.06)***"  "0.02(0.03)"     "0.13(0.05)*"   "-0.02(0.03)"
##  [8,] "-1.15(0.38)**"  "1.02(0.19)***"  "-0.87(0.4)*"   "0.32(0.21)"
##  [9,] "0.48(0.31)"     "-1.27(0.16)***" "0.65(0.32)*"   "-0.64(0.17)***"
## [10,] "0.21(2.37)"     "8.96(1.18)***"  "-0.74(2.86)"   "1.85(1.48)"
## [11,] "4.38(0.9)***"   "1.28(0.45)**"   "3.35(0.81)***" "0.7(0.42)"
## [12,] "0.04(0.09)"     "0.22(0.04)***"  "0.07(0.08)"    "0.15(0.04)***"
## [13,] NA               NA               "0.67(0.15)***" "0(0.08)"
## [14,] NA               NA               "0.17(0.18)"    "0.63(0.09)***"
```

Correlation matrix ILR and lagged ILR

```
cor(cbind(Ye, as.vector(Ye[, 1] %*% W_dep), as.vector(Ye[, 2] %*% W_dep)))
```

```
##             [,1]        [,2]       [,3]        [,4]
## [1,]  1.00000000 -0.14530796 0.49789336 -0.02004451
## [2,] -0.14530796  1.00000000 0.02563013  0.73482524
## [3,]  0.49789336  0.02563013 1.00000000  0.06500017
## [4,] -0.02004451  0.73482524 0.06500017  1.00000000
```