R codes for the paper *Predictions in spatial autoregressive models: Application to unemployment data*

Thibault Laurent & Paula Margaretic

Last Update: 2020-10-29

Contents

This document provides the **R** codes used to reproduce the results included in the paper *Predictions in spatial autoregressive models: Application to unemployment data*. The html version is available here.

To cite this work, please use :

Laurent T. and Margaretic P. (2020). Predictions in spatial autoregressive models: Application to unemployment data, *TSE WP*.

Packages needed:

```
library(ggspatial)
library(kableExtra)
library(readxl)
library(sf)
library(spatialreg)
library(spdep)
library(tidyverse)
```

Download the data:

```
download.file("http://www.thibault.laurent.free.fr/code/CT_honor/data.zip",
              paste0(getwd(), "data.zip"))
unzip(zipfile = "data.zip")
```

We modify slightly the *stargazer()* function from **stargazer** so that spatial models from **spatialreg** can be included in the table

```
source("data/stargazer.R")    # modified stargazer function to include
source("data/stargazer-internal.R")   # spatial models
```

1 Importing the data

- Spatial file

The polygons of the employment zone ("zones d'emploi" in French) can be downloaded at https://github.com/riatelab/magrit-templates-carto/tree/master/france__communes__epci__etc/layers

```
library(sf)
zones <- read_sf("data/ZEMP.geojson")
# changing CRS
zones <- st_transform(zones, 2154)
```

- Unemployment rate

The data can be found at https://www.insee.fr/fr/statistiques/1893230

```
chomage <- read_excel("data/chomage-zone-2003-2018.xls",
                      sheet = "txcho_ze",
                      skip = 4)
# dropping unused last line
chomage <- chomage[-nrow(chomage), ]
# redefining codes for Toulouse and Saint-Etienne
chomage$code <- substr(chomage$`Code de la zone (ze)`, 1, 4)
```

We consider two estimation periods. The first sample period uses the unemployment rate in 2013 and the to-be-determined explanatory variables as of 2011. The objective is to reproduce some of the results in Floch and Le Saout (2018). The second data set consists of more recent observations, with the unemployment rate, as of 2018, and the structural determinants, for 2016.

```
zones_with_data_2011 <- merge(zones, chomage %>% select(code, 13) %>% rename(chomage = 2),
                             by.x = "ZE2010", by.y = "code")
zones_with_data_2016 <- merge(zones, chomage %>% select(code, 18) %>% rename(chomage = 2),
                                by.x = "ZE2010", by.y = "code")
```

- Industrial and Public Employment (in percentages)

  Salaried employment is available according to the activity aggregated into the following five positions

  - Agriculture,
  - Industry,
  - Construction,
  - Merchant tertiary
  - Non-merchant tertiary.

We consider the share of industrial and non-merchant tertiary employment, which we refer hereafter as **part_ind** and **part_pub**, respectively. These variables can be downloaded at https://www.insee.fr/fr/statistiques/1893177

```r
public <- read_excel("data/emploi-zone-1998-2016.xls",
                     sheet = "emploi salarié - ZE",
                     skip = 5)
```

We tidy the data to keep the variables corresponding to the years 2011 and 2016:

```r
# variables observed in 2011
emploi_pub_priv_2011 <- public %>% select(1, 3, `2011`) %>%
  rename(code = 1, type_emploi = 2, nombre =3)
emploi_pub_priv_2011$code <- substr(emploi_pub_priv_2011$code, 1, 4)
base_pub_priv_2011 <- pivot_wider(emploi_pub_priv_2011,
                                  names_from = type_emploi, values_from = nombre)
base_pub_priv_2011$part_pub <- base_pub_priv_2011$`OQ-Tertiaire non marchand` /
  base_pub_priv_2011$`TT-Total` * 100
base_pub_priv_2011$part_ind <- base_pub_priv_2011$`BE-Industrie` /
  base_pub_priv_2011$`TT-Total` * 100
# variables observed in 2016
emploi_pub_priv_2016 <- public %>% select(1, 3, `2016`) %>%
  rename(code = 1, type_emploi = 2, nombre =3)
emploi_pub_priv_2016$code <- substr(emploi_pub_priv_2016$code, 1, 4)
base_pub_priv_2016 <- pivot_wider(emploi_pub_priv_2016,
                                  names_from = type_emploi, values_from = nombre)
base_pub_priv_2016$part_pub <- base_pub_priv_2016$`OQ-Tertiaire non marchand` /
  base_pub_priv_2016$`TT-Total` * 100
base_pub_priv_2016$part_ind <- base_pub_priv_2016$`BE-Industrie` /
  base_pub_priv_2016$`TT-Total` * 100
# merging the data
zones_with_data_2011 <- merge(zones_with_data_2011, base_pub_priv_2011 %>%
                                select(code, part_pub, part_ind),
                         by.x = "ZE2010", by.y = "code")
zones_with_data_2016 <- merge(zones_with_data_2016, base_pub_priv_2016 %>%
                                select(code, part_pub, part_ind),
                              by.x = "ZE2010", by.y = "code")
```

- Other covariates

We include the following additional covariates:

- **part_sans_dip**, proportion of low-educated working-age adults
- **evol_immob**, average annual growth of unoccupied houses between 2006-2011 or 2011-2016
- **dens_pop**, population density
- **part_jeunes_actifs**, proportion of working-age adults between 15 and 30
- **taux_act**, Labour force participation rate

They can be downloaded from: https://www.observatoire-des-territoires.gouv.fr/outils/cartographie-interactive/#c=indicator&f2=T&i=defm_historique.dmd_emp_abc&i2=indic_sex_rp.tx_act1564&s=2017&s2=2011&view=map12

```r
# variables observed in 2011
obs_terr_2011 <- read_excel("data/data_OT_2011.xlsx",
                            sheet = "Data",
                            skip = 3,
                            na = "N/A - résultat non disponible")
obs_terr_2011 <- obs_terr_2011 %>% select(-2) %>%
  rename(code = 1, part_sans_dip = 2,
         evol_immob = 3,
```

```r
         dens_pop = 4,
         part_jeunes_actifs = 5,
         taux_act = 6)
# merging with spatial data
zones_with_data_2011 <- merge(zones_with_data_2011, obs_terr_2011,
                         by.x = "ZE2010", by.y = "code")
# variables observed in 2011
obs_terr_2016 <- read_excel("data/data_OT_2016.xlsx",
                            sheet = "Data",
                            skip = 3,
                            na = "N/A - résultat non disponible")
obs_terr_2016 <- obs_terr_2016 %>% select(-2) %>%
  rename(code = 1, part_sans_dip = 3,
         evol_immob = 4,
         dens_pop = 5,
         part_jeunes_actifs = 6,
         taux_act = 2)
# merge with spatial data
zones_with_data_2016 <- merge(zones_with_data_2016, obs_terr_2016,
                              by.x = "ZE2010", by.y = "code")
```

We exclude Corsica of the analysis for geographical reasons.

```r
my_base_2011 <- zones_with_data_2011 %>%
  filter(substr(ZE2010, 1, 2) != 94)

my_base_2016 <- zones_with_data_2016 %>%
  filter(substr(ZE2010, 1, 2) != 94)
```

2 Exploratory analysis

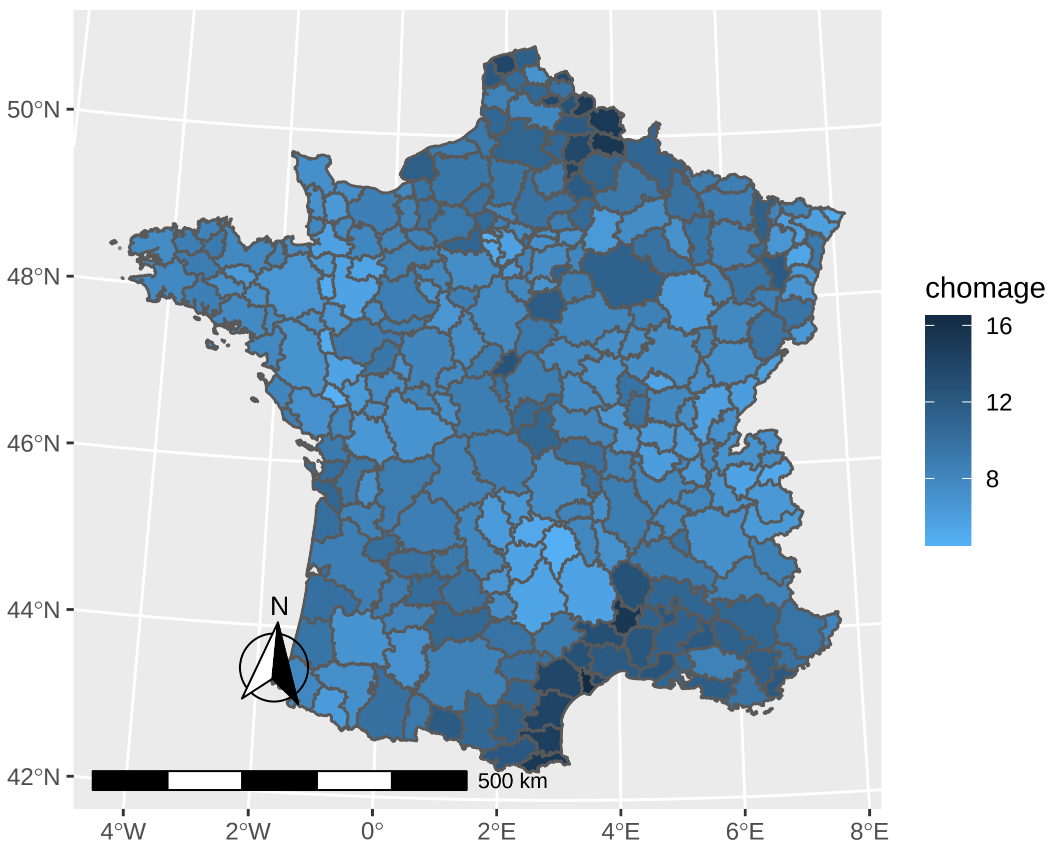
We first present the descriptive statistics of 2018 unemployment rates and the covariates observed in 2016 :

```r
my_sum <- function(x) {
  round(c(N = length(x),
    mean = mean(x),
    sd = sd(x),
    min = min(x),
    q = quantile(x, 0.25),
    q = quantile(x, 0.5),
    q = quantile(x, 0.75),
    max = max(x)), 2)
}
kable(t(sapply(st_drop_geometry(my_base_2016[, c("chomage", "taux_act", "part_sans_dip",
                    "part_jeunes_actifs", "part_ind", "part_pub",
                    "evol_immob", "dens_pop")]), my_sum)), booktabs = T) %>%
  kable_styling(bootstrap_options = "striped", full_width = F, position = "left")
```

	N	mean	sd	min	q.25%	q.50%	q.75%	max
chomage	297	8.77	2.23	4.50	7.20	8.40	9.80	16.50
taux_act	297	73.88	2.50	67.00	72.40	73.80	75.20	83.50
part_sans_dip	297	31.93	4.44	20.40	29.30	32.20	35.10	43.70
part_jeunes_actifs	297	15.51	2.29	10.50	14.00	15.10	16.70	23.90
part_ind	297	17.22	7.80	3.02	11.33	15.98	21.91	42.56
part_pub	297	34.32	6.63	15.65	30.18	34.30	38.41	54.23
evol_immob	297	3.13	1.52	-1.21	2.20	3.06	4.16	7.48
dens_pop	297	184.41	612.56	12.50	49.70	79.70	147.20	9179.00

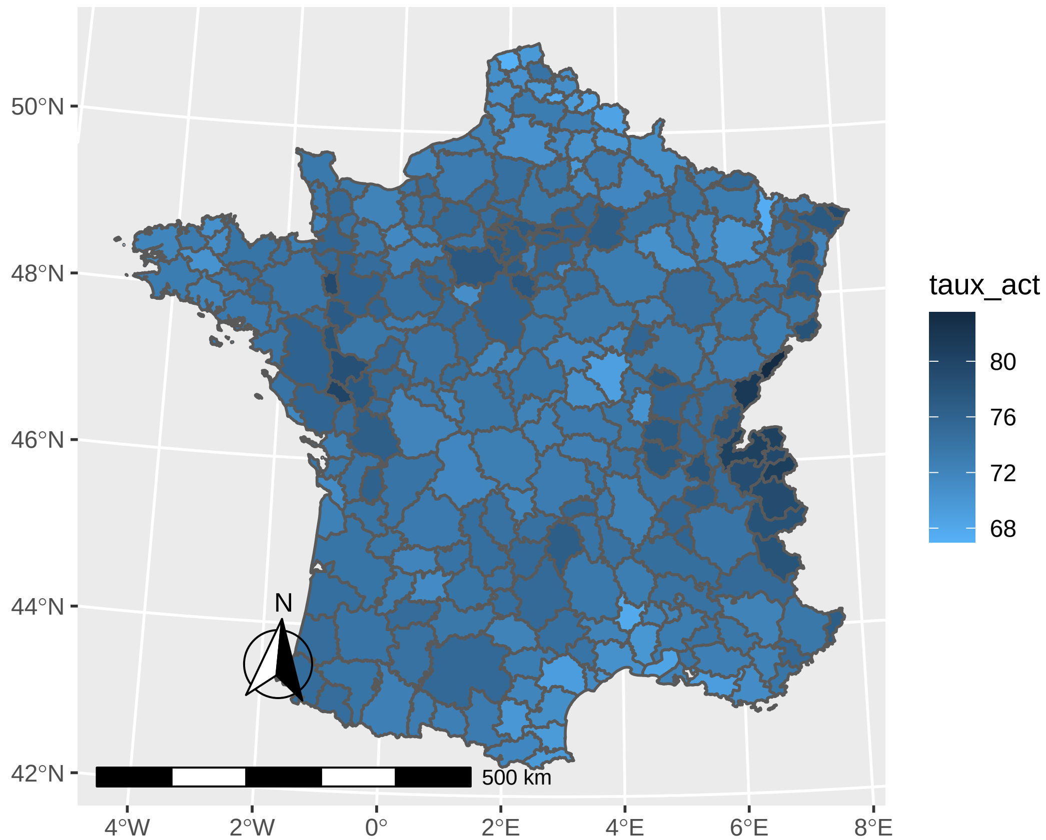
We the plot the maps of the two main variales, namely, : **chomage** and **taux_act** for 2016

```
ggplot(data = my_base_2016) +
  geom_sf(aes(fill = chomage)) +
  scale_fill_gradient(low = "#56B1F7", high = "#132B43") +
  coord_sf(crs = st_crs(2154)) +
  annotation_scale(location = "bl", width_hint = 0.5) +
  annotation_north_arrow(location = "bl", which_north = "true",
                         pad_x = unit(0.75, "in"), pad_y = unit(0.5, "in"),
                         style = north_arrow_fancy_orienteering)
```




```
ggplot(data = my_base_2016) +
  geom_sf(aes(fill = taux_act)) +
  scale_fill_gradient(low = "#56B1F7", high = "#132B43") +
  coord_sf(crs = st_crs(2154)) +
  annotation_scale(location = "bl", width_hint = 0.5) +
  annotation_north_arrow(location = "bl", which_north = "true",
```

```
                    pad_x = unit(0.75, "in"), pad_y = unit(0.5, "in"),
                    style = north_arrow_fancy_orienteering)
```



3 Spatial weight matrix and Moran plot

3.1 Spatial weight matrix definition

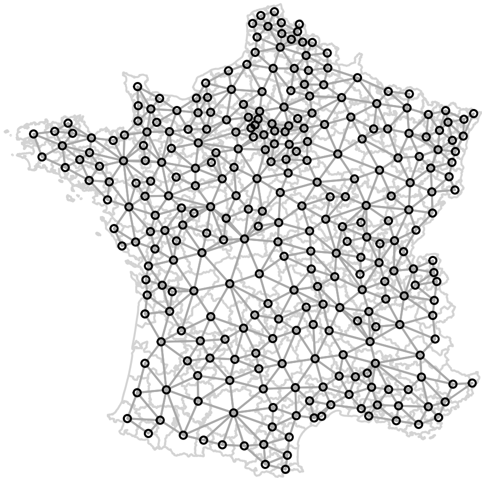
We combine two methods, namely, contiguity and two nearest neighbours. The reason for it is to allow that each employment zone has at least two neighbors. We then row-normalize the resulting spatial weight matrix:

```
library(spdep)
# method 1: contiguity
my_base_nb <- poly2nb(my_base_2016)
# method 1: 2-nearest neighbors
two_knn <- knn2nb(knearneigh(st_centroid(my_base_2016), k = 2))
# union of the methods
for (k in 1:length(my_base_nb)){
  my_base_nb[[k]] <- sort(union(my_base_nb[[k]], two_knn[[k]]))
}
# listw objet (row_normalized)
neigh.listw <- nb2listw(my_base_nb)
```

We plot the neighborhood network :

```
plot(st_geometry(my_base_2016), border = "lightgrey")
plot(my_base_nb, st_coordinates(st_centroid(my_base_2016)), add = T,
```

```
     col = "darkgrey", cex = 0.5)
```



3.2 Moran plot

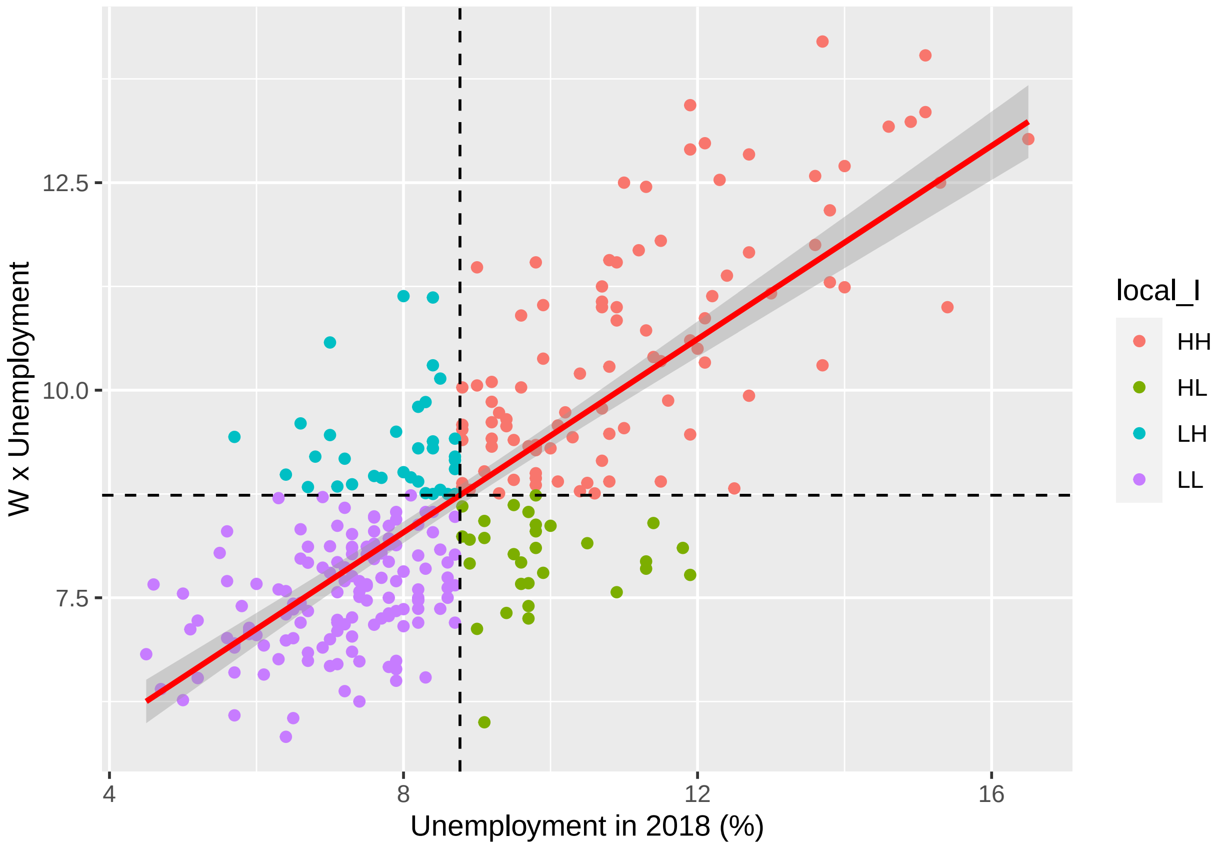
Following Anselin (1995), we define four clusters, which we denote as HH, HL, LH, LL

```
my_base_2016$w_chomage <- listw2mat(neigh.listw) %*% my_base_2016$chomage
my_base_2016$local_I <- "HH"
my_base_2016$local_I[my_base_2016$w_chomage < mean(my_base_2016$w_chomage) &
                       my_base_2016$chomage < mean(my_base_2016$chomage)] <- "LL"
my_base_2016$local_I[my_base_2016$w_chomage < mean(my_base_2016$w_chomage) &
                       my_base_2016$chomage > mean(my_base_2016$chomage)] <- "HL"
my_base_2016$local_I[my_base_2016$w_chomage > mean(my_base_2016$w_chomage) &
                       my_base_2016$chomage < mean(my_base_2016$chomage)] <- "LH"
mean_x <- mean(my_base_2016$chomage)
mean_y <- mean(my_base_2016$w_chomage)
```

We plot the Moran scatter plot :

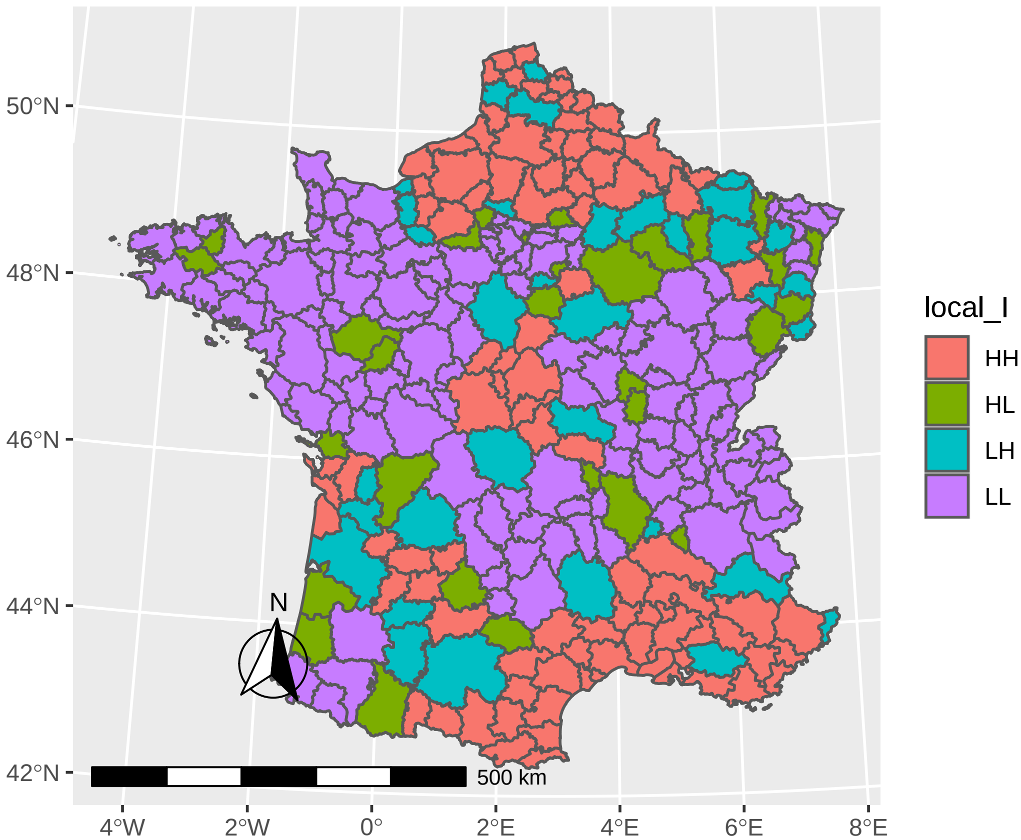
```
ggplot(data = my_base_2016, aes(x = chomage,
                           y = w_chomage,
                           color = local_I)) +
  geom_point()  +
  geom_smooth(method = "lm",
              col = "red") +
  geom_vline(xintercept = mean_x, lty = 2) +
  geom_hline(yintercept = mean_y, lty = 2) +
  xlab("Unemployment in 2018 (%)")+
  ylab("W x Unemployment")
```

```
## `geom_smooth()` using formula 'y ~ x'
```



We plot the associated map :

```
ggplot(data = my_base_2016) +
  geom_sf(aes(fill = local_I)) +
  annotation_scale(location = "bl", width_hint = 0.5) +
  annotation_north_arrow(location = "bl", which_north = "true",
                         pad_x = unit(0.75, "in"), pad_y = unit(0.5, "in"),
                         style = north_arrow_fancy_orienteering)
```



4 Models used in Floch and Le Saout (2018)

In this section, we estimate some of the model specifications considered in Floch and Le Saout (2018) (namely, OLS, SAR, SLX and SDM models) for 2013 unemployment rate, relying on the same structural factors than in they use, for 2011. Please refer to table 4 in the paper for the estimation results.

```
model_insee <- chomage ~ taux_act + part_sans_dip + part_jeunes_actifs + part_ind + part_pub
res_lm_insee <- lm(model_insee, data = my_base_2011)
res_sar_insee <- lagsarlm(model_insee, data = st_drop_geometry(my_base_2011),
                   neigh.listw, Durbin = F)
res_slx_insee <- lmSLX(model_insee, data = my_base_2011, listw = neigh.listw, Durbin = TRUE)
res_sdm_insee <- lagsarlm(model_insee, data = st_drop_geometry(my_base_2011),
                   neigh.listw, Durbin = T)
```

```
stargazer(res_lm_insee, res_sar_insee, res_slx_insee, res_sdm_insee,
           dep.var.labels = c("", "", "", ""), model.names = F, type = "latex")
```

% Table created by stargazer v.5.2.2 by Marek Hlavac, Harvard University. E-mail: hlavac at fas.harvard.edu
% Date and time: jeu., oct. 29, 2020 - 17:11:05

The spatial autocorrelation parameters of the two spatial models can be obtained like this :

```
c(res_sar_insee$rho, res_sdm_insee$rho)
```

```
##       rho       rho
## 0.4972272 0.6010751
```

Table 1:

	Dependent variable:			
	(1)	(2)	(3)	(4)
taux_act	-0.617^{***}	-0.450^{***}	-0.514^{***}	-0.507^{***}
	(0.040)	(0.038)	(0.052)	(0.043)
part_sans_dip	0.212^{***}	0.150^{***}	0.169^{***}	0.167^{***}
	(0.030)	(0.025)	(0.039)	(0.032)
part_jeunes_actifs	0.115^{***}	0.051	0.101^{**}	0.073^{**}
	(0.038)	(0.032)	(0.044)	(0.037)
part_ind	-0.062^{***}	-0.040^{***}	-0.028	-0.021
	(0.014)	(0.012)	(0.017)	(0.014)
part_pub	-0.064^{***}	-0.064^{***}	-0.047^{**}	-0.047^{***}
	(0.018)	(0.015)	(0.019)	(0.015)
lag.taux_act			-0.208^{***}	0.190^{**}
			(0.077)	(0.074)
lag.part_sans_dip			0.032	-0.098^{**}
			(0.055)	(0.047)
lag.part_jeunes_actifs			0.121^{*}	0.034
			(0.068)	(0.057)
lag.part_ind			-0.094^{***}	-0.036
			(0.028)	(0.023)
lag.part_pub			-0.038	0.003
			(0.036)	(0.030)
Constant	52.043^{***}	36.791^{***}	59.784^{***}	26.380^{***}
	(3.311)	(3.207)	(5.426)	(5.649)
Observations	297	297	297	297
R^2	0.623		0.664	
Adjusted R^2	0.617		0.652	
Log Likelihood		-487.579		-477.423
σ^2		1.479		1.340
Akaike Inf. Crit.		991.157		980.847
Residual Std. Error	1.467 (df = 291)		1.398 (df = 286)	
F Statistic	96.329^{***} (df = 5; 291)		56.403^{***} (df = 10; 286)	
Wald Test (df = 1)		110.300^{***}		108.815^{***}
LR Test (df = 1)		89.083^{***}		75.898^{***}
Note:				*p<0.1; **p<0.05; ***p<0.01

5 Our models

We now introduce two changes to the OLS, SAR, SLX and SDM model estimates in the previous section, that is, we consider more recent observations, with the unemployment rate, as of 2018, and the covariates for 2016; we augment the model specifications with the logarithm of population density and the annual growth of unoccupied houses. Please refer to table 5 in the paper for the estimation results.

```r
model_aragon <- chomage ~ taux_act + part_sans_dip + part_jeunes_actifs +
  part_ind + part_pub + evol_immob + log(dens_pop)
res_lm_aragon <- lm(model_aragon, data = my_base_2016)
res_sar_aragon <- lagsarlm(model_aragon, data = st_drop_geometry(my_base_2016),
                   neigh.listw, Durbin = F)
res_slx_aragon <- lmSLX(model_aragon, data = my_base_2016, listw = neigh.listw, Durbin = TRUE)
res_sdm_aragon <- lagsarlm(model_aragon, data = st_drop_geometry(my_base_2016),
                   neigh.listw, Durbin = T)
```

```r
stargazer(res_lm_aragon, res_sar_aragon, res_slx_aragon, res_sdm_aragon,
          dep.var.labels = c("", "", "", ""), model.names = F, type = "latex")
```

% Table created by stargazer v.5.2.2 by Marek Hlavac, Harvard University. E-mail: hlavac at fas.harvard.edu
% Date and time: jeu., oct. 29, 2020 - 17:11:06

The spatial autocorrelation parameters of the two spatial models can be obtained like this :

```r
c(res_sar_aragon$rho, res_sdm_aragon$rho)
```

```
##       rho       rho
## 0.5806759 0.6800133
```

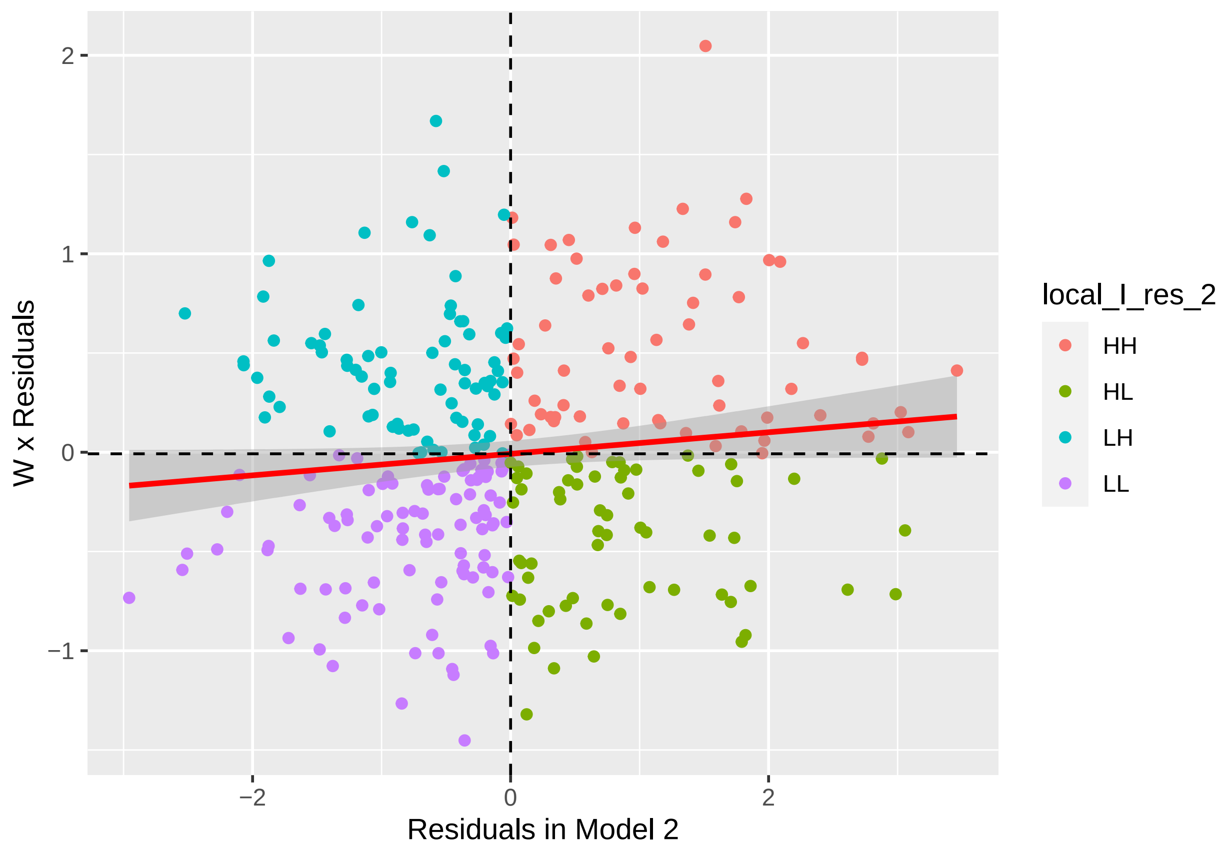
Remark: we have represented the Moran plot of the residuals in the SAR and SDM models and it appears that the models have indeed taken the correlation into account.

In the SAR model:

```
## `geom_smooth()` using formula 'y ~ x'
```

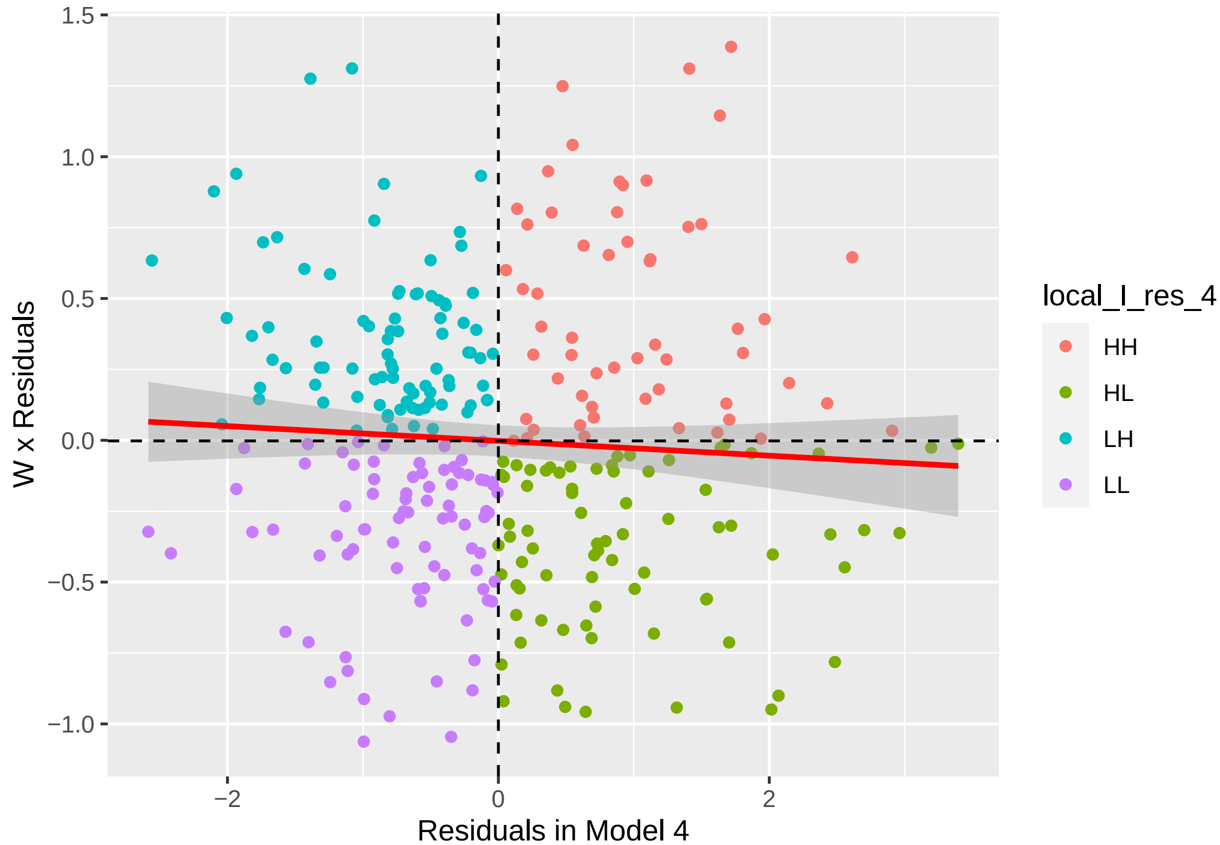
Table 2:

	Dependent variable:			
	(1)	(2)	(3)	(4)
taux_act	−0.546***	−0.366***	−0.424***	−0.396***
	(0.048)	(0.041)	(0.060)	(0.046)
part_sans_dip	0.101***	0.075***	0.109***	0.113***
	(0.028)	(0.022)	(0.036)	(0.028)
part_jeunes_actifs	−0.062	−0.066	0.008	−0.038
	(0.055)	(0.043)	(0.068)	(0.053)
part_ind	−0.045***	−0.024*	−0.023	−0.016
	(0.016)	(0.013)	(0.017)	(0.013)
part_pub	−0.020	−0.026	−0.016	−0.014
	(0.020)	(0.016)	(0.020)	(0.016)
evol_immob	−0.090	−0.060	−0.039	0.003
	(0.060)	(0.047)	(0.060)	(0.047)
log(dens_pop)	0.599***	0.426***		0.689***
	(0.149)	(0.118)		(0.163)
log.dens_pop.			0.569***	
			(0.213)	
lag.taux_act			−0.310***	0.144*
			(0.094)	(0.079)
lag.part_sans_dip			−0.043	−0.095**
			(0.054)	(0.042)
lag.part_jeunes_actifs			0.177	0.131
			(0.117)	(0.090)
lag.part_ind			−0.079**	−0.026
			(0.034)	(0.026)
lag.part_pub			−0.032	−0.007
			(0.042)	(0.032)
lag.evol_immob			−0.287**	−0.096
			(0.114)	(0.088)
lag.log.dens_pop.			−0.398	
			(0.364)	
lag.log(dens_pop)				−0.744***
				(0.280)
Constant	45.845***	28.950***	61.602***	21.420***
	(4.692)	(3.889)	(8.010)	(6.736)
Observations	297	297	297	297
R^2	0.555		0.610	
Adjusted R^2	0.544		0.591	



In the SDM model:

```
## `geom_smooth()` using formula 'y ~ x'
```



6 Predictions

6.1 In-Sample

We first compute the in-sample predictions, for the following three alternative prediction formulas, **TS**, **TC** or **BP**.

```
predictor_insample <- data.frame(
  y_ols = predict(res_lm_aragon),
  y_slx = predict(res_slx_aragon),
  y_sar_BP = as.numeric(predict.sarlm(object = res_sar_aragon,
  y_sar_TC = as.numeric(predict.sarlm(object = res_sar_aragon,
    listw = neigh.listw, zero.policy = T, pred.type = "TC")),
  y_sar_TS = as.numeric(predict.sarlm(object = res_sar_aragon,
    listw = neigh.listw, zero.policy = T, pred.type = "TS")),
  y_sdm_BP = as.numeric(predict.sarlm(object = res_sdm_aragon,
  y_sdm_TC = as.numeric(predict.sarlm(object = res_sdm_aragon,
    listw = neigh.listw, zero.policy = T, pred.type = "TC")),
  y_sdm_TS = as.numeric(predict.sarlm(object = res_sdm_aragon,
    listw = neigh.listw, zero.policy = T, pred.type = "TS"))
  )
```

```
## This method assumes the response is known - see manual page
## This method assumes the response is known - see manual page
```

We then compute the mean square errors (MSE) to compare the predictions of the OLS, SLX, SAR (with the three prediction formulas) and SDM (with the three prediction formulas) model estimates.

```
apply(predictor_insample, 2, function(x) mean((x - my_base_2016$chomage)^2))
```

```
##    y_ols    y_slx y_sar_BP y_sar_TC y_sar_TS y_sdm_BP y_sdm_TC y_sdm_TS
## 2.200253 1.928507 1.229896 2.140583 1.362383 1.138894 1.791637 1.198441
```

Conclusion : We can rank the models in decreasing order of efficiency as follows

- 1. SDM^{BP}
- 2. SDM^{TS}
- 3. SAR^{BP}
- 4. SAR^{TS}
- 5. SDM^{TC}
- 6. SLX
- 7. SAR^{TC}
- 8. OLS

6.2 Out-of-sample

We randomly split the sample into 10 subsamples i, $i = 1, ..., 10$.

For i in 1 to 10, **repeat**:

- 1. Estimate the models (OLS, SLX, SAR, SDM) on the full sample - subsample i
- 2. Predict on the subsample i. For spatial models, we compute the following methods **BP**, **BP1**, **TC**, **TC1**, **TS1**, **BPW**, **BPN**, **BPW1**, **BPN1**, **KP2**, **KP3**.

Divide the sample into 10 :

```
set.seed(1234)
samp <- sample(1:297)
q.samp <- round(quantile(samp, seq(0, 1, 0.1)), 0)
se <- matrix(0, 24, 10)
df_data <- st_drop_geometry(my_base_2016) %>%
  select(-w_chomage, -local_I)
```

Repeat the algorithm:

```
for (k in 1:10) {
  outsamp <- samp %in% (q.samp[k]:q.samp[k + 1])
  S.sample <- df_data[!outsamp, ]
  WS.listw <- subset.listw(neigh.listw, !outsamp)

  O.sample <- df_data[outsamp, ]

  # method single prediction
  O.sample.1 <- df_data[which(outsamp)[1:2], ]
  neigh.listw.1 <- subset.listw(neigh.listw, !(1:nrow(df_data) %in% which(outsamp)[-c(1:2)]))

  res_lm_aragon <- lm(model_aragon, data = S.sample)
  res_sar_aragon <- lagsarlm(model_aragon, data = S.sample,
```

```
                 WS.listw, Durbin = F)
res_slx_aragon <- lmSLX(model_aragon, data = S.sample, listw = WS.listw, Durbin = TRUE)
res_sdm_aragon <- lagsarlm(model_aragon, data = S.sample,
                 WS.listw, Durbin = T)

predictor_out_sample <- data.frame(
y_OLS = predict(res_lm_aragon, newdata = O.sample),
y_SLX = predict(res_slx_aragon, newdata = df_data, listw = neigh.listw)[outsamp],
y_sar_BP = as.numeric(predict.sarlm(res_sar_aragon, listw = neigh.listw,
                                newdata = O.sample, zero.policy = T,
                                pred.type = "BP", power = F)),
y_sar_BP1 = as.numeric(predict.sarlm(res_sar_aragon, listw = neigh.listw,
                                 newdata = O.sample, zero.policy = T,
                                 pred.type = "BP1", power = F)),
y_sar_TC = as.numeric(predict.sarlm(res_sar_aragon, listw = neigh.listw,
                                newdata = O.sample, zero.policy = T,
                                pred.type = "TC", power = F)),
y_sar_TC1 = as.numeric(predict.sarlm(res_sar_aragon, listw = neigh.listw,
                                newdata = O.sample, zero.policy = T,
                                pred.type = "TC1", power = F)),
y_sar_TS1 = as.numeric(predict.sarlm(res_sar_aragon, listw = neigh.listw,
                                 newdata = O.sample, zero.policy = T,
                                 pred.type = "TS1", power = F)),
y_sar_BPW = as.numeric(predict.sarlm(res_sar_aragon, listw = neigh.listw,
                                 newdata = O.sample, zero.policy = T,
                                 pred.type = "BPW", power = F)),
y_sar_BPN = as.numeric(predict.sarlm(res_sar_aragon, listw = neigh.listw,
                                 newdata = O.sample, zero.policy = T,
                                 pred.type = "BPN", power = F)),
y_sar_BPW1 = as.numeric(predict.sarlm(res_sar_aragon, listw = neigh.listw,
                                  newdata = O.sample, zero.policy = T,
                                  pred.type = "BPW1", power = F)),
y_sar_BPN1 = as.numeric(predict.sarlm(res_sar_aragon, listw = neigh.listw,
                                  newdata = O.sample, zero.policy = T,
                                  pred.type = "BPN1", power = F)),
y_sar_KP2 = as.numeric(predict.sarlm(res_sar_aragon, listw = neigh.listw,
                                  newdata = O.sample, zero.policy = T,
                                  pred.type = "KP2", power = F)),
y_sar_KP3 = as.numeric(predict.sarlm(res_sar_aragon, listw = neigh.listw,
                                  newdata = O.sample, zero.policy = T,
                                  pred.type = "KP3", power = F)),
y_sdm_BP = as.numeric(predict.sarlm(res_sdm_aragon, listw = neigh.listw,
                                newdata = O.sample, zero.policy = T,
                                pred.type = "BP", power = F)),
y_sdm_BP1 = as.numeric(predict.sarlm(res_sdm_aragon, listw = neigh.listw,
                                 newdata = O.sample, zero.policy = T,
                                 pred.type = "BP1", power = F)),
y_sdm_TC = as.numeric(predict.sarlm(res_sdm_aragon, listw = neigh.listw,
                                newdata = O.sample, zero.policy = T,
                                pred.type = "TC", power = F)),
y_sdm_TC1 = as.numeric(predict.sarlm(res_sdm_aragon, listw = neigh.listw,
                                newdata = O.sample, zero.policy = T,
                                pred.type = "TC1", power = F)),
```

```r
  y_sdm_TS1 = as.numeric(predict.sarlm(res_sdm_aragon, listw = neigh.listw,
                                 newdata = O.sample, zero.policy = T,
                                 pred.type = "TS1", power = F)),
  y_sdm_BPW = as.numeric(predict.sarlm(res_sdm_aragon, listw = neigh.listw,
                                 newdata = O.sample, zero.policy = T,
                                 pred.type = "BPW", power = F)),
  y_sdm_BPN = as.numeric(predict.sarlm(res_sdm_aragon, listw = neigh.listw,
                                 newdata = O.sample, zero.policy = T,
                                 pred.type = "BPN", power = F)),
  y_sdm_BPW1 = as.numeric(predict.sarlm(res_sdm_aragon, listw = neigh.listw,
                                  newdata = O.sample, zero.policy = T,
                                  pred.type = "BPW1", power = F)),
  y_sdm_BPN1 = as.numeric(predict.sarlm(res_sdm_aragon, listw = neigh.listw,
                                  newdata = O.sample, zero.policy = T,
                                  pred.type = "BPN1", power = F)),
  y_sdm_KP2 = as.numeric(predict.sarlm(res_sdm_aragon, listw = neigh.listw,
                                  newdata = O.sample, zero.policy = T,
                                  pred.type = "KP2", power = F)),
  y_sdm_KP3 = as.numeric(predict.sarlm(res_sdm_aragon, listw = neigh.listw,
                                  newdata = O.sample, zero.policy = T,
                                  pred.type = "KP3", power = F))

  )
  se[, k] <- apply(predictor_out_sample, 2, function(x) sum((x - O.sample$chomage)^2))

}
```

Finally, we rank the average MSE of the out-sample predictions in a decreasing order of efficiency:

```r
sort(apply(se, 1, sum)/nrow(df_data))
```

```
##    y_sdm_BP   y_sdm_BPN   y_sdm_BPW  y_sdm_BPW1   y_sdm_BP1  y_sdm_BPN1    y_sar_BP
##    1.245179    1.257696    1.259609    1.276387    1.282166    1.288134    1.295502
##   y_sar_BPN  y_sar_BPN1   y_sar_BP1   y_sar_BPW  y_sar_BPW1   y_sar_KP2   y_sar_KP3
##    1.299021    1.301341    1.309123    1.309353    1.313020    1.847560    1.864566
##   y_sdm_KP2   y_sdm_KP3    y_sdm_TC   y_sdm_TC1       y_SLX    y_sar_TC   y_sar_TC1
##    1.881211    1.896279    1.948042    2.014757    2.095120    2.203894    2.235359
##       y_OLS   y_sar_TS1   y_sdm_TS1
##    2.355478    2.370627    2.472252
```

7 References

Anselin, L. (1995). The Local Indicators of Spatial Association LISA. *Geographical Analysis*, **27**: 93–115.

Aragon Y., Haughton D., Haughton J., Leconte E., Malin E., Ruiz-Gazen A. and Thomas-Agnna C. (2003). Explaining the pattern of regional unemployment: The case of the Midi-Pyrénées region. *PIRS*, **82**(2).

Floch J.M and Le Saout R (2018). Spatial econometrics - common models. In V. Loonis (Ed) “Handbook of Spatial Analysis”. Insee Méthodes **131**.

Goulard M., Laurent T., and Thomas-Agnna C. (2017) About predictions in spatial autoregressive models: optimal and almost optimal strategies. *Spatial Economic Analysis*, **12**.