Spatial Econometrics

Dani Arribas-Bel

2016-04-20

This session[1] is based on the following references, which are good follow-up's on the topic:

- Session III of Arribas-Bel (2014). Check the "Related readings" section on the session page for more in-depth discussions.
- Anselin (2007), freely available to download [pdf].
- The second part of this tutorial assumes you have reviewed Lecture 5 of Arribas-Bel (2016). [html]

This tutorial is part of Spatial Analysis Notes, a compilation hosted as a GitHub repository that you can access it in a few ways:

- As a download of a `.zip` file that contains all the materials.
- As an html website.
- As a pdf document
- As a GitHub repository.

Dependencies

The illustration below relies on the following libraries that you will need to have installed on your machine to be able to interactively follow along[2]. Once installed, load them up with the following commands:

```
# Layout
library(tufte)
# For pretty table
library(knitr)
# Spatial Data management
library(rgdal)
# Pretty graphics
library(ggplot2)
# Pretty maps
library(ggmap)
# Various GIS utilities
library(GISTools)
# For all your interpolation needs
library(gstat)
# For data manipulation
library(plyr)
```


[2] You can install package `mypackage` by running the command `install.packages("mypackage")` on the R prompt or through the `Tools --> Install Packages...` menu in RStudio.

```
# Spatial regression
library(spdep)
```

Before we start any analysis, let us set the path to the directory where we are working. We can easily do that with `setwd()`. Please replace in the following line the path to the folder where you have placed this file -and where the `house_transactions` folder with the data lives.

```
setwd('/media/dani/baul/AAA/Documents/teaching/u-lvl/2016/envs453/code/GIT/kde_idw_r/')
#setwd('.')
```

Data

To explore ideas in spatial regression, we will be using house price data for the municipality of Liverpool. Our main dataset is provided by the Land Registry (as part of their Price Paid Data) but has been cleaned and re-packaged into a shapefile by Dani Arribas-Bel.

Let us load it up first of all:

```
hst <- readOGR(dsn = 'house_transactions', layer = 'liv_house_trans')
```

```
## OGR data source with driver: ESRI Shapefile
## Source: "house_transactions", layer: "liv_house_trans"
## with 6324 features
## It has 18 fields
```

```
## NOTE: rgdal::checkCRSArgs: no proj_defs.dat in PROJ.4 shared files
```

The tabular component of the spatial frame contains the followig variables:

```
names(hst)
```

```
##  [1] "pcds"       "id"         "price"
##  [4] "trans_date" "type"       "new"
##  [7] "duration"   "paon"       "saon"
## [10] "street"     "locality"   "town"
## [13] "district"   "county"     "ppd_cat"
## [16] "status"     "lsoa11"     "LSOA11CD"
```

The meaning for most of the variables can be found in the original Land Registry documentation. The dataset contains transactions that took place during 2,014:

```
# Format dates
dts <- as.Date(hst@data$trans_date)
```

```
# Set up summary table
tab <- summary(dts)[c('Min.', 'Max.')]
tab
```

```
##         Min.         Max.
## "2014-01-02" "2014-12-30"
```

Although the original Land Registry data contain some characteristics of the house, all of them are categorical: *is the house newly built? What type of property is it?* To bring in a richer picture and illustrate how continuous variables can also be included in a spatial setting, we will augment the original transaction data with Deprivation indices from the CDRC at the Lower Layer Super Output Area (LSOA) level.

Let us read the `csv` in:

```
imd <- read.csv('house_transactions/E08000012.csv')
```

The table contains not only the overall IMD score and rank, but some of the component scores, as well as the `LSOA` code:

```
names(imd)
```

```
##  [1] "LSOA11CD"   "imd_rank"   "imd_score"
##  [4] "income"     "employment" "education"
##  [7] "health"     "crime"      "housing"
## [10] "living_env" "idaci"      "idaopi"
```

That bit of information, `LSOA11CD`, is crucial to be able to connect it to each house transaction. To "join" both tables, we can use the base command `merge`, which will assign values from `imd` into `hst` making sure that each house transaction get the IMD data for the LSOA where it is located:

```
db <- merge(hst, imd)
```

The resulting table, `db`, contains variables from both original tables:

```
names(db)
```

```
##  [1] "LSOA11CD"   "pcds"       "id"
##  [4] "price"      "trans_date" "type"
##  [7] "new"        "duration"   "paon"
## [10] "saon"       "street"     "locality"
## [13] "town"       "district"   "county"
## [16] "ppd_cat"    "status"     "lsoa11"
## [19] "imd_rank"   "imd_score"  "income"
## [22] "employment" "education"  "health"
## [25] "crime"      "housing"    "living_env"
## [28] "idaci"      "idaopi"
```

For some of our analysis, we will need the coarse postcode of each house, rather than the finely specified one in the original data. This means using the available one

```
head(db@data['pcds'])
```

```
##      pcds
## 62 L1 0AB
## 63 L1 0AB
## 64 L1 0AB
## 65 L1 0AB
## 66 L1 0AB
## 67 L1 0AB
```

to create a new column that only contains the first bit of the postcode (`L1` in the examples above). The following lines of code will do that for us:

```
db$pc <- as.character(lapply(strsplit(as.character(db$pcds), split=" "),
```

Given there are 6,324 transactions in the dataset, a simple plot of the point coordinates implicitly draws the shape of the Liverpool municipality:

```
plot(db)
```

Figure 1: Spatial distribution of house transactions in Liverpool

Non-spatial regression, a refresh

Before we discuss how to explicitly include space into the linear regression framework, let us show how basic regression can be carried out in R, and how you can begin to interpret the results. By no means is this a formal and complete introduction to regression so, if that is what you are looking for, I suggest the first part of Gelman and Hill (2006), in particular chapters 3 and 4.

The core idea of linear regression is to explain the variation in a given (*dependent*) variable as a linear function of a series of other (*explanatory*) variables. For example, in our case, we may want to express/explain the price of a house as a function of whether it is new and the degree of deprivation of the area where it is located. At the individual level, we can express this as:

$$P_i = \alpha + \beta_1 NEW_i + \beta_2 IMD_i + \epsilon_i$$

where P_i is the price of house i, NEW_i is a binary variable that takes one if the house is newly built or zero otherwise and IMD_i is the IMD score of the LSOA where i is located. The parameters β_1, β_2,

and β_3 give us information about in which way and to what extent each variable is related to the price, and α, the constant term, is the average house price when all the other variables are zero. The term ϵ_i is usually referred to as "error" and captures elements that influence the price of a house but are not whether the house is new or the IMD score of its area. We can also express this relation in matrix form, excluding subindices for i[3].

Essentially, a regression can be seen as a multivariate extension of simple bivariate correlations. Indeed, one way to interpret the β_k coefficients in the equation above is as the degree of correlation between the explanatory variable k and the dependent variable, *keeping all the other explanatory variables constant*. When you calculate simple bivariate correlations, the coefficient of a variable is picking up the correlation between the variables, but it is also subsuming into it variation associated with other correlated variables –also called confounding factors[4]. Regression allows you to isolate the distinct effect that a single variable has on the dependent one, once we *control* for those other variables.

Practically speaking, running linear regressions in `R` is straightforward. For example, to fit the model specified in the equation above, we only need one line of code:

```
m1 <- lm('price ~ new + imd_score', db)
```

We use the command `lm`, for linear model, and specify the equation we want to fit using a string that relates the dependent variable (`price`) with a set of explanatory ones (`new` and `price`) by using a tilde ~ that is akin the = symbol in the mathematical equation. Since we are using names of variables that are stored in a table, we need to pass the table object (`db`) as well.

In order to inspect the results of the model, the quickest way is to call `summary`:

```
summary(m1)
```

```
##
## Call:
## lm(formula = "price ~ new + imd_score", data = db)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
##  -184254   -59948   -29032    11430 26434741
##
## Coefficients:
##             Estimate Std. Error t value
## (Intercept)   235596      13326  17.679
```

[3] In this case, the equation would look like

$$P = \alpha + \beta_1 NEW + \beta_2 IMD + \epsilon$$

and would be interpreted in terms of vectors and matrices instead of scalar values.

[4] **EXAMPLE** Assume that new houses tend to be built more often in areas with low deprivation. If that is the case, then NEW and IMD will be correlated with each other (as well as with the price of a house, as we are hypothesizing in this case). If we calculate a simple correlation between P and IMD, the coefficient will represent the degree of association between both variables, but it will also include some of the association between IMD and NEW. That is, part of the obtained correlation coefficient will be due not to the fact that higher prices tend to be found in areas with low IMD, but to the fact that new houses tend to be more expensive. This is because (in this example) new houses tend to be built in areas with low deprivation and simple bivariate correlation cannot account for that.

```
## newY            4926      19104   0.258
## imd_score      -2416        308  -7.843
##             Pr(>|t|)
## (Intercept)  < 2e-16 ***
## newY           0.797
## imd_score   5.12e-15 ***
## ---
## Signif. codes:
##   0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1  ' ' 1
##
## Residual standard error: 509000 on 6321 degrees of freedom
## Multiple R-squared:  0.009712,   Adjusted R-squared:  0.009398
## F-statistic: 30.99 on 2 and 6321 DF,  p-value: 4.027e-14
```

A full detailed explanation of the output is beyond the scope of this note, so we will focus on the relevant bits for our main purpose. This is concentrated on the `Coefficients` section, which gives us the estimates for the β_k coefficients in our model. Or, in other words, the coefficients are the raw equivalent of the correlation coefficient between each explanatory variable and the dependent one, once the polluting effect of confounding factors has been accounted for[5]. Results are as expected for the most part: houses tend to be significantly more expensive in areas with lower deprivation (an average of GBP2,416 for every additional score); and a newly built house is on average GBP4,926 more expensive, although this association cannot be ruled out to be random (probably due to the small relative number of new houses).

[5] Keep in mind that regression is no magic. We are only discounting the effect of other confounding factors that we include in the model, not of *all* potentially confounding factors.

Finally, before we jump into introducing space in our models, let us modify our equation slightly to make it more useful when it comes to interpretating it. Many house price models in the literature is estimated in log-linear terms:

$$\log P_i = \alpha + \beta_1 NEW_i + \beta_2 IMD_i + \epsilon_i$$

This allows to interpret the coefficients more directly: as the percentual change induced by a unit increase in the explanatory variable of the estimate. To fit such a model, we can specify the logarithm of a given variable directly in the formula.

```
m2 <- lm('log(price) ~ new + imd_score', db)
summary(m2)
```

```
##
## Call:
## lm(formula = "log(price) ~ new + imd_score", data = db)
##
```

```
## Residuals:
##     Min      1Q  Median      3Q     Max
## -4.3172 -0.3231 -0.0149  0.3063  5.2769
##
## Coefficients:
##               Estimate Std. Error t value
## (Intercept) 12.2037784  0.0138634  880.28
## newY         0.2456446  0.0198740   12.36
## imd_score   -0.0169702  0.0003204  -52.96
##             Pr(>|t|)
## (Intercept)   <2e-16 ***
## newY          <2e-16 ***
## imd_score     <2e-16 ***
## ---
## Signif. codes:
##   0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1  ' ' 1
##
## Residual standard error: 0.5295 on 6321 degrees of freedom
## Multiple R-squared:  0.3101, Adjusted R-squared:  0.3099
## F-statistic:  1421 on 2 and 6321 DF,  p-value: < 2.2e-16
```

Looking at the results we can see a couple of differences with respect to the original specification. First, the estimates are substantially different numbers. This is because, although they consider the same variable, the look at it from different angles, and provide different interpretations. For example, the coefficient for the IMD, instead of being interpretable in terms of GBP, the unit of the dependent variable, it represents a percentage: a unit increase in the degree of deprivation is associated with a 0.2% decrease in the price of a house.[6] Second, the variable `new` is significant in this case. This is probably related to the fact that, by taking logs, we are also making the dependent variable look more normal (Gaussian) and that allows the linear model to provide a better fit and, hence, more accurate estimates. In this case, a house being newly built, as compared to an old house, is overall 25% more expensive.

[6] **EXERCISE** *How does the type of a house affect the price at which it is sold, given whether it is new and the level of deprivation of the area where it is located?* To answer this, fit a model as we have done but including additionally the variable `type`. In order to interpret the codes, check the reference at the Land Registry documentation.

Spatial regression: a (very) first dip

Spatial regression is about *explicitly* introducing space or geographical context into the statistical framework of a regression. Conceptually, we want to introduce space into our model whenever we think it plays an important role in the process we are interested in, or when space can act as a reasonable proxy for other factors we cannot but should include in our model. As an example of the former, we can

imagine how houses at the seafront are probably more expensive than those in the second row, given their better views. To illustrate the latter, we can think of how the character of a neighborhood is important in determining the price of a house; however, it is very hard to identify and quantify "character" perse, although it might be easier to get at its spatial variation, hence a case of space as a proxy.

Spatial regression is a large field of development in the econometrics and statistics literatures. In this brief introduction, we will consider two related but very different processes that give rise to spatial effects: spatial heterogeneity and spatial dependence. For more rigorous treatments of the topics introduced here, the reader is referred to Anselin (2003), Anselin and Rey (2014), and Gibbons, Overman, and Patacchini (2014).

Spatial heterogeneity

Spatial heterogeneity (SH) arises when we cannot safely assume the process we are studying operates under the same "rules" throughout the geography of interest. In other words, we can observe SH when there are effects on the outcome variable that are intrinsically linked to specific locations. A good example of this is the case of seafront houses above: we are trying to model the price of a house and, the fact some houses are located under certain conditions (i.e. by the sea), makes their price behave differently[7].

This somewhat abstract concept of SH can be made operational in a model in several ways. We will explore the following two: spatial fixed-effects (FE); and spatial regimes, which is a generalization of FE.

[7] **QUESTION** How would you incorporate this into a regression model that extends the log-log equation of the previous section?

Spatial FE

Let us consider the house price example from the previous section to introduce a more general illustration that relates to the second motivation for spatial effects ("space as a proxy"). Given we are only including two explanatory variables in the model, it is likely we are missing some important factors that play a role at determining the price at which a house is sold. Some of them, however, are likely to vary systematically over space (e.g. different neighborhood characteristics). If that is the case, we can control for those unobserved factors by using traditional dummy variables but basing their creation on a spatial rule. For example, let us include a binary variable for every two-digit postcode in Liverpool, indicating whether a given house is located within such area (`1`) or not (`0`). Mathematically, we are now fitting the following equation:

$$\log P_i = \alpha_r + \beta_1 NEW_i + \beta_2 IMD_i + \epsilon_i$$

where the main difference is that we are now allowing the constant term, α, to vary by postcode r, α_r.

Programmatically, this is straightforward to estimate:

```
# Include '-1' to eliminate the constant term and include a dummy for every area
m3 <- lm('log(price) ~ pc + new + imd_score - 1', db)
summary(m3)
```

```
##
## Call:
## lm(formula = "log(price) ~ pc + new + imd_score - 1", data = db)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -4.2680 -0.2833 -0.0235  0.2599  5.6714
##
## Coefficients:
##            Estimate Std. Error t value
## pcL1      11.697166   0.032137  363.98
## pcL10     11.893524   0.076471  155.53
## pcL11     11.902319   0.043194  275.55
## pcL12     12.065776   0.029692  406.37
## pcL13     11.865523   0.034893  340.06
## pcL14     11.920922   0.044480  268.01
## pcL15     12.039916   0.028555  421.64
## pcL16     12.295535   0.034950  351.81
## pcL17     12.275339   0.027978  438.75
## pcL18     12.396475   0.024534  505.27
## pcL19     12.162630   0.029697  409.56
## pcL2      12.061443   0.100805  119.65
## pcL20     11.928142   0.226932   52.56
## pcL24     11.868363   0.045785  259.22
## pcL25     12.234462   0.028557  428.42
## pcL27     12.035241   0.092832  129.65
## pcL28     11.438267   0.206323   55.44
## pcL3      11.954453   0.029561  404.40
## pcL4      11.718609   0.039575  296.11
## pcL5      12.037267   0.055952  215.14
## pcL6      11.898506   0.042918  277.24
## pcL7      11.855374   0.044681  265.33
## pcL8      12.034093   0.039801  302.36
## pcL9      11.759056   0.033610  349.87
```

```
## newY       0.310829   0.020947   14.84
## imd_score -0.012008   0.000517  -23.23
##           Pr(>|t|)
## pcL1        <2e-16 ***
## pcL10       <2e-16 ***
## pcL11       <2e-16 ***
## pcL12       <2e-16 ***
## pcL13       <2e-16 ***
## pcL14       <2e-16 ***
## pcL15       <2e-16 ***
## pcL16       <2e-16 ***
## pcL17       <2e-16 ***
## pcL18       <2e-16 ***
## pcL19       <2e-16 ***
## pcL2        <2e-16 ***
## pcL20       <2e-16 ***
## pcL24       <2e-16 ***
## pcL25       <2e-16 ***
## pcL27       <2e-16 ***
## pcL28       <2e-16 ***
## pcL3        <2e-16 ***
## pcL4        <2e-16 ***
## pcL5        <2e-16 ***
## pcL6        <2e-16 ***
## pcL7        <2e-16 ***
## pcL8        <2e-16 ***
## pcL9        <2e-16 ***
## newY        <2e-16 ***
## imd_score   <2e-16 ***
## ---
## Signif. codes:
##   0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1  ' ' 1
##
## Residual standard error: 0.5007 on 6298 degrees of freedom
## Multiple R-squared:  0.9981, Adjusted R-squared:  0.9981
## F-statistic: 1.305e+05 on 26 and 6298 DF,  p-value: < 2.2e-16
```

Econometrically speaking, what the postcode FE we have introduced imply is that, instead of comparing all house prices across Liverpool as equal, we only derive variation from within each postcode[8]. Remember that the interpretation of a β_k coefficient is the effect of variable k, *given all the other explanatory variables included remain constant*. By including a single variable for each area, we are effectively forcing the model to compare as equal only house prices that share

[8] Additionally, estimating spatial FE in our particular example also gives you an indirect measure of area *desirability*: since they are simple dummies in a regression explaining the price of a house, their estimate tells us about how much people are willing to pay to live in a given area. However, this interpretation does not necessarily apply in other contexts where introducing spatial FEs does make sense. **EXERCISE** *What is the most desired area to live in Liverpool?*

the same value for each variable; in other words, only houses located within the same area. Introducing FE affords you a higher degree of isolation of the effects of the variables you introduce in your model because you can control for unobserved effects that align spatially with the distribution of the FE you introduce (by postcode, in our case).

Spatial regimes

At the core of estimating spatial FEs is the idea that, instead of assuming the dependent variable behaves uniformly over space, there are systematic effects following a geographical pattern that affect its behaviour. In other words, spatial FEs introduce econometrically the notion of spatial heterogeneity. They do this in the simplest possible form: by allowing the constant term to vary geographically. The other elements of the regression are left untouched and hence apply uniformly across space. The idea of spatial regimes (SRs) is to generalize the spatial FE approach to allow not only the constant term to vary but also any other explanatory variable. This implies that the equation we will be estimating is:

$$\log P_i = \alpha_r + \beta_{1r} NEW_i + \beta_{2r} IMD_i + \epsilon_i$$

where we are not only allowing the constant term to vary by region (α_r), but also every other parameter (β_{kr}).

In R terms, this is more straightforward to estimate if `new` is expressed as `0` and `1`, rather than as factors:

```
# Create a new variable ‘newB‘ to store the binary form of ‘new‘
db@data$newB <- 1
db[db@data$new=='N', 'newB'] <- 0
```

Also, given we are going to allow *every* coefficient to vary by regime, we will need to explicitly set a constant term that we can allow to vary:

```
db$one <- 1
```

Then, the estimation leverages the capabilities in model description of R formulas:

```
# ‘:‘ notation implies interaction variables
m4 <- lm('log(price) ~ 0 +(one + newB + imd_score):(pc)', db)
summary(m4)
```

```
##
## Call:
```

```
## lm(formula = "log(price) ~ 0 +(one + newB + imd_score):(pc)",
##     data = db)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -4.2051 -0.2506 -0.0182  0.2198  5.3507
##
## Coefficients: (8 not defined because of singularities)
##                  Estimate Std. Error t value
## one:pcL1        11.983460   0.098675 121.444
## one:pcL10       11.911281   0.504210  23.624
## one:pcL11       11.774865   0.160864  73.198
## one:pcL12       12.157627   0.068165 178.356
## one:pcL13       11.821576   0.103946 113.728
## one:pcL14       11.821895   0.115040 102.763
## one:pcL15       12.317524   0.052051 236.645
## one:pcL16       12.761711   0.098709 129.287
## one:pcL17       12.240652   0.065074 188.103
## one:pcL18       12.695103   0.056510 224.654
## one:pcL19       12.489604   0.054177 230.533
## one:pcL2        12.164848   0.321558  37.831
## one:pcL20       11.069199   0.211083  52.440
## one:pcL24       10.876938   0.135326  80.376
## one:pcL25       12.427878   0.051147 242.982
## one:pcL27       11.022835   0.435730  25.297
## one:pcL28       11.300143   1.126387  10.032
## one:pcL3        11.977708   0.054781 218.647
## one:pcL4        11.653464   0.109113 106.802
## one:pcL5        11.544085   0.277000  41.675
## one:pcL6        11.574617   0.158022  73.247
## one:pcL7        11.617544   0.128402  90.478
## one:pcL8        11.586224   0.085036 136.251
## one:pcL9        11.837544   0.080452 147.139
## newB:pcL1       -0.298430   0.051078  -5.843
## newB:pcL10             NA         NA      NA
## newB:pcL11       0.672479   0.089894   7.481
## newB:pcL12       0.977903   0.114281   8.557
## newB:pcL13      -0.659910   0.473773  -1.393
## newB:pcL14       0.610482   0.132911   4.593
## newB:pcL15       0.744844   0.139732   5.331
## newB:pcL16             NA         NA      NA
## newB:pcL17      -0.094580   0.095188  -0.994
## newB:pcL18      -0.197115   0.122109  -1.614
## newB:pcL19       0.393901   0.089607   4.396
```

```
## newB:pcL2              NA         NA      NA
## newB:pcL20             NA         NA      NA
## newB:pcL24       0.735316   0.072739  10.109
## newB:pcL25       0.196018   0.144527   1.356
## newB:pcL27       0.356678   0.189946   1.878
## newB:pcL28             NA         NA      NA
## newB:pcL3       -0.261461   0.055228  -4.734
## newB:pcL4        1.087380   0.079152  13.738
## newB:pcL5        0.519945   0.094850   5.482
## newB:pcL6        0.695856   0.062461  11.141
## newB:pcL7              NA         NA      NA
## newB:pcL8        0.334517   0.059368   5.635
## newB:pcL9              NA         NA      NA
## imd_score:pcL1  -0.010489   0.003383  -3.101
## imd_score:pcL10 -0.012413   0.011381  -1.091
## imd_score:pcL11 -0.010637   0.003005  -3.540
## imd_score:pcL12 -0.017296   0.002625  -6.588
## imd_score:pcL13 -0.011004   0.002185  -5.036
## imd_score:pcL14 -0.010432   0.002469  -4.225
## imd_score:pcL15 -0.020737   0.001413 -14.681
## imd_score:pcL16 -0.050522   0.007703  -6.559
## imd_score:pcL17 -0.009763   0.002214  -4.410
## imd_score:pcL18 -0.032289   0.003799  -8.499
## imd_score:pcL19 -0.024629   0.001860 -13.242
## imd_score:pcL2  -0.016601   0.013654  -1.216
## imd_score:pcL20        NA         NA      NA
## imd_score:pcL24  0.004090   0.002391   1.711
## imd_score:pcL25 -0.020461   0.001981 -10.327
## imd_score:pcL27  0.006626   0.007561   0.876
## imd_score:pcL28 -0.009473   0.020367  -0.465
## imd_score:pcL3  -0.006935   0.001747  -3.970
## imd_score:pcL4  -0.012032   0.001731  -6.952
## imd_score:pcL5  -0.006051   0.004009  -1.509
## imd_score:pcL6  -0.008324   0.002367  -3.517
## imd_score:pcL7  -0.007517   0.002342  -3.209
## imd_score:pcL8  -0.004064   0.001561  -2.603
## imd_score:pcL9  -0.014124   0.002052  -6.882
##                 Pr(>|t|)
## one:pcL1         < 2e-16 ***
## one:pcL10        < 2e-16 ***
## one:pcL11        < 2e-16 ***
## one:pcL12        < 2e-16 ***
## one:pcL13        < 2e-16 ***
## one:pcL14        < 2e-16 ***
```

```
## one:pcL15        < 2e-16 ***
## one:pcL16        < 2e-16 ***
## one:pcL17        < 2e-16 ***
## one:pcL18        < 2e-16 ***
## one:pcL19        < 2e-16 ***
## one:pcL2         < 2e-16 ***
## one:pcL20        < 2e-16 ***
## one:pcL24        < 2e-16 ***
## one:pcL25        < 2e-16 ***
## one:pcL27        < 2e-16 ***
## one:pcL28        < 2e-16 ***
## one:pcL3         < 2e-16 ***
## one:pcL4         < 2e-16 ***
## one:pcL5         < 2e-16 ***
## one:pcL6         < 2e-16 ***
## one:pcL7         < 2e-16 ***
## one:pcL8         < 2e-16 ***
## one:pcL9         < 2e-16 ***
## newB:pcL1       5.40e-09 ***
## newB:pcL10            NA
## newB:pcL11      8.40e-14 ***
## newB:pcL12       < 2e-16 ***
## newB:pcL13      0.163705
## newB:pcL14      4.45e-06 ***
## newB:pcL15      1.01e-07 ***
## newB:pcL16            NA
## newB:pcL17      0.320449
## newB:pcL18      0.106521
## newB:pcL19      1.12e-05 ***
## newB:pcL2             NA
## newB:pcL20            NA
## newB:pcL24       < 2e-16 ***
## newB:pcL25      0.175063
## newB:pcL27      0.060457 .
## newB:pcL28            NA
## newB:pcL3       2.25e-06 ***
## newB:pcL4        < 2e-16 ***
## newB:pcL5       4.38e-08 ***
## newB:pcL6        < 2e-16 ***
## newB:pcL7             NA
## newB:pcL8       1.83e-08 ***
## newB:pcL9             NA
## imd_score:pcL1  0.001938 **
## imd_score:pcL10 0.275485
```

```
## imd_score:pcL11 0.000404 ***
## imd_score:pcL12 4.81e-11 ***
## imd_score:pcL13 4.89e-07 ***
## imd_score:pcL14 2.42e-05 ***
## imd_score:pcL15  < 2e-16 ***
## imd_score:pcL16 5.85e-11 ***
## imd_score:pcL17 1.05e-05 ***
## imd_score:pcL18  < 2e-16 ***
## imd_score:pcL19  < 2e-16 ***
## imd_score:pcL2  0.224084
## imd_score:pcL20       NA
## imd_score:pcL24 0.087139 .
## imd_score:pcL25  < 2e-16 ***
## imd_score:pcL27 0.380911
## imd_score:pcL28 0.641850
## imd_score:pcL3  7.28e-05 ***
## imd_score:pcL4  3.96e-12 ***
## imd_score:pcL5  0.131280
## imd_score:pcL6  0.000440 ***
## imd_score:pcL7  0.001337 **
## imd_score:pcL8  0.009255 **
## imd_score:pcL9  6.47e-12 ***
## ---
## Signif. codes:
##   0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1  ' ' 1
##
## Residual standard error: 0.472 on 6260 degrees of freedom
## Multiple R-squared:  0.9984, Adjusted R-squared:  0.9983
## F-statistic: 5.966e+04 on 64 and 6260 DF,  p-value: < 2.2e-16
```

As we can see, there are a few `NA` values (e.g. `pcL10`). This has to do with the fact that there are not that many new houses, so some of the buckets in which the regimes split the data to estimate each parameter are empty. This can be readily seen by obtaining a quick cross tabulation of `pc` and `new`:

```
table(db$pc, db$new)
```

```
##
##         N   Y
##   L1  161 189
##   L10  47   0
##   L11 192  34
##   L12 326  18
##   L13 387   1
```

```
##   L14 144  19
##   L15 466  12
##   L16 212   0
##   L17 398  27
##   L18 439  16
##   L19 329  32
##   L2   25   0
##   L20   5   0
##   L24  97  84
##   L25 357  11
##   L27  22  10
##   L28   6   0
##   L3  272 101
##   L4  437  41
##   L5   97  46
##   L6  319  78
##   L7  201   0
##   L8  227 110
##   L9  329   0
```

To illustrate a correct regime estimation, we can focus only on `imd_score`[9]:

[9] Note this still returns a `NA` for the IMD estimate in `L20`. This is most likely due to the little amount of houses (five) sold in that area. The regression nevertheless serves the illustration

```
# ':' notation implies interaction variables
m5 <- lm('log(price) ~ 0 + (one + imd_score):pc', db)
summary(m5)
```

```
##
## Call:
## lm(formula = "log(price) ~ 0 + (one + imd_score):pc", data = db)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -4.4952 -0.2779 -0.0221  0.2477  5.4973
##
## Coefficients: (1 not defined because of singularities)
##                   Estimate Std. Error
## one:pcL1        11.8965065  0.1030279
## one:pcL10       11.9112807  0.5325447
## one:pcL11       11.6203268  0.1684973
## one:pcL12       12.2106579  0.0716973
## one:pcL13       11.8099973  0.1094358
## one:pcL14       12.1229804  0.0998496
## one:pcL15       12.3753226  0.0537697
## one:pcL16       12.7617106  0.1042555
```

```
## one:pcL17       12.2248054  0.0666348
## one:pcL18       12.7033951  0.0594382
## one:pcL19       12.4767606  0.0571384
## one:pcL2        12.1648479  0.3396283
## one:pcL20       11.0691989  0.2229447
## one:pcL24       11.5214275  0.1260747
## one:pcL25       12.4351502  0.0537240
## one:pcL27       11.3770205  0.4148632
## one:pcL28       11.3001428  1.1896847
## one:pcL3        11.8873513  0.0542344
## one:pcL4        11.4097020  0.1137109
## one:pcL5        10.9973874  0.2729459
## one:pcL6        11.1816689  0.1626918
## one:pcL7        11.6175445  0.1356173
## one:pcL8        11.5086161  0.0886286
## one:pcL9        11.8375435  0.0849726
## imd_score:pcL1  -0.0131286  0.0035407
## imd_score:pcL10 -0.0124128  0.0120210
## imd_score:pcL11 -0.0058373  0.0031009
## imd_score:pcL12 -0.0173734  0.0027727
## imd_score:pcL13 -0.0107902  0.0023022
## imd_score:pcL14 -0.0160870  0.0022604
## imd_score:pcL15 -0.0219205  0.0014734
## imd_score:pcL16 -0.0505217  0.0081354
## imd_score:pcL17 -0.0093980  0.0023061
## imd_score:pcL18 -0.0333885  0.0039476
## imd_score:pcL19 -0.0228257  0.0019160
## imd_score:pcL2  -0.0166013  0.0144213
## imd_score:pcL20         NA         NA
## imd_score:pcL24 -0.0020546  0.0024420
## imd_score:pcL25 -0.0205241  0.0020921
## imd_score:pcL27  0.0020943  0.0075687
## imd_score:pcL28 -0.0094733  0.0215112
## imd_score:pcL3  -0.0061811  0.0018374
## imd_score:pcL4  -0.0066455  0.0017803
## imd_score:pcL5   0.0039338  0.0037726
## imd_score:pcL6  -0.0004620  0.0023860
## imd_score:pcL7  -0.0075165  0.0024737
## imd_score:pcL8  -0.0006921  0.0015228
## imd_score:pcL9  -0.0141241  0.0021676
##                 t value Pr(>|t|)
## one:pcL1        115.469  < 2e-16 ***
## one:pcL10        22.367  < 2e-16 ***
## one:pcL11        68.964  < 2e-16 ***
```

```
## one:pcL12       170.309  < 2e-16 ***
## one:pcL13       107.917  < 2e-16 ***
## one:pcL14       121.412  < 2e-16 ***
## one:pcL15       230.154  < 2e-16 ***
## one:pcL16       122.408  < 2e-16 ***
## one:pcL17       183.460  < 2e-16 ***
## one:pcL18       213.724  < 2e-16 ***
## one:pcL19       218.360  < 2e-16 ***
## one:pcL2         35.818  < 2e-16 ***
## one:pcL20        49.650  < 2e-16 ***
## one:pcL24        91.386  < 2e-16 ***
## one:pcL25       231.464  < 2e-16 ***
## one:pcL27        27.424  < 2e-16 ***
## one:pcL28         9.498  < 2e-16 ***
## one:pcL3        219.185  < 2e-16 ***
## one:pcL4        100.340  < 2e-16 ***
## one:pcL5         40.291  < 2e-16 ***
## one:pcL6         68.729  < 2e-16 ***
## one:pcL7         85.664  < 2e-16 ***
## one:pcL8        129.852  < 2e-16 ***
## one:pcL9        139.310  < 2e-16 ***
## imd_score:pcL1   -3.708 0.000211 ***
## imd_score:pcL10  -1.033 0.301837
## imd_score:pcL11  -1.882 0.059824 .
## imd_score:pcL12  -6.266 3.95e-10 ***
## imd_score:pcL13  -4.687 2.83e-06 ***
## imd_score:pcL14  -7.117 1.23e-12 ***
## imd_score:pcL15 -14.878  < 2e-16 ***
## imd_score:pcL16  -6.210 5.63e-10 ***
## imd_score:pcL17  -4.075 4.65e-05 ***
## imd_score:pcL18  -8.458  < 2e-16 ***
## imd_score:pcL19 -11.913  < 2e-16 ***
## imd_score:pcL2   -1.151 0.249707
## imd_score:pcL20      NA       NA
## imd_score:pcL24  -0.841 0.400163
## imd_score:pcL25  -9.810  < 2e-16 ***
## imd_score:pcL27   0.277 0.782014
## imd_score:pcL28  -0.440 0.659671
## imd_score:pcL3   -3.364 0.000773 ***
## imd_score:pcL4   -3.733 0.000191 ***
## imd_score:pcL5    1.043 0.297115
## imd_score:pcL6   -0.194 0.846480
## imd_score:pcL7   -3.039 0.002387 **
## imd_score:pcL8   -0.455 0.649469
```

```
## imd_score:pcL9   -6.516 7.79e-11 ***
## ---
## Signif. codes:
##   0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1  ' ' 1
##
## Residual standard error: 0.4985 on 6277 degrees of freedom
## Multiple R-squared:  0.9982, Adjusted R-squared:  0.9982
## F-statistic: 7.282e+04 on 47 and 6277 DF,  p-value: < 2.2e-16
```

This allows us to get a separate constant term and estimate of the impact of IMD on the price of a house *for every postcode*[10].

[10] **PRO EXERCISE** *How does the effect of IMD vary over space?* You can answer this by looking at the coefficients of `imd_score` over postcodes, but it would be much clearer if you could create a choropleth of the house locations where each dot is colored based on the value of the `imd_score` estimated for that postcode.

Spatial dependence

As we have just discussed, SH is about effects of phenomena that are *explicitly linked* to geography and that hence cause spatial variation and clustering of values. This encompasses many of the kinds of spatial effects we may be interested in when we fit linear regressions. However, in other cases, our interest is on the effect of the *spatial configuration* of the observations, and the extent to which that has an effect on the outcome we are considering. For example, we might think that the price of a house not only depends on the level of deprivation where the house is located, but also whether is close to other highly deprived areas. This kind of spatial effect is fundamentally different from SH in that is it not related to inherent characteristics of the geography but relates to the characteristics of the observations in our dataset and, specially, to their spatial arrangement. We call this phenomenon by which the values of observations are related to each other through distance *spatial dependence* (Anselin 1988).

Spatial Weights

There are several ways to introduce spatial dependence in an econometric framework, with varying degrees of econometric sophistication (see Anselin 2003 for a good overview). Common to all of them however is the way space is formally encapsulated: through *spatial weights matrices (W)*[11]. These are NxN matrices with zero diagonals and every w_{ij} cell with a value that represents the degree of spatial connectivity/interaction between observations i and j. If they are not connected at all, $w_{ij} = 0$, otherwise $w_{ij} > 0$ and we call i and j neighbors. The exact value in the latter case depends on the criterium we use to define neighborhood relations. These matrices also tend to be row-standardized so the sum of each row equals to one.

[11] If you need to refresh your knowledge on spatial weight matrices, check Lecture 5 of Arribas-Bel (2016)

A related concept to spatial weight matrices is that of *spatial lag*. This is an operator that multiplies a given variable y by a spatial

weight matrix:

$$y_{lag} = Wy$$

If W is row-standardized, y_{lag} is effectively the average value of y in the neighborhood of each observation. The individual notation may help clarify this:

$$y_{lag-i} = \sum_j w_{ij} y_j$$

where y_{lag-i} is the spatial lag of variable y at location i, and j sums over the entire dataset. If W is row-standardized, y_{lag-i} becomes an average of y weighted by the spatial criterium defined in W.

Given that spatial weights matrices are not the focus of this tutorial, we will stick to a very simple case. Since we are dealing with points, we will use K-nn weights, which take the k nearest neighbors of each observation as neighbors and assign a value of one, assigning everyone else a zero. We will use $k = 150$ to get a good degree of variation and sensible results. If your computer is struggles to compute the following lines of code, you can replace 50 by a lowed number. Technically speaking is the same thing, but the probability that you will pick up only houses in the same LSOA (and hence with exactly the same IMD score) will be higher.

```
# Because some rows are different units on the same house, slightly
# jitter the locations to break ties
xy.jit <- jitter(db@coords)
# Create knn list of each house
hnn <- knearneigh(xy.jit, k=50)
# Create nb object
hnb <- knn2nb(hnn)
# Create spatial weights matrix (note it row-standardizes by default)
hknn <- nb2listw(hnb)
```

We can inspect the weights created by simply typing the name of the object:

```
hknn
```

```
## Characteristics of weights list object:
## Neighbour list object:
## Number of regions: 6324
## Number of nonzero links: 316200
## Percentage nonzero weights: 0.7906388
## Average number of links: 50
## Non-symmetric neighbours list
```

```
##
## Weights style: W
## Weights constants summary:
##      n       nn   S0       S1       S2
## W 6324 39992976 6324 230.4664 25815.45
```

Exogenous spatial effects

Let us come back to the house price example we have been working with. So far, we have hypothesized that the price of a house sold in Liverpool can be explained using information about whether it is newly built, the level of deprivation of the area where it is located, and its postcode. However, it is also reasonable to think that prospective house owners care about the larger area around a house, not only about its immediate surroundings, and would be willing to pay more for a house that was close to nicer areas, everything else being equal. How could we test this idea?

The most straightforward way to introduce spatial dependence in a regression is by considering not only a given explanatory variable, but also its spatial lag. In our example case, in addition to including the level of deprivation in the area of the house, we will include its spatial lag. In other words, we will be saying that it is not only the level of deprivation of the area where a house is located but also that of the surrounding locations that helps explain the final price at which a house is sold. Mathematically, this implies estimating the following model:

$$\log P_i = \alpha + \beta_1 NEW_i + \beta_2 IMD_i + \beta_3 IMD_{lag-i} + \epsilon_i$$

Let us first compute the spatial lag of imd_score:

```
db@data$w_imd_score <- lag.listw(hknn, db@data$imd_score)
```

And then we can include it in our previous specification. Note that we apply the log to the lag, not the reverse:

```
# ':' notation implies interaction variables
m6 <- lm('log(price) ~ new + imd_score + w_imd_score', db)
summary(m6)
```

```
##
## Call:
## lm(formula = "log(price) ~ new + imd_score + w_imd_score", data = db)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
```

```
## -4.2905 -0.3016 -0.0153  0.2822  5.2605
##
## Coefficients:
##               Estimate Std. Error t value
## (Intercept) 12.2814016  0.0145267 845.437
## newY         0.2476839  0.0195171  12.691
## imd_score   -0.0041887  0.0008918  -4.697
## w_imd_score -0.0148387  0.0009687 -15.318
##             Pr(>|t|)
## (Intercept)  < 2e-16 ***
## newY         < 2e-16 ***
## imd_score    2.7e-06 ***
## w_imd_score  < 2e-16 ***
## ---
## Signif. codes:
##   0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1  ' ' 1
##
## Residual standard error: 0.52 on 6320 degrees of freedom
## Multiple R-squared:  0.3348, Adjusted R-squared:  0.3345
## F-statistic:  1060 on 3 and 6320 DF,  p-value: < 2.2e-16
```

As we can see, the lag is not only significative and negative (as expected), but its effect seems to be even larger that that of the house itself. Taken literally, this would imply that prospective owners value more the area of the surrounding houses than that of the actual house they buy. However, it is important to remember how these variables have been constructed and what they really represent. Because the IMD score is not exactly calculated at the house level, but at the area level, many of the surrounding houses will share that so, to some extent, the IMD of neighboring houses is that of the house itself[12]. This is likely to be affecting the final parameter, and it is a reminder and an illustration that we cannot take model results as universal truth but we need to use them as tools to inform analysis, couple with theory and what we know about the particular question of analysis. Nevertheless, the example does illustrate how to introduce spatial dependence in a regression framework in a fairly straight forward way.

[12] **EXERCISE** *How do results change if you modify the number of neighbors included to compute the K-nn spatial weight matrix?* Replace the originak k used and re-run the regressions. Try to interpret the results and the (potential) differences with the original ones.

A note on more advanced spatial regression

Introducing a spatial lag of an explanatory variable, as we have just seen, is the most straightforward way of incorporating the notion of spatial dependence in a linear regression framework. It does not require additional changes, it can be estimated with OLS, and the interpretation is rather similar to interpreting non-spatial variables. The

field of spatial econometrics however is a much broader one and has produced over the last decades many techniques to deal with spatial effects and spatial dependence in different ways. Although this might be an over simplification, one can say that most of such efforts for the case of a single cross-section are focused on two main variations: the spatial lag and the spatial error model. Both are similar to the case we have seen in that they are based on the introduction of a spatial lag, but they differ in the component of the model they modify and affect.

The spatial lag model introduces a spatial lag of the *dependent* variable. In the example we have covered, this would translate into:

$$\log P_i = \alpha + \rho \log P_{lag-i} + \beta_1 NEW_i + \beta_2 IMD_i + \epsilon_i$$

Although it might not seem very different from the previous equation, this model violates the exogeneity assumption, crucial for OLS to work.

Equally, the spatial error model includes a spatial lag in the *error* term of the equation:

$$\log P_i = \alpha + \beta_{1r} NEW_i + \beta_{2r} IMD_i + u_i$$

$$u_i = u_{lag-i} + \epsilon_i$$

Again, although similar, one can show this specification violates the assumptions about the error term in a classical OLS model.

Both the spatial lag and error model violate some of the assumptions on which OLS relies and thus render the technique unusable. Much of the efforts have thus focused on coming up with alternative methodologies that allow unbiased, robust, and efficient estimation of such models. A survey of those is beyond the scope of this note, but the interested reader is referred to Anselin (1988), Anselin (2003), and Anselin and Rey (2014) for further reference.

Predicting house prices

So far, we have seen how exploit the output of a regression model to evaluate the role different variables play in explaining another one of interest. However, once fit, a model can also be used to obtain predictions of the dependent variable given a new set of values for the explanatory variables. We will finish this session by dipping our toes in predicting with linear models.

The core idea is that once you have estimates for the way in which the explanatory variables can be combined to explain the dependent one, you can plug new values on the explanatory side of the model

and combine them following the model estimates to obtain predictions. In the example we have worked with, you can imagine this application would be useful to obtain valuations of a house, given we know the IMD of the area where the house is located and whether it is a newly built house or not.

Conceptually, predicting in linear regression models involves using the estimates of the parameters to obtain a value for the dependent variable:

$$\bar{\log} P_i = \bar{\alpha} + \bar{\beta_{1r}} NEW_i^* + \bar{\beta_{2r}} IMD_i^*$$

where $\bar{\log} P_i$ is our predicted value, and we include the $\bar{}$ sign to note that it is our estimate obtained from fitting the model. We use the * sign to note that those can be new values for the explanatory variables, not necessarily those used to fit the model.

Technically speaking, prediction in linear models is fairly streamlined in R. Suppose we are given data for a new house which is to be put in the market. We know it is been newly built on an area with an IMD score of 75, but surrounded by areas that, on average, have a score of 50. Let us record the data first:

```
new.house <- data.frame(new='Y', imd_score=75, w_imd_score=50)
```

To obtain the prediction for its price, we can use the `predict` method:

```
new.price <- predict(m6, new.house)
new.price
```

```
##      1
## 11.473
```

Now remember we were using the log of the price as dependent variable. If we want to recover the actual price of the house, we need to take its exponent:

```
exp(new.price)
```

```
##        1
## 96086.16
```

According to our model, the house would be worth GBP96,060.29[13].

[13] **EXERCISE** *How would the price change if the surrounding houses did not have an average of 50 but of 80?* Obtain a new prediction and compare it with the original one.

References

Anselin, Luc. 1988. *Spatial Econometrics: Methods and Models*. Vol. 4. Springer Science & Business Media.

———. 2003. "Spatial Externalities, Spatial Multipliers, and Spatial Econometrics." *International Regional Science Review* 26 (2). Sage Publications: 153–66.

———. 2007. "Spatial Regression Analysis in R–A Workbook." Center for Spatially Integrated Social Science. http://csiss.org/GISPopSci/workshops/2011/PSU/readings/W15_Anselin2007.pdf.

Anselin, Luc, and Sergio J. Rey. 2014. *Modern Spatial Econometrics in Practice: A Guide to GeoDa, GeoDaSpace and PySAL*. GeoDa Press LLC.

Arribas-Bel, Dani. 2014. "Spatial Data, Analysis, and Regression-a Mini Course." *REGION* 1 (1). European Regional Science Association: R1. http://darribas.org/sdar_mini.

———. 2016. "Geographic Data Science'15." doi:10.5281/zenodo.46313.

Gelman, Andrew, and Jennifer Hill. 2006. *Data Analysis Using Regression and Multilevel/hierarchical Models*. Cambridge University Press.

Gibbons, Stephen, Henry G Overman, and Eleonora Patacchini. 2014. "Spatial Methods." CEPR Discussion Paper No. DP10135.