Spatial Economics – Assignment 1

Gustav Pirich (h11742049@s.wu.ac.at)
Gabriel Konecny (h11775903@s.wu.ac.at)
Jan Trimmel (h11809096@s.wu.ac.at)

2024-04-02

Contents

Exercise A

1. Load the Boston dataset from the MASS package. Think of a simple (linear) model aiming at predicting property prices involving 4–5 covariates (or interactions between some of them). Create a function that takes your dependent variable and the covariates as inputs, and return a list with:

- OLS point estimates for the intercept, slope parameters, and the error variance.
- Suitable test statistics with corresponding p-values for the relevant coefficients.
- Intervals of the coefficients for a confidence level of 95%.

As independent variables, we use per capita crime rate by town, average number of rooms per dwelling, Charles River dummy variable (= 1 if tract bounds river; 0 otherwise), nitrogen oxides concentration (parts per 10 million) and a constant.

```r
data1 <- MASS::Boston
X <- as.matrix(cbind(1,data1[,c("crim", "rm","chas","nox")]))
#X <- as.matrix(cbind(1,data1["lstat"]))
colnames(X) <- c("Constant","Crime", "Rooms", "Charles_River", "NO_pp10m")
Y <- as.matrix(data1[,"medv"])
```

Below we created a function which implements: OLS point estimates for the intercept, slope parameters, and the error variance. Suitable test statistics with corresponding p-values for the relevant coefficients. Intervals of the coefficients for a confidence level of 95%.

```r
OLS <- function(X,Y){

# OLS estimates for coefficients
beta_hat <- solve(t(X) %*% X) %*% t(X) %*% Y
Y_hat <- X %*% beta_hat   # Fitted values

e <- Y - Y_hat            # residuals
n <- nrow(X)              # Number of observations
k <- ncol(X) - 1          # Number of covariates excluding intercept

s <- as.numeric(t(e)%*%e / (n-k))           # SSR adjusted for degrees of freedom
sigma <- s*solve(t(X) %*% X)                # VCV of Beta hat

se <- sqrt(diag(sigma))                     # standard error
t_stat <- (beta_hat-0) / se                 # Compute t-statistic
p <- pt(abs(t_stat), n-k, lower.tail=FALSE) # Compute p-value

# 95% Confidence interval
th <- qt(0.975, n-k)
conf <- cbind(beta_hat-th*se,beta_hat+th*se)

colnames(beta_hat) <- "estimate"
colnames(conf) <- c("2.5%","97.5%")
colnames(t_stat) <- "t-statistic"
colnames(p) <- "p-value"

error_variance <- s

list(rbind(beta_hat,error_variance), cbind(t_stat, p), conf)
}
```

The output of the function is presented below:

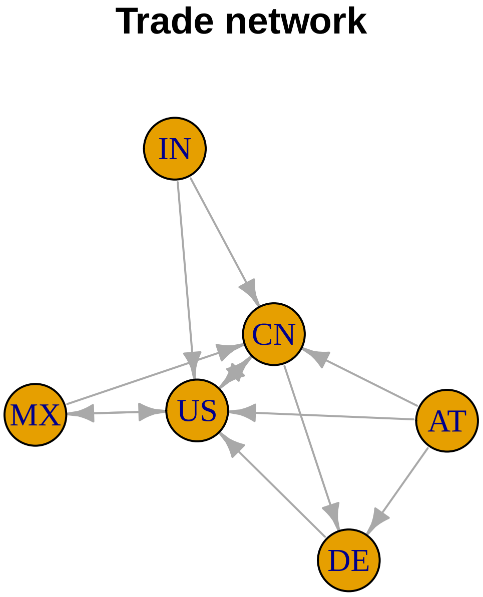
```
OLS(X,Y)
```

```
## [[1]]
##                   estimate
## Constant       -17.259625
## Crime           -0.184610
## Rooms            7.706840
## Charles_River    4.673807
## NO_pp10m       -14.960358
## error_variance  35.771378
##
## [[2]]
##                t-statistic      p-value
## Constant         -5.373025 5.935625e-08
## Crime            -5.358215 6.414580e-08
## Rooms            19.155659 4.233091e-62
## Charles_River     4.388052 6.974274e-06
## NO_pp10m         -5.674181 1.178869e-08
##
## [[3]]
##                       2.5%       97.5%
## Constant      -23.5707812 -10.9484684
## Crime          -0.2523012  -0.1169189
## Rooms           6.9163879   8.4972927
## Charles_River   2.5811627   6.7664511
## NO_pp10m      -20.1404236  -9.7802928
```

Exercise B

1. Draw a graph of the network; create the adjacency matrix in R.

We consider 6 different countries and an indicator of high trade intensity between them. For illustration, we could consider a rule where there is directed edge from A to B, if B is one of the top 5 export destinations of A. We don't actually check for biggest trading partners empirically, but lets assume that such procedure gives rise to following network (arrows are mostly made up to achieve 12 edges and nice interpretations):



The corresponding adjacency matrix is:

	US	MX	DE	AT	CN	IN
US	0	1	0	0	1	0
MX	1	0	0	0	1	0
DE	1	0	0	0	0	0
AT	1	0	1	0	1	0
CN	1	0	1	0	0	0
IN	1	0	0	0	1	0

2. Who are the most and least central agents in the network? Name, explain, and try to quantify different notions of centrality.

A very basic concept of centrality could define an agent as most central, if it has the highest number of directed edges pointing towards itself (i.e. if it is important export country for most other countries). Using this criterion we can see from the graph that or from columns of adjacency matrix that US is the most central agents in this network, because it is among the top 5 trading partner for all 5 other countries. The least central agents would be Austria and India, since they are not a top 5 exporting destination for any country in this network.

Another basic criterion of centrality could be the amount of outward pointing arrows of an agent. Thus, a country would be considered central if it exports a lot to a high number of countries in this network. In this sense, Germany is the least central with only 1 outward arrow, while Austria is the most central with 3 outward arrows. This criterion makes perhaps less sense than the first one, but we mention it to explain the concepts. Another possible centrality concept would be function of the two basic concepts above, e.g. sum of in- and outward pointing arrows of an agent.

In class we briefly discussed the intuition for eigenvector centrality. Eigenvector centrality measures a node's influence in a network by accounting not only for the number of its connections but also for the importance of those connected nodes. Unlike simpler measures that count direct connections, eigenvector centrality assigns higher scores to nodes linked to other highly scored nodes. This means a node is considered influential if it is connected to other influential nodes.

This method does not work if there are sinks. Many applied algorithms such as page rank introduce jump probabilities for sinks. Once in a sink, i.e. node which has no outward connections, a probability distribution of jumping to other nodes is specified. While defining and computing page rank would be perhaps beyond the scope of this small exercise, below we display the eigenvector centrality computed using the package igraph.

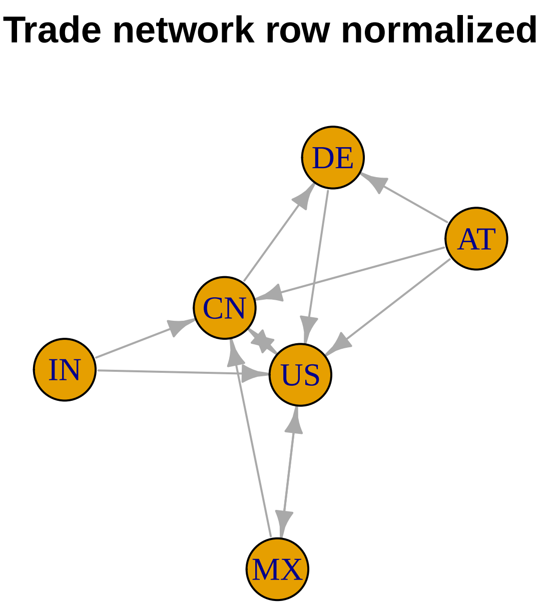
```r
eigvcen1 <- eigen_centrality(
  graph_from_adjacency_matrix(adj),
  directed = TRUE,
  scale = TRUE,
  weights = NULL,
  options = arpack_defaults
)

eigvcen1$vector
```

```
##        US        MX        DE        AT        CN        IN
## 1.0000000 0.5436890 0.4563110 0.0000000 0.8392868 0.0000000
```

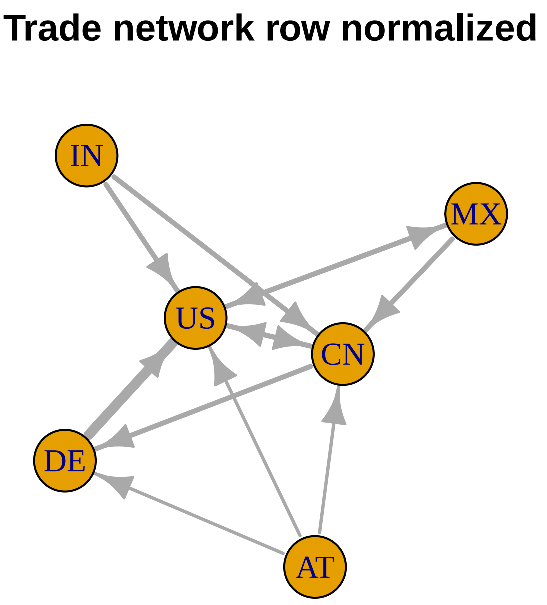
Using eigenvector centrality, We observe that US is the most central agent, followed by China. Austria and India are the least central agents with value of 0. This is because they are source only - there are no directed edges pointing to them, meaning that they are not a top 5 trading partner to any country.

- How would centralities change if you considered a row-normalized network instead?

Germany which had only 1 connection, is the only row with sum of 1. Thus if we don't use weights, all edges of nodes with more than 1 connection disappear. This would suggest depending on criterion that US or Germany would be most central agent. Other countries would be all least central. However, this approach does not give us much information. We can see that there is a strong distortion resulting from row normalization.


Trade network row normalized

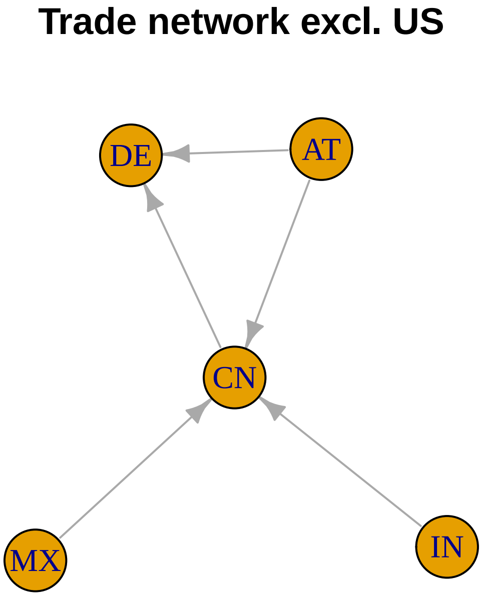
Possible solutions to this would be to use the row normalized values as weights for the connections between agents. Then row normalization would not delete the edge completely. Below we implement this and we can see that Germany which had only one top 5 trading partner in this list of countries, has the thickest edge. The values of adjacency matrix for other countries decrease by row normalization leading to thinner edges. E.g. India has 2 top 5 trading partner among those countries, resulting in thickness of edge being halved. Another possibility would be to normalize by scaling by a constant instead of row normalization. The question remains on how to judge who is the most central agent in such network. Possibly a function of number of in and out edges and weight could be defined. However, direct comparison to the concepts used above would be lost.


Trade network row normalized

```r
eigen_centrality(
  graph_from_adjacency_matrix(adj.rn),
  directed = TRUE,
  scale = TRUE,
  weights = NULL,
  options = arpack_defaults
)
```

Notice that we get a directed acyclic graph (DAG), for which eigenvector centrality is not defined, since there are no cycles at all and US is a sink.

- How would the network change if you removed or added a specific agent? Lets consider the case of removing US:



Based on inflowing arrows, China becomes the most central agent and India, Austria and Mexico the least central agents. Based on outflowing arrows, Austria is still the most central agent, while Germany is the least central agent.

Similarly to above, eigenvector centrality is undefined since there is a sink - Germany.

```r
eigen_centrality(
  graph_from_adjacency_matrix(adj[-1,-1]),
  directed = TRUE,
  scale = TRUE,
  weights = NULL,
  options = arpack_defaults
)
```

3. Simulate some agent characteristic based on a standard Normal distribution; use this characteristic to simulate responses following a liner-in-means model with your network. Repeat this a couple of times, and compare estimates of a standard linear model ($y_i = x_\beta + \varepsilon_i$) with the true values that you used to simulate the data

We simulate the data based on the row normalized adjacency matrix for trade partner connections, that we created in the preceding exercise.

Let’s say we are interested in estimating the impact of a countries fentanyl precursor production (x) on the log of a countries drug deaths (y). Then Wx denotes the average of neighboring countries fentanyl precursor production. We hypothesize that if a countries close trade partners fentanyl precursor production rate is high (as measured by Wx), then the fentanyl related drug deaths are high. Moreover, there are spillover effects with regards to the trading partners fentanyls related deaths from connected countries (Wy).

We specify the following linear-in-means model to estimate the relationship.

$$y = Wy\lambda + Wx\delta + x\beta + \varepsilon$$

Where Wy denotes the average death rate from drugs of countries that are close neighbors.

To simulate the values for y, we need to solve for the reduced form.

$$y = (I - W\lambda)^{-1}(Wx\delta + x\beta + \varepsilon)$$

```r
set.seed(123)
# number of agents
N = 6

# parameters definition
sigma2 = 1
lambda = 0.6
delta = 0.3
beta = 2 # true beta is 2
W = adj.rn

simulations <- 10000
times <- c(1:simulations)
result <- numeric(simulations)

for (i in times) {

# simulation of vecotrs
x = rnorm(N, 0, 1)
e = rnorm(N, 0, sigma2)

# caluclating means
Wx <- W %*% x

# calculating S
S = diag(N) - lambda * W

# generating y's
y = solve(S, Wx * delta + x * beta + e)

model <- lm(y ~ x)

result[i] <- coef(model)["x"]
}

inconsistent_estimate <- mean(result)

my_data <- data.frame(
  Name =  c("Simulated coef", "Real coef"),
  Estimate = c(inconsistent_estimate, beta)
)

knitr::kable(my_data, format = "markdown")
```

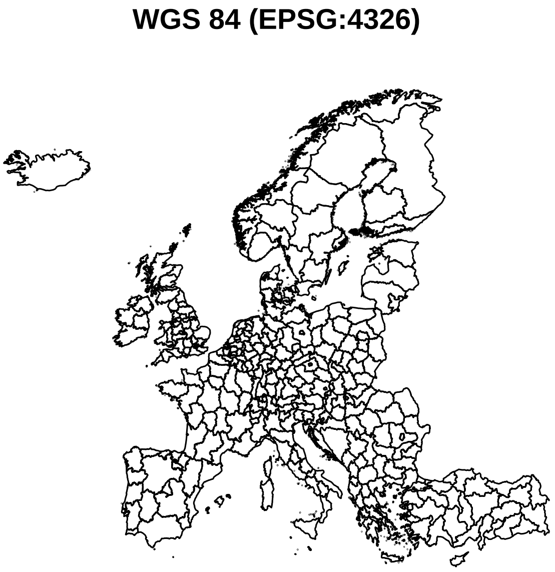
Name	Estimate
Simulated coef	1.718638
Real coef	2.000000

We set the true value for β to 2, λ to 0.6, and δ to 0.3. We then regress y on x, repeat this a 1000 times, and find that the mean of the estimand is 1.718638. Thus using the simple linear regression fails to recover the true value of β, the impact of a countries fentanyl precursor production on its fentanyl death rate.

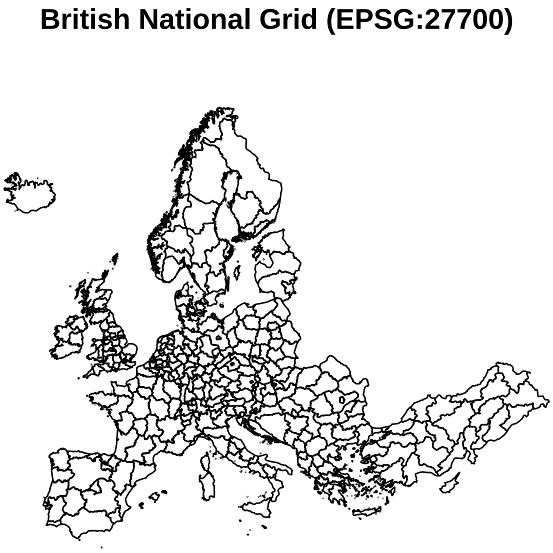
Exercise C

Download a suitable shapefile for NUTS2 regions (from here or using R directly) and some dataset of interest at the same level of aggregation (e.g. from here). Install and load the sf and ggplot2 packages together with their dependencies.

1. Read in the shapefile, and find out what projection and CRS the file uses. Map the data to use another projection and/or CRS of your choosing:



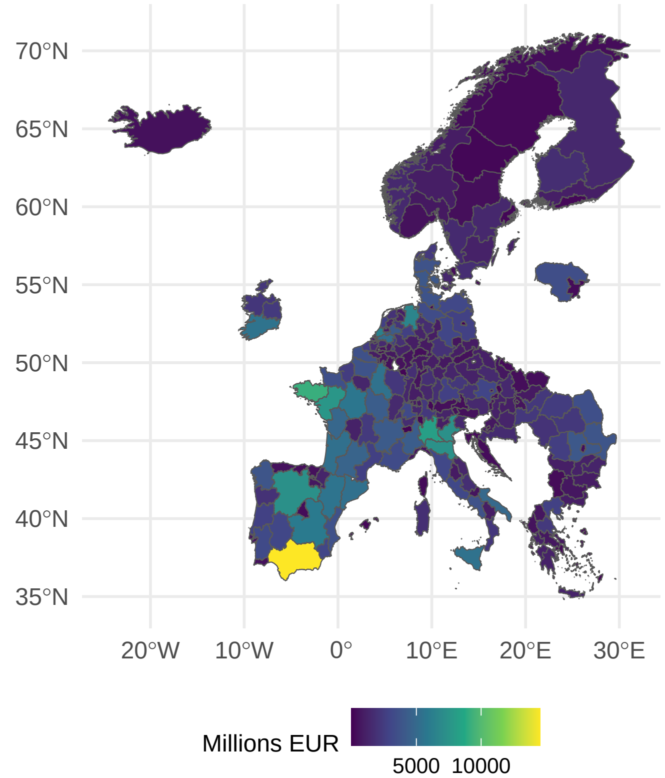
WGS 84 (EPSG:4326)



British National Grid (EPSG:27700)

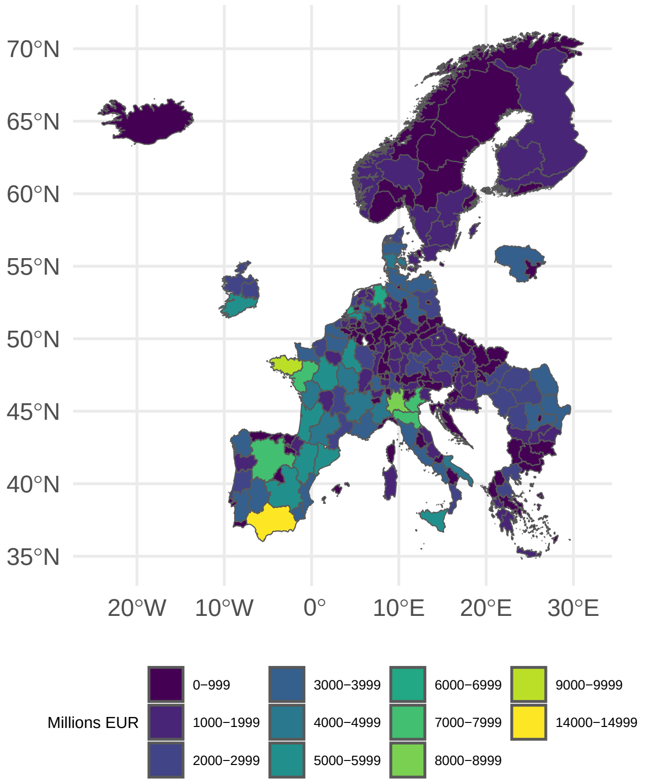
The coordinate reference system for the first map is the “World Geodetic System 1984,” or “WGS 84” (EPSG:4326) for short. The data is then re-projected to the “OSGB36 / British National Grid” coordinate reference system. The first map utilizes the Mercator projection, which is widely used in various global applications, including Google Earth. The Mercator projection is conformal, meaning it preserves angles. The second map is created using the Transverse Mercator projection, which does not have distortions along the central meridian. It’s worth noting that OSGB36 is specifically designed for mapping Great Britain and is optimized for maintaining accurate scale within this region and thus might not be suitable for mapping larger regions.

2. Merge the shapefile and the chosen dataset. Create tow meaningful visualizations of the chosen dataset using different scales (e.g. continuous versus discrete scaling of the data)



Agricultural Value Added by NUTS−2 Region in 2021

The discrete scaling shows the agricultural value added in million Euros at a NUTS-2 Level in 2021. While the first map visualizes the value added in continuous scaling. The second map portrays agricultural value added in a binned discrete scaling. Note that animal production and fishing is also included, which accounts for the high agricultural value added in Western France.



Binned Agricultural Value Added per NUTS−2 Region in

Moreover note that the NUTS-2 level visualization might induce a somewhat distorting picture. The regions with the highest output are Andalusia in Spain and some Western regions in France.

However, this is partially driven by the fact that the average NTUSS-2 region in Spain is much larger than in for example Germany. As the average size of NUTS-2 regions is more granular in Germany, the average agricultural value added might appear somewhat smaller.

```
#saving the map as raster image
#ggsave("map.tiff", plot = map, width = 10, height = 8, dpi = 300)

#saving the map as vector image
#ggsave("map.pdf", plot = map, width = 10, height = 8, dpi = 300)
```

3. Briefly explain what two conceptually different ways there are to store visualizations, and name two formats each. Which ones are more appropriate for which types of visualizations and why? Use the one you think is more appropriate to store your visualizations.

Visualizations can be broadly categorized into two distinct types based on the underlying spatial data they represent: raster and vector. Raster graphics are pixel-based, encoding images as arrays of multi-colored pixels. They are ideal for depicting continuous variables, such as elevation, temperature, or (as in the case above) agricultural output, where the variation is smoothly transitioned across the visualization. Common formats for storing raster graphics include PNG, JPEG, and TIFF files. The major caveat with raster graphics is their resolution dependency; zooming in too far reveals the pixel grid, leading to a loss of clarity.

On the other hand, vector graphics are built using geometric shapes such as points, lines, and polygons, defined mathematically by paths rather than pixels. This method enables vector images to retain crispness and clarity regardless of scaling, making them infinitely 'zoomable' without loss of resolution. Vector graphics are commonly stored in SVG (Scalable Vector Graphics), PDF (Portable Document Format), and EPS (Encapsulated PostScript) formats. Due to their scalable nature, vector graphics are typically more compact and require less storage compared to their raster counterparts.

Choosing between raster and vector formats for storing visualizations depends on the nature of the data and the intended use of the visualization. Raster images are best suited for detailed and complex imagery, such as remotely sensed measures and detailed maps with continuous data representation. They can offer a more efficient depiction of variations in data such as agricultural productivity across a map, although at the expense of larger file sizes and fixed resolution.

Vector graphics, with their ability to scale without loss of quality, are preferable for diagrams, logos, and maps where clear, clean lines and shapes are needed. Fonts for example are stored as vector graphics.

For visualizing agricultural value added in NUTS-2 regions across Europe, a PDF is suitable for publications. This choice supports a detailed and accurate representation of agricultural productivity data, accommodating the depth and variation in the dataset. PNGs and JPEG can be very efficient when file size is a strict constraint. For high-quality use cases, a tiff file might be most appropriate. However, such a file can have a very large size.

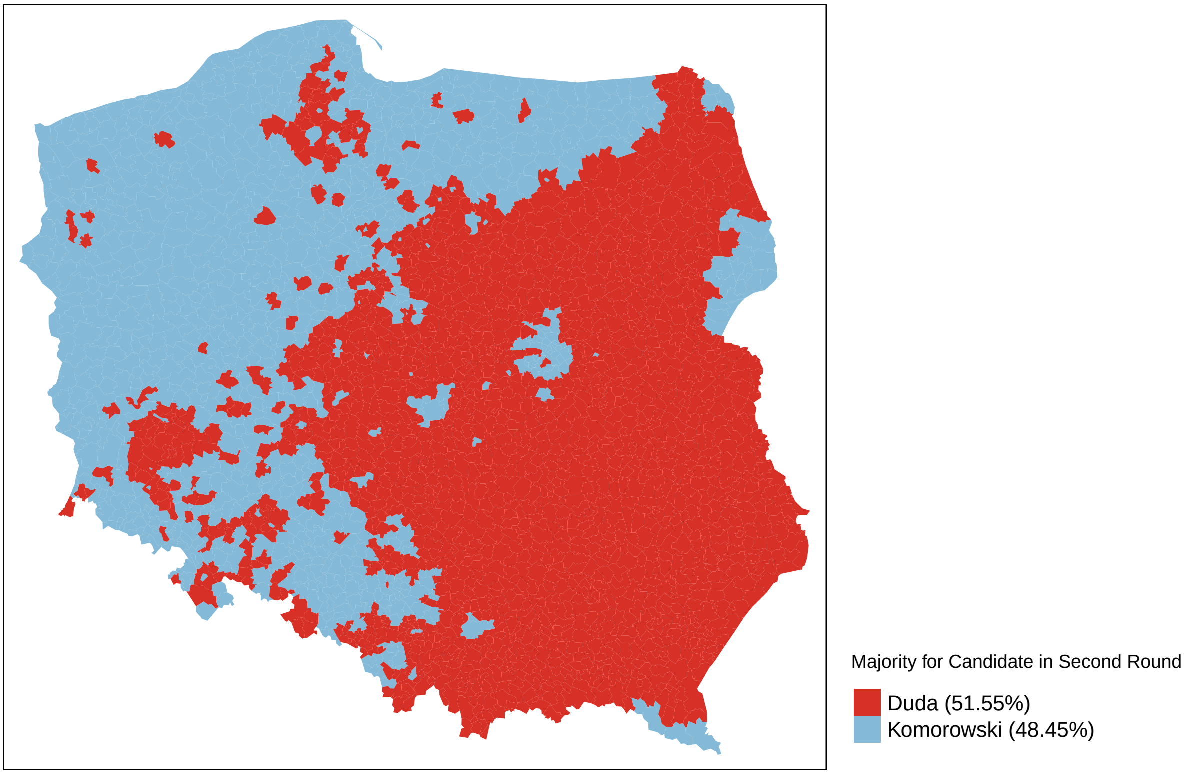
Exercise D

1. Install and load the tmap and spDataLarge packages (available from GitHub). Load and review the pol_pres15 dataset on the Polish Presidential election in 2015 (see ?pol_pres15). Create three different, insightful visualizations of the underlying data.

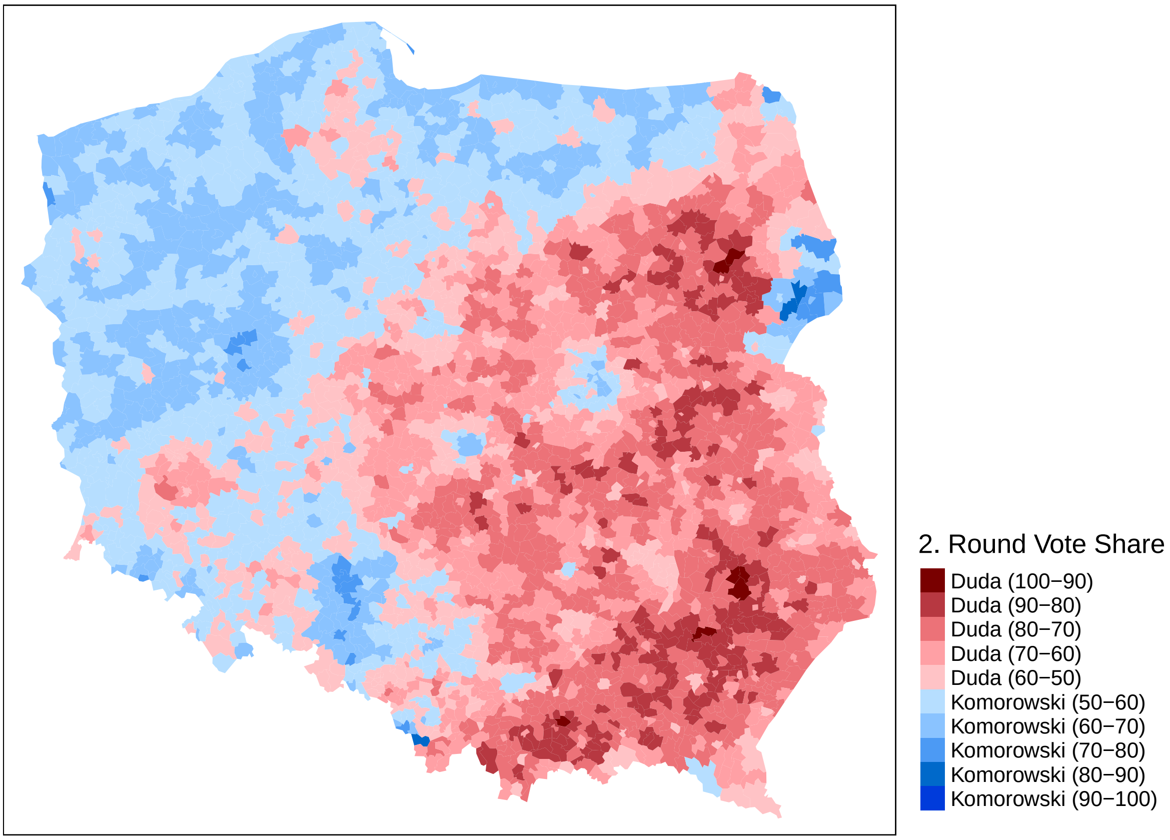
- One visualization should compare the support for Komorowski and Duda.

- One visualization should investigate possible issues with postal voting envelopes.
- At least one visualization should use the tmap package.

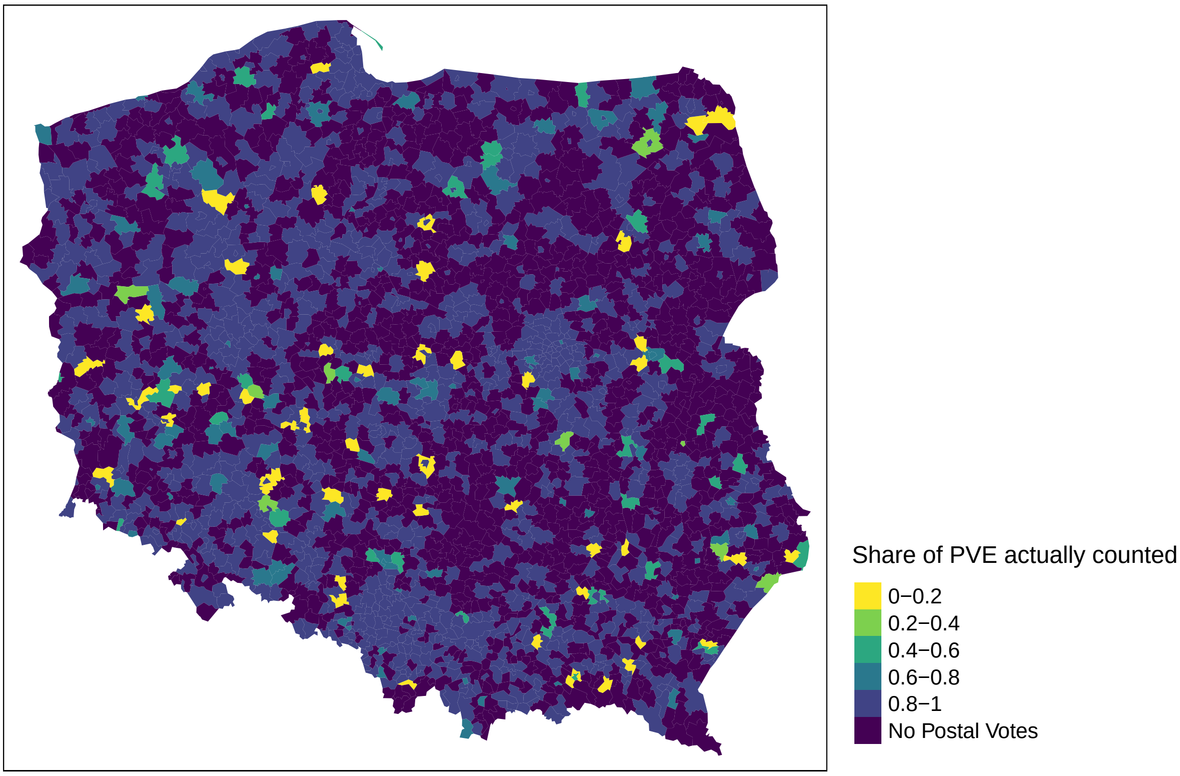
We start by visualizing the results of the second round of the Polish election. The first map displays the winner of the respective district. Overall we can see that the election was extremely close, with Duda winning just 51.55% and Komorowski winning 48.45% of the popular vote. One can clearly observe an east-west pattern, where the Western part of the country predominantly voted for Duda and the eastern part of the country for Komrowski.



We then plot the percentage points of the candidates in the second runoff. This gives a detailed picture of the election outcome. One can clearly see the strongholds of the respective candidates.

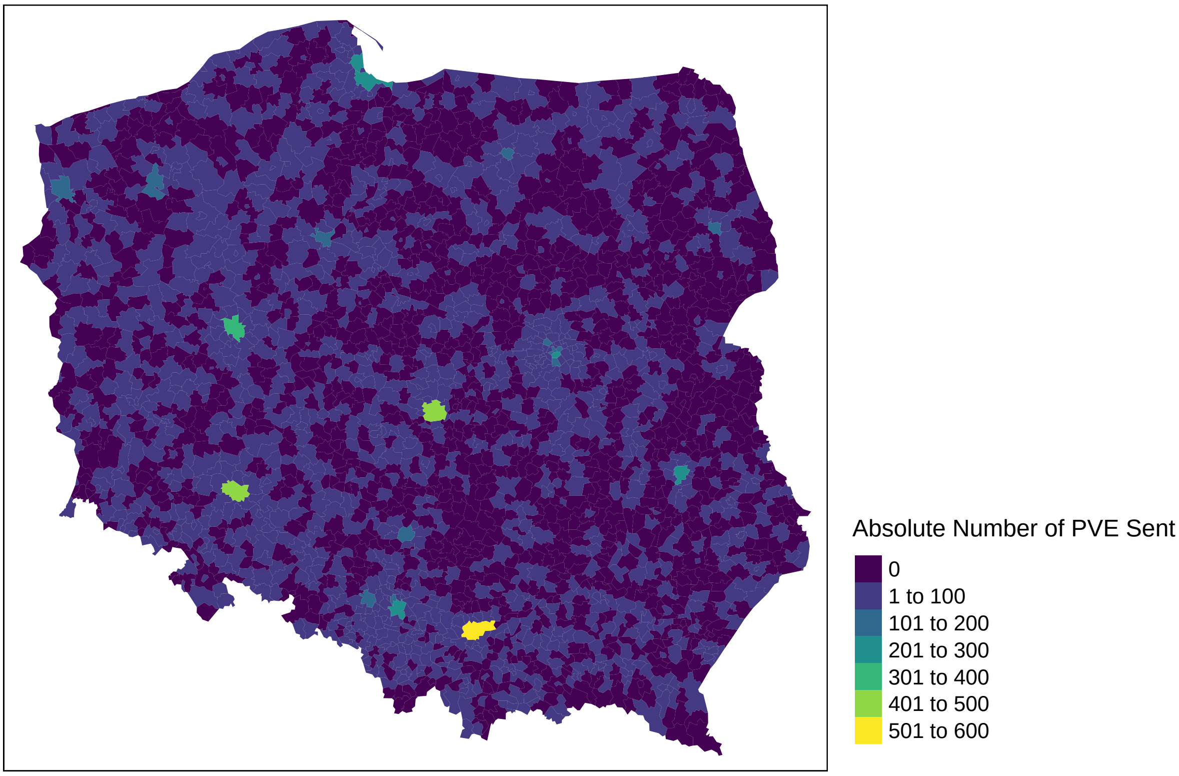


We now investigate possible issues with voting envelopes. To uncover any potential oddities regarding postal voting in the second round, we calculate the share of postal votes received over the postal voting packages that were sent in. A low share indicates that a lot of postal votes were not counted for any reason. This might be preliminary and somewhat suggestive evidence into some sort of inconsistency or even fraud.



The figure shows districts in higher darkblue, green, or yellow if the share of postal votes that were received was lower. There seems to be no systematic relationship between any districts and candidate allegiance. We conclude that there is no evidence for inconsistencies in the postal voting system based on this measure.

However, note that the map above disguises the low absolute number of votes in some districts. We see that the overall number of postal votes sent is very low in most rural districts and thus the inconsistencies can be driven by only a few incorrectly sent postal votes.



,