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)

2024-03-25

Exercise A

For our 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.

Below we wrote a function which computes: 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%.

```
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)
```

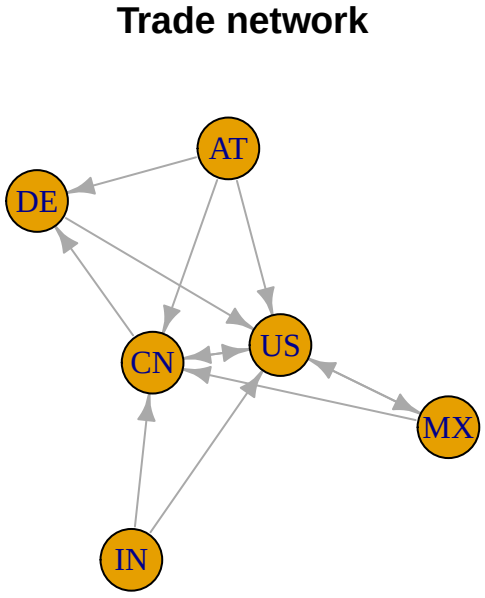
```
## [[1]]
##                  estimate
## Constant       -17.259625
## Crime           -0.184610
## Rooms            7.706840
```

```
## 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 could consider 6 different countries and an indicator of a high trade 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 10 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 the most central agents in this network, because it is 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 export country for any country from this network.

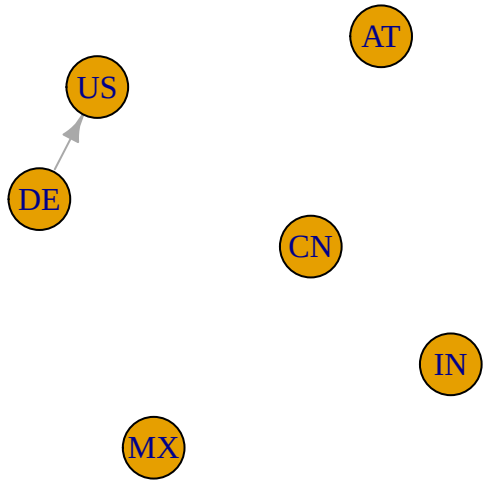
Another basic criterion of centrality could be the amount of outwards pointing arrows of an agent. Thus a country would be considered central if it exports to highest number of countries from this network. In this sense, Germany is least central with only 1 outward arrow, while Austria is most central with 3 outward arrows.

Eigenvector centrality: There is no sink, but not possible to get to Austria. ? Page Rank: weights (probs) need to be defined?

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

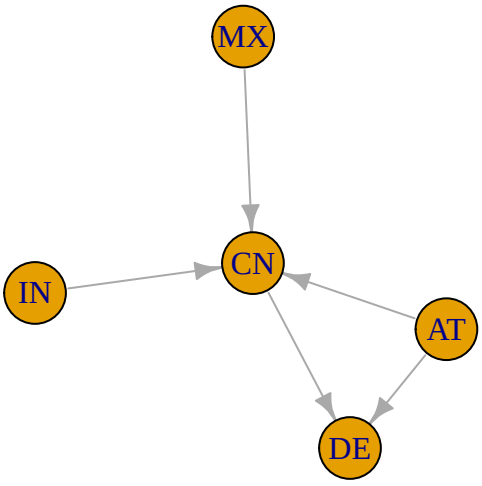
Germany which had only 1 connection, is the only row with sum of 1. Thus its the only edge which survives row normalization. Thus here depending on criterion US or Germany would be most central agent. Other countries would be all least central.

Trade network row normalized



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

Trade network excl. 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.

3.Simulate some agent charaacteristic 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

```
# 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)

# sampling from reduced form to generate y's

# caluclating means
Wx <- W %*% x

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

# create 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"), # Creating a column named ID with values from 1 to 4
  Estimate = c(inconsistent_estimate, beta) # Creating a Name column
)

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

Name	Estimate
Simulated coef	1.713612
Real coef	2.000000

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.

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

We set the true value for β to 2, λ to 0.9, and δ to 2.2. We then regress y on x, repeat this a 1000 times, and find that the mean of the estimand is 1.443618 Thus using the simple linear regression fails to recover the true value of β.

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.

• 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:

```
shp <- st_read(dsn = "./data", layer ="NUTS_RG_03M_2021_4326_LEVL_2")
```

```
## Reading layer `NUTS_RG_03M_2021_4326_LEVL_2' from data source
##   `/Users/gustavpirich/Desktop/GITHUB/spatial_econ/assignment_1/data'
##   using driver `ESRI Shapefile'
## Simple feature collection with 334 features and 9 fields
## Geometry type: MULTIPOLYGON
## Dimension:     XY
## Bounding box:  xmin: -63.15176 ymin: -21.38696 xmax: 55.83509 ymax: 80.83426
## Geodetic CRS:  WGS 84
```

```
old_crs <- st_crs(shp)
print(old_crs)
```

```
## Coordinate Reference System:
##   User input: WGS 84
##   wkt:
## GEOGCRS["WGS 84",
##     DATUM["World Geodetic System 1984",
##         ELLIPSOID["WGS 84",6378137,298.257223563,
##             LENGTHUNIT["metre",1]]],
##     PRIMEM["Greenwich",0,
##         ANGLEUNIT["degree",0.0174532925199433]],
##     CS[ellipsoidal,2],
##         AXIS["latitude",north,
##             ORDER[1],
##             ANGLEUNIT["degree",0.0174532925199433]],
##         AXIS["longitude",east,
##             ORDER[2],
##             ANGLEUNIT["degree",0.0174532925199433]],
##     ID["EPSG",4326]]
```

```
shp_27700 <- st_transform(shp, 27700)

new_crs <- st_crs(shp_27700)
print(new_crs)
```

```
## Coordinate Reference System:
##   User input: EPSG:27700
##   wkt:
## PROJCRS["OSGB36 / British National Grid",
##     BASEGEOGCRS["OSGB36",
```

```
##         DATUM["Ordnance Survey of Great Britain 1936",
##             ELLIPSOID["Airy 1830",6377563.396,299.3249646,
##                 LENGTHUNIT["metre",1]]],
##         PRIMEM["Greenwich",0,
##             ANGLEUNIT["degree",0.0174532925199433]],
##         ID["EPSG",4277]],
##     CONVERSION["British National Grid",
##         METHOD["Transverse Mercator",
##             ID["EPSG",9807]],
##         PARAMETER["Latitude of natural origin",49,
##             ANGLEUNIT["degree",0.0174532925199433],
##             ID["EPSG",8801]],
##         PARAMETER["Longitude of natural origin",-2,
##             ANGLEUNIT["degree",0.0174532925199433],
##             ID["EPSG",8802]],
##         PARAMETER["Scale factor at natural origin",0.9996012717,
##             SCALEUNIT["unity",1],
##             ID["EPSG",8805]],
##         PARAMETER["False easting",400000,
##             LENGTHUNIT["metre",1],
##             ID["EPSG",8806]],
##         PARAMETER["False northing",-100000,
##             LENGTHUNIT["metre",1],
##             ID["EPSG",8807]]],
##     CS[Cartesian,2],
##         AXIS["(E)",east,
##             ORDER[1],
##             LENGTHUNIT["metre",1]],
##         AXIS["(N)",north,
##             ORDER[2],
##             LENGTHUNIT["metre",1]],
##     USAGE[
##         SCOPE["Engineering survey, topographic mapping."],
##         AREA["United Kingdom (UK) - offshore to boundary of UKCS within 49°45'N to 61°N and 9°W to 2
##         BBOX[49.75,-9,61.01,2.01]],
##     ID["EPSG",27700]]
```

The projection used is the "World Geodetic System 1984" or for short "WGS 84" (EPSG 4326). We then map the data to the "North American Datum 1983" (NAD83) projection, which is extremely similar to the WGS 84.

• 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)

```
agr_r_accts_page_spreadsheet <- read_excel("/Users/gustavpirich/Desktop/GITHUB/spatial_econ/data/agricu
           sheet = 3, range = "A11:C472")
```

```
## New names:
## * `` -> `...3`
```

```
agr_r_accts_page_spreadsheet_1 <- agr_r_accts_page_spreadsheet %>%
  rename("NUTS_ID" = "GEO (Codes)")

shp_merged <- left_join(shp, agr_r_accts_page_spreadsheet_1)
```

```
## Joining with `by = join_by(NUTS_ID)`
```

Exercise D

Another way: 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.

Combine support for Komorowski and Duda into a single categorical variable

pol_pres15$Support <- ifelse(pol_pres15I_Komorowski_share > pol_pres15$I_Duda_share, "Komorowski", "Duda")

Visualize support for Komorowski and Duda in the first run

tm_shape(pol_pres15) + tm_borders(lwd = 0.5, alpha = 0.4) + tm_fill(col = "Support", style = "cat", palette = "-RdYlBu", title = "Support for Candidates") + tm_layout(legend.position = c("left", "bottom"))

• One visualization should investigate possible issues with postal voting envelopes. We want to investigate if Issues occured because voting envelopes got lost on the way to the voters.

Visualization investigating possible issues with postal voting envelopes

Calculate the proportion of postal voting envelopes received to postal voting packages sent

pol_pres15$Envelope_Proportion <- pol_pres15I_postal_voting_envelopes_received / pol_pres15$I_voters_sent_postal_voti

Define a threshold for the proportion indicating possible issues

threshold <- 0.99 # You can adjust this threshold as needed

Create a new column indicating possible issues with postal voting envelopes based on the proportion

pol_pres15$Envelope_Issues <- ifelse(pol_pres15Envelope_Proportion < threshold, "Possible Issues", "No Issues")

tm_shape(pol_pres15) + tm_borders(lwd = 0.5, alpha = 0.4) + tm_fill(col = "Envelope_Issues", style = "cat", palette = c("green", "red"), title = "Possible Issues with Postal Voting Envelopes") + tm_layout(legend.position = c("left", "bottom"))

• Third vizualization # Visualize support for Komorowski and Duda in the second run pol_pres15$Support <- ifelse(pol_pres15II_Komorowski_share > pol_pres15$II_Duda_share, "Komorowski", "Duda") tm_shape(pol_pres15) + tm_borders(lwd = 0.5, alpha = 0.4) + tm_fill(col = "Support", style = "cat", palette = "-RdYlBu", title = "Support for Candidates") + tm_layout(legend.position = c("central", "bottom"))