Spatial Economics – Assignment 1

Max Heinze (h11742049@s.wu.ac.at) Kevin Kain (h12232066@s.wu.ac.at)
Jaime Miravet (h12235992@s.wu.ac.at)

April 2, 2024

Contents

The executable code that was used in compiling the assignment is available on GitHub at https://github.com/maxmheinze/spatial.

Task A

Preliminaries

First, we load the `MASS` package and check what variables there are in the `Boston` dataset.

```
# Header ----------------------------------------------------------------

rm(list = ls())
gc()

pacman::p_load(MASS)

# Check Column Names ----------------------------------------------------

colnames(Boston)
```

Creating the Function

Next, we create the desired function.

```
# Create the function ---------------------------------------------------

boston_quick_ols <- function(dependent, ...) {

    # Create a formula string from the inputs
    independents <- paste(c(...), collapse = " + ")
    formula_string <- paste(dependent, "~", independents)

    # Fit the model
    fitted_model <- lm(as.formula(formula_string), data = Boston)

    # Get the summary
    fitted_model_summary <- summary(fitted_model)

    # Get point estimates and confidence intervals
    list_coef <- fitted_model_summary$coefficients
    list_conf <- confint(fitted_model, level = 0.95)
    list_ervr <- fitted_model_summary$sigma^2

    # Output a list
    return(list(coefficients = list_coef[, 1], error_variance = list_ervr, test_statistic_t
    = list_coef[,
        3], test_statistic_p = list_coef[, 4], confidence_intervals = list_conf))

}
```

A Simple Linear Model

Next, we apply the function, using a collection of four independent variables.

```
boston_quick_ols("medv", "rm", "age", "dis", "nox")
```

```
## $coefficients
##  (Intercept)           rm          age          dis          nox
##  -6.61135440   8.00051949  -0.06932587  -1.08526888 -22.10858455
##
## $error_variance
## [1] 37.35166
##
## $test_statistic_t
```

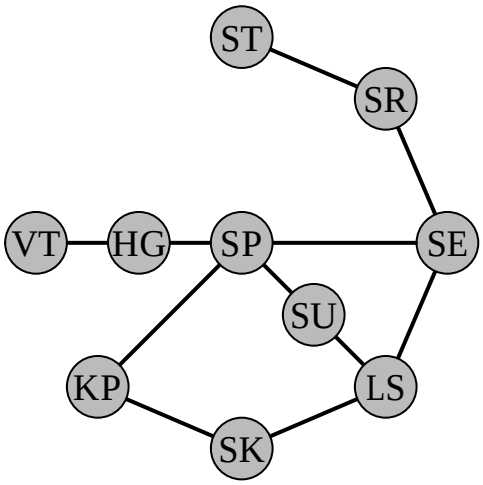
```
## (Intercept)          rm         age         dis         nox
##   -1.590287   19.654992   -4.422259   -4.850184   -5.486524
##
## $test_statistic_p
##  (Intercept)           rm          age          dis          nox
## 1.124008e-01 3.520952e-64 1.198751e-05 1.649044e-06 6.516890e-08
##
## $confidence_intervals
##                    2.5 %       97.5 %
## (Intercept) -14.7793094   1.55660058
## rm            7.2007886   8.80025034
## age          -0.1001258  -0.03852595
## dis          -1.5248891  -0.64564866
## nox         -30.0256124 -14.19155674
```

Task B

Creating a Graph and an Adjacency Matrix

We chose the network of all first-district Vienna subway stations. The graph and the adjacency matrix can be found below. Nodes represent individual stations, and edges represent direct subway connections between two stations, without passing another station or changing to another line. We abstract from the existence of different subway lines and from the existence of other stations outside the first district as well as links to these stations. The two-character node labels are to be read as follows: ST is Schottentor, SR is Schottenring, SE is Schwedenplatz, LS is Landstraße, SK is Stadtpark, KP is Karlsplatz, SU is Stubentor, SP is Stephansplatz, HG is Herrengasse, and VT is Volkstheater.

```r
# Header ----------------------------------------------------------------
pacman::p_load(igraph, extrafont)


# Create Matrix ---------------------------------------------------------

# Create the adjacency matrix
adj_matrix <- matrix(c(0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 1, 0, 0, 0, 0, 0, 0, 0,
    0, 1, 0, 1, 0, 0, 0, 1, 0, 0, 0, 0, 1, 0, 1, 0, 1, 0, 0, 0, 0, 0, 0, 1, 0, 1,
    0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 1, 0, 0, 0, 0, 0, 1, 0, 0, 0, 1, 0, 0, 0, 0,
    1, 0, 0, 1, 1, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0,
    1, 0), nrow = 10, byrow = TRUE)

# Define node names
node_names <- c("ST", "SR", "SE", "LS", "SK", "KP", "SU", "SP", "HG", "VT")
dimnames(adj_matrix) <- list(node_names, node_names)

coords_matrix <- matrix(c(0, 0, 3.5, -1.5, 5, -5, 3.5, -8.5, 0, -10, -3.5, -8.5,
    1.75, -6.75, 0, -5, -2.5, -5, -5, -5), ncol = 2, byrow = TRUE)


# Create graph ----------------------------------------------------------

graph_1 <- graph_from_adjacency_matrix(adj_matrix, mode = "undirected")


# Output ----------------------------------------------------------------

plot(graph_1, layout = coords_matrix, vertex.size = 30, vertex.color = "#BBBBBB",
    vertex.label.cex = 1.2, vertex.label.font = 2, vertex.label.family = "Lato",
    vertex.label.color = "black", edge.color = "black", edge.width = 2)
```




```
knitr::kable(adj_matrix)
```

	ST	SR	SE	LS	SK	KP	SU	SP	HG	VT
ST	0	1	0	0	0	0	0	0	0	0
SR	1	0	1	0	0	0	0	0	0	0
SE	0	1	0	1	0	0	0	1	0	0
LS	0	0	1	0	1	0	1	0	0	0
SK	0	0	0	1	0	1	0	0	0	0
KP	0	0	0	0	1	0	0	1	0	0
SU	0	0	0	1	0	0	0	1	0	0
SP	0	0	1	0	0	1	1	0	1	0
HG	0	0	0	0	0	0	0	1	0	1
VT	0	0	0	0	0	0	0	0	1	0

Centrality

Our graph is undirected, and thus the simplest notion of centrality that we can investigate is the nodes' **degree**. We can compute the degree by simply calculating the row sums of the adjacency matrix. The result, given in the table below, is that Stephansplatz is the most central station, having 4 links, and Volkstheater and Schottentor are the least central stations, having one link each.

```
knitr::kable(rowSums(adj_matrix), col.names = c("degree centrality"))
```

	degree centrality
ST	1
SR	2
SE	3
LS	3
SK	2
KP	2
SU	2
SP	4
HG	2
VT	1

Alternatively, we can calculate the nodes' **Eigenvector centrality**. Eigenvector centrality is a centrality measure that takes into account how influential (central) the nodes bordering some node are, where being connected to a more influential node is "rewarded" with a higher centrality measure. To describe the notion mathematically, let N be the set of nodes in the graph from above, and let $H(n)$ be the set of neighbors of some node n (H in this case stands for the *hood*). Let now i and j be two nodes that are in N. We can define a centrality measure c_n such that

$$c_i^{\text{eigenvector}} = \alpha \sum_{j \in H(i)} c_j = \alpha \sum_{j \in N} a_{i,j} c_j,$$

where $a_{i,j}$ is an element of the adjacency matrix and α is some constant. The latter equality follows quite straightforwardly from the fact that $a_{i,j} = 0$ if $j \notin H(i)$ and $a_{i,j} = 1$ if $j \in H(i)$, i.e., the definition of the adjacency matrix. If we now let $\alpha = \frac{1}{\lambda}$, this can be written as

$$\boldsymbol{A}\boldsymbol{c} = \lambda \boldsymbol{c},$$

which means that $\boldsymbol{c}$ is an eigenvector of $\boldsymbol{A}$ corresponding to eigenvalue λ. It follows from the Perron–Frobenius Theorem that there is exactly one (except multiples of itself) eigenvector with all non-negative entries (which is what we desire for the centrality measure) and that it corresponds to the largest of the eigenvalues. Conveniently, the `igraph` package has a function that does the calculations for us. We get the following centrality measures:

```
eigen_centrality(graph_1, directed = FALSE, scale = TRUE, weights = NULL, options =
arpack_defaults()) %>%
    `$`(vector) %>%
    knitr::kable(col.names = "eigenvector centrality")
```

	eigenvector centrality
ST	0.1480101
SR	0.3832787
SE	0.8445071
LS	0.8036100
SK	0.5399797
KP	0.5946913
SU	0.6964970
SP	1.0000000
HG	0.4538490
VT	0.1752621

Again, Stephansplatz is the most central station. Schottentor is now the uniquely least central station.

There exist extensions and variations of eigenvector centrality, such as PageRank centrality, but there are also other approaches. One of these other approaches is the notion of **closeness centrality**, a concept where having shorter average shortest path lengths to all other nodes is rewarded. It is defined as

$$c_i^{\text{closeness}} = \frac{N-1}{\sum_{j \in N, j \neq i} d(i,j)},$$

where $d(\cdot)$ refers to the length of the shortest average path. Again, we are happy to use the implementation the `igraph` package provides to calculate closeness centrality of our stations.

```
knitr::kable(closeness(graph_1), col.names = "closeness centrality")
```

	closeness centrality
ST	0.0333333
SR	0.0454545
SE	0.0625000
LS	0.0526316
SK	0.0454545
KP	0.0500000
SU	0.0500000
SP	0.0666667
HG	0.0476190
VT	0.0344828

Surprise, surprise: Stephansplatz is the most central station and Schottentor is the least central station.

Centrality in a Row-Normalized Network

Row-normalizing means that we divide every element in our adjacency matrix by the corresponding row sum. If we do this, we can see that our adjacency matrix is no longer symmetric:

```
adj_matrix_2 <- adj_matrix/rowSums(adj_matrix)

knitr::kable(round(adj_matrix_2, 2))
```

	ST	SR	SE	LS	SK	KP	SU	SP	HG	VT
ST	0.0	1.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.0
SR	0.5	0.00	0.50	0.00	0.00	0.00	0.00	0.00	0.00	0.0
SE	0.0	0.33	0.00	0.33	0.00	0.00	0.00	0.33	0.00	0.0
LS	0.0	0.00	0.33	0.00	0.33	0.00	0.33	0.00	0.00	0.0
SK	0.0	0.00	0.00	0.50	0.00	0.50	0.00	0.00	0.00	0.0
KP	0.0	0.00	0.00	0.00	0.50	0.00	0.00	0.50	0.00	0.0
SU	0.0	0.00	0.00	0.50	0.00	0.00	0.00	0.50	0.00	0.0
SP	0.0	0.00	0.25	0.00	0.00	0.25	0.25	0.00	0.25	0.0
HG	0.0	0.00	0.00	0.00	0.00	0.00	0.00	0.50	0.00	0.5
VT	0.0	0.00	0.00	0.00	0.00	0.00	0.00	0.00	1.00	0.0

This means that we can no longer treat our network as an undirected graph, since in-connections and out-connections are differently weighted.

Regarding **degree centrality**, we therefore have to split up our measure into **in-degree** and **out-degree** centrality. Since we normalized row sums, i.e., the sum of outward connections of a node, the out-degree is a somewhat useless measure, since calculating it will always return unity by definition. Calculating the in-degree, however, yields a result:

```
knitr::kable(colSums(adj_matrix_2), col.names = c("in-degree centrality"))
```

	in-degree centrality
ST	0.5000000
SR	1.3333333
SE	1.0833333
LS	1.3333333
SK	0.8333333
KP	0.7500000
SU	0.5833333
SP	1.8333333
HG	1.2500000
VT	0.5000000

A high measure can be interpreted as a station being relatively important for its neighbor stations. If a station has only one neighbor station, and that neighbor station has another station that it neighbors, the first station will have a measure of 0.5. In our example, Stephansplatz is the most central station and Schottentor and Volkstheater are the least central stations.

So, we get the same most and least central stations as in the original degree centrality case. Will that always be the case that stations that are more central in the one measure are also more central in the other? No. For a smooth disproof by counterexample, look at nodes SR and SE.

Our measure of **eigenvector centrality** is affected in so far as that in the original formula,

$$c_i^{\text{eigenvector}} = \alpha \sum_{j \in N} a_{i,j} c_j,$$

$a_{i,j}$ can now assume values between 0 and 1 instead of *just* 0 and 1. Every value that was 0 before is still 0, but all connections that do exist are now weighted by how many other outgoing connections from a station there are. This makes for a slightly different eigenvector:

```
adj_matrix_2 <- adj_matrix/rowSums(adj_matrix)

graph_2 <- graph_from_adjacency_matrix(adj_matrix_2, mode = "directed", weighted = TRUE)



eigen_centrality(graph_2, directed = FALSE, scale = TRUE, weights = NULL, options =
arpack_defaults()) %>%
    `$`(vector) %>%
    knitr::kable(col.names = "eigenvector centrality")
```

	eigenvector centrality
ST	0.5030657
SR	0.6942537
SE	0.8190652
LS	0.8004645
SK	0.6486233
KP	0.6756410
SU	0.6845443
SP	1.0000000
HG	0.7628555
VT	0.5527755

For our measure of **closeness centrality**, we need a definition of "distance" in a weighted graph. Conventionally, weights are in this case interpreted as the "length" of a node, meaning that a low-weighted connection is related to the notion of two nodes being "closer," and that the distance equals the sum of weights along a path. Since weights *from* a station that has many outgoing connections will be lower (even if incoming connections' weights need not be), having many outgoing connections is rewarded using this measure of centrality with a row-normalized adjacency matrix. We can also see that Stephansplatz gains relatively more compared to the original closeness measure:

```
knitr::kable(closeness(graph_2), col.names = "closeness centrality")
```

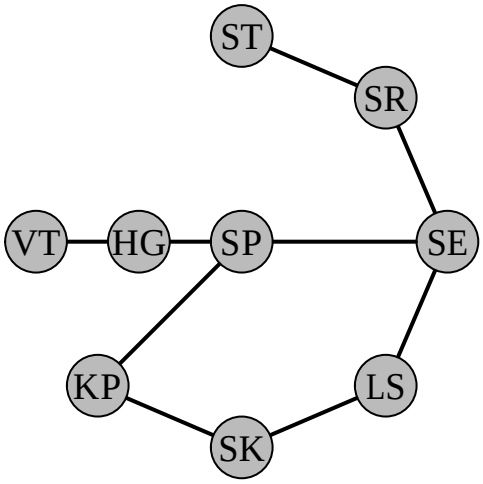
	closeness centrality
ST	0.0582524
SR	0.1153846
SE	0.1875000
LS	0.1500000
SK	0.1052632
KP	0.1224490
SU	0.1250000
SP	0.2105263
HG	0.1212121
VT	0.0597015

Removing a node

We remove Stubentor because it might be interesting to see what happens if we remove one of Stephansplatz's connections. Let's see:

```
# Header ----------------------------------------------------------------
pacman::p_load(igraph, extrafont)


# Create Matrix ---------------------------------------------------------

# Create the adjacency matrix
adj_matrix_3 <- matrix(c(0, 1, 0, 0, 0, 0, 0, 0, 0, 1, 0, 1, 0, 0, 0, 0, 0, 0, 0,
```

```
    1, 0, 1, 0, 0, 1, 0, 0, 0, 0, 1, 0, 1, 0, 0, 0, 0, 0, 0, 0, 1, 0, 1, 0, 0, 0,
    0, 0, 0, 0, 1, 0, 1, 0, 0, 0, 0, 1, 0, 0, 1, 0, 1, 0, 0, 0, 0, 0, 0, 0, 1, 0,
    1, 0, 0, 0, 0, 0, 0, 0, 1, 0), nrow = 9, byrow = TRUE)

# Define node names
node_names_3 <- c("ST", "SR", "SE", "LS", "SK", "KP", "SP", "HG", "VT")
dimnames(adj_matrix_3) <- list(node_names_3, node_names_3)

coords_matrix_3 <- matrix(c(0, 0, 3.5, -1.5, 5, -5, 3.5, -8.5, 0, -10, -3.5, -8.5,
    0, -5, -2.5, -5, -5, -5), ncol = 2, byrow = TRUE)


# Create graph ------------------------------------------------------------

graph_3 <- graph_from_adjacency_matrix(adj_matrix_3, mode = "undirected")


# Output ------------------------------------------------------------------

plot(graph_3, layout = coords_matrix_3, vertex.size = 30, vertex.color = "#BBBBBB",
    vertex.label.cex = 1.2, vertex.label.font = 2, vertex.label.family = "Lato",
    vertex.label.color = "black", edge.color = "black", edge.width = 2)
```




```
knitr::kable(adj_matrix_3)
```

	ST	SR	SE	LS	SK	KP	SP	HG	VT
ST	0	1	0	0	0	0	0	0	0
SR	1	0	1	0	0	0	0	0	0
SE	0	1	0	1	0	0	1	0	0
LS	0	0	1	0	1	0	0	0	0
SK	0	0	0	1	0	1	0	0	0
KP	0	0	0	0	1	0	1	0	0
SP	0	0	1	0	0	1	0	1	0
HG	0	0	0	0	0	0	1	0	1
VT	0	0	0	0	0	0	0	1	0

```
knitr::kable(rowSums(adj_matrix_3), col.names = c("degree centrality"))
```

	degree centrality
ST	1
SR	2
SE	3

	degree centrality
LS	2
SK	2
KP	2
SP	3
HG	2
VT	1

```
eigen_centrality(graph_3, directed = FALSE, scale = TRUE, weights = NULL, options =
arpack_defaults()) %>%
    `$`(vector) %>%
    knitr::kable(col.names = "eigenvector centrality")
```

	eigenvector centrality
ST	0.2412297
SR	0.5471942
SE	1.0000000
LS	0.7211592
SK	0.6358438
KP	0.7211592
SP	1.0000000
HG	0.5471942
VT	0.2412297

```
knitr::kable(closeness(graph_3), col.names = "closeness centrality")
```

	closeness centrality
ST	0.0384615
SR	0.0526316
SE	0.0714286
LS	0.0555556
SK	0.0500000
KP	0.0555556
SP	0.0714286
HG	0.0526316
VT	0.0384615

We can see that Stephansplatz suffers and is now exactly as central as Schwedenplatz. Also, there is no longer a difference between Schottentor and Volkstheater as least central stations using any measure of centrality. This makes intuitive sense since the new graph is now symmetrically consisting of a central "circle" and two "appendices" of length two.

Simulation

We say that the coolness factor $\boldsymbol{x}$ is a characteristic of each station. Some stations are cooler and some are less cool. We also assume that coolness x affects the crime rate $\boldsymbol{y}$ at a station. However, crime rates at stations are also influenced by neighboring stations' crime rates and coolness factors, in the style of a linear-in-means model:

$$\boldsymbol{y} = \boldsymbol{x}\beta + \boldsymbol{W}\boldsymbol{x}\gamma + \lambda\boldsymbol{W}\boldsymbol{y} + \varepsilon,$$

where $\varepsilon \sim \mathbf{N}(\mathbf{0}, \boldsymbol{I}\sigma^2)$. Letting $\boldsymbol{S} = (\boldsymbol{I} - \lambda\boldsymbol{W})$, this becomes

$$\boldsymbol{S}\boldsymbol{y} = \boldsymbol{x}\beta + \boldsymbol{W}\boldsymbol{x}\gamma + \varepsilon,$$

which we can simulate as shown in the following. For the simulation, we let $\beta = -1$ (cooler stations have less crime), $\gamma = 1$, $\lambda = 0.65$, and $\sigma^2 = 1$.

```
set.seed(1234)

# Set parameters
N <- length(node_names)
beta <- -1
gamma <- 1
lambda <- 0.65
sigmasquared <- 1


reps <- 1000
estims <- vector("numeric", reps)

for (i in 1:reps) {

    # Create the coolness vector
    x <- rnorm(length(node_names))
    names(x) <- node_names

    # Rename the adj. matrix W
    W <- adj_matrix_2

    errs <- rnorm(N, 0, sigmasquared)

    Wx <- W %*% x

    # Calculate S = (I - \lambda W)
    S = diag(N) - lambda * W

    # Solve for y (the crime variable)
    y = solve(S, Wx * gamma + x * beta + errs)

    # Fit a linear model
    model_1 <- lm(y ~ x)

    # Store fitted estimates
    estims[i] <- coef(model_1)["x"]
}
```

We then fit the following linear model

$$\boldsymbol{y} = \boldsymbol{x}\beta + \boldsymbol{\varepsilon},$$

and store the coefficients for β.

```
avg_estimate <- mean(estims)
print(avg_estimate)
```

```
## [1] -0.9238921
```

The average β coefficient of 1,000 simulations was -0.9238921, and we conclude the estimate is downward biased in magnitude, since the true value of β was −1.

Task D

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

```
# install required packages
install.packages("tmap", repos = "https://r-tmap.github.io/tmap/", type = "source")
```

```
## Warning: unable to access index for repository https://r-tmap.github.io/tmap/src/contrib:
##   cannot open URL 'https://r-tmap.github.io/tmap/src/contrib/PACKAGES'
```

```
## Warning: package 'tmap' is not available for this version of R
##
## A version of this package for your version of R might be available elsewhere,
## see the ideas at
## https://cran.r-project.org/doc/manuals/r-patched/R-admin.html#Installing-packages
```

```
install.packages("spDataLarge", repos = "https://nowosad.github.io/drat/", type = "source")

library(tmap)
```

```
## The legacy packages maptools, rgdal, and rgeos, underpinning the sp package,
## which was just loaded, will retire in October 2023.
## Please refer to R-spatial evolution reports for details, especially
## https://r-spatial.org/r/2023/05/15/evolution4.html.
## It may be desirable to make the sf package available;
## package maintainers should consider adding sf to Suggests:.
## The sp package is now running under evolution status 2
##      (status 2 uses the sf package in place of rgdal)
```

```
## Breaking News: tmap 3.x is retiring. Please test v4, e.g. with
## remotes::install_github('r-tmap/tmap')
```

```
library(spDataLarge)

# load the tmap and spDataLarge packages
pacman::p_load(ggplot2, spatstat, spatialreg, spgwr, spdep, adehabitatHR, googleway,
    gmapsdistance, leaflet, dismo, raster, sp, leaflet, tmap, foreign, sf, terra,
    RColorBrewer, stargazer, rmapshaper)

# load the dataset
data("pol_pres15")

summary(pol_pres15)
```

```
##     TERYT              TERYT0             name0              name
##  Length:2495        Length:2495        Length:2495        Length:2495
##  Class :character   Class :character   Class :character   Class :character
##  Mode  :character   Mode  :character   Mode  :character   Mode  :character
##
##
##
##      gm0                         types      I_entitled_to_vote
##  Length:2495        Rural          :1563   Min.    :  1308
##  Class :character   Urban          : 303   1st Qu.:  4026
##  Mode  :character   Urban/rural    : 611   Median :  6033
##                     Warsaw Borough:  18   Mean    : 12221
##                                           3rd Qu.: 10524
##                                           Max.    :594643
##  I_voting_papers_received I_unused_voting_papers
##  Min.    :  1000          Min.    :   204
##  1st Qu.:  3500           1st Qu.:  1708
```

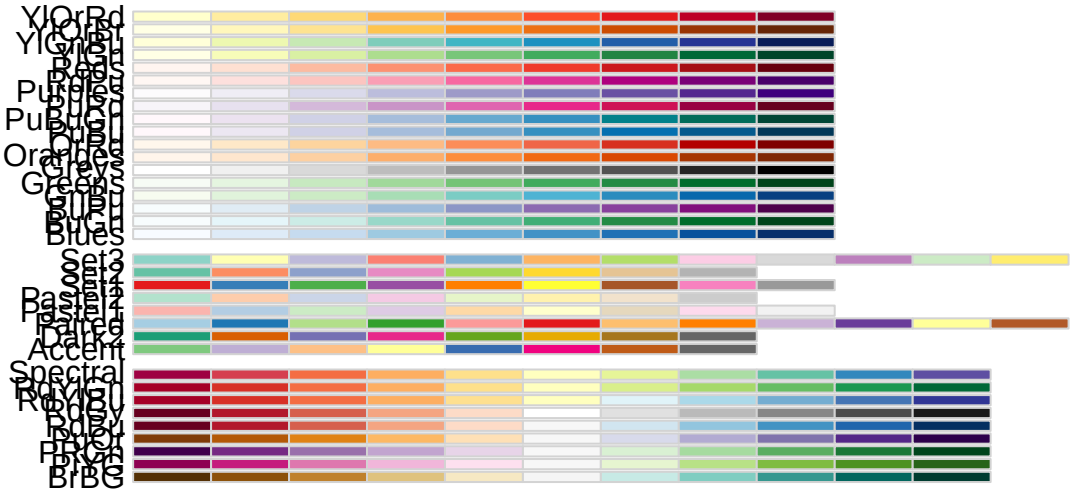
```
##  Median :  5200          Median :  2545
##  Mean   : 10530          Mean   :  4577
##  3rd Qu.:  9104          3rd Qu.:  4112
##  Max.   :505747          Max.   :200431
##  I_voting_papers_issued_to_voters I_voters_voting_by_proxy
##  Min.   :   561                   Min.   :  0.000
##  1st Qu.:  1698                   1st Qu.:  0.000
##  Median :  2654                   Median :  1.000
##  Mean   :  5954                   Mean   :  2.387
##  3rd Qu.:  4988                   3rd Qu.:  3.000
##  Max.   :338252                   Max.   :119.000
##  I_voters_voting_by_declaration I_voters_sent_postal_voting_package
##  Min.   :   0.00                Min.   :  0.000
##  1st Qu.:   7.00                1st Qu.:  0.000
##  Median :  15.00                Median :  0.000
##  Mean   :  49.98                Mean   :  3.524
##  3rd Qu.:  31.00                3rd Qu.:  2.000
##  Max.   :7583.00                Max.   :577.000
##  I_postal_voting_envelopes_received I_PVE_of_which_no_declaration
##  Min.   :  0.000                    Min.   : 0.0000
##  1st Qu.:  0.000                    1st Qu.: 0.0000
##  Median :  0.000                    Median : 0.0000
##  Mean   :  3.066                    Mean   : 0.1455
##  3rd Qu.:  2.000                    3rd Qu.: 0.0000
##  Max.   :281.000                    Max.   :34.0000
##  I_PVE_of_which_no_signature I_PVE_of_which_no_voting_envelope
##  Min.   :0.00000             Min.   :0.000000
##  1st Qu.:0.00000             1st Qu.:0.000000
##  Median :0.00000             Median :0.000000
##  Mean   :0.00481             Mean   :0.004008
##  3rd Qu.:0.00000             3rd Qu.:0.000000
##  Max.   :4.00000             Max.   :2.000000
##  I_PVE_of_which_voting_envelope_open I_voting_envelopes_placed_in_ballot_box
##  Min.   :0.000000                    Min.   :  0.000
##  1st Qu.:0.000000                    1st Qu.:  0.000
##  Median :0.000000                    Median :  0.000
##  Mean   :0.007615                    Mean   :  2.897
##  3rd Qu.:0.000000                    3rd Qu.:  2.000
##  Max.   :2.000000                    Max.   :272.000
##  I_voting_papers_taken_from_ballot_box
##  Min.   :   561
##  1st Qu.:  1698
##  Median :  2656
##  Mean   :  5956
##  3rd Qu.:  4990
##  Max.   :338444
##  I_of_which_voting_papers_taken_from_voting_envelopes I_invalid_voting_papers
##  Min.   :  0.000                                      Min.   :  0.0000
##  1st Qu.:  0.000                                      1st Qu.:  0.0000
##  Median :  0.000                                      Median :  0.0000
##  Mean   :  2.906                                      Mean   :  0.5623
##  3rd Qu.:  2.000                                      3rd Qu.:  0.0000
##  Max.   :272.000                                      Max.   :134.0000
##  I_valid_voting_papers I_invalid_votes   I_valid_votes    I_candidates_total
##  Min.   :   561        Min.   :   0.00   Min.   :   550   Min.   :   550
##  1st Qu.:  1698        1st Qu.:  13.00   1st Qu.:  1682   1st Qu.:  1682
##  Median :  2656        Median :  22.00   Median :  2631   Median :  2631
##  Mean   :  5955        Mean   :  49.57   Mean   :  5906   Mean   :  5906
##  3rd Qu.:  4990        3rd Qu.:  41.00   3rd Qu.:  4946   3rd Qu.:  4946
##  Max.   :338410        Max.   :2785.00   Max.   :335625   Max.   :335625
##  I_Grzegorz.Michal.Braun I_Andrzej.Sebastian.Duda I_Adam.Sebastian.Jarubas
```

```
##  Min.   :   0.00         Min.   :    44           Min.   :   3.00
##  1st Qu.:  10.00         1st Qu.:   665           1st Qu.:  33.00
##  Median :  18.00         Median :  1094           Median :  55.00
##  Mean   :  47.82         Mean   :  2060           Mean   :  95.49
##  3rd Qu.:  38.00         3rd Qu.:  1956           3rd Qu.:  89.00
##  Max.   :4010.00         Max.   :107367           Max.   :7154.00
##  I_Bronislaw.Maria.Komorowski I_Janusz.Ryszard.Korwin.Mikke
##  Min.   :    55.0             Min.   :    7.0
##  1st Qu.:   412.5             1st Qu.:   45.0
##  Median :   727.0             Median :   76.0
##  Mean   :  2002.5             Mean   :  189.6
##  3rd Qu.:  1455.5             3rd Qu.:  153.0
##  Max.   :127361.0             Max.   :15505.0
##  I_Marian.Janusz.Kowalski I_Pawel.Piotr.Kukiz I_Magdalena.Agnieszka.Ogorek
##  Min.   :   0.00          Min.   :   69       Min.   :   7
##  1st Qu.:   8.00          1st Qu.:  305       1st Qu.:  40
##  Median :  15.00          Median :  498       Median :  62
##  Mean   :  30.53          Mean   : 1217       Mean   : 141
##  3rd Qu.:  27.00          3rd Qu.: 1008       3rd Qu.: 114
##  Max.   :1966.00          Max.   :65702       Max.   :7605
##  I_Janusz.Marian.Palikot I_Pawel.Jan.Tanajno  I_Jacek.Wilk
##  Min.   :   0.00         Min.   :   0.00     Min.   :   0.00
##  1st Qu.:  15.00         1st Qu.:   1.00     1st Qu.:   5.00
##  Median :  26.00         Median :   3.00     Median :  10.00
##  Mean   :  83.58         Mean   :  11.69     Mean   :  27.05
##  3rd Qu.:  53.00         3rd Qu.:   7.00     3rd Qu.:  20.00
##  Max.   :5942.00         Max.   :1278.00     Max.   :2166.00
##  II_entitled_to_vote II_voting_papers_received II_unused_voting_papers
##  Min.   :  1302      Min.   :  1099            Min.   :    50
##  1st Qu.:  4042      1st Qu.:  3500            1st Qu.:  1419
##  Median :  6030      Median :  5200            Median :  2115
##  Mean   : 12205      Mean   : 10543            Mean   :  3801
##  3rd Qu.: 10510      3rd Qu.:  9106            3rd Qu.:  3431
##  Max.   :596651      Max.   :505401            Max.   :162901
##  II_voting_papers_issued_to_voters II_voters_voting_by_proxy
##  Min.   :   630                    Min.   :  0.0
##  1st Qu.:  1992                    1st Qu.:  0.0
##  Median :  3080                    Median :  2.0
##  Mean   :  6743                    Mean   :  3.6
##  3rd Qu.:  5667                    3rd Qu.:  4.0
##  Max.   :378687                    Max.   :173.0
##  II_voters_voting_by_declaration II_voters_sent_postal_voting_package
##  Min.   :    0.0                 Min.   :  0.000
##  1st Qu.:   18.0                 1st Qu.:  0.000
##  Median :   32.0                 Median :  0.000
##  Mean   :  100.5                 Mean   :  5.049
##  3rd Qu.:   68.0                 3rd Qu.:  3.000
##  Max.   :11427.0                 Max.   :537.000
##  II_postal_voting_envelopes_received II_PVE_of_which_no_declaration
##  Min.   :  0.000                     Min.   : 0.0000
##  1st Qu.:  0.000                     1st Qu.: 0.0000
##  Median :  0.000                     Median : 0.0000
##  Mean   :  4.589                     Mean   : 0.1611
##  3rd Qu.:  2.000                     3rd Qu.: 0.0000
##  Max.   :489.000                     Max.   :46.0000
##  II_PVE_of_which_no_signature II_PVE_of_which_no_voting_envelope
##  Min.   :0.00000              Min.   :0.000000
##  1st Qu.:0.00000              1st Qu.:0.000000
##  Median :0.00000              Median :0.000000
##  Mean   :0.01764              Mean   :0.008016
##  3rd Qu.:0.00000              3rd Qu.:0.000000
```

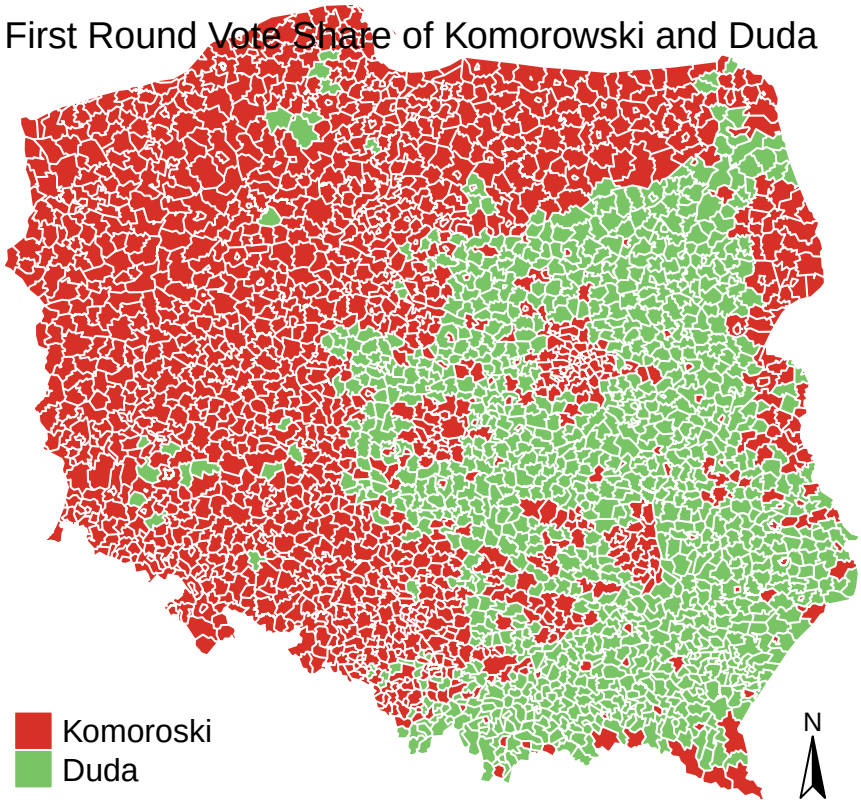
```
##  Max.   :8.00000              Max.   :2.000000
##  II_PVE_of_which_voting_envelope_open II_voting_envelopes_placed_in_ballot_box
##  Min.   :0.000000                     Min.   :  0.000
##  1st Qu.:0.000000                     1st Qu.:  0.000
##  Median :0.000000                     Median :  0.000
##  Mean   :0.008818                     Mean   :  4.386
##  3rd Qu.:0.000000                     3rd Qu.:  2.000
##  Max.   :6.000000                     Max.   :468.000
##  II_voting_papers_taken_from_ballot_box
##  Min.   :   630
##  1st Qu.:  1994
##  Median :  3081
##  Mean   :  6747
##  3rd Qu.:  5670
##  Max.   :379097
##  II_of_which_voting_papers_taken_from_voting_envelopes II_invalid_voting_papers
##  Min.   :  0.000                                       Min.   : 0.0000
##  1st Qu.:  0.000                                       1st Qu.: 0.0000
##  Median :  0.000                                       Median : 0.0000
##  Mean   :  4.385                                       Mean   : 0.3872
##  3rd Qu.:  2.000                                       3rd Qu.: 0.0000
##  Max.   :467.000                                       Max.   :66.0000
##  II_valid_voting_papers II_invalid_votes  II_valid_votes
##  Min.   :   630         Min.   :   4.00   Min.   :   616
##  1st Qu.:  1994         1st Qu.:  25.00   1st Qu.:  1967
##  Median :  3081         Median :  41.00   Median :  3041
##  Mean   :  6746         Mean   :  99.39   Mean   :  6647
##  3rd Qu.:  5670         3rd Qu.:  78.00   3rd Qu.:  5592
##  Max.   :379059         Max.   :5823.00   Max.   :373236
##  II_Andrzej.Sebastian.Duda II_Bronislaw.Maria.Komorowski          geometry
##  Min.   :   133            Min.   :    80.0              MULTIPOLYGON :2495
##  1st Qu.:  1142            1st Qu.:   676.5              epsg:2180    :   0
##  Median :  1820            Median :  1174.0              +proj=tmer...:   0
##  Mean   :  3424            Mean   :  3223.4
##  3rd Qu.:  3200            3rd Qu.:  2378.5
##  Max.   :176990            Max.   :201473.0
##    I_turnout        II_turnout      I_Duda_share     II_Duda_share
##  Min.   :0.2634   Min.   :0.3363   Min.   :0.06433   Min.   :0.1540
##  1st Qu.:0.3989   1st Qu.:0.4661   1st Qu.:0.30362   1st Qu.:0.4520
##  Median :0.4448   Median :0.5102   Median :0.38993   Median :0.5775
##  Mean   :0.4455   Mean   :0.5112   Mean   :0.40091   Mean   :0.5809
##  3rd Qu.:0.4891   3rd Qu.:0.5559   3rd Qu.:0.48993   3rd Qu.:0.7091
##  Max.   :0.6768   Max.   :0.7548   Max.   :0.78566   Max.   :0.9508
##  I_Komorowski_share II_Komorowski_share
##  Min.   :0.03747    Min.   :0.04917
##  1st Qu.:0.20285    1st Qu.:0.29087
##  Median :0.29594    Median :0.42254
##  Mean   :0.29540    Mean   :0.41915
##  3rd Qu.:0.38709    3rd Qu.:0.54796
##  Max.   :0.66901    Max.   :0.84602
```

One visualization should compare the support for Komorowski and Duda

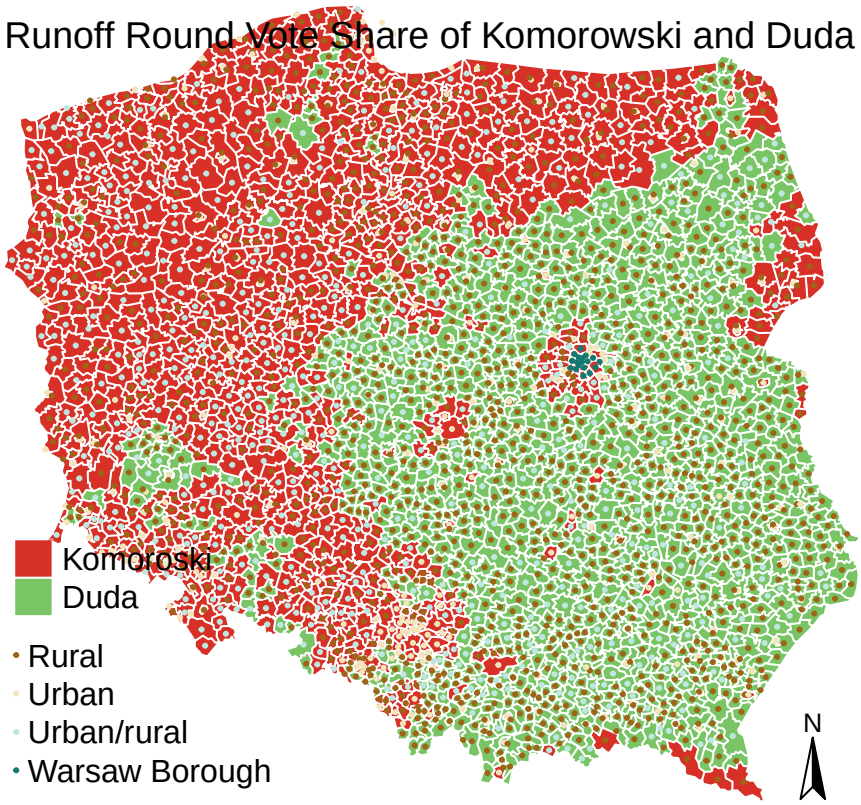
```
# setting color palette
display.brewer.all()
```



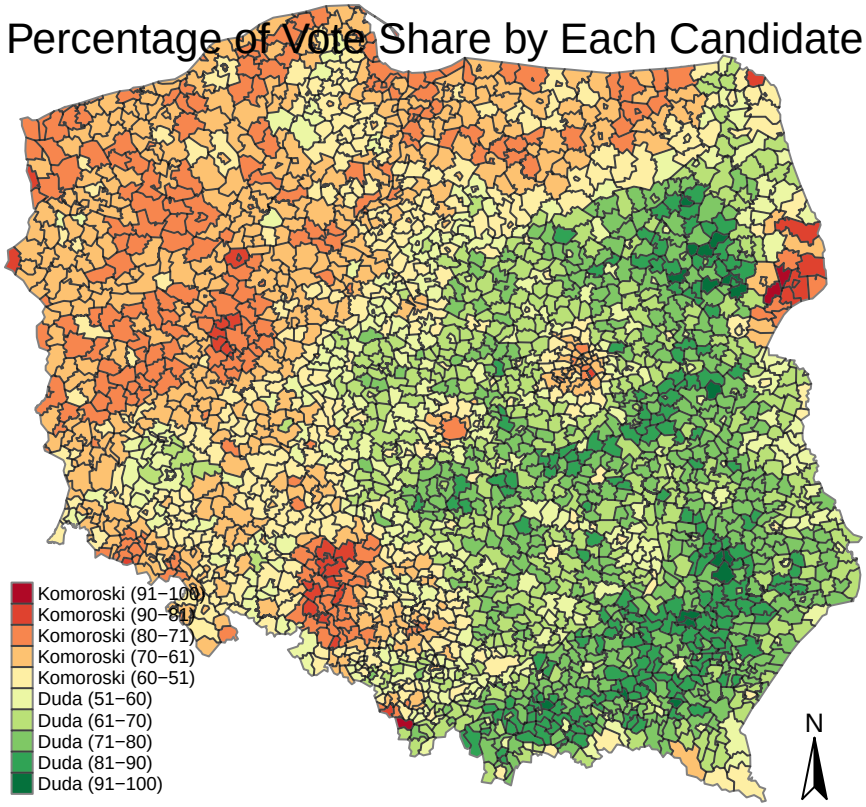

```
# first round vote share of komorowski and duda on choropleth
tm_shape(pol_pres15) + tm_fill("I_Duda_share", palette = "RdYlGn", style = "equal",
    n = 2, breaks = c(0, 0.49, 1), labels = c("Komoroski", "Duda")) + tm_borders(alpha =
    0.5,
    col = "white") + tm_compass() + tm_layout(title = "First Round Vote Share of Komorowski
    and Duda",
    legend.text.size = 1, legend.title.size = 0.01, legend.position = c("left", "bottom"),
    legend.title.color = "white", frame = FALSE)
```




```
# runoff vote share of komorowski and duda on choropleth
tm_shape(pol_pres15) + tm_fill("II_Duda_share", palette = "RdYlGn", style = "equal",
    n = 2, alpha = 1, breaks = c(0, 0.49, 1), labels = c("Komoroski", "Duda")) +
    tm_borders(alpha = 0.5, col = "white") + tm_dots("types", palette = "BrBG") +
    tm_compass() + tm_layout(title = "Runoff Round Vote Share of Komorowski and Duda",
    legend.text.size = 1, legend.title.size = 0.01, legend.position = c("left", "bottom"),
    legend.title.color = "white", frame = FALSE)
```

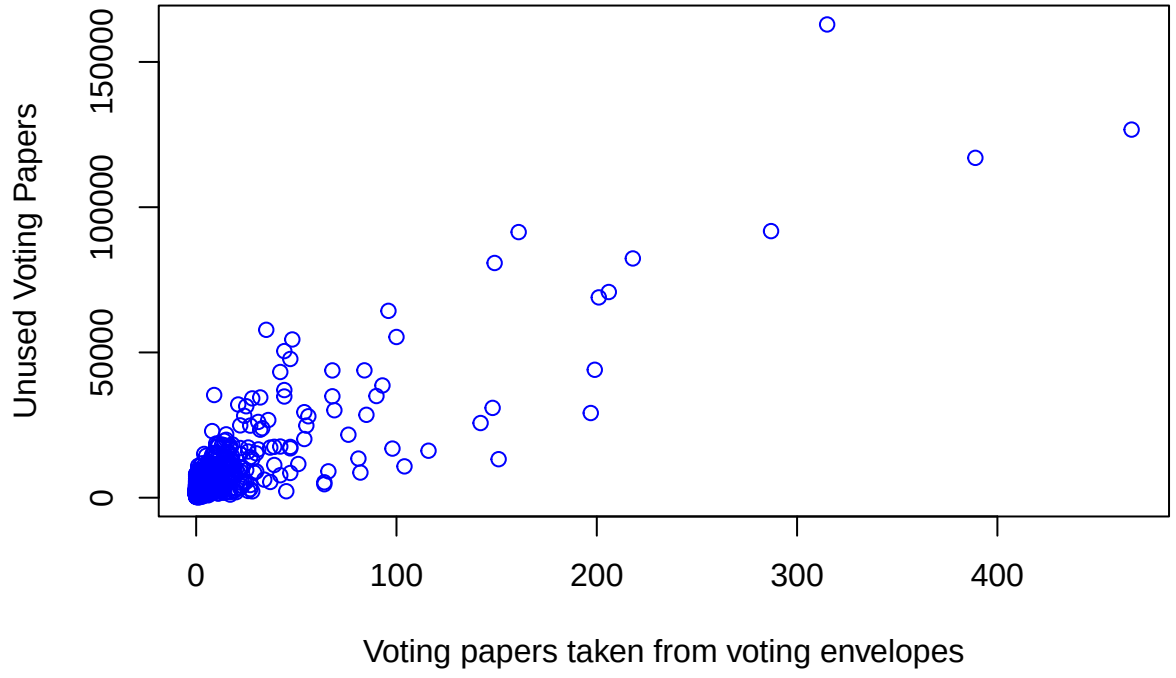



```
# percentage of vote share by each candidate
tm_shape(pol_pres15) + tm_fill("II_Duda_share", palette = "RdYlGn", style = "equal",
    n = 10, breaks = c(0, 0.49, 1), labels = c("Komoroski (91-100)", "Komoroski (90-81)",
        "Komoroski (80-71)", "Komoroski (70-61)", "Komoroski (60-51)", "Duda (51-60)",
        "Duda (61-70)", "Duda (71-80)", "Duda (81-90)", "Duda (91-100)")) + tm_borders(alpha
        = 0.5,
    col = "black") + tm_compass() + tm_layout(title = "Percentage of Vote Share by Each
    Candidate",
    legend.text.size = 0.55, legend.title.size = 0.01, legend.position = c("left",
        "bottom"), legend.title.color = "white", frame = FALSE)
```

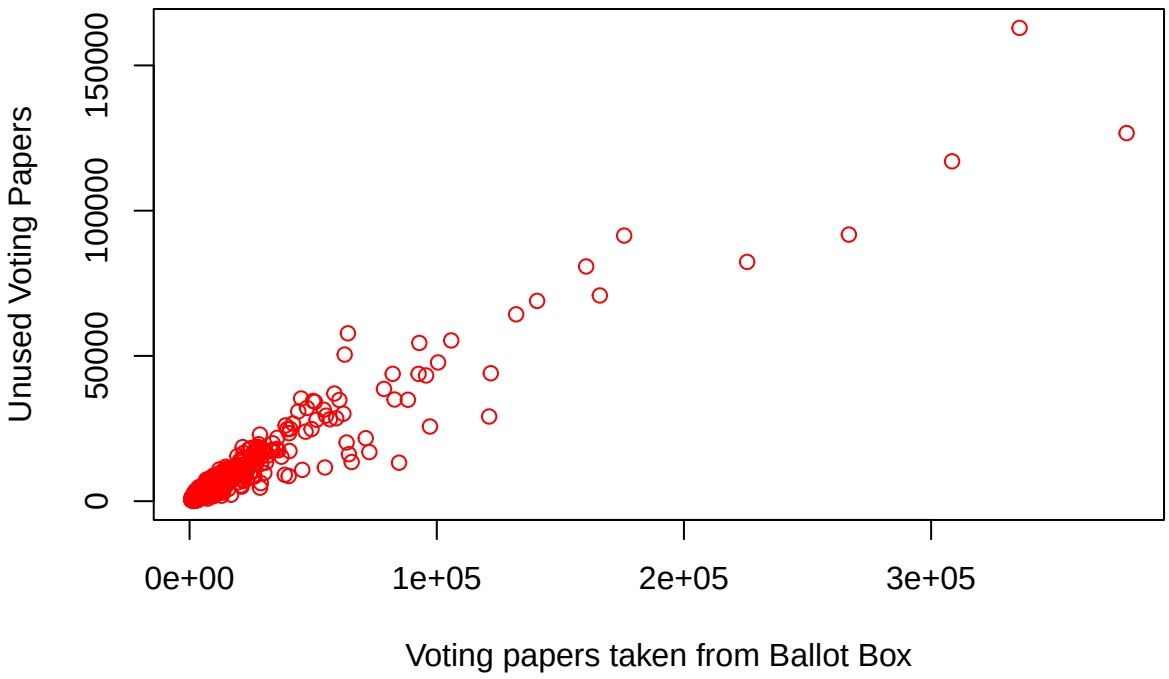


One visualization should investigate possible issues with postal voting envelopes

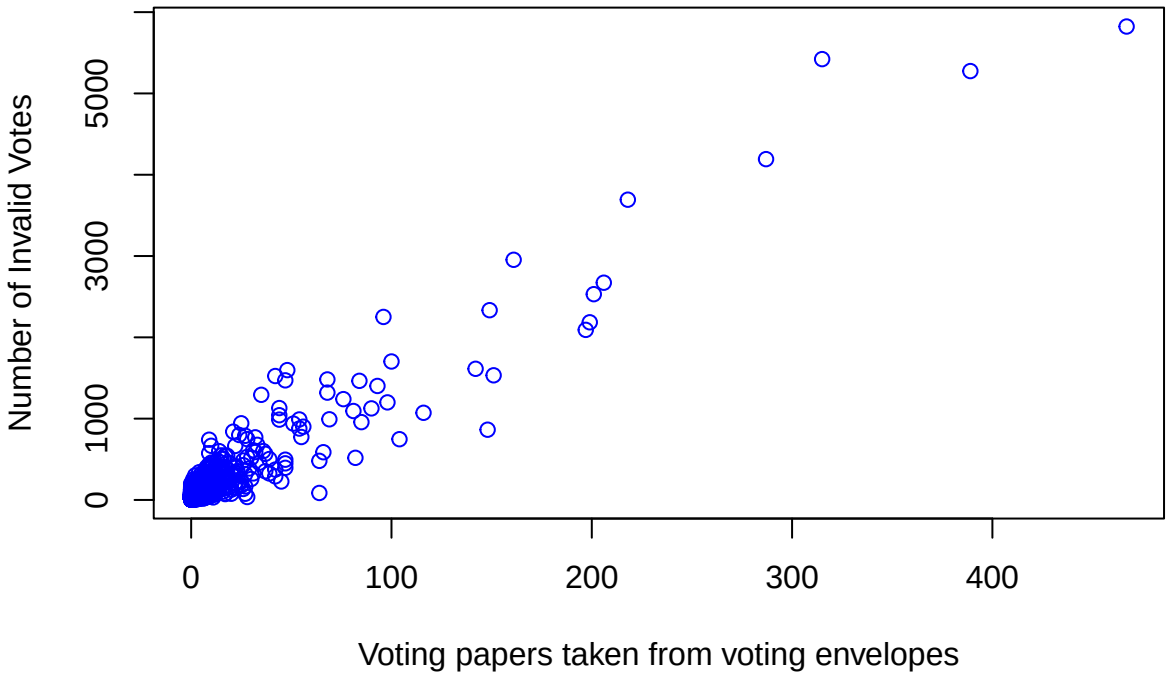
```
# correlation of unused voting papers
plot(II_unused_voting_papers ~ II_of_which_voting_papers_taken_from_voting_envelopes,
    data = pol_pres15, xlab = "Voting papers taken from voting envelopes", ylab = "Unused
    Voting Papers",
    col = "blue")
```



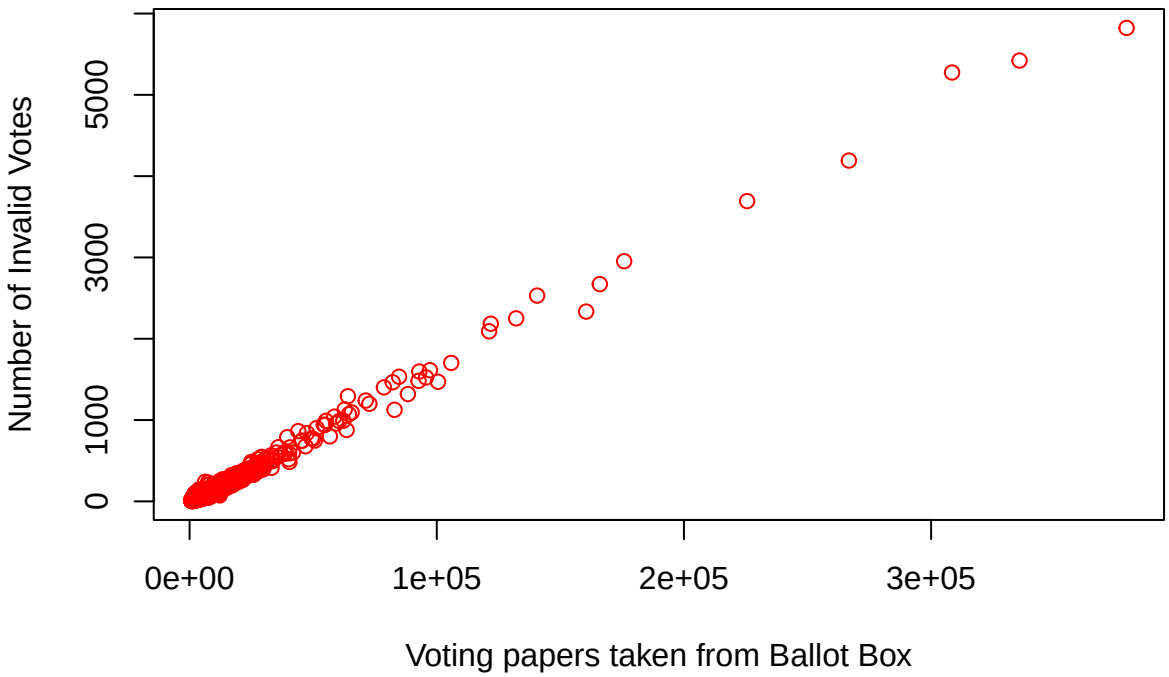

```
plot(II_unused_voting_papers ~ II_voting_papers_taken_from_ballot_box, data = pol_pres15,
    xlab = "Voting papers taken from Ballot Box", ylab = "Unused Voting Papers",
    col = "red")
```




```
# correlation of invalid votes
plot(II_invalid_votes ~ II_of_which_voting_papers_taken_from_voting_envelopes, data =
pol_pres15,
    ylab = "Number of Invalid Votes", xlab = "Voting papers taken from voting envelopes",
    col = "blue")
```




```
plot(II_invalid_votes ~ II_voting_papers_taken_from_ballot_box, data = pol_pres15,
    ylab = "Number of Invalid Votes", xlab = "Voting papers taken from Ballot Box",
    col = "red")
```


Number of Invalid Votes
0
1000
3000
5000
0e+00
1e+05
2e+05
3e+05
Voting papers taken from Ballot Box