Spatial Economics – Assignment 2

Gustav Pirich (h11910449) Peter Prlleshi () Filip Lukijanovic ()

April 2, 2024

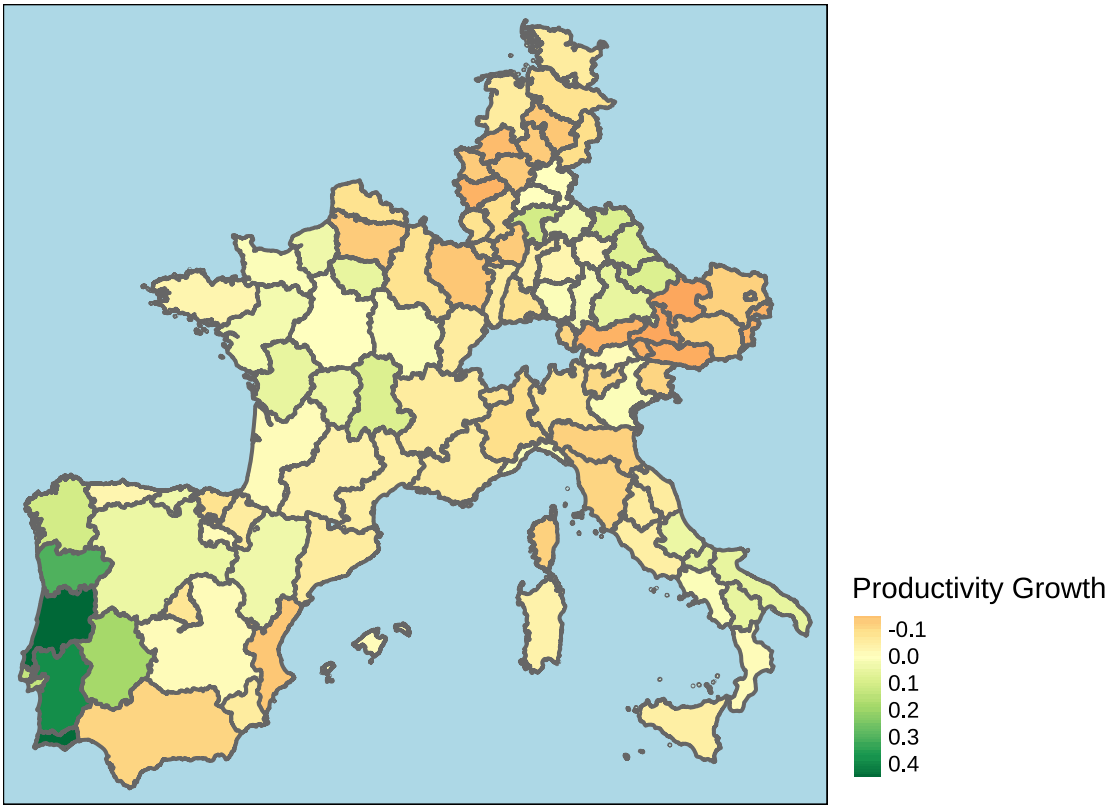
Contents

The code that was used in compiling the assignment is available on GitHub at https://github.com/gustavpirich/spatial_econ/blob/main/02_assignment/02_assignmnet.Rmd.

Exercise A

Calculate the growth rate of productivity from 1980 to 2013 and create a map that shows the productivity growth for each region.

The map shows the productivity growth rates in the NUTS-2 regions for the selected countries. We can see that many regions especially in Germany, Austria, and France exhibited negative productivity growth over the selected time period. Notably, Portugal's productivity has been growing the fastest. We suspect that the negative growth rates can be explained by the fact that high-income countries had a high baseline productivity to being with, while Portugal had a low baseline productivity. This could be evidence of convergence among productivity differences across Europe.



Generate three different spatial weights matrixes using (i) a distance threshold, (ii) smooth distance-decay, and iii) a contiguity-based measure.

(i) Distance Threshold

First, we create a spatial weights matrix based on the distance threshold criterion. Any region is being assigned a '1', if the center of any other region is less than 3 km away. Note that we have chosen this value so that every region has a neighbor. We use the nb2mat function from the 'spdep' package. We directly row-normalize the matrix.

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##  0.4377  0.8376  1.0191  1.1436  1.4705  2.6527
```

(ii) Smooth-Distance Decay

Next, we create a spatial weights matrix based on a smooth distance-decay. We use the negative exponential decay function $w_{i,j}(d) = exp(-d)$. We calculate the weights for each neighboring region based on the k=2 nearest neighbors. We do not row-normalize the matrix.

```
## Characteristics of weights list object:
## Neighbour list object:
## Number of regions: 103
## Number of nonzero links: 103
## Percentage nonzero weights: 0.9708738
```

```
## Average number of links: 1
## 31 disjoint connected subgraphs
## Non-symmetric neighbours list
## Link number distribution:
##
##   1
## 103
## 103 least connected regions:
## AT11 AT12 AT13 AT21 AT22 AT31 AT32 AT33 AT34 DE11 DE12 DE13 DE14 DE21 DE22 DE23 DE24 DE25 DE26 DE27
## 103 most connected regions:
## AT11 AT12 AT13 AT21 AT22 AT31 AT32 AT33 AT34 DE11 DE12 DE13 DE14 DE21 DE22 DE23 DE24 DE25 DE26 DE27
##
## Weights style: B
## Weights constants summary:
##     n    nn       S0       S1       S2
## B 103 10609 105.2174 214.6678 599.7385
```

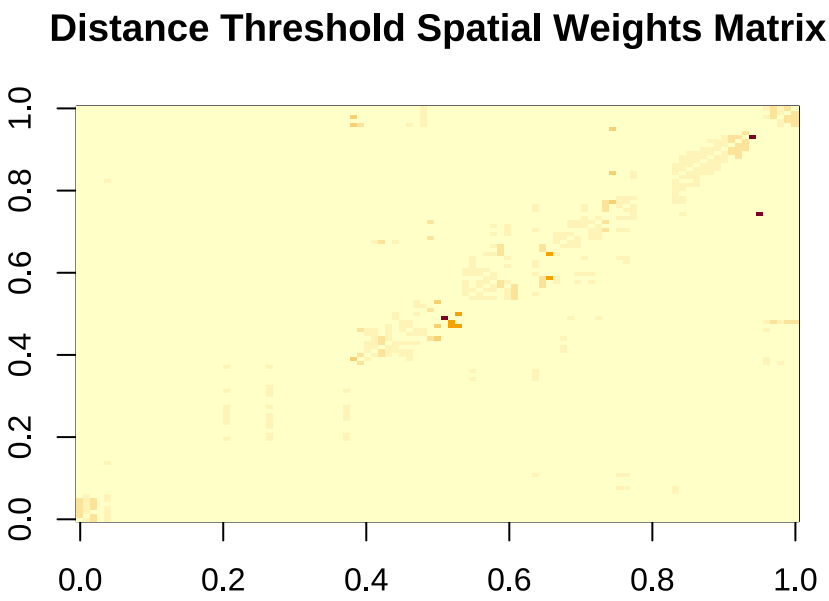
(iii) Contiguity-based measure

Finally, we calculate a contiguity based measure, which we row normalize as well.

Compare the matrices; use your knowledge of graph theory and linear algebra

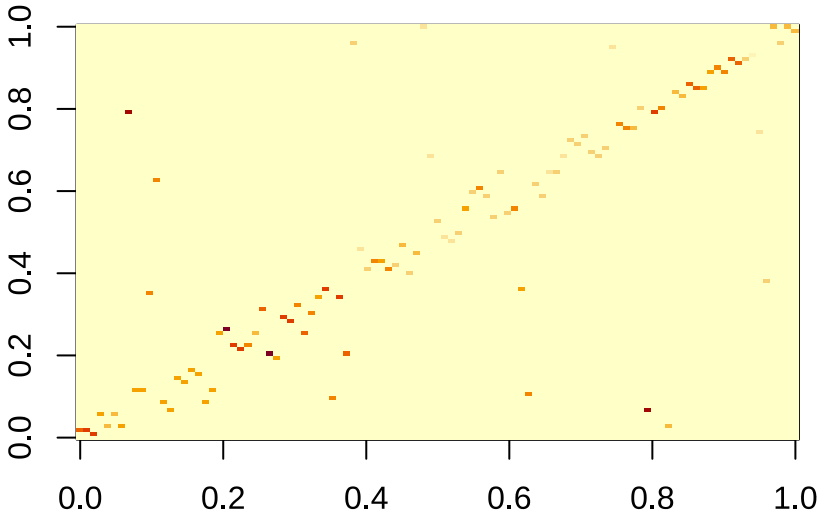
Plot the matrix

```
image(dist_w_matrix, main="Distance Threshold Spatial Weights Matrix")
```



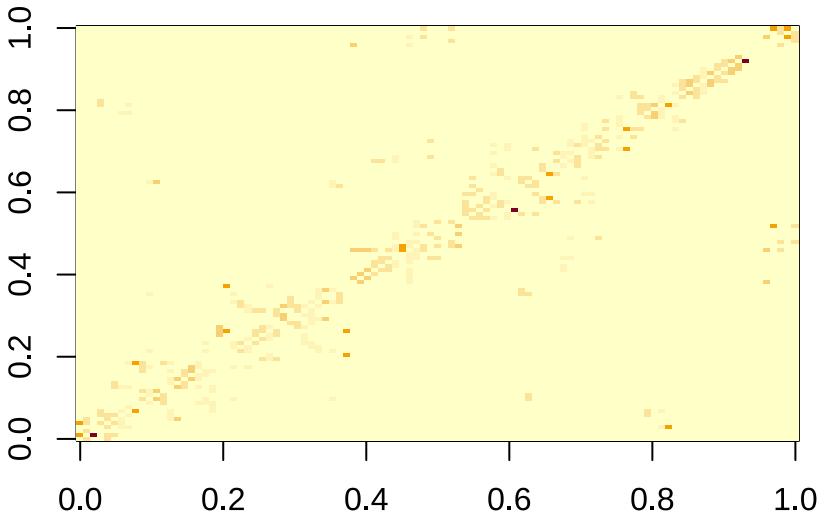

```
image(decay_weights_matrix, main="Smooth Distance-Decay Spatial Weights Matrix")
```

Smooth Distance-Decay Spatial Weights Matrix

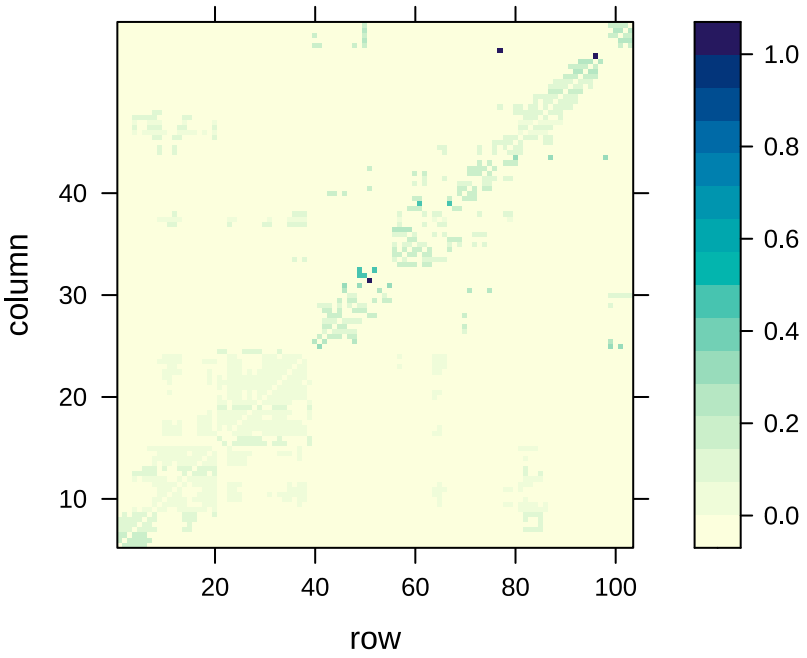



```
image(contig_w_matrix, main="Contiguity-Based Spatial Weights Matrix")
```

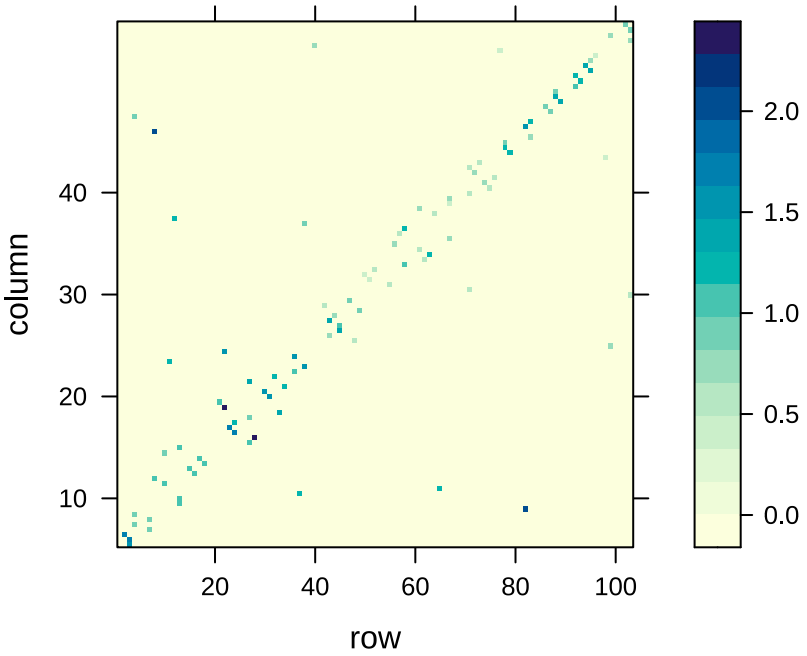
Contiguity-Based Spatial Weights Matrix



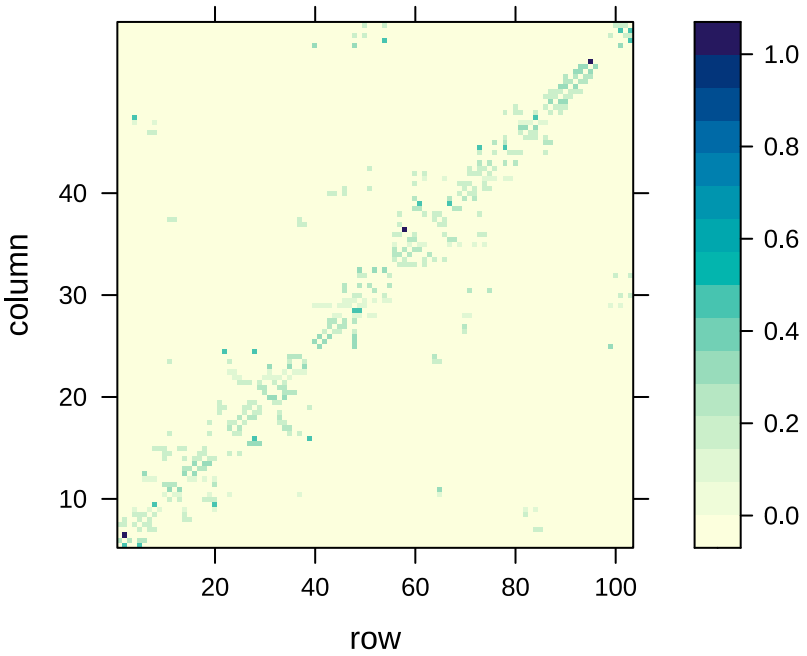

```
lattice::levelplot(t(dist_w_matrix),
scales = list(y = list(at = c(10, 30, 50, 70),
                       labels = c(10, 20, 30, 40))))
```



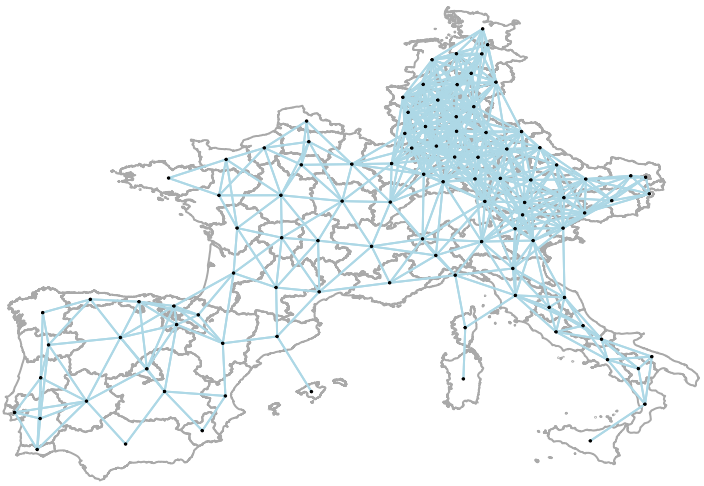

```
lattice::levelplot(t(decay_weights_matrix),
scales = list(y = list(at = c(10, 30, 50, 70),
                       labels = c(10, 20, 30, 40))))
```



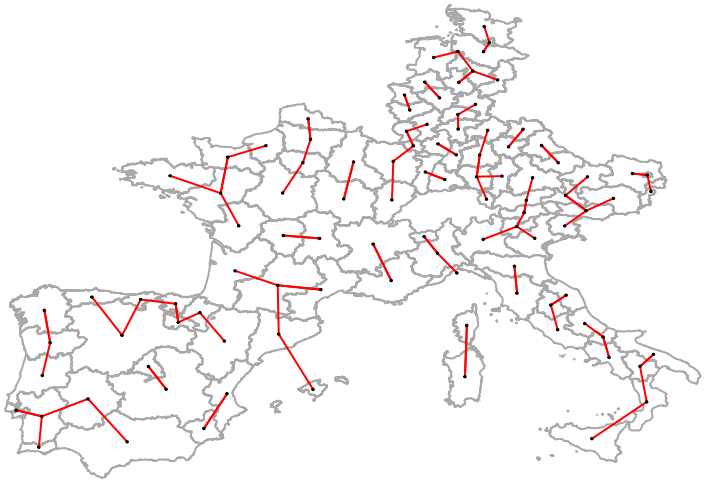

```
lattice::levelplot(t(contig_w_matrix),
scales = list(y = list(at = c(10, 30, 50, 70),
                       labels = c(10, 20, 30, 40))))
```



Try to visualize the network they represent



Let us first visualize the distance based spatial matrix.



```
# Convert to igraph objects for graph analysis
graph_dist <- graph_from_adjacency_matrix(dist_w_matrix, mode = "undirected", weighted =
TRUE)
graph_decay <- graph_from_adjacency_matrix((decay_weights_matrix), mode = "undirected")
graph_contig <- graph_from_adjacency_matrix(contig_w_matrix, mode = "undirected", weighted =
TRUE)

# Function to summarize graph properties
```

```r
summarize_graph <- function(g) {
  cat("Number of vertices:", vcount(g), "\n")
  cat("Number of edges:", ecount(g), "\n")
  cat("Average path length:", average.path.length(g, directed = FALSE), "\n")
  cat("Graph density:", edge_density(g), "\n")
  cat("Average degree:", mean(degree(g)), "\n")
  cat("Components:", components(g)$no, "\n")
}

# Analyze graph properties
cat("Distance Threshold Graph:\n")
summarize_graph(graph_dist)
cat("\nSmooth Distance-Decay Graph:\n")
summarize_graph(graph_decay)
cat("\nContiguity-Based Graph:\n")
summarize_graph(graph_contig)
```

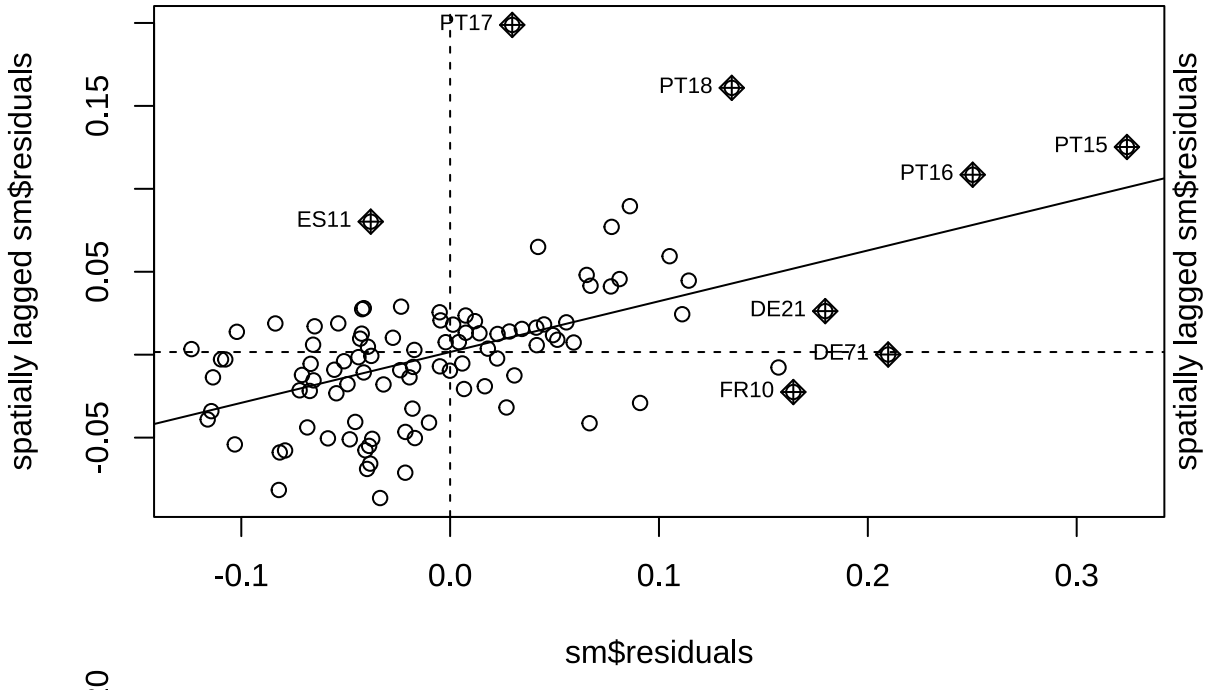
Compute a suitable measure of spatial autocorrelation for productivity growth using these matrices. Point out differences, if there are any.

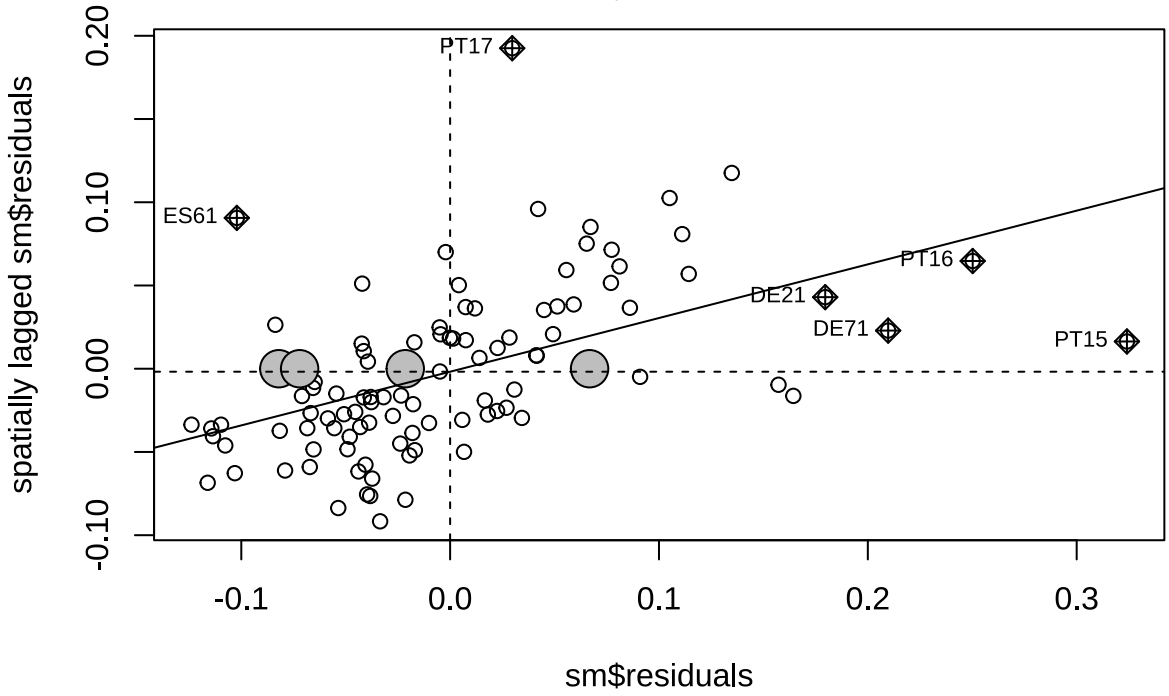
```
##             Length Class  Mode
## statistic   1      -none- numeric
## p.value     1      -none- numeric
## estimate    3      -none- numeric
## alternative 1      -none- character
## method      1      -none- character
## data.name   1      -none- character
```

Estimate a linear regression model using OLS.

Table 1:

	Dependent variable:
	prod_growth
pr80b	−0.253***
	(0.025)
lninv1b	0.032***
	(0.008)
lndens.empb	0.007
	(0.009)
Constant	0.314***
	(0.061)
Observations	103
R^2	0.528
Adjusted R^2	0.514
Residual Std. Error	0.080 (df = 99)
F Statistic	36.983*** (df = 3; 99)
Note:	*p<0.1; **p<0.05; ***p<0.01


PT17
PT18
PT15
PT16
ES11
DE21
DE71
FR10
spatially lagged sm$residuals
sm$residuals
-0.05
0.05
0.15
-0.1
0.0
0.1
0.2
0.3


PT17
ES61
PT16
DE21
DE71
PT15
spatially lagged sm$residuals
sm$residuals
-0.10
0.00
0.10
0.20
-0.1
0.0
0.1
0.2
0.3