Analysis of Northern Europe's Rail Network

Mikko Rinta-Homi, Julius Laitala and Jarkko Karttunen

28. February 2020

1 Introduction

In this paper we will analyze the rail network of Northern Europe. As our nodes we have operational railroad stations from northern Europe, and as edges the connections between the stations. Our first major analysis topic is the correlation of the rail network communities and country borders. The second major analysis topic is the correlation of the network to the population density around the stations.

In section 2, we describe the data source and how we transformed it to be suitable for our needs. In section 3, we collect the primary attributes of the network and explain what kind of network we're dealing with.

After we are familiar with our network, we start deeper analysis of it in section 4. We look into communities in section 4.2, and we deal with population densities in section 4.3. After the analysis is done, we are ready to draw our conclusions.

This project is done in Helsinki University course DATA16001 - Network Analysis. All the code used for the results here can be found in the Github repository `https://github.com/laitalaj/railanalysis`.

2 Data source and preprocessing

As our rail network source, we used OpenStreetMap. From that data we excluded every other data type except railroad- and railroad station related data. Due to memory limitations, we decided to only use northern Europe as our dataset, and coordinates used are latitudes 52 to 72 and longitudes 4 to 42.

The OpenStreetMap data consists of three kinds elements: nodes, ways and relations. Since we were not interested in relations, we just skipped them. Ways are ordered list of nodes and basically a line of edges: we selected them using the tag `railway=rail`. This way we got the railway edges and nodes, sometimes also the stations. This was fairly easy to do and also was the point when we decided our topic.

Shortly after deciding on our topic, we noticed that some of the stations are not part of the ways, but are separate nodes overlying the railroads. This was the case especially with bigger railway station complexes, which were the

Figure 1: From left to right: Helsinki area produced by the failed, geometry based simplification; Helsinki area produced by the successful, image based simplification

most interesting for us. Adding them to our rail data was luckily easy: we just included all nodes that were tagged with `railway=station`.

Since we wanted to focus on the connections between stations, we also had to simplify the network. In it's raw form, it contained every junction, spur, etc. in the network, and many routes also have more than one rail running in parallel. For our topic, this was bad since routes between two stations were often very messy, and we had to find a way to simplify these into single edges between stations. The railway stations not being an integral part of the rails also added to these problems.

We tried different kinds of methods to overcome this problem. First was to use OSMnx Python library to find the nearest nodes of the stations with reasonable distance and combine them. This required modifying the original OSMnx code a little bit, and in the end it didn't produce too good results and many stations like Pasila had side rails which bypassed the station in the final result, which we didn't want. The left hand side of figure 1 shows Helsinki area simplified with the failed simplification tactic.

Later we tried a new strategy where the whole network first got rendered as a very high resolution picture, and then based on the picture re-rendered the data. This worked out better and even though it's a little bit obscure, we decided to go with it. At the same time we got rid of the railway nodes between railway stations, which was what we initially wanted: edges representing railways and nodes representing stations. The data achieved this way seemed to be a pretty good representation of the ground truth. The right-hand side of figure 1 shows Helsinki area as it is in our final dataset, simplified using a high resolution image.

Next we noticed that mainly Sweden, but also some other countries were missing smaller stations from the final data. After some investigation, we noticed that OSMnx, the Python library, has a feature which trimmed some nodes

Figure 2: Before simplification

Figure 3: After simplification

Nodes	5014
Edges	7162
Average degree	2,8568

Table 1: Basic attributes of the network

it classified unnecessary, for example if the node was between two bidirectional edges. Without going too much into details, we added temporary self loops to those stations and managed to solve the problem by "cheating" OSMnx.

In addition to the railroad data we also wanted to have population density data in order to have more data to analyze. The source of this data is the European commission global human settlement data. The original data contains population density in 250m resolution, and since we want to combine that data to our current railroad network, we calculated the population density of the station and their adjacent tiles and saved it to the node attributes. By adding adjacent tiles we got a better picture of the size and importance of the stations. Some parts of the map had missing data - those are labelled as 0 in our final data.

3 Fundamental attributes of the network

The network is an undirected network, where nodes are operational railway stations and edges are railway lines between the stations. Edges have weight which represents the length of the railway between stations in kilometers. Nodes also contain population density of the station area. In total, the data contains about 5000 nodes and about 7200 edges. We've collected some basic attributes of the network in table 1.

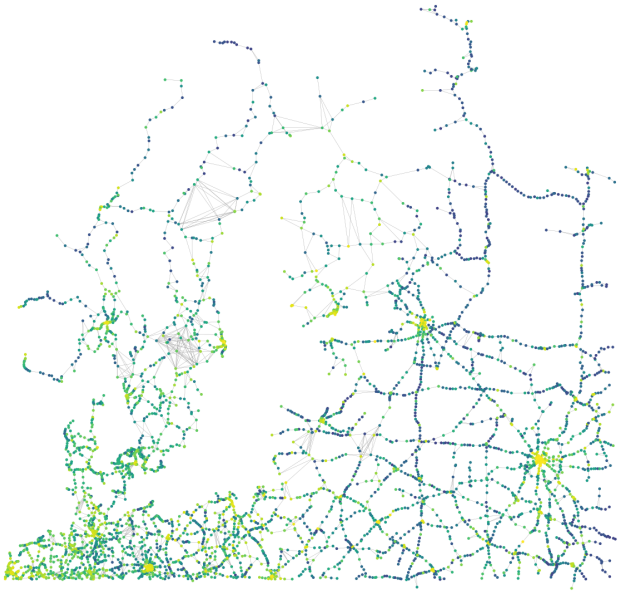

Figure 4: The north European rail network coloured according to population density. Brighter nodes signify higher population densities.

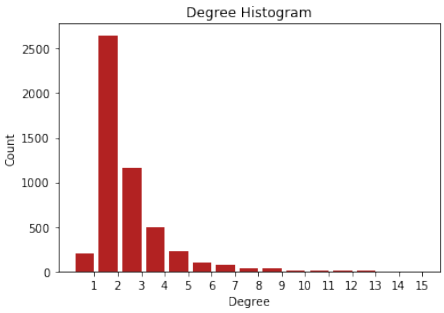

Figure 5: Degree distribution.

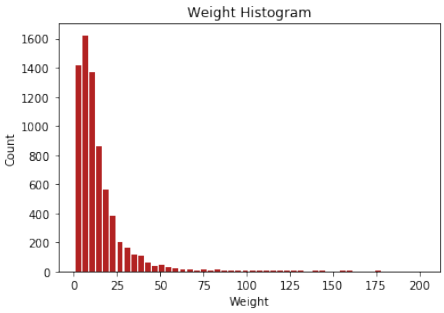

Figure 6: Weight distribution.

Mean	14.68940
Standard deviation	17.51037
Min	0.591086
Max	202.1141
Total	105205.5

Table 2: Weight statistics

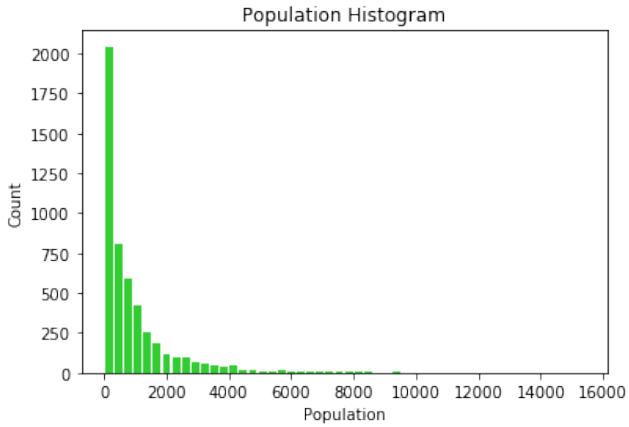

Figure 7: Population distribution.

The average degree of the nodes is 2,8. Most common degree is 2, which is the case in over half of the nodes. This is not surprising, since when visually inspecting the map, most of the stations seem to be in on the way of a long railway line. The highest degree is 15. Degree distribution can be seen in figure 5.

The mean edge length is about 15 kilometers. The shortest edge is about 0.6km and the longest is about 200km long. In total, the network contains 105000 kilometers of railroad. Most of the edges are shorter ones, and only 15% are over 25km long. Weight distribution can be seen in figure 6, and useful weight statistics are collected in table 2.

Like the other histograms, the population density is heavily focused on the smaller values. Mean is about 1000 when the maximum value is 15000. Popula-

Mean	935.4480
Standard deviation	1397.976
Min	0
Max	15387.81

Table 3: Population statistics

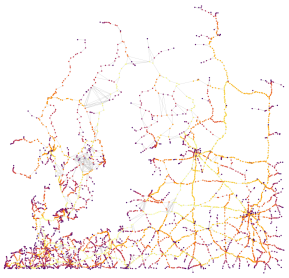

Figure 8: Betweenness centrality

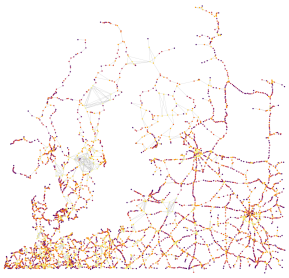

Figure 9: Degree centrality

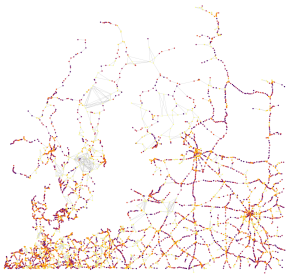

Figure 10: Weighted clustering coefficient

tion distribution can be seen in 7, and some population statistics are collected in the table 3.

4 Analyzing the network

4.1 Visualizing the network

We made several different kinds of plots of the network, aiming to highlight correlations with the population density data showcased in figure 4, but most of them didn't seem to be any good. We tried coloring the nodes based on betweenness centrality, degree centrality and clustering coefficient centrality, but none of them provided an interesting visualization of the data. We had more luck with average weighted neighbour degree graph.

Betweenness centrality (figure 8) highlights some railroads, but we couldn't see any good connection to population density. Degree centrality (figure 9) and weighted or non-weighted clustering coefficient (figure 10 are not really interesting, since high degrees and cliques randomly appear around the map and they don't provide any interesting information to us. Smaller three-node cliques happen easily when the railroad splits, and larger ones happen when this is done multiple times in the same railroad.

Weighted closeness centrality seems a little bit useless for us, too, since our data is just a portion of the Europe and basically it just highlights the south coastal area of the Baltic sea. because the distance is so great up in north and most of the data is centralized to the south.

Average weighted neighbour degree, however, does provide a more interesting graph. This graph greatly highlights some bigger cities from the map. This data inspired us to look more into the regression and try to predict population density of nodes. The network colored according to average weighted neighbour degree is included in figure 11.

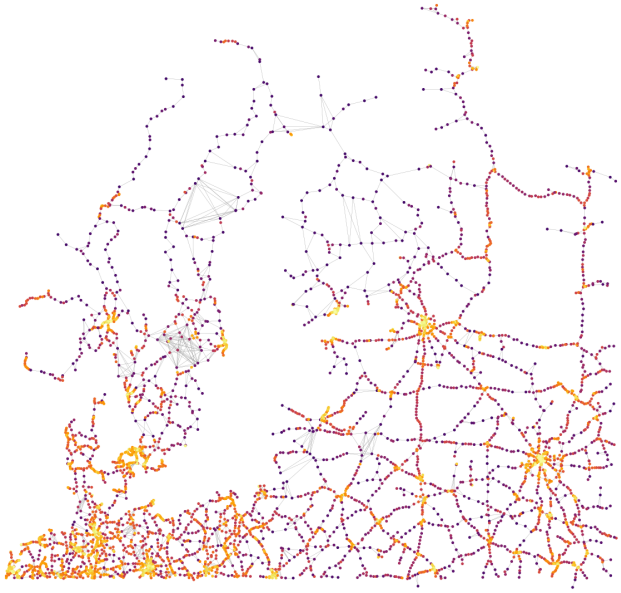

Figure 11: Average weighted neighbor degree. Brighter nodes signify higher value.

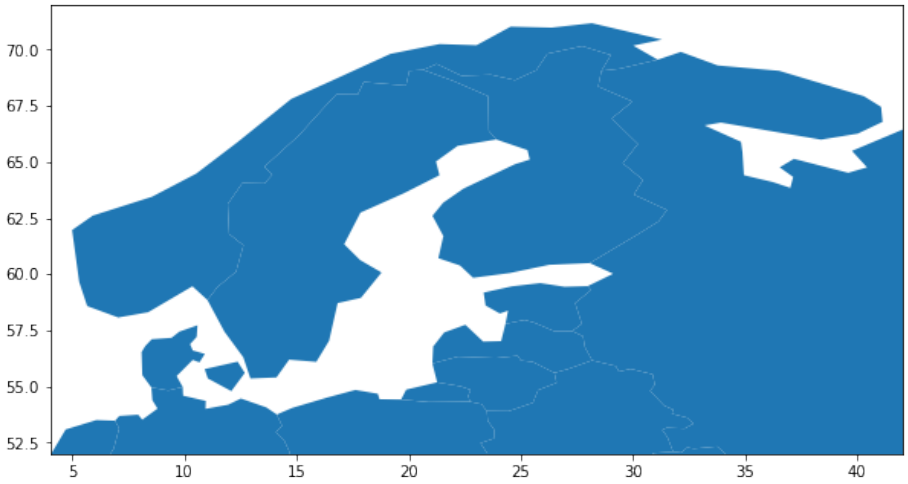

Figure 12: Map of Northern Europe.

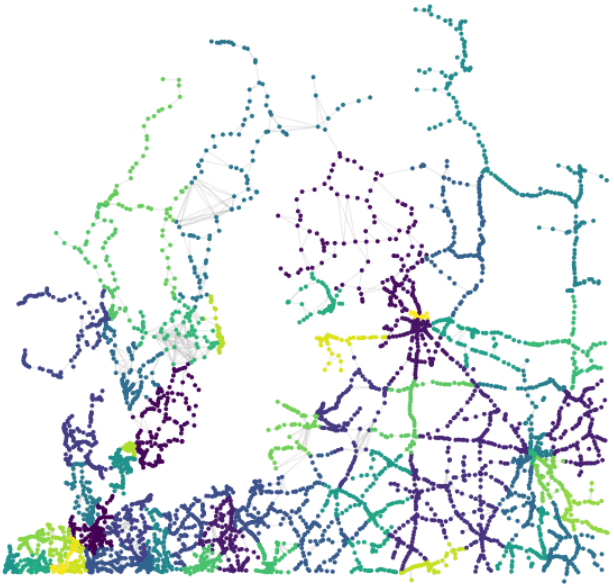

Figure 13: Clauset-Newman-Moore.

4.2 Communities

We compared and evaluated the communities generated by three different algorithms: Clauset-Newman-Moore, Girvan–Newman and Fluid communities. We then evaluated how well their results compare to the real borders between countries. For reference, a picture of northern Europe can be found in 12.

4.2.1 Clauset-Newman-Moore

Clauset-Newman-Moore finds communities via greedy modularity maximization. This method produces a mostly nonsensical map with our data; for example, it often splits real-word cities in half, as has happened in Moscow and Stockholm. The network coloured according to Clauset-Newman-Moore clusters is shown in figure 13.

4.2.2 Fluid communities

The fluid communities -algorithm finds communities by attempting to simulate the way fluids interact with each other. This method produces better results, and can be qualified as successful in some cases. For example, when generating three communities, as seen in 14, the border between Germany and Poland, and also Sweden and Denmark is clearly visible. Despite this, the borders are more often than not, completely off.

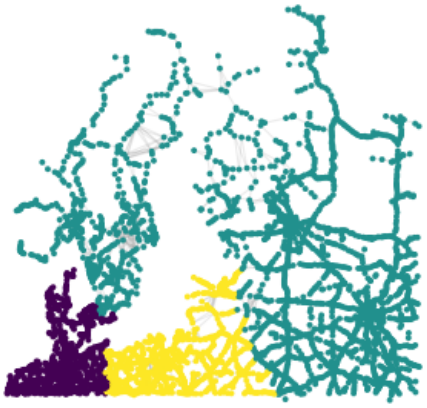

Figure 14: Three communities.

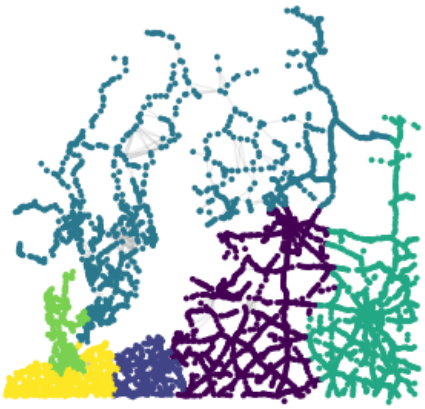

Figure 15: Six communities.

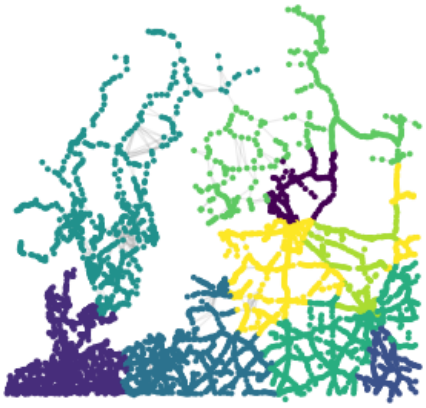

Figure 16: Nine communities.

Communities generated by fluid community detection

4.2.3 Girvan–Newman

Girvan–Newman identifies and removes edges with the highest betweenness centralities, arguing that edges between communities tend to have high betweenness centrality.

After a visual evaluation by comparing figures 12 and 19, we found that this method produced communities that fit country borders by far the most accurately. It even seems to separate the former border between East and West Germany (see 19, the shades of purple/blue may be hard to distinguish) with a fair bit of accuracy. Other examples of good predictions are the borders of the Scandinavian countries in general, and Germany, even managing to roughly split the Netherlands into it's own community. On the other hand, this does has some difficulties with Eastern Europe, making St. Petersburg a part of Finland, and splitting Russia into smaller communities on later iterations.

Because of the visually great quality that Girvan-Newman produced, we decided to delve into it further. We deduced the country each node resides in from the node's coordinates, and then calculated how many nodes in each country is in each community. As a measure of how well the communities fit country borders, we used the *total number of nodes in non-majority countries*. This measure is calculated as

$$\sum_{c \in C} n_c - m_c \,, \tag{1}$$

where C is the set of communities, n_c is the total number of nodes in community c and m_c is the maximum number of nodes belonging to just one country in community c. This measure can be seen as a function of the number of communities in figure 21.

As evident from figure 21, at 16 communities only about 500 of our around 5000 nodes belong to a community that has the majority of nodes in some other country that those 500 nodes are in. We can therefore safely say that the Girvan-Newman communities of the Northern European rail network capture country borders very well. Figure 22 further emphasizes this fact; we can see that every community has more than one third of it's nodes in a single country.

4.3 Population

We started by comparing three node attributes to node population: degree, weighted clustering coefficient and latitude. Scatter plots of these comparisons can be found in figure 23.

Degree seems to have no correlation whatsoever with node population. In order to find a possible reason as to why this is the case, let us compare two pieces of track: Kehärata and Kemi-Rovaniemi -track - both are mostly filled with nodes with a degree of two (that is, they are linear tracks with no branches), yet Kehärata has a far higher population density, whereas the Kemi-Rovaniemi -track is very sparsely populated.

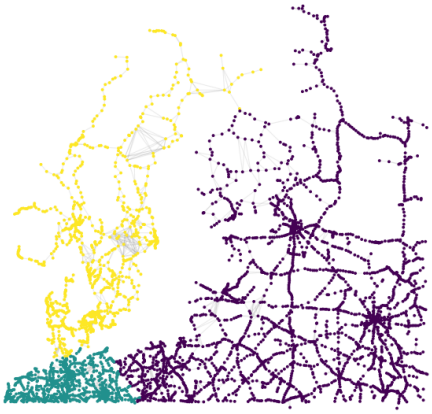

Figure 17: Three communities.

Figure 18: Six communities.

Figure 19: Twelve communities

Figure 20: Sixteen communities

Communities generated by Girvan-Newman

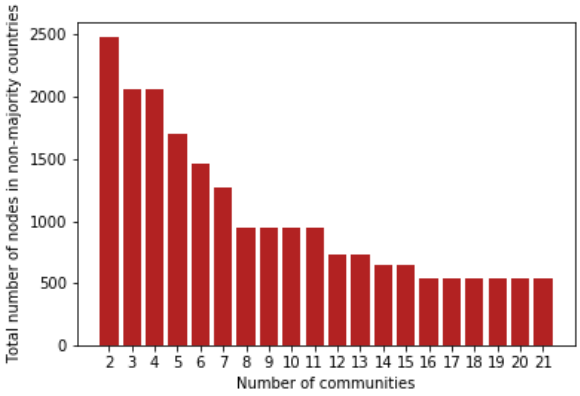

Figure 21: The number of nodes in non-majority countries (as defined in equation 1) as a function of the number of Girvan-Newman communities.

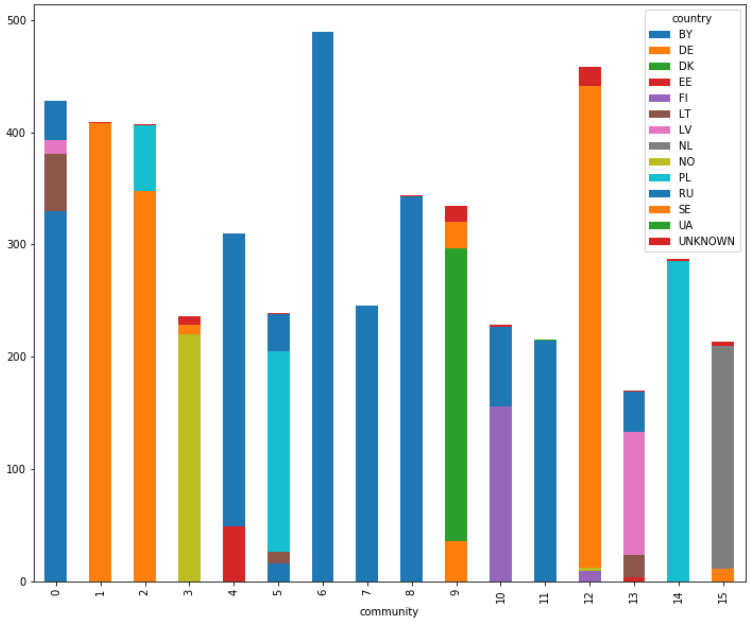

Figure 22: A bar chart of the number of countries in each community. The legend gives the ISO code of each country, and the x-axis consists of the indices of the communities.

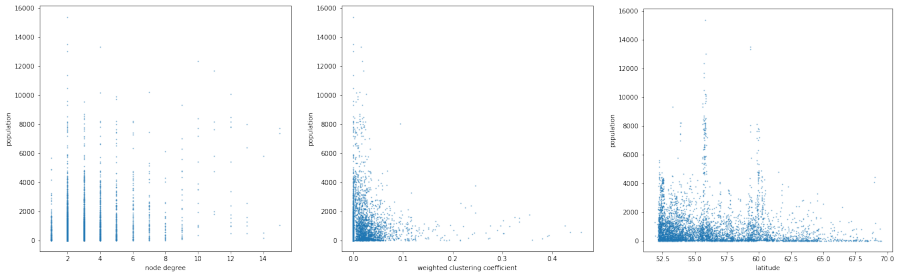

Figure 23: From left to right: population to degree, weighted clustering coefficient and latitude

Variable	**Explanation**
y	Latitude
x	Longitude
degree	The degree of the nodes
avg_neighbor_degree	Average degree of the node's neighbors
degree_centrality	Degree centrality
clustering_coefficient	Clustering coefficient
clustering_coefficient_weighted	Weighted clustering coefficient
pagerank	Pagerank

Table 4: Variables used in population prediction

Weighted clustering coefficient and population seem to be inversely proportional, as the nodes with the highest population have a coefficient close to zero, but population quickly drops off after that. For example, nodes with population greater than 5000 have a coefficient less than 0.04 in all cases but one.

Latitude and population seem to be directly proportional: as latitude increases, population decreases linearly. There are two notable spikes at latitudes 56 and 60. These are most likely explained by Copenhagen and Moscow at latitude 56, and St. Petersburg, Stockholm and Oslo at latitude 60.

4.3.1 Predicting population

As there seemed to be some link between the population density and the network structure and node attributes, we decided to try to predict the former using the latter.

We started by gathering a bunch of variables based on network structure and node attributes into one table. The variables we used are shown in table 4. We then calculated interaction terms for these variables, bringing the total number of predictors to 45. We split the nodes into a training- and a test dataset, with 4512 nodes in the training- and 502 nodes in the test dataset.

We decided on using 50-fold cross validated Lasso regression as our model. The MSE as a function of various values of α can be seen in figure 24.

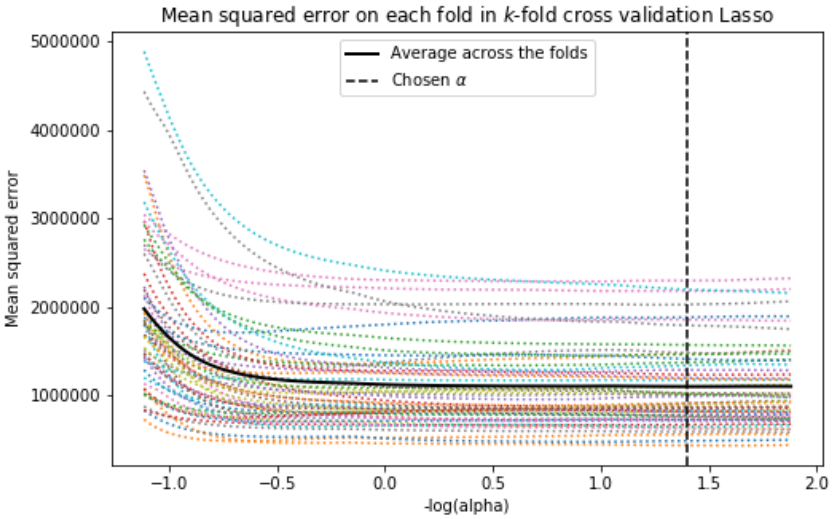

Figure 24: The Lasso cross-validation MSE with various values of α. The coloured, dashed lines represent each of the 50 folds. The α chosen by our model was ≈ 0.03991.

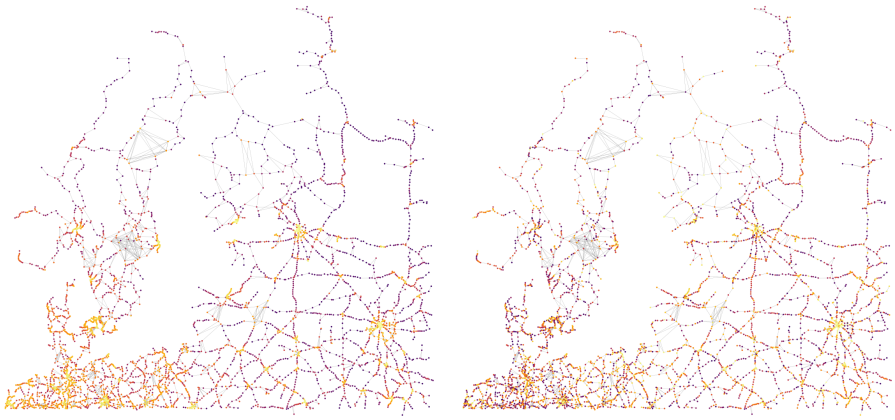

Figure 25: From left to right: Population predicted by our Lasso model, the squared error for each node. Brighter nodes mean higher values.

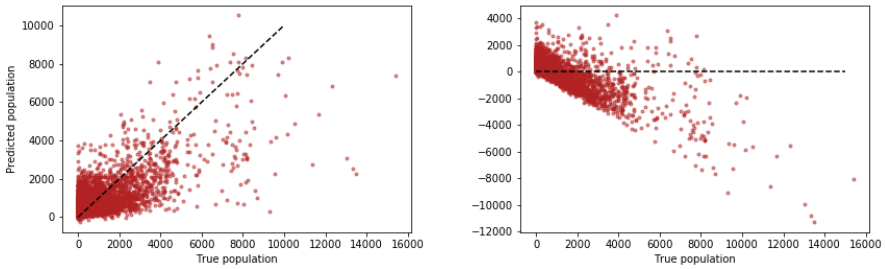

Figure 26: From left to right: Predicted population as a function of the true population, prediction error as a function of the true population. On the right hand side scatter plot the y-axis corresponds to how much and to which direction our prediction errored; a value of 1000, for example, means that our model predicted a population density that's 1000 higher than the truth, and a value of -2000 means our model predicted a population density that's 2000 smaller than the truth.

The results were better than we expected: the test MSE for our final model was ≈ 872714, less than half of the population data's variance (1953947). The model's R^2 score of 0.47 also tells us that this model can give us some idea of the population to be expected based on the railway network. Figure 25 shows us the model in action – in the left hand side we can clearly see that urban, high population areas are highlighted, while rural, low population areas are dimmer. The right hand side shows where our model's predictions are the most inaccurate.

Figure 26 gives more insight into the inaccuracies of our Lasso model. We can see that the direction of errors is quite regular: we consistently overestimate small populations and underestimate large ones. It seems like we could have achieved even better results with some further tuning of our model.

4.3.2 Population prediction accuracy and communities

As can be seen from the right hand side of figure 25, our Lasso model seems to handle the western parts of northern Europe pretty well, while Russia's area seems to have the highest error. To further investigate this and to bring together our two main analysis subjects – the communities and the population predicting model – we decided to find out how high the MSE of our model is in each of the 16 Girvan–Newman communities that we found to follow country borders very well in section 4.2.3.

As we predicted based on figure 25, the MSE in the western communities of our rail network is significantly lower than that of the eastern communities, as evidenced by figure 27. From figure 28 we can see that community number 6 (situated around Moscow) has a especially high MSE.

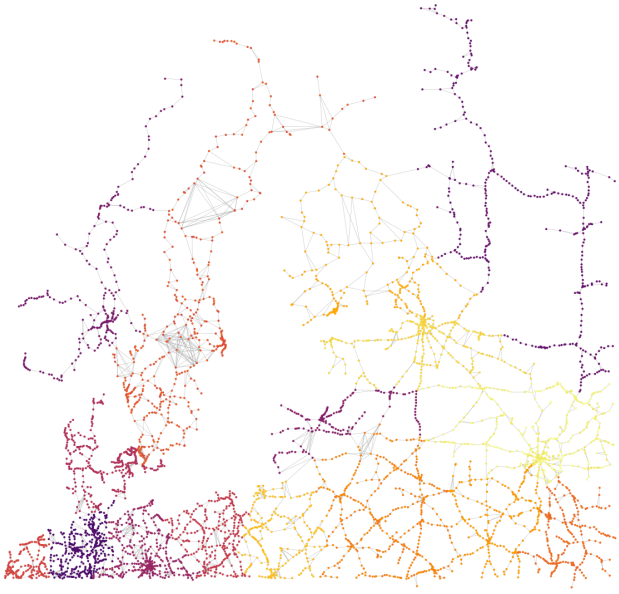

Figure 27: Our rail network data coloured according communitywise MSE. Brighter areas are communities with a higher MSE.

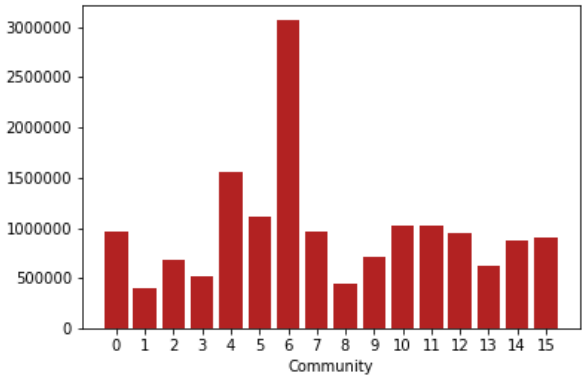

Figure 28: The MSE of our Lasso model in each of the 16 Girvan-Newman communities.

5 Conclusions

Despite the initial difficulties with transforming the OpenStreetMap data to an actually useful form, we found out, amongst other things, that it is possible to correlate population with the network topology to some extent, but it is somewhat harder in Eastern Europe. We also found out that the Girvan–Newman -algorithm is able to predict borders surprisingly well.