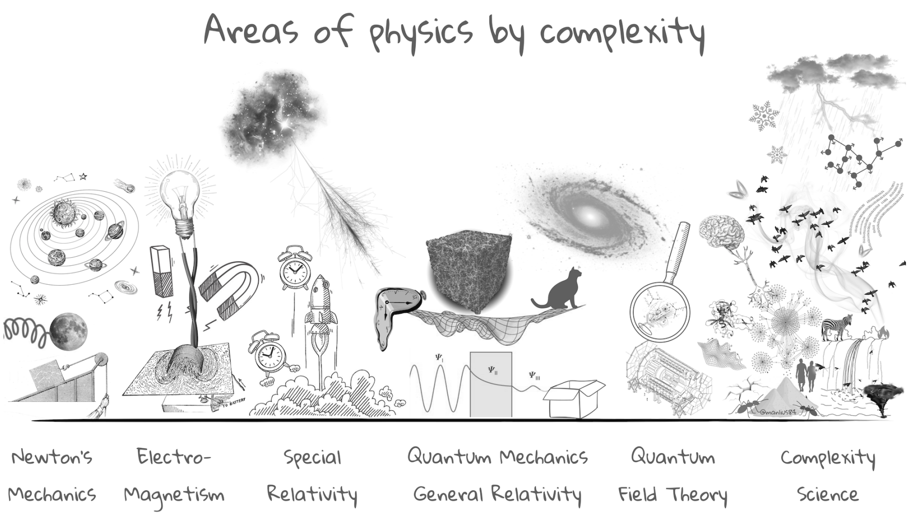
Università degli Studi di Padova



Projects PoCN: # 3, 16, 45

Chan, Robin

Last update: July 18, 2024

Final Report

Physics of Complex Networks: Structure and Dynamics

Contents

1 | #3: Potts model on arbitrary topologies

Task leader(s): *Robin Chan*

1.1 | Introduction

The Potts model forms a natural extension to the Ising model. Instead of looking at model where every element can have one of two values (*up* or *down*), the Potts model has p possible states. The Ising model is most well-known for its application to lattices, however, this task will analyse the behaviour of the Potts model on arbitrary topologies. This project is based on a paper by Dorogovtsev et al. [2] and a review paper by Dorogovtsev et al. [3].

To generate an arbitrary network, the configuration model is used where the degree distribution is parametrised as a power law $P(k) \propto k^{-\gamma}$. The Hamiltonian of this Potts model is given by:

$$H = -J \sum_{<ij>} \delta_{\alpha_i,\alpha_j} - H \sum_i \delta_{\alpha_i,1} \tag{1.1}$$

where the first sum goes over all edges and the second over all vertices. α_i refers to the state of node i ($\alpha_i = (1, ..., p)$). This project will work with $J = 1$ and $H = 0$.

To simulate this system the Metropolis Hastings Monte Carlo algorithm was used. The network starts off as a random realisation of the configuration model with random "spins" (running from 1 to p) assigned to each node. At each time step a random node is assigned a new random spin. The energy of the system is then computed. If this action has lowered the energy, the new configuration is accepted. If it does not lower the energy, the new configuration is accepted with probability $e^{-\frac{\Delta E}{kT}}$. This helps to prevent getting stuck in local minima while still sampling in a directed manner (i.e. not exploring the phase space at random).

1.2 | Results

All simulations were run on a network of 500 nodes. Let us first perform a sanity check. Does the implemented MHMC algorithm actually minimise the energy? For this the procedure is repeated 10 times for random realisations of the network. All quantities are then averaged out over these 10 iterations, resulting in Fig. 1.1.

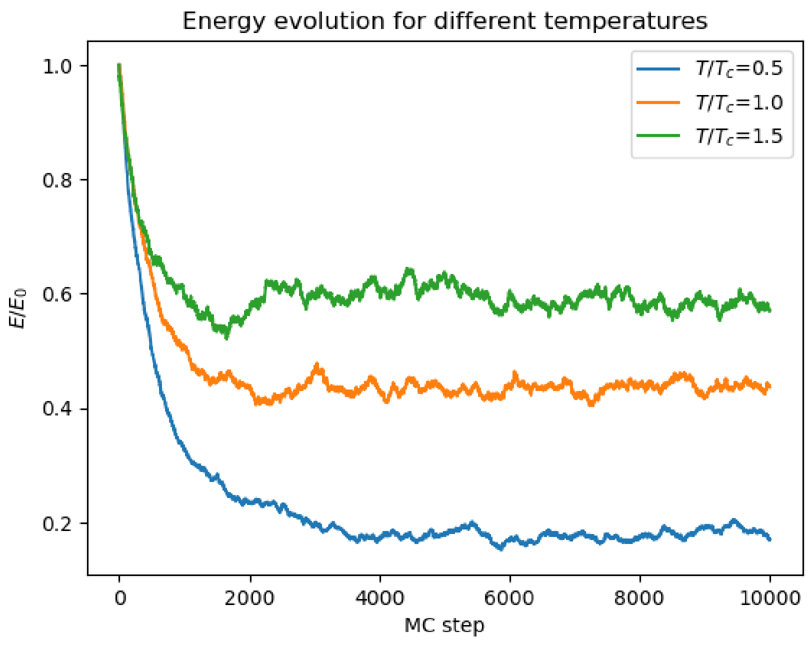


Figure 1.1: *Energy evolution of the Potts model for different temperatures*

The energy can be seen levelling off and an increase of the equilibrium energy and its fluctuations is seen for higher temperatures. This is the expected behaviour for this system indicating that our algorithm is implemented correctly.

Trying to reproduce the critical behaviour of this system resulted in some strange results. When looking at the magnetisation of the system as a function of the temperature, no clear result could be obtained. I am still not sure why. The paper's definition for the average magnetisation is used, but I am unable to interpret the result in Figure 1.2.

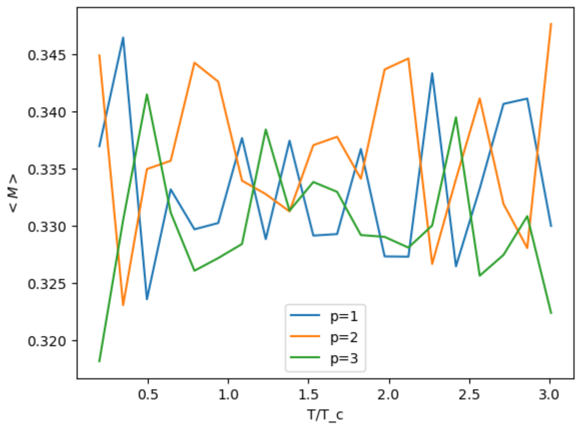


Figure 1.2: *Average magnetisation for a three state Potts model*

2 | #16: Traffic Congestion

2.1 | Introduction

For this task the transportation of packages over a network, more specifically the internet, will be modelled. To do this two approaches will be used. The first follows Arenas et al. [1]. Here packages are generated at each time step with probability p on a hierarchical tree network. These networks are characterised by z, the branching factor, and m, the number of layers. Every package gets assigned a random node as its destination and they are left to perform a random walk until they reach their destination. The probability to jump from node i to j is given by the *quality of communication* q_{ij}:

$$q_{ij} = \sqrt{k_{ij} k_{ji}} \tag{2.1}$$

with k_{ij} representing the capability of nodes i and j to communicate with each other.

$$k_{ij} = \xi_{ij} f(n_i) \tag{2.2}$$

ξ_{ij} is a uniformly distributed random between [0,1]. In the first part of the paper $\xi_{ij} = 1$, which is used in this analysis. $f(n_i)$ will be used to describe congestion in the system, depending on the number of packages in node i (n_i) at a given time step. Here we will use the following form:

$$f(n) = \begin{cases} 1, & \text{for } n = 0 \\ n^{-\gamma}, & \text{for } n = 1, 2, 3, \ldots \end{cases} \tag{2.3}$$

If the probability to generate a package p is not too high, the system should tend towards an equilibrium where package generation and delivery balance each other out.

The second approach by Echenique et al. [5], [4] uses a directed method to propagate the packages. Here the following Hamiltonian is introduced:

$$H_i = hd_i - (1 - h)c_i \tag{2.4}$$

d_i is the amount of hops required to go from node i to the destination node. c_i is the number of packets in queue on a given node. At every time step, only the top node of a queue is allowed to move, the rest have to wait. The probability for a node to move to the next is then given by:

$$\Pi_i = \frac{e^{-\beta H_i}}{\sum_j e^{-\beta H_j}} \tag{2.5}$$

For $h = 1$ the packets follow the shortest path to their destination. For $h < 1$ they start avoiding more congested nodes.

2.2 Results

Just like the previous task, quantities are averaged over 10 iterations. This task was plagued by long computation times and (for the second part) numerical instability. This means that the networks used are quite small with about 100 nodes. Still, a few results could be obtained. Let us first study the model by Arenas et al. [1]. First, the behaviour of the model is tested by studying the influence of the parameter p, the probability of generating a package in a node. It can clearly be seen that for low values of p, equilibrium can be reached, whereas larger values result in a continuous increase of the number of packages.

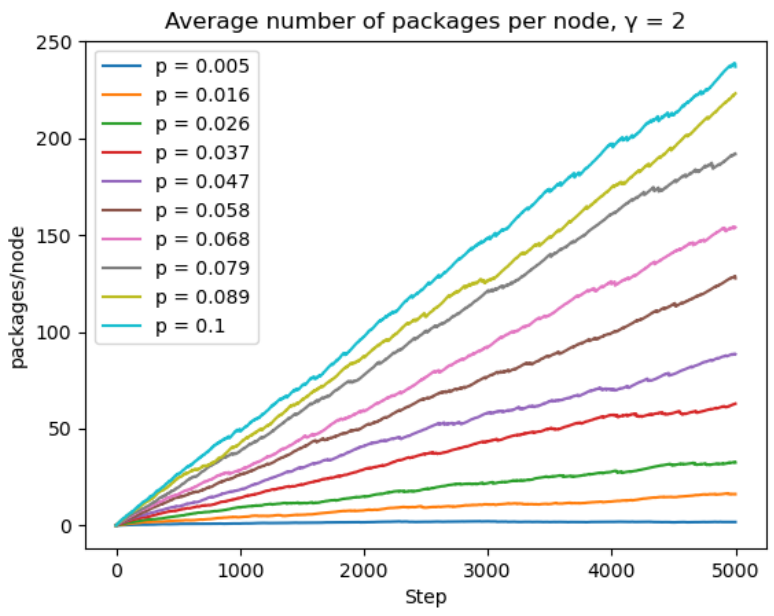


Figure 2.1: *Evolution of package numbers for different generation probabilities p*

Next, we can explore the influence of both the parameters p and γ. After a set number of steps (about 3000) the size of deviations from the number of nodes at that point was calculated at each step and averaged out. This serves of a measure of whether or not a system has reached (or can reach) equilibrium. This is better than looking at the size of the system at the end of the simulation and comparing it to the value at a set point as fluctuations near the end might significantly influence this.

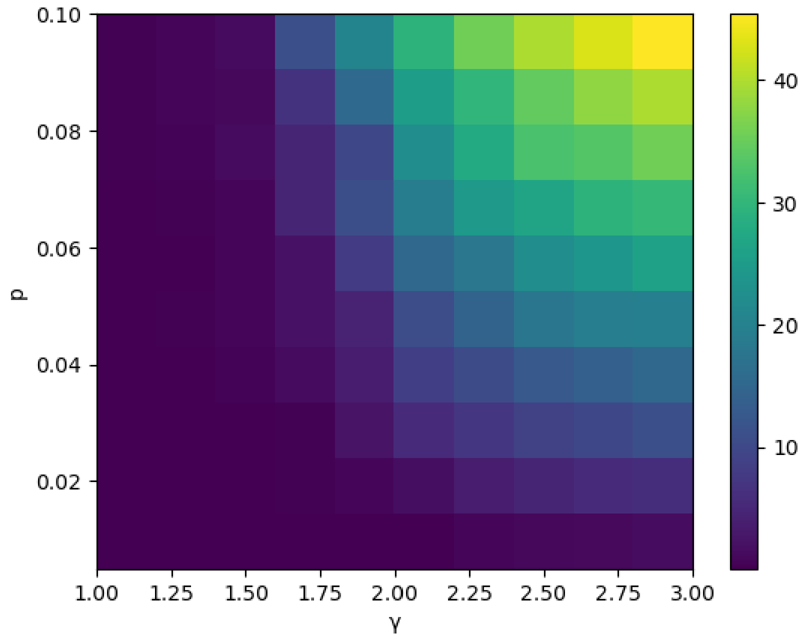


Figure 2.2: *Influence of both p and γ. The colour bar represents the average size of fluctuations around the package number at a certain step. This package value is roughly where equilibrium starts for the lower combinations of p and γ. The higher combinations result in divergence and will keep growing.*

Next the dynamics on an ER network is studied. One would expect the ER network (with a high enough wiring probability) results in a lower number of packages at equilibrium. This is what is observed in Figure 2.3.

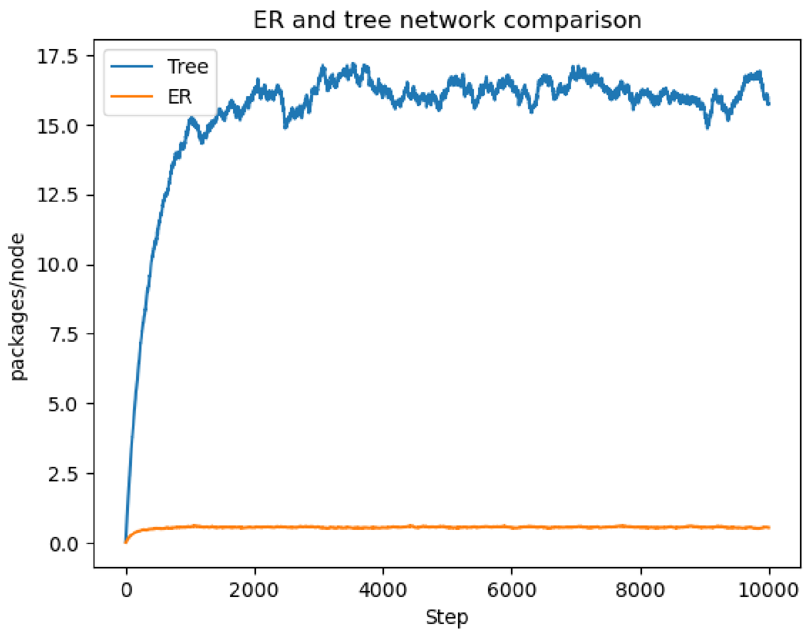


Figure 2.3: *Comparison between the package dynamics on a tree and Erdös-Renyi network*

Finally the critical behaviour of this model was studied by using the analytical expression for p_c, the critical value for p:

$$p_c = \frac{\sqrt{z}}{\frac{z(z^{m-1}-1)^2}{z^m-1} + 1} \tag{2.6}$$

and the order parameter η defined as $\eta = \frac{p-1}{p}$. The results are shown in Figure 2.4. The results are not like those obtained in the paper. I am not sure what went wrong in my modelling.

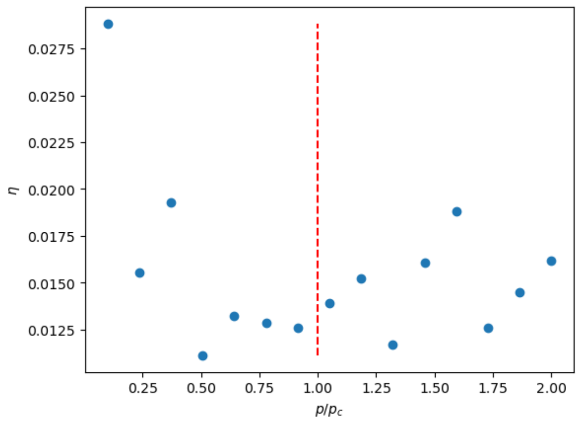


Figure 2.4: *Critical behaviour of the traffic model for z=3, m=5*

The network employed in the second model is the Barabási-Albert model. This method is giving ok results, but not fully compatible with the research papers. For congested networks one would expect the traffic aware scheme to perform better but it appears to not make a difference in my implementation. This is shown in Figure 2.5. The scaling with the number of initial packages seems reasonable as a lower density of packages allows for faster delivery.

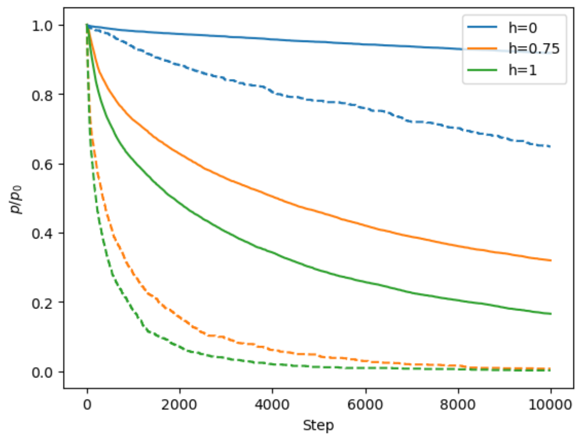


Figure 2.5: *Evolution of the number of packages normalised to the number at step 0 on a network of 1000 nodes. The full lines are for* 10^4 *initial packages, dashed for* 10^3

2.3 Runtime scaling

As mentioned, the runtime was a large obstacle for this task. To measure its scaling with the system size the simulation code was run over just a single iteration (normally 10 would be done to average out fluctuations) for differing values of z and a set value of $m = 4$. The results were then fitted with both an exponential $t = e^{\alpha z}$ and a power scaling fit $t = z^{\beta}$. The results are presented in Figure 2.6. From the fits the following values were found: $\alpha \approx 1.14$, $\beta \approx 4.05$. Extrapolating with these parameters we find that a system with branching factor $z = 10$ ($N = 1111$) would take 3 hrs (power function) or 24 hrs (exponential) for a single iteration.

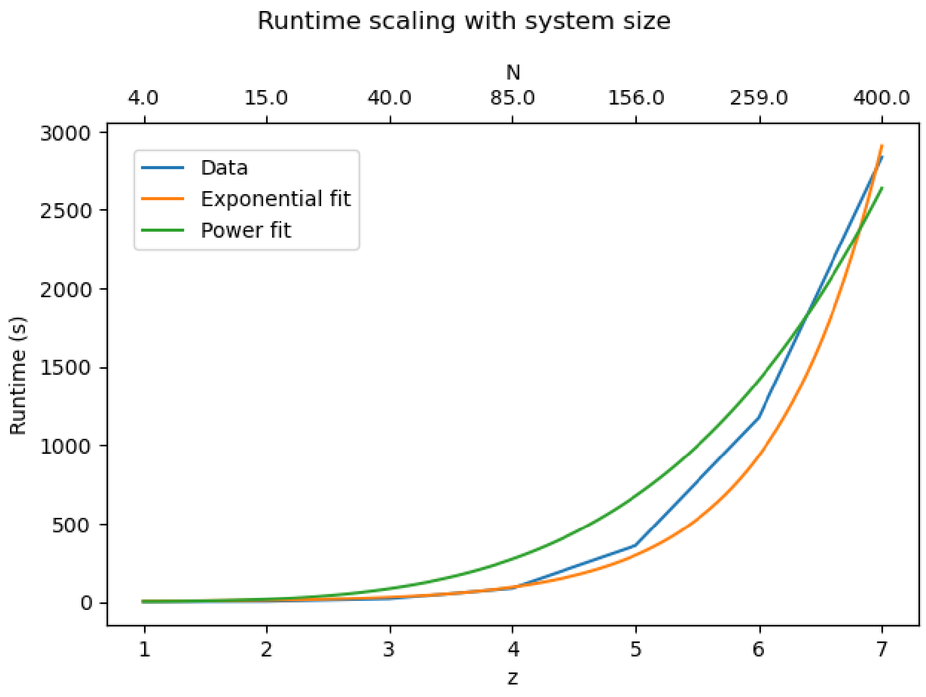


Figure 2.6: *Scaling of runtime of a single iteration for differing values of the branching factor z with a fixed number of levels m*

3 | #45: European transportation networks I

Task leader(s): *Robin Chan*

3.1 | Introduction

For this task the rail network of Europe and a selection of its countries will be generated using GIS-data provided by the professor. The data originates from EuroGeographics. It contains GIS data for administrative boundaries, hydrography, named locations, settlements and transportation. The latter is of interest for this project. The transportation data includes (not exclusively) ferry lines, airports, roads, level crossings, ... and most interesting for this project, rail lines.

To generate the rail networks, python and mainly the GeoPandas and Pandas libraries were used as the former is well-suited to working with GIS data.

3.2 | How to extract the relevant data?

The task is now to find the files containing information about the train stations (nodes) and train lines (edges). Among the wealth of GIS-files these can be identified as *RailrdC* and *RailrdL* respectively.

All the relevant information can be found in the shapefiles (.shp). At first it seemed that there was a mismatch between the coordinates of railway stations and railroad boundaries. So it was decided to link nodes to edges based on their vicinity. This, however, would not yield a physical network. Railway lines are not straight lines between stations, but curve around geographic features, population centres, etc. So not all of their boundaries will coincide with railway stations. This can result in railway lines being connected to stations that they are not actually connected to.

The next strategy was to construct the nodes dataframe using all allowed connections to edges. According to the data user manual, RailrdL can connect to RailrdC, ExitC, FerryC and LevelcC (level crossing). Then this dataframe can be filtered by looking at which coordinates correspond to edge coordinates. However, some edges have bounderies on nodes that have no label, they just indicate a location where the rail line changes direction. This causes issues linking edges to to and from coordinates of nodes as not all nodes that rail lines connect are included.

The solution is to construct the nodes dataframe based on the coordinates of the rail line boundaries. Then this can be referenced to the existing (allowed) node coordinates. This allows for labelling of the known nodes, those that do not appear in the searched nodes are labelled as 'unknown'.

3.3 Results

To get an idea if the generated network is correct, we can use matplotlib and geopandas to directly plot the network edges from the shapefile. This can then be compared to the network generated by the procedure described in the previous section.

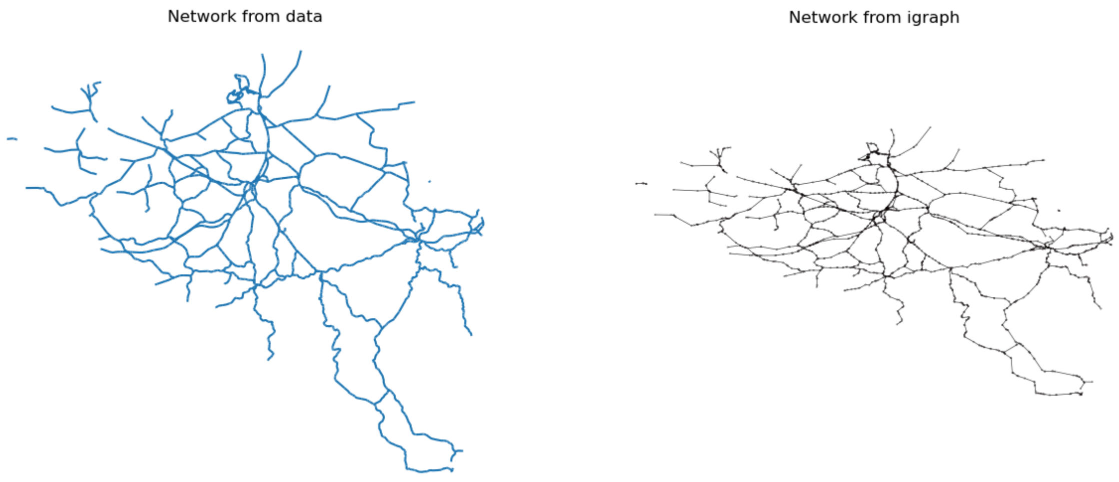
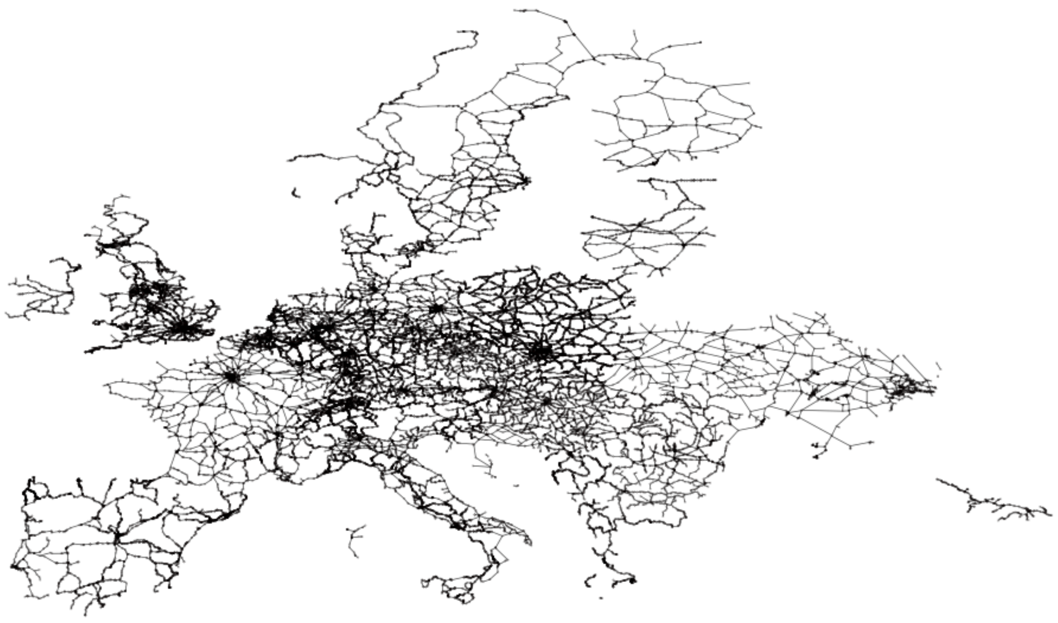


Figure 3.1: *Network shown straight from the data (left) and after the filtering procedure, visualised with igraph (right). The strange aspect ratio of the right image is probably due to the way igraph interfaces with matplotlib.*

The Belgian network was used as a way of testing the code as I am most familiar with it, being from Belgium myself. I noticed that some lines are not complete (see the northwestern corner). One of these is a major line connecting the coast to the rest of the country. This might be due to the data being taken during a period where works were being performed on the line.

European rail network



4 | Bibliography

[1] A. Arenas, A. Díaz-Guilera, and R. Guimerà. Communication in networks with hierarchical branching. *Phys. Rev. Lett.*, 86:3196–3199, Apr 2001. doi: 10.1103/PhysRevLett.86.3196. URL https://link.aps.org/doi/10.1103/PhysRevLett.86.3196.

[2] S. N. Dorogovtsev, A. V. Goltsev, and J. F. F. Mendes. Potts model on complex networks. *The European Physical Journal B*, 2004. doi: https://doi.org/10.1140/epjb/e2004-00019-y.

[3] S. N. Dorogovtsev, A. V. Goltsev, and J. F. F. Mendes. Critical phenomena in complex networks. *Rev. Mod. Phys.*, 80:1275–1335, Oct 2008. doi: 10.1103/RevModPhys.80.1275. URL http.

[4] P. Echenique, J. Gómez-Gardeñes, and Y. Moreno. Dynamics of jamming transitions in complex networks. *Europhysics Letters*, 71(2):325, jun 2005. doi: 10.1209/epl/i2005-10080-8. URL https://dx.doi.org/10.1209/epl/i2005-10080-8.

[5] Pablo Echenique, Jesús Gómez-Gardeñes, and Yamir Moreno. Improved routing strategies for internet traffic delivery. *Phys. Rev. E*, 70:056105, Nov 2004. doi: 10.1103/PhysRevE.70.056105. URL https://link.aps.org/doi/10.1103/PhysRevE.70.056105.