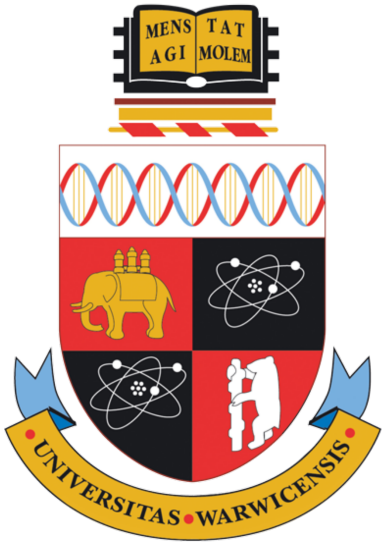


Dynamic Network Analysis and its Application to Global Trade

by

David Hooton

(1607149)

ST415 Statistics Masters Dissertation

May 2020

Supervisor:

Professor Chenlei Leng

Department of Statistics

Abstract

Political structures, such as those that dictate international trade, naturally form networks from the social and formal relationships they are comprised of. However, the tools of network science remain an underused tool in their analysis. This dissertation studies the myriad techniques for visualising and analysing dynamic networks and how these can be applied to gain insight into the ever-evolving global trade network.

We give an outline of various aspects of network science before developing a novel algorithm for dynamic graph visualisation. This is applied to both to the global trade network and to the network of international alliances. Statistical analyses are then conducted to quantify the changes to the trade network, specifically to address the changing roles geographical and political factors have in influencing it.

Contents

1 Introduction

> The power of network analysis resides in its fundamental approach to the study of social structure, and not as a bag of tools and techniques.
>
> Barry Wellman [1]

The connected and changing nature of modern society requires ever more powerful tools of analysis to understand and interrogate the relationships between key networks. Networks play important roles in the modern world, from the ubiquity of social networks to the complex trade networks that govern global commerce.

Political structures inherently form networks through their social and formal relationships, but the application of network science to them has been historically underutilised as a tool of analysis. Research will often focus on simpler aspects of the network, ignoring the significant breadth of knowledge that could be used to facilitate deeper analysis.

By leveraging the tools of network science we hope to see quantitatively in what actors, and what connections, political power resides. For instance, in a world where the only transatlantic trade was done between the USA and the UK, this link would be of great importance to global politics, and the countries representing it would draw increased power from it.

This dissertation studies the myriad challenges of analysing networks that change over time. This introduces us to the cutting-edge field of dynamic network analysis, which allows us to account for shifting power structures and provides tools to visualise these changes.

Visualisation is not only a nicety to help garner intuition of the network structure, it is necessary to foster deep theoretical insights about the data. The challenge of how to best display a network to effectively convey information has long been discussed and optimised, with many competing visualisation algorithms. This problem becomes significantly harder when adding the complexity of a network that changes over time.

In this report, sections 2 through 4 provide background on the network analysis and visualisation techniques used in this dissertation. This begins with a brief introduction to graphs and their mathematical terminology before covering more advanced network statistics, and then the AME model for network data. We then discuss different network visualisation techniques, and how they have been combined to form our novel implementation. We demonstrate our algorithms on the network of intercountry alliances. In section

5 we introduce the global trade network, and present visualisations to learn about its structures. We then conduct more formal analyses on the networked data in section 6, seeking to answer whether the effect distance has on countries' propensity to trade has changed.

Technical note

Several of the visualisations presented are in video format. Sadly PDF technology has not progressed far enough for these to be embedded directly, so still images from these videos are used as figures in the appropriate locations, with the caption providing the relevant file location in the Github repository [1]. With most PDF viewers, clicking on the captions should link there directly. These visualisations represent countries, which are referred to by their 3-letter ISO codes. A reference for these can be found in the Github as ISOcodes.md.

This Github repository also contains the code required to reproduce all results. A copy of this Github repository is submitted as the technical appendix, and this can also be used to find all the relevant visualisations and code.

[1] `https://github.com/u1607149/dynamicnetworks`

2 Network analysis

Many of the standard tools of statistical inference cannot be applied to networked data, since the usual assumption of independent and identically distributed data does not hold. Crucially, though, the network itself is what encodes the dependence structure, and so by incorporating its structure into our models we can still hope to learn from them.

Firstly, we provide a basic formulation of graphs and networks to be used throughout this dissertation. For a more comprehensive introduction, see Chapter 2 of Barabási's book *Network Science* [2].

A *graph* G is a set of vertices V, together with an edge set $E \subseteq V \times V$. We will interchangeably use the word graph, which usually refers to a mathematical abstraction, and *network*, which refers to the physical system of nodes and links, equivalent to vertices and edges. A familiar example is a social network, where the nodes are people and there are links between them if they are friends.

For our purposes, each edge must contain two distinct vertices, and the edges must form a true set in that no edge can appear twice. In a directed network the edges are ordered tuples $(u, v), u, v \in V$ with $(u, v) \neq (v, u)$, whereas in an undirected network $(u, v) \in E \Leftrightarrow (v, u) \in E$ and we can think of these as a single edge. These edges can be collected into an adjacency matrix $Y \in \{0, 1\}^{|V| \times |V|}$, where $y_{i,j} = 1$ if there is an edge from v_i to v_j and 0 otherwise.

An edge $(u, v) \in E$ may also be assigned a *weight* $w(u, v)$, a positive real number. These allow us to quantify the strength of a relationship between two nodes. These can be incorporated into the adjacency matrix, with $y_{i,j} = w(v_i, v_j)$ if there is an edge from i to j, and 0 otherwise.

If we fix a vertex u, the set of vertices it is connected to $N(u) = \{v \in V : (u, v) \in E\}$ is called the neighbourhood of u, and the vertices in it are u's neighbours.

2.1 Node importance statistics

Importance statistics can quickly outline key actors and relationships in a network. The simplest of these would be the degree of a node - the more connections it has, the more important it is to the network.

Eigenvector centrality [3] extends this idea with the notion that a node is important if it is linked to by other important nodes. This self-referential definition can be made mathematically rigorous as follows:

Let Y be the adjacency matrix of a graph G. Let λ be the largest positive eigenvalue of Y. The associated eigenvector $\mathbf{x}$ is the centrality vector, containing the eigenvector centralities of every node. In matrix form $\lambda\mathbf{x} = \mathbf{x}Y$. This is a popular measure which forms the basis of Google's PageRank algorithm for ranking search results.

2.2 AME models

AME models facilitate predictive and explanatory analysis of networked data beyond graph statistics. They are a recent development in network modelling from Peter Hoff et al. [4]. The aspect of networked data we wish to model is *dyadic data*, that is data measured over pairs of variables, naturally representing the edges of a network. Dyadic data for n actors in a network is represented by the $n \times n$ *sociomatrix* Y, equivalent to an adjacency matrix with edge weights.

The classical approach to modelling the values of the sociomatrix is an ANOVA decomposition, where the $y_{i,j}$ are considered to vary only according to row and column effects a and b, with random error ε:

$$y_{i,j} = \mu + a_i + b_j + \varepsilon_{i,j}$$

The first extension we need to make to this model is that the row and column effects are likely correlated. For example consider our $y_{i,j}$ to be trade from country i to country j. The GDP of both countries will affect trade, although possibly at different levels. Furthermore, the variables themselves are likely correlated within each pair: $y_{i,j}, y_{j,i}$ represent imports and exports, fundamentally similar quantities.

To remedy this, we keep the above model but specify distributions on a, b and ε to create the *social relations covariance model*:

$$\forall i,\ (a_i, b_i) \sim N\big(\mathbf{0}, \Sigma_{ab}\big) \qquad \text{where } \Sigma_{ab} = \begin{pmatrix} \sigma_a^2 & \sigma_{ab} \\ \sigma_{ab} & \sigma_b^2 \end{pmatrix}$$

$$\forall i \neq j,\ (\varepsilon_{i,j}, \varepsilon_{j,i}) \sim N\big(\mathbf{0}, \Sigma_{\varepsilon}\big) \qquad \text{where } \Sigma_{\varepsilon} = \sigma_{\varepsilon}^2 \begin{pmatrix} 1 & \rho \\ \rho & 1 \end{pmatrix}$$

Under this new model we can specify that the ith row mean has mean $\mu + a_i$, and the variance across the row means is σ_a^2. Similarly, the jth column has mean $\mu + b_j$ and the variance in these column means is σ_b^2.

The covariance σ_{ab} describes the linear association between the row and column means. The within-dyad variability beyond this is described by ρ, and the additional variation across dyads is approximated by σ_ε^2.

Next we would like to extend this to the *social relations regression model*, which combines the above with a linear regression approach. The dyadic variables Y are modelled as

$$y_{i,j} = \beta_d^T \mathbf{x}_{i,j}^{(d)} + \beta_r^T \mathbf{x}_i^{(r)} + \beta_c^T \mathbf{x}_j^{(c)} + a_i + b_j + \varepsilon_{i,j}$$

Here we start to incorporate predictor variables of various types. $\mathbf{x}^{(d)}$ is a 3-dimensional array incorporating d dyad-level characteristics for each (i, j), for instance the distance between countries or some measure of their diplomatic ties. $\mathbf{x}^{(r)}, \mathbf{x}^{(c)}$ represent nodal covariates, for instance GDP or population for each country. These are separated into row and column effects as it is possible that some variable could only affect connection in or out of the node - for instance tariffs affecting imports but (at least not directly) exports. These all have associated parameter vectors β.

This model can effectively represent the dyadic dependencies in the data that are lacking in a simpler regression model, however there are still network structures that this fails to account for. These are triadic and higher order dependencies, including transitivity: if $y_{i,j}, y_{j,k}$ are high then it is likely that $y_{i,k}$ is also. There is also stochastic equivalence, where groups of nodes with similar covariates tend to have similar connections in the network. These complex network effects must be accounted for by multiplicative effects between node-level characteristics.

If the required characteristics are not known, we can include latent variables $U, V \in \mathbb{R}^{n \times R}$, where R is the *rank* of these latent factors, and requires specification. The value of $y_{i,j}$ then depends on the dot product of u_i and v_j, which can be decomposed into the latent factors' size and cosine similarity. These allow adjustments to the model which improve representation of complex network characteristics, and represent the degree to which the dyadic variable cannot be explained by the existing predictors.

Including these, we get the full AME (Additive and Multiplicative Effects) model:

$$y_{i,j} = \beta_d^T \mathbf{x}_{i,j}^{(d)} + \beta_r^T \mathbf{x}_i^{(r)} + \beta_c^T \mathbf{x}_j^{(c)} + a_i + b_j + \mathbf{u_i}^T \mathbf{v_j} + \varepsilon_{i,j}$$

In order to estimate these parameters, the `R` package `amen` is used [5], which uses a Markov Chain Monte Carlo approach. This simultaneously updates the parameter values and provides Bayesian inference about them in a diagnostic plot which updates as the model is fit. These diagnostics show how well the model accounts for row, column, dyadic and higher order effects in the networked data. The Markov Chains for each parameter have a burn-in period which is discarded, and are thinned before the average value across this smaller chain is taken as the final parameter estimate.

3 Network visualisation

With the ubiquity and popularity of network modelling, the ability to analyse and digest large networks in an efficient manner has become critically important. For political networks, gaining an understanding of a network topology can prove crucial to understanding global tensions and conflicts. Relying on summary statistics is rarely sufficient to gain deep insights, and the fields of network analysis and visualisation are converging as a result.

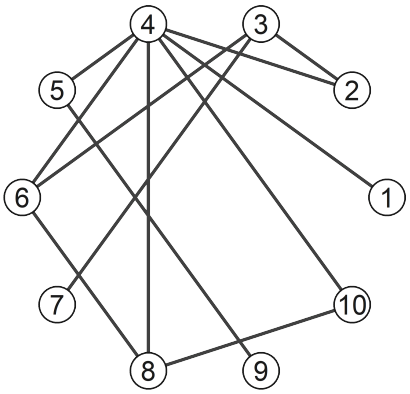
Since the sociogram was first formulated by Moreno in the 1930s [6], his actor-relationship diagram of nodes with lines between them to represent their relationships has been the cornerstone of network visualisation.

The layout of these diagrams is of critical importance to creating a clear intuition of the network. An ideal layout should position the nodes in order to reflect some aspect of the underlying graph structure, while keeping the diagram legible at a glance. Some common criteria for determining a 'good' graph visualisation are even distribution of vertices, minimal edge crossings, and uniform edge lengths. We will only consider using straight lines for the edges, and so a graph layout consists only of a placing of each node in the plane. An intuitive impression of the network topology should then emerge from the proximities of nodes in the image.

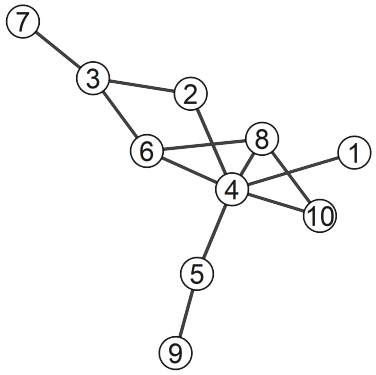
The most common layout algorithms are *force-directed*, and borrow from physics and mechanics to reformulate the problem. Some of the earliest and most widely used algorithms are Fruchterman-Reingold and Kamada-Kawai, which frame the network structure as a physical model with forces between the nodes according to their edge weights. By minimising the energy in this system, an aesthetically pleasing layout is constructed. For both algorithms iterative optimisation is used to find the minima.

Another method is the technique of multidimensional scaling whereby the nodes are laid out evenly in a high-dimensional space and then projected back down onto a 2D plane.

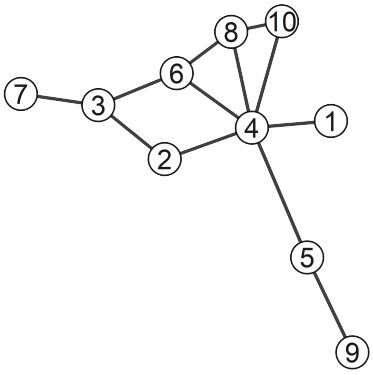
Various visualisation techniques are shown in Figure 1. Firstly, the circle layout fixes each node's position before displaying the edges, which can be useful if there is significance to these initial positions. However, in this case it appears that the force-directed algorithms give a better view of the 'shape' of the graph. The graph is planar (can be drawn without edge crossings), which the Fruchterman-Reingold layout achieves, and this



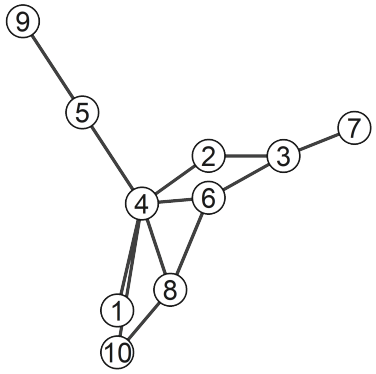
(a) Circle layout



(b) Kamada-Kawai layout



(c) Fruchterman-Reingold layout



(d) Multidimensional scaling layout

Figure 1: A graph plotted using 4 different layout algorithms

allows us to easily see the cycles in the graph. The Kamada-Kawai layout does not do this, but the edge lengths are more uniform which gives a better idea of distances within the graph. The multidimensional scaling algorithm has not performed as well, giving an overlapping edge and varying edge lengths. Often when this is used in practice it has to be adjusted to stop node and edge overlap.

The force-directed methods give pleasing results, as well as being intuitive and relatively easy to use. For this reason we will focus on this class of graph layout algorithm, and explore two of them in more detail.

3.1 Fruchterman-Reingold

This method for graph drawing [7] extends earlier work by Eades on force-directed placement; there are also commonalities with the Kamada-Kawai algorithm. Attractive forces are present between nodes connected by edges, and the strength of this force is proportional to the weight of this edge (if there are none, unit edge weights can be considered). This creates a layout where strongly linked nodes appear closest to each other, and where similar nodes will cluster together. The forces are as though there are springs for each edge, and therefore are derived by Hooke's law. There are repulsive forces introduced between nodes in close proximity, thus ensuring they do not overlap.

The attractive force between connected nodes $x, y \in V$ is

$$f_a(x, y) = \frac{w(x, y) \times d(x, y)^2}{k}$$

while the repulsive force between any two vertices is

$$f_r(d(x, y)) = \frac{-k^2}{d(x, y)}$$

Note that while the attractive force grows polynomially with the distance, the repulsive force can become arbitrarily large at short distances, imposing a harsh penalty on very close vertices to prevent them overlapping.

k is a parameter specified as $k = C\sqrt{\frac{\text{display area}}{\#\text{ vertices}}}$, where C may be changed experimentally.

The total force between any two connected vertices is then given by $f_a + f_r$, and it

can be seen that the total force is in equilibrium when

$$\frac{w(x,y) \times d(x,y)^2}{k} = \frac{k^2}{d(x,y)}$$
$$d(x,y)^3 = \frac{k^3}{w(x,y)}$$
$$d(x,y) = \frac{k}{\sqrt[3]{w(x,y)}}$$

Figure 2 shows the total effect of these forces, and how they might act on a small network.

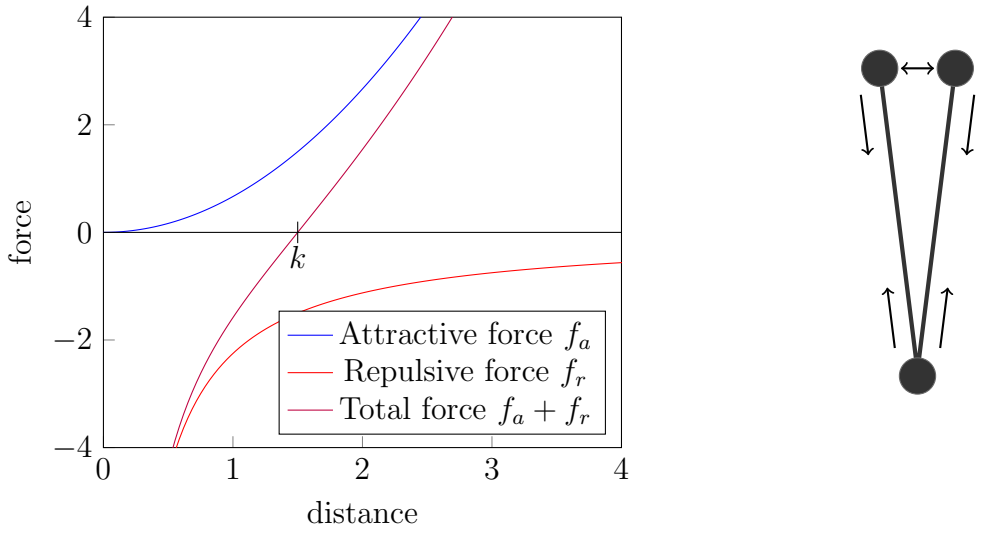


Figure 2: Forces between nodes in the Fruchterman-Reingold model

Algorithm 1 details the Fruchterman-Reingold method. It is important to note that while true forces induce acceleration on objects, in this algorithm they only induce velocities at each timestep, ensuring static rather than dynamic equilibria in the system.

Another aspect of the implementation is that repulsive forces are only calculated where $d(p_u, p_v) < 2k$. Due to the rapid decrease of the repulsive force at this distance this simplification speeds up calculations without impeding the quality of the output. Finally, in order to stop excessive movement of the vertices at any iteration, the displacement is limited by a parameter t, which decreases with each iteration to finely tune the positions. In this implementation t is decreased linearly to zero, although exponential schedules could also be used.

input : Graph $G = (V, E)$, initial node positions p_v, $v \in V$, maximum iterations $niter$, display area $A = W \times H$, max displacement T
output: Graph layout **p**

$k = \sqrt{\frac{A}{|V|}}$
$f_a : x \mapsto \frac{x^2}{k}$
$f_r : x \mapsto \frac{k^2}{x}$
for i *in* $1 : niter$ **do**
  `/* Calculate repulsive forces */`
  for v *in* V **do**
    $d_v = 0$ `/* node displacement */`
    for u *in* V **do**
      if $d(p_u, p_v) < 2k$ **then**
        $\Delta = p_v - p_u$
        $d_v = d_v + \frac{\Delta \times f_r(|\Delta|)}{|\Delta|}$
      end
    end
  end
  `/* Calculate attractive forces */`
  for (u, v) *in* E **do**
    $\Delta = p_v - p_u$
    $d_v = d_v - \frac{\Delta \times f_a(|\Delta|)}{|\Delta|} \times w_{(u,v)}$
    $d_u = d_u + \frac{\Delta \times f_a(|\Delta|)}{|\Delta|} \times w_{(u,v)}$
  end
  `/* Limit maximum displacement */`
  $t = \left(1 - \frac{i}{niter}\right)T$
  for v *in* V **do**
    $p_v = p_v + \frac{d_v}{|d_v|} \times \min(d_v, T)$
  end
end

Algorithm 1: Fruchterman-Reingold force-directed placement algorithm

3.2 Kamada-Kawai

This graph drawing algorithm was first proposed by Tomihisa Kamada and Satoru Kawai in 1989 [8], as one of the first such algorithms applicable to general undirected graphs. The basic idea is that the desired Euclidean distance between two vertices should be proportional to the graph-theoretic distance between them. The graph-theoretic distance $d_{i,j}$ between two vertices v_i, v_j is defined as the length of the shortest path between them.

A physical model is introduced, where the nodes $1, \ldots, n$ in a graph are represented by particles with positions $p_1, \ldots, p_n$ that are mutually connected by springs. The degree of imbalance in this system is therefore the energy of the springs:

$$\mathcal{E} = \sum_{i=1}^{n-1} \sum_{j=i+1}^{n} \frac{k_{i,j}}{2} \big(|p_i - p_j| - l_{i,j}\big)^2$$

Here $l_{i,j}$ is the ideal length of the spring between i and j, and can be set according to the desired distance between them in the drawing: $l_{i,j} = L \times d_{i,j}$, where L is a parameter that fixes the length of a single edge in the layout.

$k_{i,j}$ is the strength of the spring between nodes i and j. We relate this to the graph-theoretic distance earlier as

$$k_{i,j} = \frac{K}{d^2_{p_i,p_j}}$$

with K a constant.

The formula for $\mathcal{E}$ can also be interpreted as the sum of squared differences between the actual edge lengths $|p_i - p_j|$ and ideal edge lengths $l_{i,j}$.

Expanding the formula, and writing each p_i as a coordinate pair (x_i, y_i),

$$\begin{aligned}
\mathcal{E} &= \sum_{i=1}^{n-1} \sum_{j=i+1}^{n} \frac{k_{i,j}}{2} \left(d(p_i, p_j)^2 - 2l_{i,j} d(p_i, p_j) + l_{i,j}^2\right) \\
&= \sum_{i=1}^{n-1} \sum_{j=i+1}^{n} \frac{k_{i,j}}{2} \left((x_i - x_j)^2 + (y_i - y_j)^2 - 2l_{i,j}\sqrt{(x_i - x_j)^2 + (y_i - y_j)^2} + l_{i,j}^2\right)
\end{aligned}$$

We now need to minimise this energy, which is done by finding a local minimum using a method based on Newton-Raphson. The partial derivatives of energy with respect to

each coordinate are

$$\frac{\partial \mathcal{E}}{\partial x_m} = \sum_{i \neq m} k_{m,i} \left((x_m - x_i) - \frac{l_{m,i}(x_m - x_i)}{\sqrt{(x_m - x_i)^2 + (y_m - y_i)^2}} \right)$$
$$\frac{\partial \mathcal{E}}{\partial y_m} = \sum_{i \neq m} k_{m,i} \left((y_m - y_i) - \frac{l_{m,i}(y_m - y_i)}{\sqrt{(x_m - x_i)^2 + (y_m - y_i)^2}} \right)$$

We have $2n$ simultaneous non-linear equations to solve for the partial derivatives, but they are not independent so we cannot apply Newton-Raphson in $2n$ dimensions. Instead we move each node in turn using 2-dimensional Newton-Raphson, fixing all others. The node v_m with the highest value of

$$\Delta_m = \sqrt{\left(\frac{\partial \mathcal{E}}{\partial x_m}\right)^2 + \left(\frac{\partial \mathcal{E}}{\partial y_m}\right)^2}$$

is chosen to move to $p_m^* = p_m + (\delta_x, \delta_y)$, where the variables (δ_x, δ_y) satisfy

$$\begin{pmatrix} \frac{\partial^2 \mathcal{E}}{\partial x_m^2} & \frac{\partial^2 \mathcal{E}}{\partial x_m \partial y_m} \\ \frac{\partial^2 \mathcal{E}}{\partial x_m \partial y_m} & \frac{\partial^2 \mathcal{E}}{\partial y_m^2} \end{pmatrix} \begin{pmatrix} \delta_x \\ \delta_y \end{pmatrix} = \begin{pmatrix} -\frac{\partial \mathcal{E}}{\partial x_m} \\ -\frac{\partial \mathcal{E}}{\partial y_m} \end{pmatrix} \tag{1}$$

These computations terminate when the maximal Δ_m falls below a certain threshold.

input : Graph $G = (V, E)$, initial node positions p_v, $v \in V$, Δ threshold ε, constants K, L, maximum iterations *niter*
output: Graph layout **p**

Compute initial $d_{i,j}$, $l_{i,j}$, $k_{i,j}$ $\forall\, i \neq j$
while $\max_i \Delta_i > \varepsilon$ **do**
  $m = \operatorname{argmax} \Delta_m$
  while $\Delta_m > \varepsilon$ **do**
    Compute δ_x, δ_y by solving (1)
    $p_m = p_m + (\delta_x, \delta_y)$
    Update $d_{i,j}$, $l_{i,j}$, $k_{i,j}$
  end
end

Algorithm 2: Kamada-Kawai force-directed placement algorithm

4 Dynamic network visualisation

Extending the principles of graph visualisation to generate a representation over time is a challenging problem that lies at the cutting edge of current research. Many methods for dynamic visualisation are adaptations of the force-directed methods described previously, some reduce the problem to animating clusters of the nodes, and others eschew an animated approach to show the entire timeline at once.

For our purposes, a dynamic graph can be thought of as a series of graphs over the same vertex set with different edges and edge weights at each time step.

A dynamic graph visualisation needs to serve the same function as a static one - to position the nodes in such a way as to reveal an underlying structure in the graph. The novel challenge is that the local structures of the individual graphs need to be balanced with displaying the global structure of how they evolve over time.

In a sparse (with few edges) network it may be possible to leave the node position fixed and only adapt the edges; however, the networks we will consider are too dense for this method to convey useful information and so the nodes must also be moved over time. The most coherent way to visualise a succession of graphs is then an animation, where at each timestep the graph layout is changed. Simply applying static algorithms independently at each step is unlikely to lead to a satisfactory result, even when anchoring the layouts such that they all have the same centering and orientation.

The key issue is that the *mental map* [9] of the viewer is disrupted by large layout updates, and they cannot easily process the changes to the system. The nature of this has been evaluated throughout the literature by studying the performance of human participants in retaining information from various visualisations. Archambault et al. find in their meta-study [10] that preserving the mental map is only of importance where the information being presented cannot be easily understood without the aid of the graph, such as if the graph is too large. However, they also stress that this may be outweighed by the need for generally legible layouts that follow the guidelines for static graphs.

With this considered, the focus of this section is to develop an algorithm that creates a list of layouts from an input list of graphs, balancing mental map preservation with high-quality static layouts.

An exhaustive overview of dynamic graph visualisation techniques has been presented by Beck et al. [11], where they create a taxonomy of the different approaches. Static timeline-based approaches, where the whole sequence of graphs are shown at once, lose

their usefulness when considering a large number of complex graphs and so we do not consider them. Of the animation-based approaches, we focus on the more flexible class of general-purpose layouts that can be readily applied to a wide range of graphs. Many of the ideas surveyed are appropriate only in specific domains, or for certain classes of graphs. The need to visualise large undirected graphs necessitates a general algorithm.

4.1 The algorithm

The algorithm we develop is inspired by the work of Frishman and Tal [12]. Much of their work involves computational optimisations for very large graphs which we may ignore, however the key concepts are taken from their paper. An outline of the method is given in Algorithm 3. More layouts are generated than graphs; intermediate updates and interpolation are used to assist animation and mental map preservation.

This description has been left intentionally vague to capture the core structure of the algorithm. Firstly, there is the *layout* phase, where for each time t the node positions are updated R times to reflect the structure of the new graph. Next there is the adjustment phase, which is to prevent rapid node movement and assist with mental map preservation. The layouts are interpolated for a smooth animation. The final layouts $\hat{p}$ can then be used to plot the graphs and create a video.

The layout phase uses force-directed updates of the current layout. We pass on the node positions from the previous graph, and update them according to the structure of the new graph. A single update can be thought of as Algorithm 1 for Fruchterman-Reingold with $niter = 1$ and a small maximum displacement, or Algorithm 2 for Kamada-Kawai where the number of iterations of the while loop is capped at 1. This will move the old layout towards the optimum for the new graph. Here we can use the parameter R, as well as the maximum displacement, in order to balance strong individual layouts with mental map preservation. We also center every layout by subtracting the median x and y coordinate positions of nodes in the layout.

The layout phase is the most crucial one to the algorithm's design, since it is where we encode the graph structure into the layouts. Since force-directed algorithms work by iteratively updating the layouts until they converge, by making a few small updates for each graph we should capture any structural changes to the graph while being able to keep track of the node movements.

input : Graphs $G^{(t)} = (V^{(t)}, E^{(t)}), 0 \leq t \leq T$, initial node layout $p^{(0)}$, number of intermediate updates R, number of interpolated layouts K
output: Graph layouts $\hat{p}^{(j)}, 1 \leq j \leq RKT$

for $t \in \{0, \ldots, T-1\}$ **do**
  for $r \in \{1, \ldots, R\}$ **do**
    `/* Force-directed update of the old layout with information from the new graph */`
    $p^{(R \cdot t + r)} = update(p^{(R \cdot t + r - 1)}, G^{(t+1)})$
  end
end

`/* Adjust the layout sequence for better stability */`
$\tilde{p} = adjust(p)$

for $i \in \{0, \ldots, RT\}$ **do**
  for $k \in \{1, \ldots, K\}$ **do**
    `/* Interpolate the layouts for smooth animation */`
    $\hat{p}^{(iK+k)} = \frac{k}{K}\tilde{p}^{(i+1)} + (1 - \frac{k}{K})\tilde{p}^{(i)}$
  end
end

Algorithm 3: Outline of the dynamic graph layout algorithm

For the adjustment phase, polynomial smoothing splines are fit to the x and y coordinate paths of each node separately, and their fitted values are used as the new coordinates of the layout. This is a practical approach which greatly improves the appearance of the resulting visualisations and mental map preservation.

Spline regression is a non-parametric regression technique that gives flexible, smooth fits. The input space is divided into regions at points called knots, then a separate polynomial regression model is fitted in each region, such that all the models smoothly join up at the knots.

An example of how spline regression works on the layout data can be seen in Figure 3. The initial layout updates lead to rapid oscillations in node position, but the spline smoothing curve alleviates this problem. The fit is smooth enough that we can use linear interpolation between the fitted values rather than make more predictions, saving computation time.

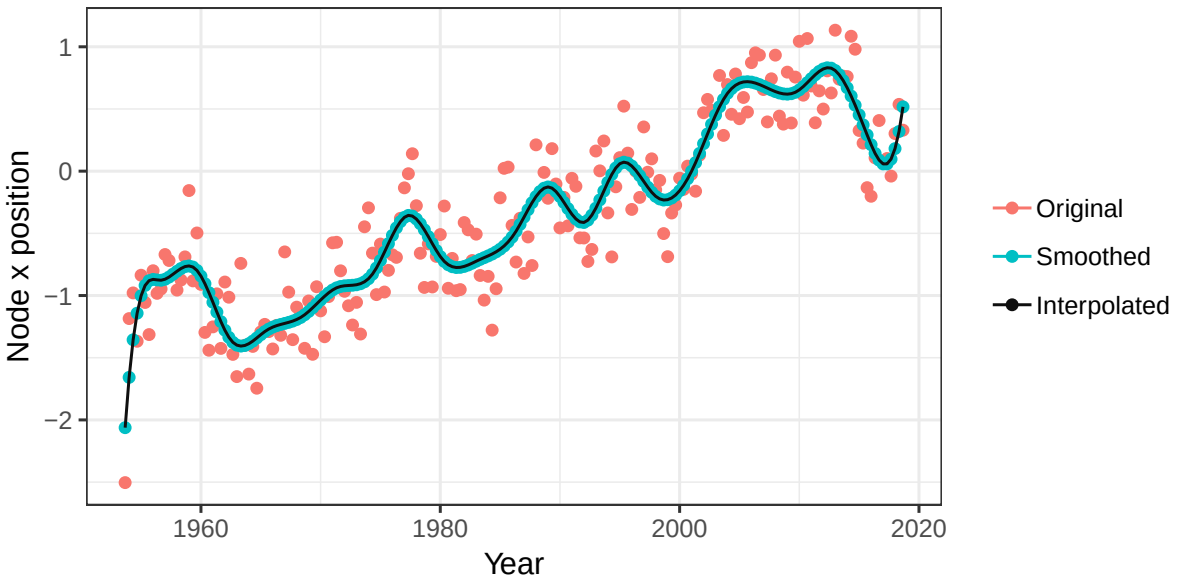


Figure 3: Spline smoothing on a node's x coordinate

We now discuss some specific choices made in our implementation to improve the aesthetics of the visualisations.

Nodes will fade out if they become disconnected and fade in if they join the graph, alongside any incident edges. The initial positions of new nodes are set to the average positions of their neighbours. This allows us to use as input graphs that do not have the same vertex set.

Many graphs, such as the trade networks we will be considering, have meaningful edge weights that affect the visualisation. Using the Fruchterman-Reingold method, a higher edge weight brings nodes closer together, whereas the opposite occurs in Kamada-Kawai. We take the reciprocal of the edge weights in the latter, so as to ensure that when using either method a higher edge weight should bring nodes closer together.

Since it is appropriate for our choice of graphs, the initial node placements are set to the coordinates of the corresponding country's capital city, although this can be left unspecified in the program and it will compute an initial layout using a full run of a force-directed layout algorithm (until convergence).

The size of the plotting frame also needs to be chosen. We take the 1% and 99% quantiles of the x and y coordinates for each layout, then take the 1% and 99% quantiles of these over all layouts. This means that rather than taking the minimum and maximum positions we initially exclude some nodes. We scale these limits up by a small multiplicative factor. This is intended to make sure that any large components of the graph are

guaranteed to remain in frame, but individual vertices far from the center of the layout will be allowed to move out of bounds without the need to expand the plot window to accommodate them. This provides a compromise between guaranteeing every node is visible and remaining appropriately zoomed in on the graph to see it in detail.

The algorithm has been implemented using `R`. The `igraph` package [13] is used to handle many graph operations, including the force-directed updates, as well as plotting the visualisations. `R` does not have native support for creating video, so we must make separate images for each frame of the animation before converting them into a video.

Trying to render all the `R` output is prohibitively slow for even moderate quality output; with a modestly powered laptop the process had to run overnight to generate the largest ones. Better software and/or hardware would be required to create the visualisations fast enough for interactive use. Conversion to video is handled by the open-source program `FFmpeg`. The video creation from the still images is quite fast, taking at most a minute.

4.2 Example: Visualising the global alliance network

We demonstrate the algorithm using the ATOP intercountry alliance data [14], which records alliances between nations from 1900 to 2016. The networks are formed from slices of the global alliance data taken at each year from 1935 to 2016. Countries (represented by their 3-letter ISO codes) are linked in the network if there is a codified alliance between them. As well as illuminating the features of the algorithm, the visualisations in themselves provide interesting insights.

Figures 4 and 5 show the network visualisation with Fruchterman-Reingold updates in the algorithm, before and after smoothing. Figure 6 shows a visualisation with Kamada-Kawai updates and smoothing. Each year there are 3 force-directed updates, and there are 4 intermediate frames between each of these. In each figure we can see the core principles of the algorithm at work. As more edges are added to the graph, the nodes are pulled closer together, reflecting their new status as political allies.

It is clear from a glance at Figure 4 that the force-directed updates require smoothing. The nodes rapidly oscillate back and forth with each update, as the force-directed algorithms attempt to make the positions converge to local minima. The spline smoothing technique entirely solves this, although it does not avoid the issue of nodes with a short lifespan rapidly shooting across the screen before disappearing. To fix this further work

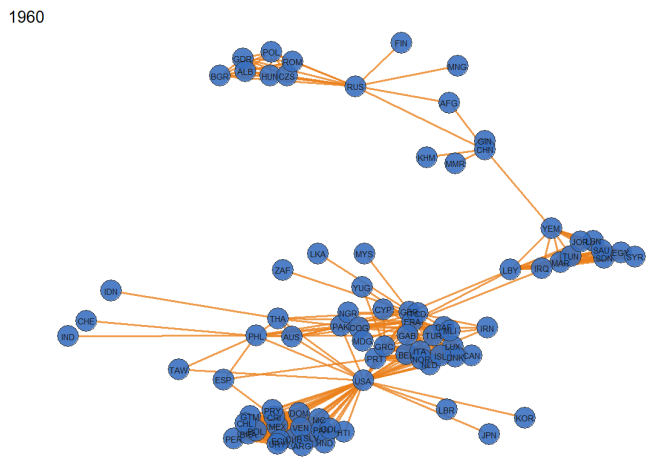


Figure 4: Snapshot of the alliance network visualisation, Fruchterman-Reingold updates
Video on Github on filepath Visualisations/alliancenetwork/alliancenetwork_fr_24fps

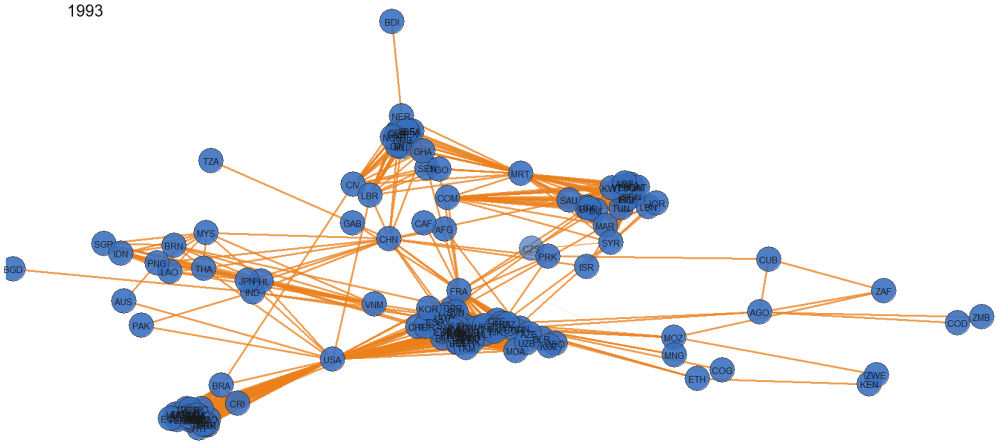


Figure 5: Snapshot of the alliance network visualisation, Fruchterman-Reingold updates (smoothed)
Video on Github on filepath
Visualisations/alliancenetwork/alliancenetwork_fr_smoothed_24fps

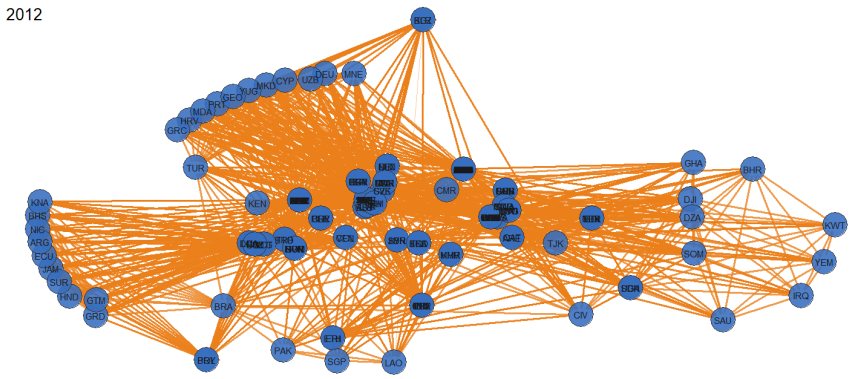


Figure 6: Snapshot of the alliance network visualisation, Kamada-Kawai updates (smoothed)
Video on Github on filepath
Visualisations/alliancenetwork/alliancenetwork_kk_smoothed_24fps

is needed, possibly anchoring the splines or introducing a maximum velocity to prevent excess movement.

Looking at the ouput of the algorithm with Kamada-Kawai updates, we see a very different structure has emerged. The Kamada-Kawai algorithm forces nodes with equal or similar neighbourhoods to appear close to each other, or in many cases in exactly the same position, making the graph unreadable. Adding a small amount of noise to the overlapping positions could help with the problem, or adding repulsive forces as in the Fruchterman-Reingold model. The output also seems to convey less useful information about clusters within the graph, although in other circumstances the alternate structural view it provides of the network may prove useful.

It was hoped Fruchterman-Reingold would avoid the problem of node overlap with its repulsive forces between the nodes, but despite the theoretical benefits the implementation used considers the vertices to be small points, far smaller than the size required for displaying them, and so only pushes them apart by a small amount. Sadly the `igraph` implementation of Fruchterman-Reingold also deprecated some useful parameters that could

easily fix this issue. It should be noted that the algorithms were explicitly optimised for the trade network, and overlap is less of an issue in those visualisations.

We can make inference about international power structures from the visualisation (focusing now on figure 5), even without leaning on historical context. A large American alliance bloc forms in the 1940s, quickly becoming its own cluster in the graph. Throughout the 1950s, the effects of the Soviet sphere of influence can be seen in the cluster forming towards the top of the frame, with Russia/USSR being the central hub. The USA is a very important node at this time, providing the only link between the American alliance web and the rest of the world. A Western European cluster has also formed in the center of the frame, and the Middle East can be seen clearly on the right.

New alliances are forged in the 1960s which bring the graph closer together; these quickly dissipate but reappear in the 1970s as the Soviet and Western European clusters collide. Many African countries have been included in the network by 1980, and some can be seen clustering towards the top right. Several countries of the former USSR appear upon its dissolution in 1991, and immediately move into the European cluster through their new alliances. South and East Asian countries form a cluster during the 1990s, which can be seen to the top left of the frame.

Around 2000, a slew of new alliances bring together all of the African and Middle Eastern nodes. Two key nodes can be identified here by their unique positions in the network. The USA is still facilitating a key link to the rest of the Americas (Brazil has also begun to do this and can be seen moving apart from the rest of the Americas), as well as linking to Europe and much of Asia, and China can be seen just above the European cluster, providing a structural link between the Asian and African/Middle Eastern clusters. By the 2010s the Asian and European clusters have all but combined. We end in 2016 with a graph composed mainly of 3 distinct clusters.

Some outliers can also be seen in the final graph. Cuba can be seen between the European and African/Middle Eastern clusters; due to its significance as a former socialist state in the Americas it has a unique set of allies almost unrelated to its geographical position. Iran and Afghanistan are seen isolated, and Syria lies outside of the Middle Eastern cluster due to its lack of allies there.

This example goes to illustrate the power of visualisation in qualitatively explaining network change, and a use of this in political science.

5 The global trade network

Here we introduce the global trade network through the lens of network analysis, which allows us to gain new insights into its structure and the roles of countries within it.

We look at IMF data on the Direction of Trade (DOT) [15], which tracks annual intercountry trade flows from 1947 onwards. Much preprocessing was required to get the data into a workable format.

The data contains the total (directed) trade between countries every year in $USD. Due to a large amount of missing data in the early years, we do not consider data from before 1953. The networks are formed by splitting the data by year, then creating an edge from country i to country j if there is trade in this direction. The weight of this edge is related to the volume of trade. While we want to preserve the original data, the volume of trade in a single link can reach the trillions and so transformations are needed.

The trade values were adjusted for inflation using the CPI index and brought to 2020 $USD values. Log scaling was used to bring the data to a better distribution - amongst the positive values the distribution is approximately normal, but there are a large number of zeroes where no trade is measured (we add 1 to the values before log scaling such that the weight of these non-links remains zero). Further transformations of the trade value were used in different contexts. In the modelling the trade was left in this form or scaled by a constant so that the results could be interpreted easily. The trade volumes also dictate the width of the edges in the network visualisations, and here the widths are an increasing function of the original trade value that restricts the edge widths so as to be visible but not overpowering.

5.1 Visualisations

Here we present several dynamic network visualisations of the IMF Direction of Trade dataset. This is a highly connected and dense network, so some aspects had to be simplified for a coherent image. The edges are made undirected, representing the trade flowing both ways across the link.

The force-directed methods struggle with the full graphs, and will move the densely connected nodes into an unintelligible cluster at the center - there is in effect too much attractive force between the nodes. This cannot be easily remedied by changing layout parameters; too many constraints would be needed. Instead, we can alter the graphs without destroying too much of the structure we are trying to convey.

We remove some edges from the graph before display. Only each country's top 5 trade partners by $USD value are included in the graph (note that high-trade countries such as the USA can be the top trading partner for several others, and so will have degree larger than 5).

We also rank each country's outgoing edges according to trade value, and use these to determine the opacity of the edges. This means that a country's largest export destination will be the most visible, and the opacity will decrease for each subsequent export destination country. When making the graphs undirected, these ranks are averaged, representing a measure of the trade link's importance to both countries involved. This helps to declutter the frame.

There are however, still a great many edges competing for real estate in the frame. Crucially, since the edges are rendered before the nodes are, the latter appear on top in the image and so we can rely on the node positions for insight.

In figure 7 we can see the dynamic visualisation with these settings. Rather than the clustering visible in the alliance network graph, the structure appears to have more important nodes as hubs in the center of the frame, with other vertices circling on the outside. With that said, some geographical structure does remain. While not a perfect description, moving clockwise from the top left in the last frame of the video we can see Central Asia, Africa, Southeast Asia and Oceania, the Caribbean, Northern Europe and Eastern Europe. In the centre lie the powerful trading hubs in Western Europe, China, and the USA, as well as some countries merely in transit across the image.

Highlighted in red are China, France, Germany, Great Britain, The Netherlands and the USA, all highly important to global trade. China does not begin the animation at the centre of global trade, but moves there over time, reflecting its growing power and influence. The Western European trade powers remain close due to the high volumes of trade between them, and this group remains central in the network.

Looking at the end of the animation, China, the USA, and Western Europe can be seen as three separate hubs that connect to different areas of the global network. Unsurprisingly, the USA has links to many countries in the Americas and appears close to them in the frame, and the highlighted European countries likewise are positioned alongside the rest of Europe. China does have links to many other Asian countries, but interestingly can also be seen near many African countries.

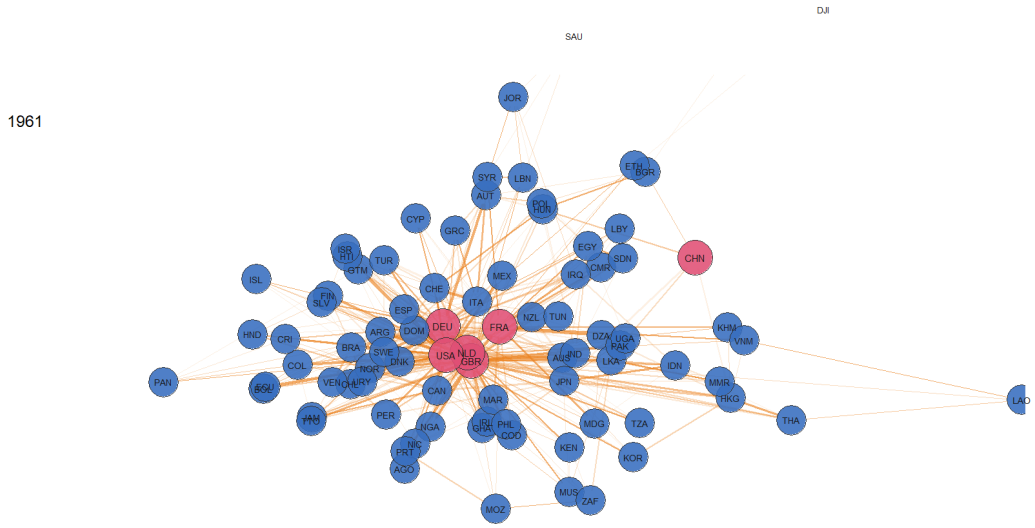


Figure 7: Snapshot of the trade network visualisation
Video on Github on path Visualisations/tradenetwork/tradenetworkK2_smoothed_24fps

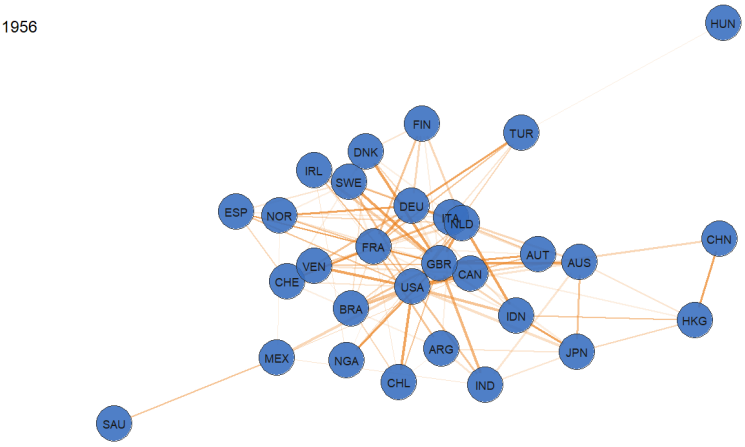


Figure 8: Snapshot of the trade network visualisation, top 40 countries
Video on Github on path
Visualisations/tradenetwork/tradenetworksmallK_smoothed_24fps.mp4

It is interesting to note the advent of new trading blocs throughout the visualisation, and the impact they have had on the dynamic mapping. For example, the formation of the Association of South East Asian Countries (ASEAN) in 1967, the Gulf Cooperation Council (GCC) in 1981, European Union (EU) in 1993 and the World Trade Organisation (WTO) in 1995 have all created strong new linkages. Equally, political and economic shocks (such as the Chinese economic reforms of 1978, the stock market crash of 1987, the Twin Towers attack in 2001 and the global financial crisis that began in 2007) will have impacted on trade. It will be interesting to see whether the effects of brexit will become visible, or whether the UK's position will remain linked with the rest of the EU.

6 Analysis

The goal of this analysis is to quantify the changing landscape of global trade, and the roles different political structures have in influencing it.

Some of the changes in global trade can already be seen by looking at aggregate statistics. Figure 9 shows the top 10 countries ranked by their imports plus exports in each decade. China, Hong Kong and Japan have advanced their position, while Great Britain, Canada and Brazil have seen their relative trade power fall.

Figure 10 ranks the top 10 countries according to their eigenvector centrality in the unweighted trade network, averaged over each decade. Eigenvector centrality is a measure of the influence a node has on a network. Intuitively, if many countries export to another country, that country will have high eigenvector centrality.

In the unweighted network the volume of trade does not affect the centrality, so this looks purely at the network topology to determine importance. Great Britain and other European countries appear to have much more importance to the network using this metric, with the USA and China falling down the rankings.

In the weighted network (Figure 11) the volume of trade is accounted for, and the USA is once again at the top of the rankings with Britain also holding an important role. China again rises quickly to become the second most important nation in the 2010s, and there is more European presence than there was when simply looking at the volume of trade.

6.1 AME modelling

We now move on to a more formal analysis. AME models are the most powerful tool we can use here, allowing us to model network structures in a statistically sound way. We seek to discover if there is evidence for globalisation in the trade network, which we can measure by the influence that the geographical distance between two countries has on their propensity to trade and how this has changed over time. The advance of globalisation is, essentially, an exercise in network construction, as states create links around the world untethered by geographical distance. We have seen this already in the visualisations, as distant countries converge to form a densely connected network.

We use trade data from the IMF's Direction of Trade dataset [15], adjusted for inflation. The intercountry covariates used in the model are distance between capitals, formal military alliances [14], and former colonial relations [16].

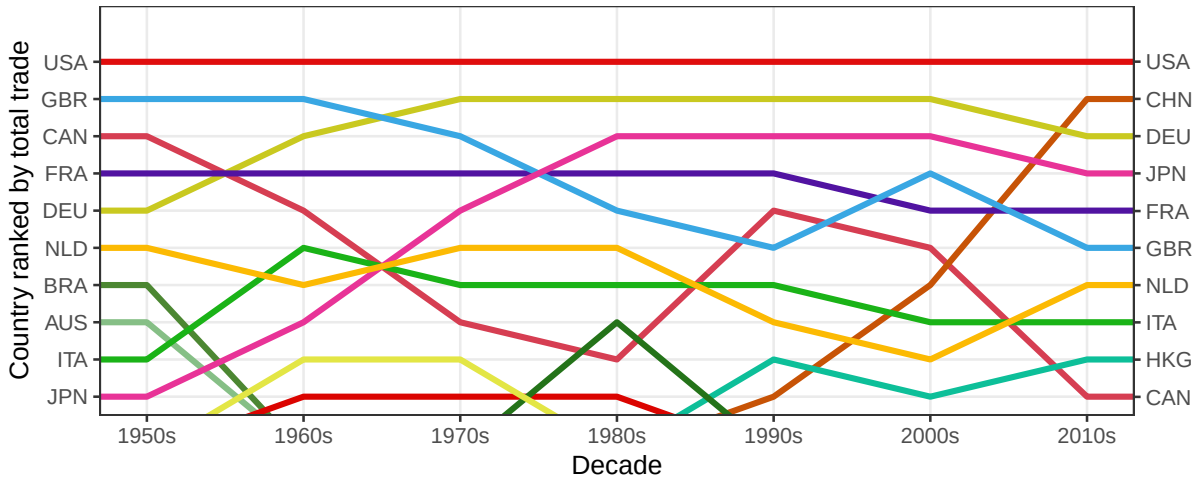


Figure 9: Top 10 countries by total trade in each decade

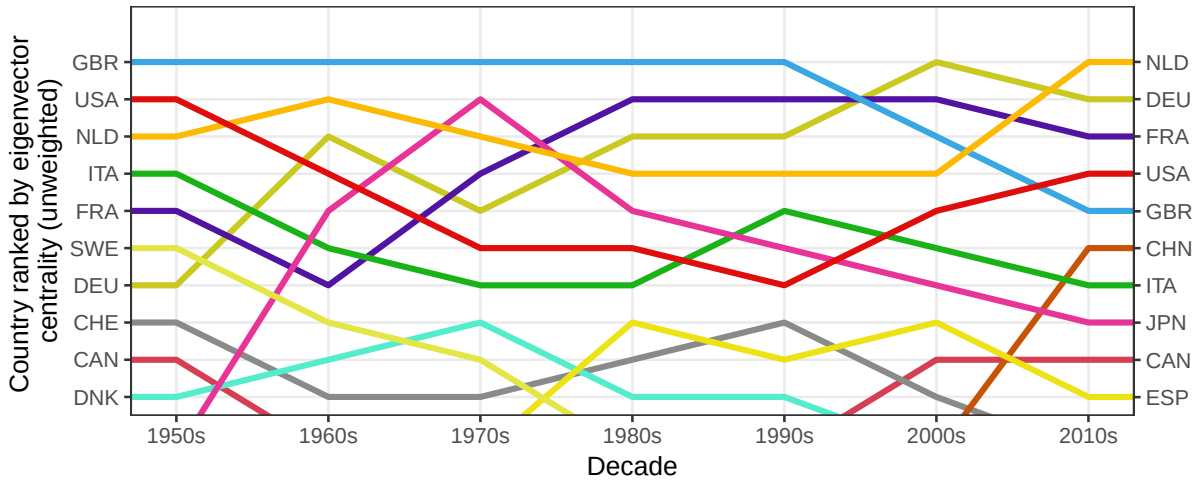


Figure 10: Top 10 countries by average unweighted trade network eigenvector centrality in each decade

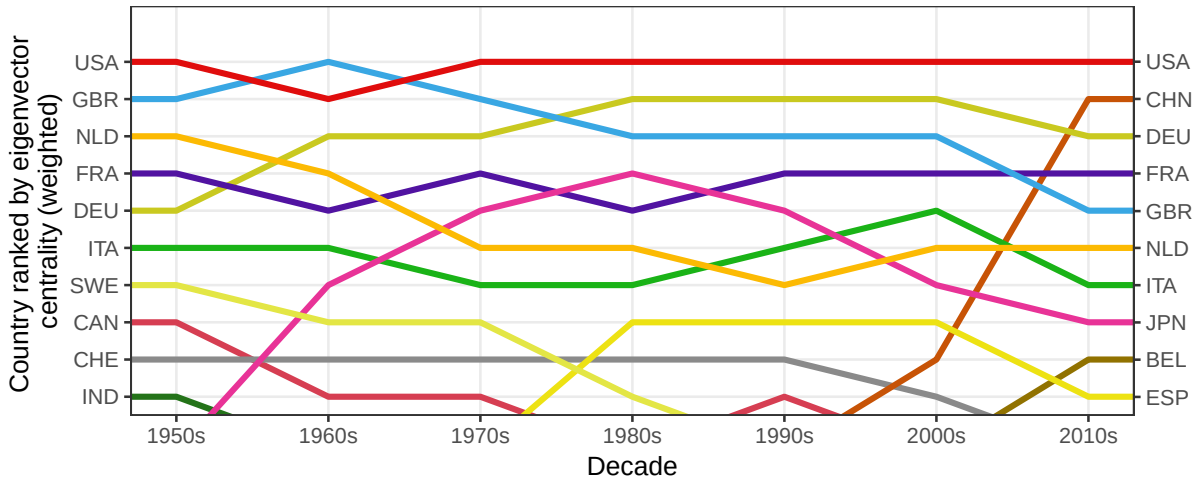


Figure 11: Top 10 countries by average weighted trade network eigenvector centrality in each decade

The alliance data comes from the Alliance Treaty Organisations and Provisions (ATOP) Project [14]. Each codified alliance given is treated equally, and the network is built from these. Some manual data cleaning and country matching is required to match this dataset to the others.

We also use COLDAT [16], a dataset which aggregates information about European colonial empires. Two countries are linked if one was at some point a colonial overlord of the other.

Distance is a geographic barrier to trading, but one which can be overcome ever more easily. The military alliances in our model are a proxy for political relations, as well as providing a possible avenue for material and arms trade. Colonial relations are included in the model to account for long-distance trade relations that can be more easily explained through this effect, and might confound the results.

An AME model is fitted to the dataset for every two-year period, and we analyse the coefficients over time.

In our first analysis, trade is encoded in the model as a binary variable (and binary probit regression is used) - this only represents the presence of trade between two countries and says nothing about the volume.

The results of this can be seen in Figure 12. This only shows the relative contribution of each variable to likelihood to trade (note that these are not the actual parameter values). We find that the dependence of trade on distance has slightly lessened over time. States closer to each other are still more likely to trade. The actual coefficient of distance in the model is still negative, unlike for the other two variables. Allied countries are more likely to trade, and this effect has slightly increased.

Interestingly, states are more becoming more likely to trade with their former colonies and vice versa, suggesting a potential driver for the changes to trade.

We now expand our analysis to account for the volume of trade. This means that the response variable is now the value of trade between each country, adjusted for inflation and log transformed. The predictor variables remain the same. The results have been plotted in the same way as before and can be seen in Figure 13.

The effect of a political alliance on the volume of trade between two countries has fluctuated greatly, especially around 2000, with an overall increase in its effect. We see the same trend as before that countries are trading more with their former colonies.

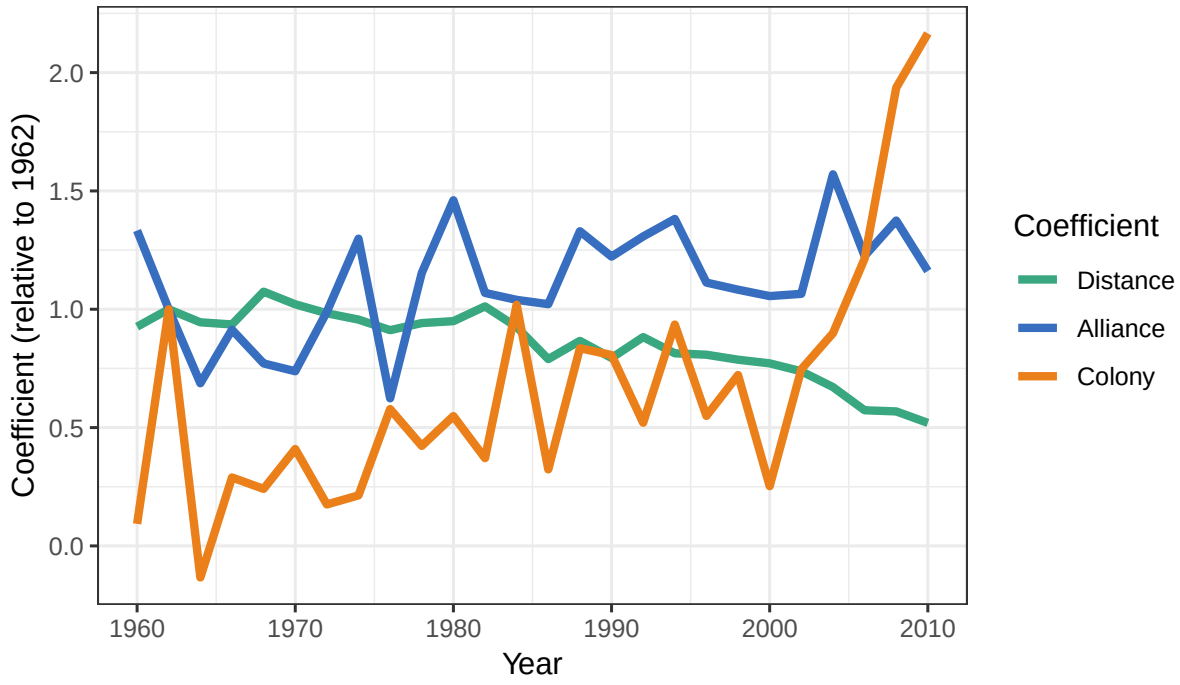


Figure 12: Relative effects of factors on likelihood to trade in the binomial probit AME model

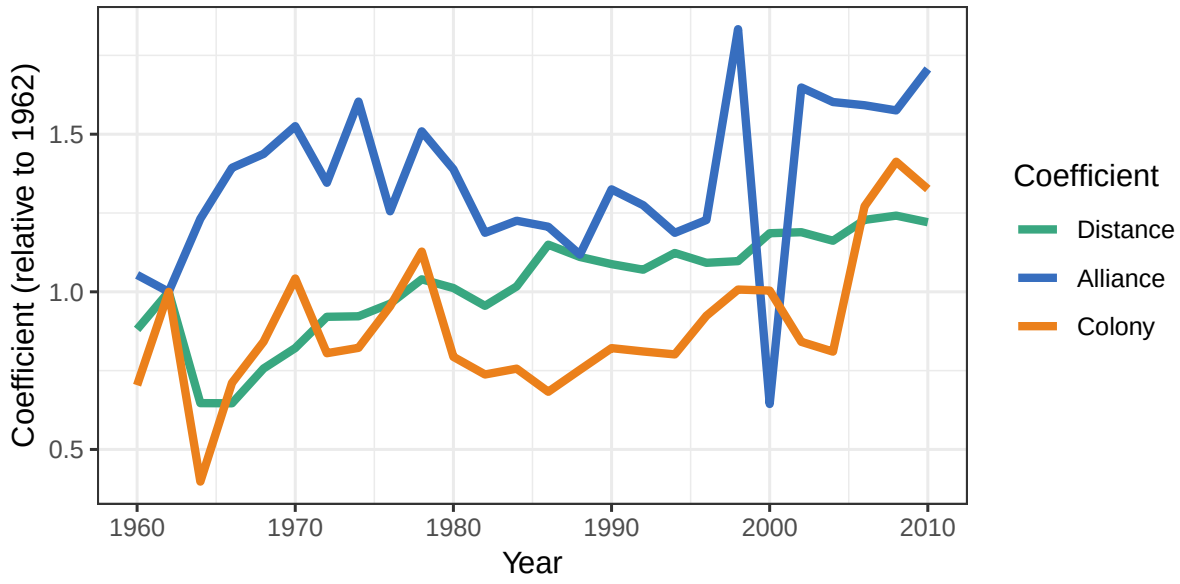


Figure 13: Relative effects of factors on volume of trade in the normal AME model

Interestingly, the effect of distance on trade value has decreased, suggesting that the volume of trade between countries is now more dependent on geographic closeness than it was before. This would suggest that countries are expanding their trade networks around the world, but are sending greater volumes of trade to countries nearer to them. Overall, it seems that globalisation is manifesting as an increasing propensity for countries to trade around the world, but that an increasing proportion of trade is focused on local partners.

7 Conclusion

Many of the challenges of this dissertation stemmed from working in the relatively unexplored area of dynamic network science, which meant having to create new methods to solve problems and answer questions. Formal analysis of dynamic networked data remains a challenge. The AME model can technically be extended to use temporal data, however it assumes time does not affect the distribution of the model, only allowing for a narrow selection of autoregressive models to be used. Further work is still needed to design a more useful repeated measures model.

Network visualisation is as much an art as a science. Our layout algorithm is designed flexibly enough to allow experimentation with various visual parameters. The output created from it was clear and useful enough to gain understanding of the networks shown, and so the algorithm has worked as intended. There are shortcuts taken in the implementation specific to our graph inputs, but without a great deal of further work the programs could be restructured to accommodate general graphs.

Overall, the techniques studied in this dissertation have provided deep analysis of the global trade network, and would provide an excellent tool in future work in political science. For instance, modelling of the global trade network could be used in decisions regarding trade policy.

References

[1] Barry Wellman. Network analysis: Some basic principles. *Sociological Theory*, 1:155–200, 1983.

[2] Albert-László Barabási et al. *Network science*. Cambridge university press, 2016.

[3] Mark EJ Newman. The mathematics of networks. *The new palgrave encyclopedia of economics*, 2(2008):1–12, 2008.

[4] Shahryar Minhas, Peter D Hoff, and Michael D Ward. Inferential approaches for network analysis: Amen for latent factor models. *Political Analysis*, 27(2):208–222, 2019.

[5] Peter D Hoff. Dyadic data analysis with amen. *arXiv preprint arXiv:1506.08237*, 2015.

[6] Jacob Levy Moreno. Who shall survive?: A new approach to the problem of human interrelations. 1934.

[7] Thomas MJ Fruchterman and Edward M Reingold. Graph drawing by force-directed placement. *Software: Practice and experience*, 21(11):1129–1164, 1991.

[8] Tomihisa Kamada, Satoru Kawai, et al. An algorithm for drawing general undirected graphs. *Information processing letters*, 31(1):7–15, 1989.

[9] Kazuo Misue, Peter Eades, Wei Lai, and Kozo Sugiyama. Layout adjustment and the mental map. *Journal of Visual Languages & Computing*, 6(2):183–210, 1995.

[10] Daniel Archambault and Helen C Purchase. The "map" in the mental map: Experimental results in dynamic graph drawing. *International Journal of Human-Computer Studies*, 71(11):1044–1055, 2013.

[11] Fabian Beck, Michael Burch, Stephan Diehl, and Daniel Weiskopf. The state of the art in visualizing dynamic graphs. In *EuroVis (STARs)*. Citeseer, 2014.

[12] Yaniv Frishman and Ayellet Tal. Online dynamic graph drawing. *IEEE Transactions on Visualization and Computer Graphics*, 14(4):727–740, 2008.

[13] Gabor Csardi and Tamas Nepusz. The igraph software package for complex network research. *InterJournal*, Complex Systems:1695, 2006.

[14] Brett Ashley Leeds. Alliance treaty obligations and provisions (ATOP) codebook, version 4.01. http://www.atopdata.org/data.html, 2018.

[15] General Statistics Division International Monetary Fund. Direction of trade statistics. https://data.imf.org/?sk=9D6028D4-F14A-464C-A2F2-59B2CD424B85, 2018.

[16] Bastian Becker. Colonial Dates Dataset (COLDAT). https://doi.org/10.7910/DVN/T9SDEW, 2019.