Imperial College London
Department of Mathematics

M2R Project

June 2023

Network Graph Embeddings

Authors:

Zak Aladin	za120	01866293
Zongqi Han	zh1520	01845302
Henry Nye	htn19	01734221
Hamis Rizwan	hr620	01869464
Zhekai Weng	zw1321	02043365

Supervisor: Nick Heard

Abstract

Spectral clustering is a popular method for community detection in network graphs, which can be represented by an adjacency matrix and then embedded into fewer dimensions. The nodes can be modelled as samples from a finite number of communities described by a stochastic block model (SBM), which can be optimised through degree correction. The Gaussian mixture model can be used to model these communities as samples of different multivariate normal distributions. Alternatively, spectral clustering can be used to iteratively cluster the eigenvectors of each node, minimising the total distance between each eigenvector and the mean of its cluster. In this paper, the performance of each method will be evaluated in the context of clustering UK Members of Parliament by party membership based on public voting records. The inferences from this case study will then be further analysed by applying the same methods to simulations of SBMs.

Contents

1 Introduction

In this project, there will be detailed discussions on community detection in bipartite graphs, different algorithms for detecting the communities, and comparisons to the 'true' community partition. To conceptualise this, data publicly available at the public whip will be the source data, consisting of 674 Members of the UK Parliament (with some repeats for special cases) and how they voted on the previous 794 divisions in some specified period. By letting 1 represent an 'aye' vote, -1 a 'no' vote and 0 an abstention from voting, an adjacency matrix for this graph is obtained. Under the stochastic block model, attempts will be made to cluster MPs into their respective parties, with the true party data compared to predicted results. This will be accomplished by representing each of the MPs as nodes in a graph, of which each will be assigned a 794-dimensional vector representing their voting record (i.e. a row in the adjacency matrix). This will then be dimensionally reduced in order to perform community detection (which will be referred to as clustering), and predicted results will be compared with the true results.

For each section, the theory will be set out first, followed by motivations behind the theory and necessary assumptions, and then the application of this theory to the voting records dataset. The dataset is used to assess the utility of the theory and clarify any topics that are particularly hard to understand. The theory and the application are not completely separated, because presenting the results of the application helps clarify what the theory means, and gives empirical reasons for choosing different methods.

At the start, a review of the literature that is the foundation for this project, and the algorithms used for community detection Section 2 will be included. This will address references that have been used, and similar papers/articles that produce results conducive to the findings of this report.

Section 3 gives an overview of the source data. There will be manipulations of the data, such as the proportion of majority votes per party and the agreement in votes between parties. This will provide detailed context and help form expectations for how well these clustering algorithms should work. It is intended also to provide a clear conceptualisation of the data for the reader. (Athreya et al., 2016)

In Section 4, there will be a discussion on graph embedding as well as the Stochastic Block Model (SBM). Further considerations will show how embedded graph sets can be visualised in lower-dimensional space, and this will motivate a definition for clustering - a producible result of graph embedding - and what an ideal cluster will mean spatially. After introducing embedding, the assumptions of the SBM and their justifications and possible limitations will be discussed, specifically to the given dataset of MPs. There will be a brief overview of what is needed for this particular model (such as the assumed number of clusters), and thus optimise the model via a degree-correction, using the degree-corrected stochastic block model (DCSBM).

This will lead into the next section on Methods of Clustering (Section 5) with a given number of clusters (equal to the number of parties or some variation of that in the case of the dataset). There will be analyses of how each method works, and which parameters work best. Focusing largely on two main clustering methods - GMM and spectral clustering - it is shown the Gaussian distribution and eigen-properties respectively play significant roles. Preceding obtaining the results, methods for analysing these results will be introduced (i.e. comparing the party estimates to the true parties), using a measure called the Rand index.

After implementing the methods discussed in the previous section, an analysis will be conducted for the dataset results (Section 6), comparing to the actual party membership. There is expected to be some error due to party members voting against the party, which intuitively could indicate them being placed into the 'wrong' cluster/party. Thus, it will be highlighted how these issues can be dealt with. As shown in the Contents, further considerations for changes to the model will be discussed and justified, such as how many dimensions the latent space is reduced into, and how many clusters are produced.

Next, a spherical transformation of the graph embedding in Section 7 is considered as a form of degree correction, along with justifications on why this is appropriate. Therefore results can be produced using both clustering algorithms on this transformed data. The results are again analysed and compared to previous findings, with the outcomes discussed.

Simulation results are included in Section 8 which will be representative of each of the different methods, applying the same approaches to randomly generated data and seeing what results are yielded. The simulations are repeated to provide stronger evidence for and against different methods of clustering, and comparing a variety of simulated models with the real data set provides context for which model the real data set should be assumed to follow.

In Section 9 possible model adjustments are briefly touched upon, alongside what could be done differently and the possible outcomes. The analysis measure of the Rand index will be discussed, and what could be done to improve it. Bayesian approaches to the project will be mentioned, and what different results this could produce.

In the conclusion (Section 10) the findings in the report will be stated and a critical review of these modelling efforts for the dataset provided will be given.

The code for generating all the plots and simulation data will be included in the supplementary Material section before the references.

2 A Review of Literature on the SBM and Graph Embedding

There are many ways to analyse the source data, but this report will focus primarily on working with the SBM and clustering data. This field has been well studied with advancements in technology allowing modelling to be significantly more accessible through programming libraries and publicly available documentation, with much active research still in progress. Several journal papers have been published particularly recently with newer developments and extensions to previous research. This section will investigate clustering methods detailed in these papers and evaluate both their relevance and usefulness for the dataset analysed in this report.

A piece of literature that is fundamental to the base understanding of clustering is the 2019 journal article, "A review of stochastic block models and extensions for graph clustering" Lee and Wilkinson (2019). This paper provides a comprehensive overview of SBMs and the various clustering approaches that can be used for an SBM, such as hard clustering. Additionally, the mathematical foundations are established throughout the paper, which will be indispensable when understanding the results of this report and suggesting steps to improve them.

"A Comprehensive Survey of Clustering Algorithms" (Xu and Tian, 2015) goes into depth on how the different methods can be compared.

In this article, the formulae and algorithms from the aforementioned articles will be applied to the data and will be expanded on by measuring and analysing their effectiveness in this use case.

3 Voting Patterns and Party Discipline in the House of Commons

All votes taken by MPs on all divisions between 20/12/2019 (the date of the 2019 United Kingdom general election) and 24/05/2023 were analysed. This data was retrieved from the public whip, and stored in an adjacency matrix A where A_{ij} represents the i^{th} MP's vote in the j^{th} division with 1 representing an aye vote, -1 representing a no vote and 0 representing an absence. There are 11 political parties (excluding independent MPs and the Speaker) represented in the House of Commons, but only 3 parties have more than 20 of the 650 total seats. However, around 20 MPs switched parties between those dates and are therefore recorded multiple times. The full breakdown of the data used is below:

Party	**Number of Seats**
Conservative	370
Labour	213
Scottish National Party	49
Liberal Democrats	14
Democratic Unionist	8
Sinn Féin	7
Independent	7
Plaid Cymru	4
Social Democratic Labour	2
Alba	2
Speaker	1
Green	1
Alliance	1

Table 1: Party representation in the House of Commons

The proportion of each party that voted with the party majority, averaged over all divisions, is listed below. In this table, absences are ignored, and the proportion of votes is measured based on how many MPs from the party voted either aye or no in each division.

Party	**Average Proportion of Majority Vote**
Conservative	0.9902
Labour	0.9987
Scottish National Party	0.9999
Liberal Democrats	0.9995
Democratic Unionist	0.9877
Sinn Féin	0*
Independent	0.8606
Plaid Cymru	1.0000
Social Democratic Labour	1.0000
Alba	1.0000
Speaker	0*
Green	1.0000
Alliance	1.0000

Table 2: Party Discipline in the House of Commons

To clarify, this table shows that on average, 99.02% of Conservative MPs voted with the majority of their party on any given vote. The calculation is weighted for each division based on how many votes were taken by the party in that division. Over the 794 divisions analysed, there were only 3 times that the Liberal Democrats did not vote unanimously, and if the only two independent MPs that have served as Conservative MPs are removed, the proportion of independent MPs that voted with the majority of independent MPs increases from 86.06% to 99.88%. Sinn Féin abstain from voting so the data on them is misleading, and the parties below them have a maximum of 3 MPs so it is unsurprising that they always vote unanimously.

The following table gives an agreement index for all parties. This looks at how the majority of each party votes in each division, and then how often that majority agrees with the majority of each other party. For example, the majority vote of Labour is the same as the majority vote of the SNP 97.9% of the time. Divisions in which a party did not vote are ignored when calculating the agreement index between 2 parties.

	Conservative	Labour	SNP	Lib Dems	DUP	Sinn Féin	Independent	Plaid Cymru	Social Democrat Labour Party	Alba	Speaker	Green	Alliance
Conservative	1.0	0.077	0.023	0.055	0.583	0	0.108	0.026	0.038	0.014	0	0.048	0.046
Labour	0.077	1.0	0.979	0.982	0.438	0	0.926	0.988	0.989	0.982	0	0.985	0.986
SNP	0.023	0.979	1.0	0.983	0.421	0	0.955	0.993	0.996	1.0	0	0.996	0.991
Lib Dems	0.055	0.982	0.983	1.0	0.439	0	0.938	0.992	0.993	0.996	0	0.993	1.0
DUP	0.583	0.438	0.421	0.439	1.0	0	0.458	0.427	0.424	0.462	0	0.419	0.434
Sinn Féin	0	0	0	0	0	0	0	0	0	0	0	0	0
Independent	0.108	0.926	0.955	0.938	0.458	0	1.0	0.95	0.937	0.977	0	0.954	0.938
Plaid Cymru	0.026	0.988	0.993	0.992	0.427	0	0.95	1.0	0.998	0.993	0	0.998	0.996
Social Democrat Labour Party	0.038	0.989	0.996	0.993	0.424	0	0.937	0.998	1.0	1.0	0	0.994	1.0
Alba	0.014	0.982	1.0	0.996	0.462	0	0.977	0.993	1.0	1.0	0	0.996	1.0
Speaker	0	0	0	0	0	0	0	0	0	0	0	0	0
Green	0.048	0.985	0.996	0.993	0.419	0	0.954	0.998	0.994	0.996	0	1.0	0.998
Alliance	0.046	0.986	0.991	1.0	0.434	0	0.938	0.996	1.0	1.0	0	0.998	1.0

Table 3: Political Parties Agreement Index

Alternatively, the 'aye agreement index' (AAI) can be used. The AAI is computed for each pair of parties, using data from all voting instances in which at least one of the two parties has cast a vote of 'aye'.

Specifically, the index is defined as the number of times both parties have voted 'aye', divided by the total number of instances in which at least one of the parties has voted 'aye'.

As is shown in the data below, parties generally have a marginally lower aye agreement index than an agreement index. The lower scores are the motivation behind the AAI, as they potentially make the AAI a better metric for distinguishing between two parties.

	Conservative	Labour	SNP	Lib Dems	DUP	Sinn Féin	Independent	Plaid Cymru	Social Democrat Labour Party	Alba	Speaker	Green	Alliance
Conservative	1.0	0.052	0.013	0.044	0.412	0.0	0.051	0.022	0.028	0.014	0.0	0.043	0.039
Labour	0.052	1.0	0.962	0.968	0.32	0.0	0.87	0.979	0.981	0.967	0.0	0.973	0.975
SNP	0.013	0.962	1.0	0.97	0.3	0.0	0.919	0.987	0.992	1.0	0.0	0.992	0.984
Lib Dems	0.044	0.968	0.97	1.0	0.314	0.0	0.89	0.986	0.987	0.993	0.0	0.988	1.0
DUP	0.412	0.32	0.3	0.314	1.0	0.0	0.312	0.309	0.304	0.355	0.0	0.292	0.313
Sinn Féin	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
Independent	0.051	0.87	0.919	0.89	0.312	0.0	1.0	0.909	0.886	0.958	0.0	0.918	0.888
Plaid Cymru	0.022	0.979	0.987	0.986	0.309	0.0	0.909	1.0	0.997	0.988	0.0	0.997	0.993
Social Democrat Labour Party	0.028	0.981	0.992	0.987	0.304	0.0	0.886	0.997	1.0	1.0	0.0	0.989	1.0
Alba	0.014	0.967	1.0	0.993	0.355	0.0	0.958	0.988	1.0	1.0	0.0	0.993	1.0
Speaker	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
Green	0.043	0.973	0.992	0.988	0.292	0.0	0.918	0.997	0.989	0.993	0.0	1.0	0.997
Alliance	0.039	0.975	0.984	1.0	0.313	0.0	0.888	0.993	1.0	1.0	0.0	0.997	1.0

Table 4: Political Parties Aye Agreement Index

Mutual agreement between separate parties introduces issues in clustering the dataset, particularly in the case of smaller parties. The clustering algorithms performance decreases for closely agreeing parties, so when analysing the data using these algorithms, it is likely that small parties will have to be grouped together in order to gather significant results.

4 Graph Embedding and the Stochastic Block Model (SBM)

This section will include a brief introduction to graph embedding and the SBM, as well as a brief analysis of the result of adjacency spectral embedding (ASE) using voting records of MPs.

4.1 Graph Embedding for the Dataset

When using and manipulating node-edge graphs, it is often convenient to quantify the data as vectors, in some latent vector space. Clustering algorithms can then be applied to this vector space to estimate the latent centroid vectors and assign each node to one of the clusterings around these centroid vectors (discussed in Section 6 and Section 7).

It remains to consider which latent spaces should be used, and how to embed the nodes into them. It should be noted that in the dataset used in this report, the graph can be represented by a bipartite adjacency matrix, of N MPs and D divisions. Thus, singular value decomposition can be used to estimate latent positions for the nodes. (Sanna Passino, Heard, and Rubin-Delanchy, 2022). Let:

$\mathbf{A} \in \mathbb{R}^{N \times D}$ be the adjacency matrix

$\mathbf{U}\mathbf{\Sigma}\mathbf{V}^T$ be the SVD of $\mathbf{A}$, reduced into

$\hat{\mathbf{U}}\hat{\mathbf{\Sigma}}\hat{\mathbf{V}}^T = \mathbf{U}_{:d}\mathbf{\Sigma}_{:d}\mathbf{V}_{:d}^T$, where now $\hat{\mathbf{\Sigma}}$ is square and diagonal.

The number of dimensions, d, that the nodes should embed into must be considered. This will be the number of columns in $\hat{\mathbf{\Sigma}}$, $\hat{\mathbf{U}}$, and $\hat{\mathbf{V}}$. The elements along the diagonal of $\hat{\mathbf{\Sigma}}$ will be decreasing in magnitude. Thus, $\hat{\mathbf{\Sigma}}$ can be truncated until it is $d \times d$ and the adjacency spectral embedding (ASE) of $\mathbf{A}$ can be obtained by

$$\hat{\mathbf{X}} = \hat{\mathbf{U}}\hat{\mathbf{\Sigma}}^{1/2}$$

The rows of $\hat{\mathbf{X}}$ are the latent vector positions for each of the MPs. (Sanna Passino, Heard,

and Rubin-Delanchy, 2022)

Before spectral embedding is discussed, it is helpful to know why graph embedding is useful. Clearly, one of the main uses is to quantify data in a way that can be analysed. This is especially convenient for machine learning processes - for example in Higgs, Weller, and Solka (2006) various medical images can be modelled as nodes in a graph alongside similarity weights to "teach" a program to categorise these pictures. This could, for example, be done via clustering methods. This would in turn mean that, when new images are input into the program, it would be able to deduce similarities and effectively categorise the picture depending on specific features of the image. The task would then be to embed these medical images onto a latent space, where the more similar the features, the "closer" the node embeddings are in some latent space.

Another key idea in graph embedding is dimensionality reduction Rubin-Delanchy et al. (2022). This is generally used when graphs naturally have a high dimension, such as a social media network where an edge exists between two nodes if and only if the social media accounts corresponding to each of the nodes follow each other. Thus, for a large social media network, the adjacency matrix will be of very high dimension, making clustering algorithms and processes very computationally heavy and of high cost, unless significant and reliable dimension reduction is applied. Information about the graph structure needs to be retained, so further processes such as clustering can work effectively.

While the adjacency matrix is small enough (674×794) to perform calculations in a reasonable time, the methods described in this paper would be useful on much larger matrices. For example, the voting intentions of civilians based on survey results. Therefore, graph embedding is essential to decrease the dimensions and therefore calculation time. There is also the potential for noise reduction in graph embedding, which can lead to more accurate clusterings (Hegger, Kantz, and Matassini, 2001).

Ordering the diagonal matrices in the eigen (or singular value) decomposition by descending value allows retention of the most relevant information about the graph structure. As mentioned previously, there are two different sets of nodes (MPs and Divisions),

which means the graph is bipartite. However, there have been considerations to graph networks for Harry Potter character enmities, which will require the eigendecomposition of a symmetric adjacency matrix. (Rubin-Delanchy et al., 2022)

When decomposing into SVD, the matrices $\mathbf{U}$ and $\mathbf{V}^T$ are both composed of orthogonal columns. Since $\mathbf{A}$ describes the relationships between the MPs and their divisions, when decomposing, $\hat{\mathbf{X}}$ will represent the latent positions of the MPs, with the weighting of each column included via the $\hat{\mathbf{\Sigma}}^{1/2}$ term. Since only the MPs are being clustered, only $\hat{\mathbf{X}}$ will be considered.

The goal is to cluster the data into K distinct clusters, so it makes sense to dimensionally reduce data into K dimensions. This is due to the assumption in the model that each node is an observation from one of K distinct distributions, and therefore, taking into account the uncertainty of which cluster each node is in, can be described by some combination of K numbers.

On the other hand, a frequentist or a Bayesian approach could be considered. These test multiple values of the dimensions and model the dimension size as a new, random parameter respectively. One relevant example of this parameter selection appears in Sanna Passino, Heard, and Rubin-Delanchy (2022), where the dimensions of the embedded latent positions are determined by minimising the Bayesian information criterion (BIC).

4.2 Visualization of an Embedded Set

By setting the dimension, d, of ASE $\hat{\mathbf{X}}$ to 2 and plugging in adjacency matrix A, an embedded matrix $\hat{\mathbf{X}} \in \mathbb{R}^{N \times d}$ can be obtained, with each row vector corresponding to the latent position of an MP. Therefore, a 2-dimensional scatter plot of MPs in latent space can be produced:

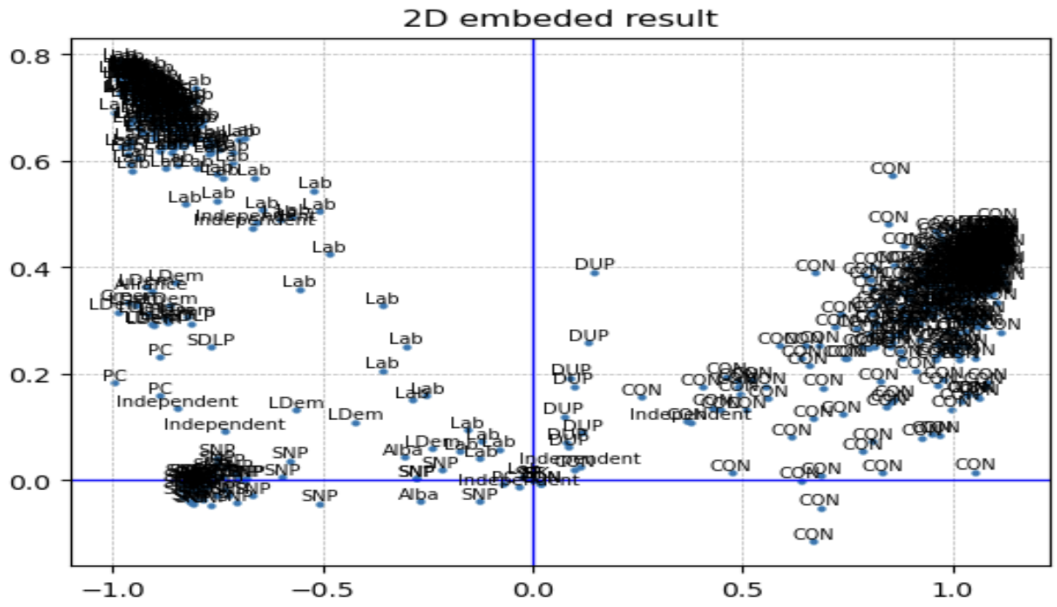

Figure 1: 2-dimensional scatter plot of an embedded matrix, with real party affiliation attached to each node

It should be noted that Independents are a collection of MPs outside of a party, so should not be considered as a strongly organized party. Therefore, the overall tendency of left or right on the political spectrum does not apply to Independents. In the following analysis of the scatter plot, all Independents will be excluded.

From the agreement index in Table 3 all parties except DUP almost always disagree with Conservatives, with their agreement index only around 5%. However, most parties except DUP and Conservatives almost always agree with the majority of Labour, with agreement indices all higher than 95%.

The clear disagreement between left and right is also represented in the embedded data set. It can be seen that the y-axis in the scatter plot separates the left-wing and right-wing parties. Both right-wing parties, the Conservatives and DUP are on the right of the y-axis and other left-wing parties are on the left.

Notice that the DUP is closer to the origin than the Conservatives, and seems to follow their own line from the origin which could also be explained by the agreement index. The agreement index between the DUP and Conservatives is 58%, which is significantly lower than 100%.

Overall, it can be seen that the main features of the source data are preserved by the

ASE, even if the dimension is low, with only 2 dimensions.

4.3 The Stochastic Block Model (SBM)

In this project, the data will be modelled in accordance with the stochastic block model (SBM). Under this model, some key assumptions hold. The general idea behind the SBM is that the data set can be modelled by K blocks, where all nodes each belong to one block and all nodes in a block have the same probability of having a specified edge. The model can be represented by a probability matrix **B** with dimension determined by the number of blocks, of which the $(ij)^{th}$ element determines the probability of an edge existing from an arbitrary node in the i^{th} block to the j^{th} block. In an undirected graph, this matrix will be symmetric, and in a graph with only one set of nodes linking to each other, this matrix will always be square. To formalise this, assuming the blocks are labelled from 0 to $K-1$, z_i represents the block the i^{th} node is assigned to, and the existence of an edge between an arbitrary node in the ${z_i}^{th}$ block and the ${z_j}^{th}$ block is assumed to be an observation of a random variable, with the following distribution:

$$A_{ij} \sim \text{Bernoulli}(B_{z_i z_j}) \tag{1}$$

An assumption not required for this model, but generally assumed in practice is that the edge probability between two distinct nodes in the same cluster is higher than the edge probability between two distinct nodes from distinct clusters (Lee and Wilkinson, 2019). Intuitively this provides a good foundation for the clustering approaches, as the edges usually represent some level of connection, and one would expect more connections within clusters than between clusters.

4.3.1 Degree-corrected Stochastic Block Model

The degree-corrected stochastic block model (DCSBM) is a sub-model of the SBM. It considers degree-corrected terms in each membership by introducing node-specific pa-

rameters that scale the expected number of connections for each node. ρ_i represents the degree correction parameter of the i^{th} node and the distribution of each entry of A can now be assumed to be distributed as follows:

$$A_{ij} \sim \text{Bernoulli}(\rho_i \rho_j B_{z_i z_j}) \tag{2}$$

This way, a better representation of the graph can be obtained because the edge probability between nodes will incorporate the degrees of each node.

4.3.2 SBMs of Bipartite Graphs

When working on a network represented by a bipartite graph, there will be two distinct sets of nodes, X and Y, and a node in one set will exclusively have links to the other set of nodes (i.e. none to its own set). For this reason, block probability matrix $\mathbf{B}$, is derived from the binary data: for each distinct pair of nodes (x, y) from (X, Y) '1' is assigned if x links to y, '0' if x doesn't link to y. Then $\mathbf{B}$ is constructed as follows:

$\mathbf{B}_{ij}$ = probability an arbitrary node in the i^{th} cluster of X links to node j in the set Y.

Clearly $\mathbf{B}$ is of dimension $K \times D$, where K is the number of blocks in X, and D is the cardinality of Y.

4.3.3 SBMs of Weighted Graphs

Given a finite set of possible weightings in $\{a, ..., b\}$, $a, b \in \mathbb{Z}$ with cardinality $c = b - a$, a set of c SBMs, $W_i, i = a, a+1, ..., b$ is needed to represent all possible edges. $(W_i)_{jk}$ represents the probability of an edge of weight i between a node in block j and a node in block k.

Generally, only one edge is possible between two nodes, so $\sum_{i=a}^{i=b} (W_i)_{jk} \leq 1$.

4.3.4 The Dataset

As mentioned before, the graph is bipartite with one set of nodes being the MPs and the other set being the divisions they vote on. The objective is only to cluster the MPs into their respective parties. Let X be the set of MPs and Y be the set of divisions. A node in X can link 'positively' (if the MP votes 'yes', assign '1') to $y \in Y$, can link 'negatively' (if they vote 'no' assign '-1') to $y \in Y$, or can abstain from any link (assigning '0'). Therefore, this can be conceptualised as a weighted graph.

Thus, from the section above, under the SBM it is expected to obtain two block probability matrices, $\mathbf{B}_1$ and $\mathbf{B}_{-1}$, detailing how likely it is that a specific block votes 'aye' or 'no' (valued by '1' and '-1' respectively) on a specific division. MPs have now been grouped into their respective blocks which are assumed to have the same block-wise voting probability (instead of all MPs having an individual voting record). So under this model, all MPs that have been clustered into the i^{th} group have probability $(\mathbf{B}_1)_{ij}$ of voting 'aye' and probability $(\mathbf{B}_{-1})_{ij}$ of voting 'no' on the j^{th} division.

5 Methods Of Clustering and Analysis

In this section, the predictive outcomes of multiple clustering algorithms will be examined and compared.

Clustering algorithms are used to divide the nodes in a graph into different clusters, and in the context of this project, different clusters are the different party affiliations of MPs. Thus the predictive result of a clustering algorithm is a list of the predicted party affiliations of each MP. This result is stored as an n-dimension label vector where the i^{th} component has a value between 0 and K-1 representing which one of the K parties, the i^{th} MP is predicted to belong to.

The two main classes of clustering methods used are Gaussian mixture models and spectral clustering. The name 'spectral clustering' is not perfectly descriptive, as the Gaussian mixture model can be considered a form of spectral clustering, but it is used in this report for convenience.

An overview of these methods for clustering and analysis will be given, and then in Section 6 they will be utilised with the data.

5.1 Methods of Clustering

A Gaussian mixture model (GMM) is a probabilistic model that represents a dataset as a collection of Gaussian distributions. It assumes that the dataset is generated from a mixture of multiple Gaussian distributions, each representing a different cluster. GMM assigns a probability to each data point, indicating its likelihood of belonging to each cluster. By iteratively optimizing the model parameters, such as means, covariances, and weights of the Gaussian components, GMM finds the best fit for the given dataset.

Spectral clustering, on the other hand, is a graph-based clustering algorithm that operates on the affinity matrix of a dataset. It views the dataset as a graph, where each data point is a node, and the affinity between points is represented by edge weights calculated with the Gaussian radial basis function.

The Gaussian radial basis function (RBF) (Fornberg, Larsson, and Flyer, 2011) is

defined as follows:

$$rbf(x, y) = \exp\{-||x - y||^2\} \tag{3}$$

The normalized graph Laplacian (Chung, 1997) is defined as follows:

$$L_{sym} := D^{-1/2} L D^{-1/2} \tag{4}$$

Spectral clustering constructs a similarity graph, applies a dimensionality reduction technique such as spectral embedding, and then performs clustering on the reduced space using techniques like K-means. This algorithm leverages the eigenvectors of the affinity matrix to capture the underlying structure of the dataset and group similar data points together.

5.2 Specific Algorithms for Clustering Algorithms

Algorithm 1 Spectral Clustering

Input Latent position matrix $\hat{\mathbf{X}} \in \mathbb{R}^{N \times d}$ as calculated in Graph Embedding for the Dataset, optionally spherically transformed as in Spherical Coordinate Transformation and Degree Corrected SBM

Output Predicted label vector v $\in \mathbb{R}^N$

1: Generate an affinity matrix $\mathbf{W}$, where W_{ij} represents a 'distance' between the i^{th} and j^{th} nodes represented in $\hat{\mathbf{X}}$, calculated by rbf($\hat{\mathbf{X}}_\mathbf{i}, \hat{\mathbf{X}}_\mathbf{j}$) using (3). $\mathbf{W} \in \mathbb{R}^{N \times N}$.
2: Calculate the diagonal matrix $\hat{\mathbf{D}}$ of W, and construct L_{sym} using (4)
3: Calculate $N \times K$ matrix $\mathbf{B}$ equal to the orthogonal eigenvectors of L_{sym} corresponding to the K largest eigenvalues of L_{sym}.
4: Cluster the rows of $\mathbf{B}$ using K-means clustering, returning a predicted label vector v $\in \mathbb{R}^N$ where v_i is the predicted cluster of the i^{th} node of $\hat{\mathbf{X}}$

This approach really involves two stages. The first step is using SVD to embed the nodes, truncating and performing spherical transformation to account for the degree differences within clusters. The second step is applying the spectral clustering algorithm as described above in this section. Normally, this spectral clustering algorithm would be applied directly to an adjacency matrix, not an already embedded matrix. However, as will be explored later in Section 7, the results obtained from this combination outperform any other algorithm that has been tried.

Gaussian Mixture Model:

The Gaussian mixture model varies most significantly from other forms of clustering in the fact that it incorporates information about the covariance structure of the clusters, as well as position vectors of the centres of each cluster. It also does not perform hard clustering, giving each node a vector that assigns a probability of being in each cluster instead of just assigning each node to a single cluster. It assumes that each cluster is distributed around its centroid vector by a multivariate normal distribution, where both the mean vector and covariance matrix needs to be estimated.

GMM uses the expectation maximisation (EM) algorithm (Bishop, 2006) to fit the nodes to clusters. The EM algorithm relies on the assumption that the probability of a given latent position is calculated with the following formula:

$$\Pr(\hat{X}_i = x) = \sum_{k=1}^{K} \pi_k \cdot \Pr(X_i = x | z_i = k))$$

And that with Bayes's theorem:

$$\Pr(Z_i = k | \hat{X}_i = x) = \frac{\pi_k \cdot \Pr(\hat{X}_i = x | z_i = k)}{\sum_{j=1}^{K} \pi_j \cdot \Pr(\hat{X}_i = x | z_i = j))}$$

Where $\hat{X}_i$ is the i^{th} row of the latent position matrix, x is a value of this row, z_i is the cluster that the i^{th} node is assigned to, π_k is the probability of $\hat{X}_i$ being assigned to cluster k and K is the total number of clusters.

Given it is assumed each cluster is normally distributed, the joint probability distribution of all N nodes is:

$$\Pr(\hat{X}_1 = x_1, \ldots, \hat{X}_N = x_N | \boldsymbol{\pi}, \boldsymbol{\mu}, \boldsymbol{\Sigma}) = \prod_{i=1}^{N} \left(\sum_{k=1}^{K} (\pi_k \cdot \mathcal{N}(x_i; \mu_k, \Sigma_k)) \right)$$

Where μ_k is the mean vector of cluster k and Σ_k is the covariance matrix of cluster k

The log-likelihood function is given by:

$$\ln\left(\Pr(\hat{X}_1 = x_1, \ldots, \hat{X}_N = x_N | \boldsymbol{\pi}, \boldsymbol{\mu}, \boldsymbol{\Sigma})\right) = \sum_{i=1}^{N} \ln \left(\sum_{k=1}^{K} (\pi_k \cdot \mathcal{N}(x_i; \mu_k, \Sigma_k)) \right) \quad (5)$$

Now the goal is to get maximum likelihood estimations (MLE) for $\boldsymbol{\pi}_k, \boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k$. Differentiating the log-likelihood function with respect to $\boldsymbol{\mu_k}$, and setting that equal to 0 gets:

$$0 = -\sum_{n=1}^{N} \frac{\pi_k \cdot \mathcal{N}(x_n|\boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k)}{\sum_j \pi_j \cdot \mathcal{N}(x_n|\boldsymbol{\mu}_j, \boldsymbol{\Sigma}_j)} \boldsymbol{\Sigma}_k(\boldsymbol{x}_n - \boldsymbol{\mu}_k)$$

For convenience, write

$$\gamma(z_{nk}) = \frac{\pi_k \cdot \mathcal{N}(x_n|\boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k)}{\sum_j \pi_j \cdot \mathcal{N}(x_n|\boldsymbol{\mu}_j, \boldsymbol{\Sigma}_j)} \tag{6}$$

Through rearranging, the following can be obtained:

$$\boldsymbol{\mu}_k = \frac{1}{N_k} \sum_{n=1}^{N} (\gamma(z_{nk})\boldsymbol{x}_n) \tag{7}$$

where

$$N_k = \sum_{n=1}^{N} \gamma(z_{nk})$$

Then do a similar process, differentiating with respect to Σ_k to obtain its MLE as:

$$\boldsymbol{\Sigma}_k = \frac{1}{N_k} \sum_{n=1}^{N} (\gamma(z_{nk})(\boldsymbol{x}_n - \boldsymbol{\mu}_k)(\boldsymbol{x}_n - \boldsymbol{\mu}_k)^T \tag{8}$$

To obtain the MLE of $\boldsymbol{\pi}_k$ the fact that $\sum_{k=1}^{K} \pi_k = 1$ is taken into account, i.e. each node is in exactly 1 cluster. Therefore a Lagrange multiplier is included and the following must be maximised:

$$\ln\left(\Pr(\hat{\boldsymbol{X}}|\boldsymbol{\pi}, \boldsymbol{\mu}, \boldsymbol{\Sigma})\right) + \lambda(\sum_{k}^{K} \pi_k - 1)$$

which when differentiated, set to 0 and rearranged gives:

$$\boldsymbol{\pi}_k = \frac{N_k}{N} \tag{9}$$

The EM algorithm then consists of 4 steps:

Algorithm 2 EM Algorithm

Input Latent position matrix $\hat{\mathbf{X}} \in \mathbb{R}^{N \times d}$ spherically transformed as in Spherical Coordinate Transformation and Degree Corrected SBM

Output MLEs for $\boldsymbol{\pi}, \boldsymbol{\mu}, \boldsymbol{\Sigma}$, the probabilities of assigning a node to a specific cluster, the mean vectors of each cluster, and the covariance matrices of each cluster.

1: Initialise the values of $\boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k$ and $\boldsymbol{\pi}_k$ using an algorithm, either through a random choice of numbers within a reasonable range, or through a K-means algorithm.
2: **E step.** Evaluate $\gamma(z_{nk})$ as in (6) using the current parameter values
3: **M step.** Reevaluate values for $\boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k$ and $\boldsymbol{\pi}_k$ using the respective MLE algorithms given above (7, 8, 9). It is important to do it in order so that the newly evaluated $\boldsymbol{\mu}_k$ is used in the evaluation of $\boldsymbol{\Sigma}_k$.
4: Evaluate the log-likelihood as in 5 and check for convergence by comparing with the log-likelihood of the previous iteration, and repeating until the difference is less than some decided threshold value.

The labels for each node are obtained by finding the values that maximise the likelihood function with the parameters calculated by the EM algorithm.

5.3 Methods of Analysis

Visualization methods for analysing results have limited value given the size of the data set - a scatterplot of points can only be easily visualised for two dimensions and a bar chart of the size of each cluster does not include data on whether the right nodes are in each cluster. To get a consistent and comparable metric, the Rand index (RI) will be computed to measure and compare the quality of predictions across different algorithms (Rand, 1971) (Hubert and Arabie, 1985). The RI is used to measure the similarity of the predicted cluster vector, with the real cluster affiliation vector given. The score of RI ranges from 0 to 1, with higher numbers indicating better results. Hubert and Arabie (1985) explores the calculations for the RI, defining:

$\mathbf{S} = (s_1, \ldots, s_n)$ as the set of n nodes;

$\mathbf{P} = (P_1, \ldots, P_r)$ the calculated partition of the data into r clusters, and

$\mathbf{T} = (T_1, \ldots, T_r)$ the true partition of the data into r clusters.

Then let

$\mathbf{a}$ = the number of pairs of nodes in $\mathbf{S}$ that are in the same subset in $\mathbf{P}$ and in the same subset in $\mathbf{T}$. In other words, if two distinct nodes from the same cluster are correctly

placed into their true cluster under the model (i.e. they 'pairwise' agree), this counts as 1. All these such pairs are then summed up.

$\mathbf{b}$ = the number of pairs of nodes in $\mathbf{S}$ that are in different subsets in $\mathbf{P}$ and in different subsets in $\mathbf{T}$.

The proposed measure for comparing partitions, the RI, is then described by:

$$R = \frac{\mathbf{a} + \mathbf{b}}{\binom{n}{2}}$$

An intuitive approach would be that $\mathbf{a} + \mathbf{b}$ is the number of pair agreements between nodes, and $\binom{n}{2}$ is the total pairings of nodes. It is clear that this measure, therefore, can be thought of as an agreement ratio which in itself is a good measure for partition comparisons.

It is therefore clear to see why the RI sits between 0 and 1, with RI = 1 implying a perfect partition estimate for the data.

6 Simple Clustering Analysis

In this section, the results of the two algorithms introduced in Section 5 to cluster matrix $\hat{\mathbf{X}}$ are analysed and improvements are suggested.

Initial prediction results are given by clustering $\hat{\mathbf{X}}$ into 13 clusters, where $\hat{\mathbf{X}}$ is obtained directly from the original data set using the chosen clustering algorithms.

The prediction outcomes are enhanced by truncating the source data. This involves removing certain portions of the original data set. A reduced $\hat{\mathbf{X}}$ matrix based on the truncated data is produced and clustering is applied using the same algorithms.

6.1 Clustering The Data into 13 Communities

As there are 13 parties in the data set, including the speaker, initially the spectrally embedded 13-dimensional data set, $\hat{\mathbf{X}}$, is clustered into 13 groups. The number 13 is used in the spectral clustering and GMM algorithms as the number of components.

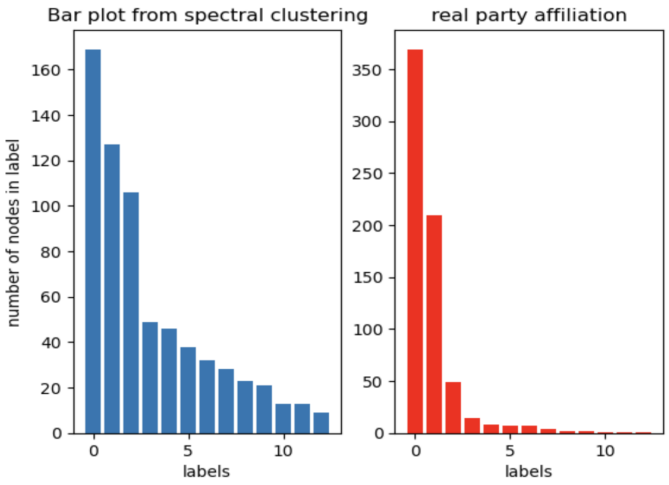

Figure 2: Bar chart of the number of nodes in labels sorted obtained by spectral clustering

The bar chart above is obtained by counting the number of nodes with each value in the predicted label vector and sorting by decreasing value. The sorting is just for visualization purposes, and it can be done because the labels generated are arbitrary.

A bar plot with similar values to the true party labels does not guarantee a good result. It does not provide clarity on whether individual MPs are correctly clustered. For instance, if the number of nodes labelled as 0 closely matches the true count of

Conservative MPs (370), it cannot be determined how many of those MPs are actually Conservatives. They could belong to other parties but have been grouped with Conservatives. However, a significant difference between the true party label bar plot and the predicted result bar plot indicates that the ideal outcome has not been achieved, and they give a clear and readable reason to reject a clustering method. Therefore, all bar plots in this report serve solely as auxiliary visualizations for RI.

By comparing the bar charts of the prediction vector and the real party affiliation, it is immediately clear the spectral clustering algorithm fails to predict the small clusters, as the smallest predicted party has over 10 members, something only true of the biggest 4 parties in the real data. The small parties like SDLP and Alba have lower than 5 party members (from Table 1). It is very unlikely that a clustering algorithm would accurately detect parties with such a small number of members. Moreover, from Table 3, all parties (except Speaker) with lower than 5 party members have an agreement index with the Labour party which is higher than 90%, which makes them even harder to detect, as a high agreement index implies that they are close to each other in latent space. As shown in Figure 1 for example, there are clearly independent MPs closer to the centroid vector of the Labour and Conservative clusters than some Labour and Conservative MPs.

The RI of 0.7311 for this algorithm further indicates that the prediction is not very accurate.

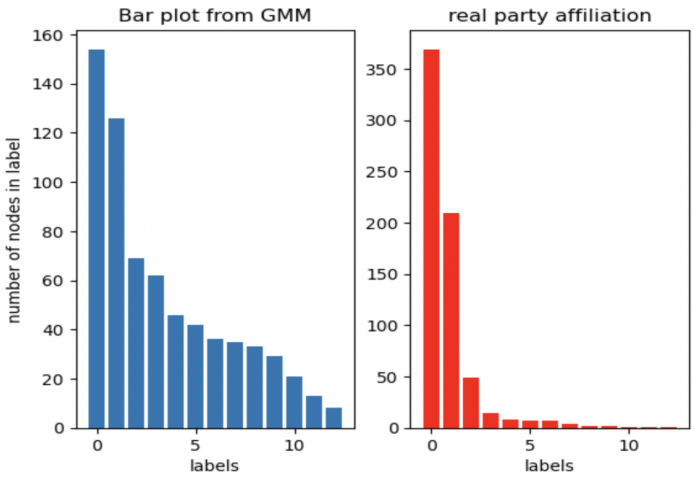

Figure 3: Bar chart of the number of nodes in labels sorted obtained by GMM

The conclusions obtained from the bar chart for the GMM algorithm are the same as

the conclusions for Spectral clustering. GMM also fails to detect small clusters.

The RI for GMM is slightly lower than that of spectral, at 0.7157.

The performance of the two algorithms is not ideal for grouping the data into 13 clusters. Through an analysis of the RIs and the size of the smaller parties in question, it is clear that the data needs to be filtered more carefully before accurate predictions can be made. Note finally that this clustering cannot be represented visually as it would have to be reduced to at most 3 dimensions.

6.2 Reducing The Cluster Number to 5

To improve the prediction score the required clusters can be reduced to 5 by removing all MPs except those affiliated with the Conservatives, Labour, Scottish National Party, Liberal Democrats and the Democratic Unionist Party from the data.

Sinn Féin is not considered, as the party abstains from all voting, which makes it hard to perform spherical coordination transformation which will be discussed in the following section. Independents and Speaker are removed because they are not strictly parties. The other small parties (Plaid Cymru, Social Democratic Labour, Alba, Green and Alliance) are removed because of their small party size and high similarity with Labour, mentioned in the previous section.

After removing these party members, the adjacency matrix is 649×794 and the real party affiliation vector is a 649-dimensional vector. The spectral embedded adjacency matrix, which is the input of clustering algorithms, is now 649×5.

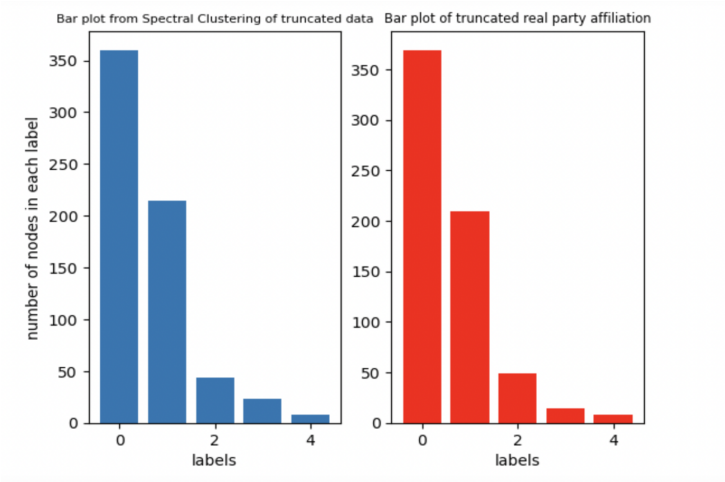

Figure 4: Truncated version of Bar chart obtained by Spectral Clustering

From a simple inspection of the bar charts, the performance seems to have improved significantly. The differences between the predicted result and the real affiliation data on the right are now hard to distinguish solely by comparing these visualizations bar charts.

The RI agrees with the conclusion from the visualization. The score increased from 0.7311 to 0.9438.

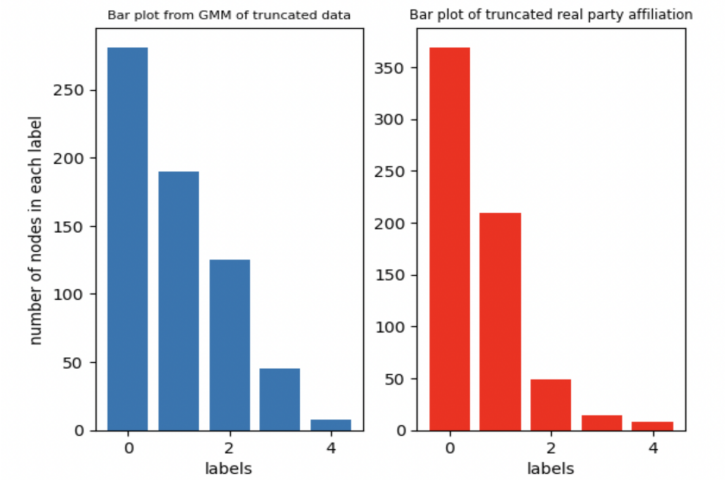

Figure 5: Truncated version of Bar chart obtained by GMM

The improvement of GMM is not as significant as that in spectral clustering. For example, the largest clustering should have over 350 nodes, but from the prediction of GMM, there are only around 300.

The change in RI is from 0.7157 to 0.8470. The increase is less significant than that of spectral clustering. Note again that this clustering cannot be represented visually as

it would have to be reduced to at most 3 dimensions.

In conclusion, based on the analysis conducted so far, spectral clustering has generally outperformed GMM. This is evident in both the 5-cluster and 13-cluster scenarios, where GMM produced worse prediction results compared to spectral clustering.

Furthermore, the data truncation procedure implemented resulted in a significant improvement in the performance of both GMM and spectral clustering. As a result, for the subsequent analysis of this data set, the truncated data consisting of 5 parties will be utilized instead of the original full data.

7 Spherical Coordinate Transformation and Degree Corrected SBM

As mentioned previously, there is a submodel under the SBM, DCSBM, which works on the assumption that any difference in magnitude of the embedded nodes is noise, assumed to be generated from a Uniform(0,1) distribution in (Sanna Passino, Heard, and Rubin-Delanchy, 2022), and steps should be taken to reduce this noise.

Following the results in (Section 6), degree correction using a spherical coordination transformation function (Sanna Passino, Heard, and Rubin-Delanchy, 2022) will be attempted in this section and further analysis will be built based on the result produced by degree correction: For each d-dimensional vector $\hat{\mathbf{x}}_i$ representing the latent embedded position of the i^{th} node, it is transformed into a $(d-1)$-dimensional vector $\boldsymbol{\theta}$ by:

$$\begin{aligned} \theta_1 &= \begin{cases} \arccos\left(\frac{x_2}{\|\mathbf{x}_{:2}\|}\right) & \text{if } x_1 \geq 0 \\ 2\pi - \arccos\left(\frac{x_2}{\|\mathbf{x}_{:2}\|}\right) & \text{if } x_1 < 0 \end{cases} \\ \theta_j &= 2\arccos\left(\frac{x_{j+1}}{\|\mathbf{x}_{:j+1}\|}\right), \quad \text{for } j = 2, \ldots, m-1 \end{aligned} \tag{10}$$

of which is presented in the aforementioned paper, (Sanna Passino, Heard, and Rubin-Delanchy, 2022). The first element of $\boldsymbol{\theta}$ is determined by the sign of $\hat{x}_1$ and the arccos value of $\hat{x}_2$ normed. To understand the motivation behind this choice, it is first worth noting that $-1 \leq \frac{x_2}{\|\mathbf{x}_{:2}\|} \leq 1$ and therefore $0 \leq \arccos\left(\frac{x_2}{\|\mathbf{x}_{:2}\|}\right) \leq \pi$. This implies:

$$0 \leq \arccos\left(\frac{x_2}{\|\mathbf{x}_{:2}\|}\right) \leq \pi \text{ if } x_1 \geq 0 \tag{11}$$

$$\pi \leq \arccos\left(\frac{x_2}{\|\mathbf{x}_{:2}\|}\right) \leq 2\pi \text{ if } x_1 < 0 \tag{12}$$

This creates a strict division between the spherical coordinates of nodes with positive and negative first singular values, and then sets all subsequent spherical coordinates to double the arccos value of the respective normed vectors so that they range between 0

and 2π.

The effect of this transformation is to ignore the magnitude of each component. The new spherical components only store information on the angle between $\hat{\mathbf{X}}$ and its respective axes, thus separating the 'direction' of each vector from its magnitude. This is the latent positions are effectively being spatially 'relocated' onto a unit $(d-1)$-sphere. It is therefore clear to see that there is a potential for better clustering algorithms, as there is now no variation in the magnitude of the latent position vectors.

7.1 Clustering results

The ASE will be performed first to obtain the embedded $\hat{\mathbf{X}}$ of dimension $\mathbf{649 \times 5}$. Then the spherical transformation function will be applied to $\hat{\mathbf{X}}$, obtaining $\hat{\mathbf{\Theta}}$ matrix of dimension $\mathbf{649 \times 4}$. The spectral clustering algorithm and GMM are then performed on $\hat{\mathbf{\Theta}}$.

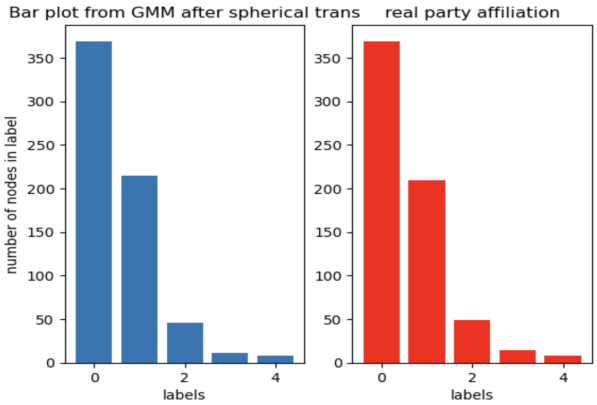

Figure 6: Result from GMM after spherical transformation

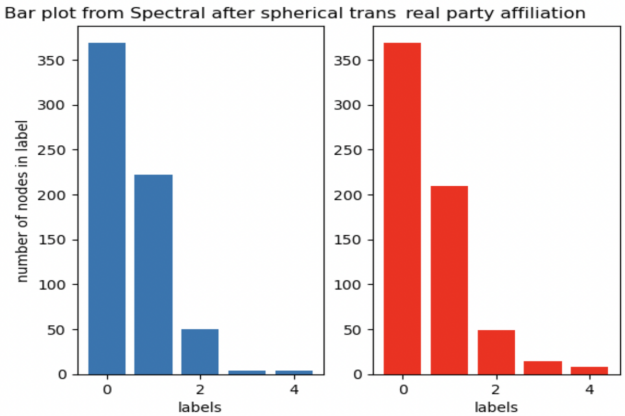

Figure 7: Result from Spectral Clustering after spherical transformation

The result from spectral clustering is improved in a minor way by the spherical transformation. But by comparing Figure 6 with Figure 5, it is observed that the quality of prediction of GMM is significantly improved. The RI demonstrates that as well.

The RI of the results from spectral clustering and GMM are 0.9848 and 0.9777 respectively. The score increases in both cases. The change in score from GMM is more significant, increasing from 0.8470 to 0.9777, compared to spectral clustering increasing from 0.9438 to 0.9848. However, the score from spectral clustering is still higher than that of GMM. The spherical transformation has both equalized and improved the performance of GMM and spectral clustering.

The RI of over 0.97 means that both algorithms clustered most nodes correctly. The few nodes, i.e. MPs, that are clustered wrongly may have special properties that distinguish them from the majority of their party members. The nodes that were allocated to the wrong clusters in each algorithm will be analysed further in Section 7.3.

7.2 Exploration of the effects of spherical transformation through visualization

This section will provide an initial analysis of how degree correction using spherical transformation improves the accuracy of the predictions by comparing the 2-dimensional scatter plots of nodes before and after spherical transformation.

As a point of reference, Figure 8 shows a scatter plot of 2-dimensional $\hat{\mathbf{X}}$ obtained by ASE.

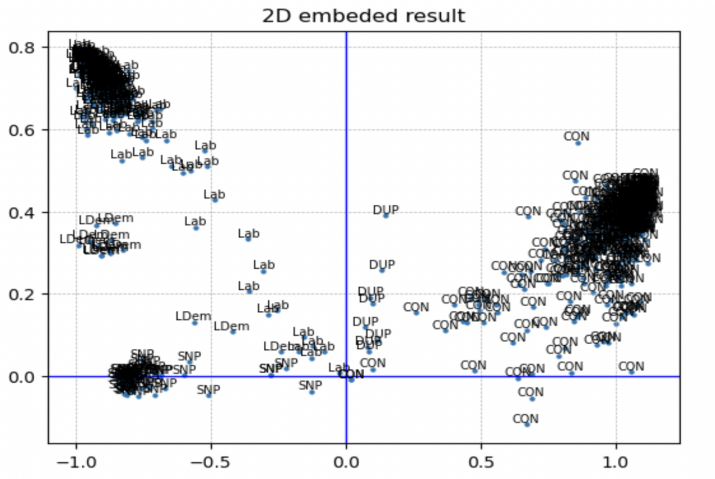

Figure 8: Visualization of 2D embedded results with only 5 parties obtained in the same way as Figure 11

From Figure 8 it is clear that among Labour and Conservative MPs, there are multiple MPs that have significantly smaller distances from the origin than the majority of MPs in their parties which also means that they tend to have smaller distances in latent space from other parties like LDem, SNP and DUP compared to the most Labour MPs and Conservatives.

Applying a spherical transformation algorithm to a 3-dimensional ASE produces a 649×2 $\hat{\mathbf{\Theta}}$ matrix. Each 2-dimensional row vector will then be plotted in a scatter plot.

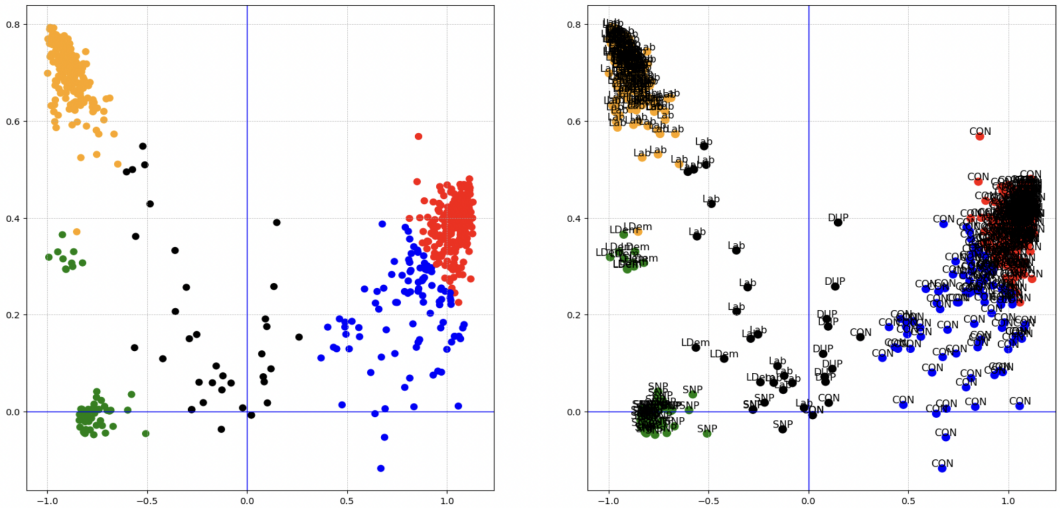

Figure 9: Left: a 2D scatter of result from spectral, with different colours representing different clusters; Right: True party affiliation attached on each node with the same colour in right-hand side plot

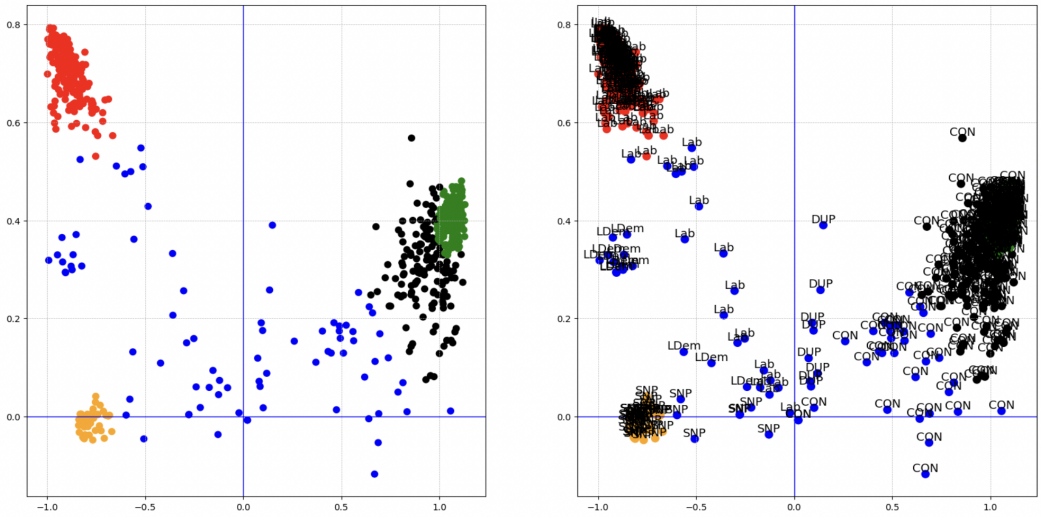

Figure 10: Left: a 2D scatter of result from GMM, with different colours representing different clusters; Right: True party affiliation attached on each node with the same colour in right-hand side plot

The lower magnitude of the nodes of these outlying MPs in the Labour and Conservative parties causes them to be clustered differently from the majority in their parties. In both spectral clustering and GMM, these outlying MPs in the two major parties tend to be clustered together into the same group with MPs from other small parties.

For example, in Figure 9, the black cluster encompasses the Labour MPs who are positioned closer to the origin than their party average, along with MPs from the Lib Dem and DUP parties. On the other hand, Figure 10 displays a larger cluster in blue

centred around the origin, with a notable inclusion of Conservatives who are similarly close to the origin.

What follows is a look at results after spherical transformation is applied and see how the method improves the prediction result.

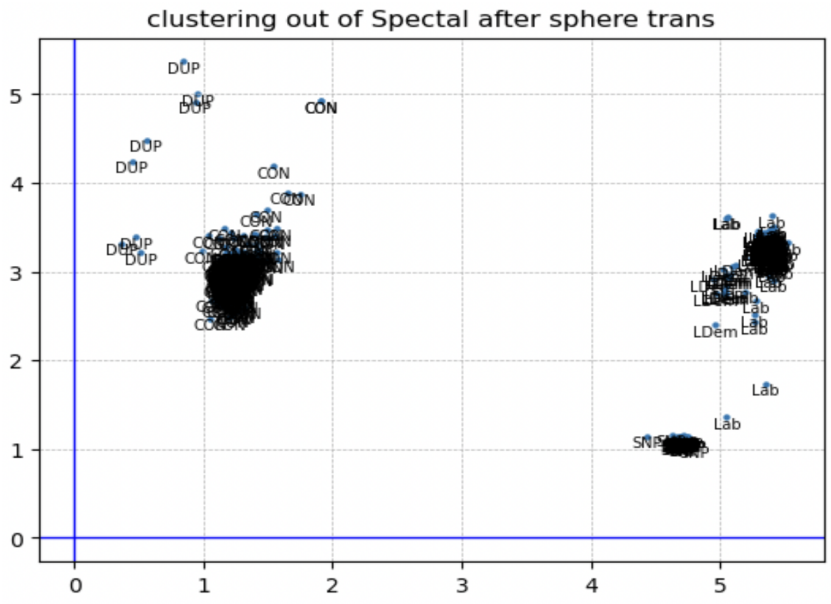

Figure 11: 2-dimensional Scatter plot of $\hat{\mathbf{\Theta}}$ matrix

From Figure 11, after applying the transformation function to the 649×3 embedded matrix $\hat{\mathbf{X}}$, it is observed that the points in the latent space are more tightly clustered together. Additionally, the MPs (nodes in the graph) who previously exhibited different behaviour compared to the majority now have a magnitude closer to the average magnitude of the group. Overall, the distances between nodes appear to be normalized, resulting in a higher density in the latent space within parties, and larger distances between parties. The reasoning behind why the transformation achieves this is covered above in Spherical Coordinate Transformation and Degree Corrected SBM.

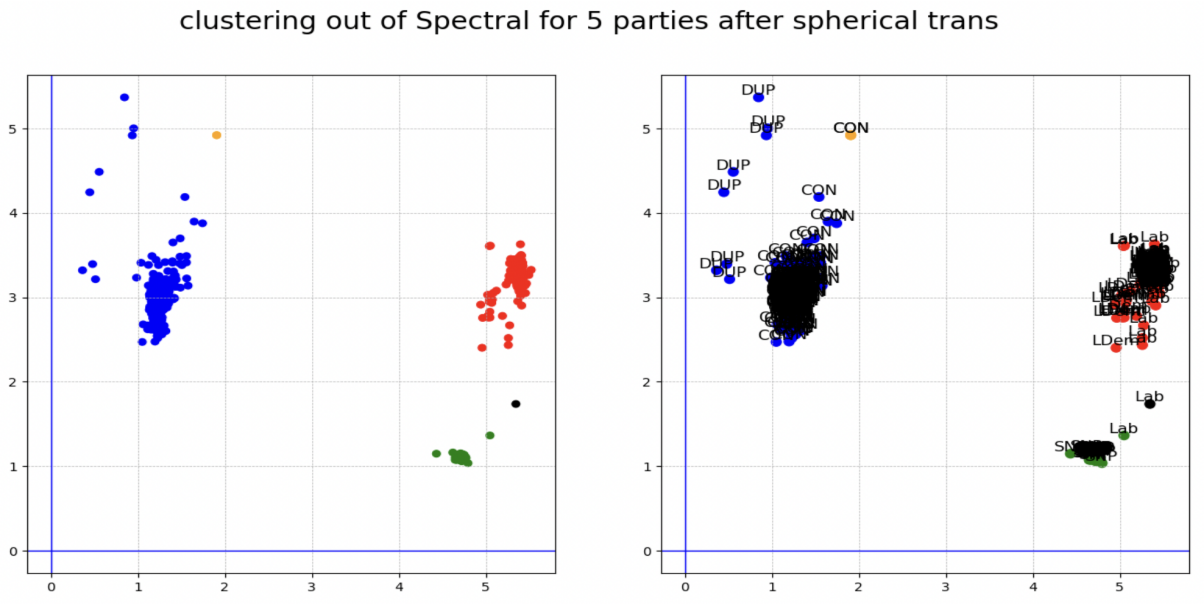

Figure 12: Left: a 2D scatter of result from spectral applied to $\hat{\mathbf{\Theta}}$, with different colours representing different clusters; Right: True party affiliation attached on each node with the same colour in right-hand side plot

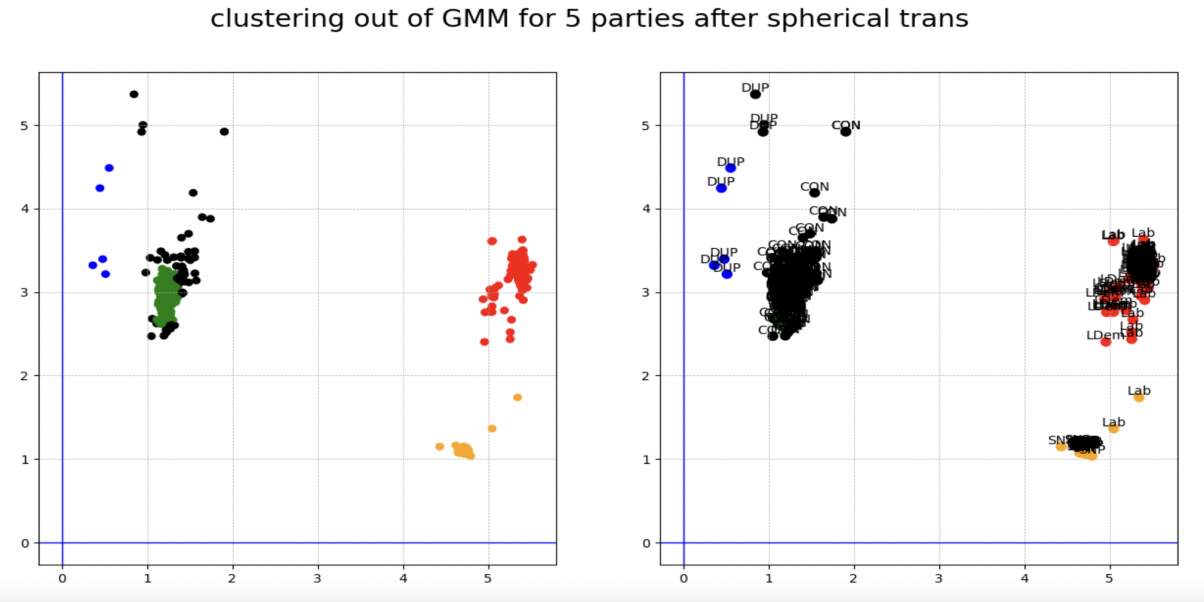

Figure 13: Left: a 2D scatter plot of result from GMM applied to $\hat{\mathbf{\Theta}}$, with different colours representing different clusters; Right: True party affiliation attached on each node with the same colour in right-hand side plot

From the scatter plots, it is also clear that the results have become more accurate. Compared with Figure 9, the result in Figure 12 clusters many more Conservative and Labour MPs correctly with the majority of their party members. While GMM still separates the Conservatives into two groups from Figure 13, most Labour MPs are identified as one group (except for 2 points), which results in a better result than Figure 10.

7.3 Analysis of the Misclustered Nodes

As discussed in Section 7.1, the best result obtained for the 5-cluster model was using degree correction by applying the spherical coordination transformation and then spectral clustering. The RI was 0.98, with only 19 MPs misclassified. These MPs and information on their misclassifications are displayed below. This was also using 5 components.

Table 5: Information on misclassified MPs

Name	Number of Votes	Real Party	Predicted Party
Munira Wilson	658	Lib Dem	Labour
Sarah Olney	636	Lib Dem	Labour
Daisy Cooper	618	Lib Dem	Labour
Wera Hobhouse	617	Lib Dem	Labour
Alistair Carmichael	610	Lib Dem	Labour
Wendy Chamberlain	604	Lib Dem	Labour
Tim Farron	587	Lib Dem	Labour
Jamie Stone	568	Lib Dem	Labour
Christine Jardine	581	Lib Dem	Labour
Layla Moran	565	Lib Dem	Labour
Edward Davey	542	Lib Dem	Labour
Kim Leadbeater	383	Labour	Lib Dem
Christian Wakeford	248	Labour	Lib Dem
Paulette Hamilton	194	Labour	Lib Dem
Simon Lightwood	177	Labour	Lib Dem
Samantha Dixon	104	Labour	Lib Dem
Andrew Western	82	Labour	Lib Dem
Ashley Dalton	56	Labour	Lib Dem
Rosie Winterton	12	Labour	SNP

There are a few immediate points:

The only misclassifications were between Labour and Lib Dem except for one Labour MP misclassified as SNP. This means every Conservative, SNP and DUP MP was correctly classified and no MP was incorrectly classified as Conservative or DUP.

Of the 14 Lib Dem MPs in the dataset, 11 of them were misclassified as Labour, and the three that were properly classified took the fewest votes of the 14. In contrast, the 7 Labour MPs that were misclassified as Lib Dem all took fewer votes than any of the misclassified Lib Dem MPs. However there are 16 Labour MPs who took fewer than 384

votes, and only 8 were misclassified. Having a low vote number is clearly a necessary condition for a Labour MP to be misclassified, but it isn't sufficient.

The one Labour MP misclassified as SNP was the Labour MP with the fewest votes

It appears that the similarity between Labour and Lib Dem voting records, as shown by their agreement index of 0.979 in Table 3, makes it very hard for any algorithm to differentiate them. Therefore, when forced into making 5 clusters, there exists a more significant divide between MPs in Labour and Lib Dem than party affiliations, so the two clusters are more accurately described as 'Labour or Lib Dem MPs with X characteristic' and 'Labour or Lib Dem MPs without X characteristic'. It seems likely this divide is related to the number of votes taken by each Labour or Lib Dem MP.

Comparing the 16 Labour MPs who voted fewer than 384 times, the date of their election separates them perfectly into misclassified MPs and correctly classified MPs. The 8 MPs who voted from the start of the data set in 2019, but left their positions before the end (or just don't vote frequently in the case of Keir Starmer and Siobhain McDonagh) are: Tracy Brabin, Rosie Cooper, Jeremy Corbyn, Jack Dromey, Mike Hill, Chris Matheson, Siobhain Mcdonagh and Keir Starmer. They were all correctly classified as Labour. The 8 who were elected in by-elections in 2022 and 2023 were all misclassified as Lib Dem. Furthermore, the three Lib Dem MPs who were accurately classified: Richard Foord, Sarah Green, and Helen Morgan, are the only three Lib Dem MPs who did not vote from the start of the data set. Rosie Winterton, the Labour MP misclassified as SNP, is the only outlier to this pattern, but she only voted 12 times which would make it extremely difficult, if not impossible, to classify her accurately.

Hence, it appears that the two clusters which together contain all Labour and Lib Dem MPs are not split between parties, but split between MPs who voted from the start and MPs who did not.

If all Lib Dem MPs are classified as Labour MPs in the source data, and the number of clusters in the model is decreased to 4, all MPs are correctly classified as one of 'Conservative', 'Labour or Lib Dem', 'SNP', or 'DUP'. Except again for Rosie Winterton.

7.4 Effects and Justification of using a 4 cluster model

The results change when you change the number of components and the number of clusters that the spectral clustering and Gaussian clustering algorithms should consider from 5 to 4. This change can be justified by the fact that 4 is the number of spherical components, since $\hat{\mathbf{\Theta}}$ has one fewer column than $\hat{\mathbf{X}}$, and therefore 4 is the natural choice for the number of components that the clustering algorithms should consider. It can also be justified by looking at the results described in Analysis of the misclustered nodes, and seeing that Labour and Lib Dem are not meaningfully separated when the data is split into 5 clusters, and instead, they are split into MPs who were elected in 2019, and those who were not.

It is not sensible to increase the number of initial components to 6. In order to get 5 spherical components to match up with the 5 clusters the real data has been filtered into. This would be over-fitting the data. If $\hat{\mathbf{X}}$ had six columns, the 6th column would only contribute noise that would be compressed into $\hat{\mathbf{\Theta}}$ and decrease the quality of the model. This is exemplified by the RI of this approach: 0.6093

Changing the number of components that the spectral clustering algorithm should consider to 4 while keeping the number of clusters at 5 does not achieve ideal results in this case. It does cluster all of Labour and Lib Dem together into one cluster, which is arguably better than splitting them into 2 based on the date of the election, but it forces a 5^{th} cluster by splitting the DUP up into two clusters of 4 MPs each.

The final results of a model with 4 clusters and 4 components are almost perfect if you consider Labour and Lib Dem as the same party. The Conservative and DUP parties are clustered perfectly, one cluster matches up with SNP perfectly except for the inclusion of 1 Labour MP, Rosie Winterton, who only voted 12 times, and then one cluster includes all of Labour and Lib Dem. The RI is now 0.985, with the main reason for its imperfection being the 14 Lib Dem MPs clustered in with Labour.

8 Performing Clustering on Simulated Data

In this section clustering analysis will be performed on simulated data from both SBM and DCSBM instead of on real data. And the results from simulated SBM and DCSBM will then be compared to the results obtained in the real data set.

The same procedure as the real data set - applying ASE to the data set, performing clustering algorithms on the data set and measuring the predictions from the algorithms using RI, will be applied to the simulated data set.

The analysis will include examining how changes in the number of blocks and number of nodes within the block affect algorithm performance. Additionally, one example in Sanna Passino, Heard, and Rubin-Delanchy (2022) about the DCSBM will be reproduced.

8.1 Simulating using the SBM

In this section, simulated data will be obtained through SBM.

The probability matrix $\mathbf{B} \in \mathbb{R}^{K \times K}$ mentioned in section 4.3 will be first generated using a $Beta(\alpha, \beta)$ distribution with $\alpha = 3$, $\beta = 2$. K is the number of clusters in the model. After obtaining $\mathbf{B}$, the adjacency matrix $\mathbf{A} \in \mathbb{R}^{N \times N}$ of a graph with N nodes is generated by the aforementioned *Bernoulli* distribution in Section 4.3. $\hat{\mathbf{X}}$ is then obtained by performing ASE on $\mathbf{A}$.

Changing the number of blocks affects the performance of each algorithm differently. In Figure 14 the number of blocks increases from 5 to 52 with 50 nodes in each block.

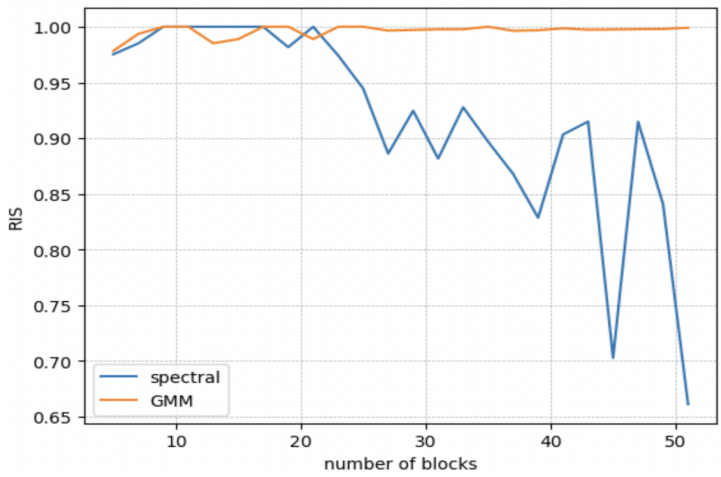

Figure 14: The changes of RI of both spectral clustering and GMM when the number of blocks changes

Both algorithms performed well on a small number of blocks. While the RI of GMM is close to 1 regardless of the number of blocks, the RI of spectral clustering decreases as the block number increases. When the cluster number exceeds 20, the RI of GMM become stable, having less significant fluctuation around 1.0.

The variation in RI for 10 clusters varying in size between 50 and 800 nodes is also notable.

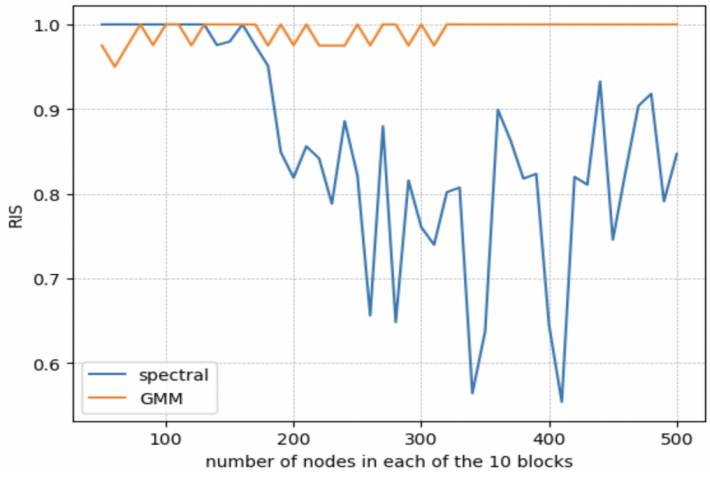

Figure 15: The changes of RI of both spectral clustering and GMM when the number of nodes in each block changes

Similarly, both algorithms perform well when the number of nodes in each of the 10 clusters is small. However, for larger numbers, the RI of the GMM algorithm stabilises around 1.0, but the RI of spectral clustering starts to fluctuate significantly. It is also notable that when the number of nodes in each of the 10 clusters exceeds 300, the result

of GMM is constant at 1.0.

The possible conclusions that can be drawn from the two plots are the following.

The performance of GMM is more stable and better than that of spectral clustering on SBM-generated data.

The reason for the good performance of GMM is the central limit theorem(CLT) (Athreya et al., 2016). When the number of blocks or the number of nodes within each block increases, the total number of nodes in the graph increases. The distribution of the latent positions of the nodes can then be modelled increasingly accurately using a Gaussian distribution, giving a better result using GMM.

The suboptimal performance of spectral clustering in SBM could be attributed to the dense positioning of points in the latent space. As described in Section 5.1, spectral clustering relies on the structure of the affinity matrix. Since probability matrix B is generated using a constant distribution regardless of the number of nodes, the central points of each block in the latent space tend to have similar positions to those of other blocks. This similarity leads to a higher likelihood of clashes between clusters in the latent space when the number of blocks increases. Likewise, when the number of nodes in each block increases, the proximity between blocks results in overlaps of nodes in different clusters, making it difficult for spectral clustering to distinguish them as separate clusters.

8.2 Simulating using the DCSBM

In this section, data will be simulated using DCSBM. The method to generate data is similar to the previous section under SBM. However, there are two major differences.

Firstly, in Sanna Passino, Heard, and Rubin-Delanchy (2022), in the section that is to be reproduced, B is the following 2×2 matrix.

$$B = \begin{bmatrix} 0.1 & 0.05 \\ 0.05 & 0.15 \end{bmatrix} \tag{13}$$

Therefore, the B matrix will be this constant 2×2 matrix instead of being generated from a *Beta* distribution.

Secondly, as introduced in Section 4.3, the degree correction parameter ρ_i is included in the parameter of the *Bernoulli* distribution used to generate $\mathbf{A}$. The ρ_i parameter of each node will be generated using $Beta(\alpha, \beta)$ distribution with $\alpha = 2$, $\beta = 1$.

8.2.1 Visualizations of DCSBM on $\hat{\mathbf{X}}$ and $\hat{\Theta}$

The differences between the structure of SBM-generated data and DCSBM are analysed by comparing the visualizations of the embedded 2-dimensional $\hat{\mathbf{X}}$ matrices. Further analysis of the simulated DCSBM data will mainly focus on the effect of spherical transformation (10) on the results of both algorithms.

Both SBM and DCSBM use the same 2×2 $\mathbf{B}$ matrix mentioned before, with 2 blocks represented in different colours containing 400 nodes each. Following is the visualization of the result.

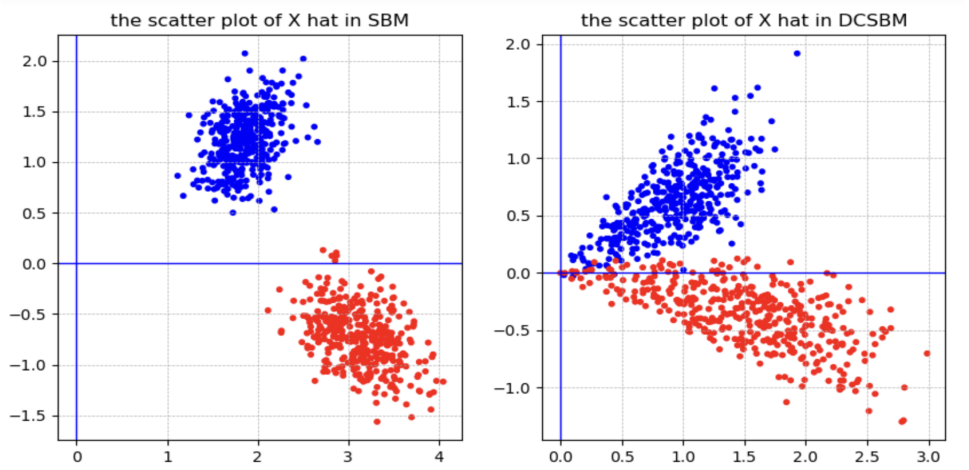

Figure 16: Left: visualization of $\hat{\mathbf{X}}$ from SBM; Right: visualization of $\hat{\mathbf{X}}$ from DCSBM; Different colors represent nodes in different blocks of 2×2 $\mathbf{B}$

There is a distinction between the two clusters in the visualisations of $\hat{\mathbf{X}}$ from both SBM and DCSBM in Figure 16. However, in DCSBM there is a clear drift from the respective centroid vectors towards the origin. This drift appears linear and causes the nodes near the origin to be much harder to classify, because there is a smaller distance between nodes in different clusters near the origin, and also because the structure of the clusters has been distorted, decreasing the accuracy of a Gaussian model that looks for elliptical clusters.

Figure 17 compares the ASE of a 3×3 $\hat{\mathbf{X}}$ obtained from the same adjacency matrix A in Figure 16 with that of $\hat{\mathbf{\Theta}}$ obtained by applying the spherical transformation function on $\hat{\mathbf{X}}$.

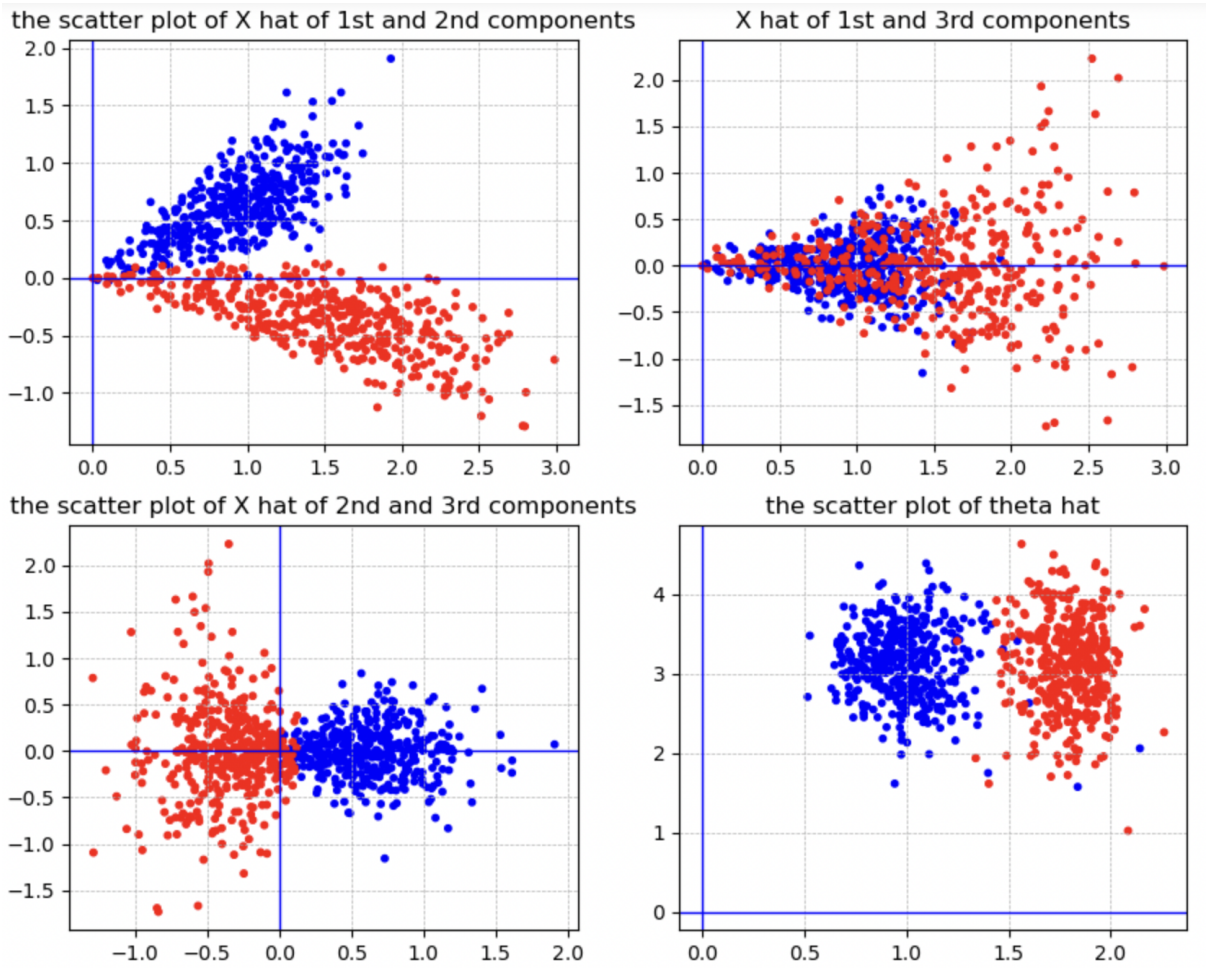

Figure 17: First 3 plots: combinations of different column vectors in the 3×3 $\hat{\mathbf{X}}$ matrix plotted against each other; Bottom Right plot: two column vectors of 2×2 $\hat{\mathbf{\Theta}}$ plotted against each other

To obtain the desired 2-dimensional $\hat{\mathbf{\Theta}}$ matrix, the ASE algorithm in three dimensions is applied to the adjacency matrix A from Figure 16. This results in a 3×3 $\hat{\mathbf{X}}$ matrix. Since the $\hat{\mathbf{X}}$ matrix is three-dimensional, three distinct plots are used to represent the combinations of different column vectors from the $\hat{\mathbf{X}}$ matrix. The i^{th} column of the $\hat{\mathbf{X}}$ matrix is denoted $\hat{\mathbf{X}}_i$.

As the same adjacency matrix A is used, the first plot of $\hat{\mathbf{X}}_\mathbf{1}$ against $\hat{\mathbf{X}}_2$ is the same as the plot on the right-hand side in Figure 16. However, the second and third plot obtained by plotting $\hat{\mathbf{X}}_1$ against $\hat{\mathbf{X}}_3$ and $\hat{\mathbf{X}}_2$ against $\hat{\mathbf{X}}_3$ respectively, present different result from each other. Both plot 1 and plot 3 have presented a clear division between the nodes

in different blocks, while the division in plot 2 is much less clear. The drifting feature described under Figure 16 of the first plot is also evident in both the second and third plots.

The results from the first 3 plots agree with the effect of SVD described in Section 4.1. The use of SVD in ASE ensures that the first 2 components of $\hat{\mathbf{X}}$, $\hat{\mathbf{X}}_1$ and $\hat{\mathbf{X}}_2$ are more important than other components, storing the most prominent features in the adjacency matrix A. However, other components like $\hat{\mathbf{X}}_3$ provide extra information apart from the first two components by introducing a new dimension to latent space.

The results obtained from the first three plots align with the impact of SVD discussed in 4. By using SVD in ASE, the first two components of the $\hat{\mathbf{X}}$ matrix, $\hat{\mathbf{X}}_1$ and $\hat{\mathbf{X}}_2$, are prioritized as they capture the most significant features present in the adjacency matrix A. However, it is important to note that the additional components, such as $\hat{\mathbf{X}}_3$, provide supplementary information beyond the first two components by introducing an additional dimension to the latent space.

The impact of the spherical transformation is evident when comparing the last plot of the 2×2 $\hat{\mathbf{\Theta}}$ matrix with the first three plots obtained prior to the transformation. The application of the spherical transformation causes the disappearance of the drifting feature observed in the DCSBM data. Instead, elliptical clusters become apparent, indicating more prominent Gaussian features mentioned in Section 7 compared to the original $\hat{\mathbf{X}}$ matrix. This highlights the effectiveness of the spherical transformation in accentuating the Gaussian characteristics of the data, leading to improved cluster representation and separation.

8.2.2 Further Analysis about the Spherical Transformation on DCSBM

Figure 18 shows how the RI of the results from both algorithms changes varies before applying the spherical transformation algorithm. The number of nodes in each of the two blocks varies from 500 to 1900.

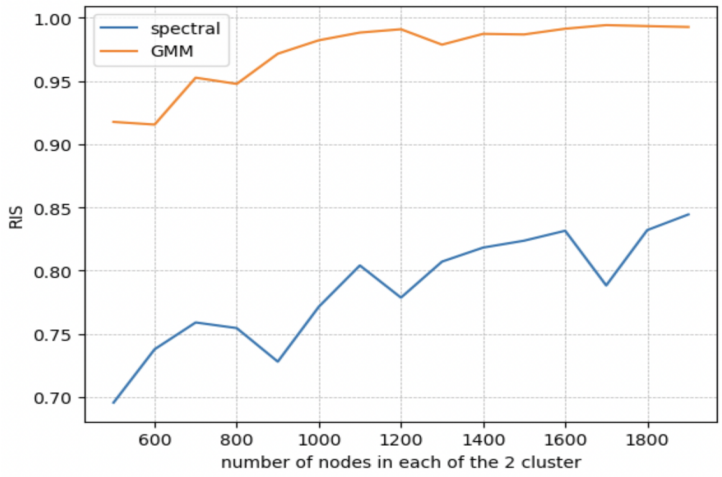

Figure 18: RI of both spectral clustering and GMM before applying spherical transformation

Before applying the spherical transformation, it is evident that GMM consistently outperforms spectral clustering. The RI of GMM is consistently higher than that of spectral clustering. The difference in RI scores between the two algorithms remains around 0.2 across various numbers of nodes within each block.

Additionally, in SBM, the performance of GMM is positively correlated with the number of nodes, as explained by the CLT discussed earlier. This positive correlation can be also found in the result of DCSBM. The RI increases from around 0.92 to close to 1.0 as the number of nodes in each of the 2 blocks grows from 500 to 1900. However, spectral clustering exhibits the opposite trend. In SBM, spectral clustering tends to perform worse as the number of nodes increases, but in DCSBM, the pattern reverses. The RI of spectral clustering shows a positive correlation with the number of nodes in the graph, similar to GMM.

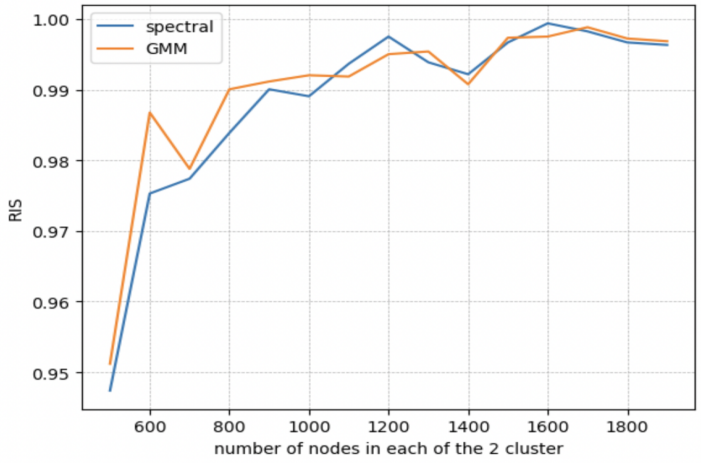

Figure 19: RI of both spectral clustering and GMM after applying the spherical transformation to DCSBM generated data

The performance of both algorithms has been improved by applying the spherical transformation function. Prior to the transformation, the RI of GMM was below 0.95 when the number of nodes in each cluster was less than 800. However, after the transformation, the RI exceeds 0.95 when the number of nodes is 500. Similarly, spectral clustering also improves. Before the transformation, spectral clustering had a lower RI than GMM. However, after the transformation, the RI of spectral clustering become very similar to that of GMM.

In the current results obtained from both GMM and Spectral clustering, a clear correlation between the number of nodes and the RI can be observed. As the number of nodes in each cluster increases from 500 to 1900, the RI of both algorithms rises significantly, reaching a value close to 1 (perfect agreement).

The obtained results and conclusions align with the findings presented in Sanna Passino, Heard, and Rubin-Delanchy (2022) study, despite the difference in the clustering algorithm employed (spectral clustering instead of k-means). Both GMM and spectral clustering initially demonstrate suboptimal performance when applied directly to the embedded matrix $\hat{\mathbf{X}}$. However, after applying the spherical transformation to obtain $\hat{\mathbf{\Theta}}$, the performance of both algorithms shows a substantial improvement. This improvement can be attributed to the CLT, as the two-dimensional $\hat{\mathbf{\Theta}}$ tends in probability to a Gaussian distribution, enhancing the clustering performance (Sanna Passino,

Heard, and Rubin-Delanchy, 2022). Additionally, the Gaussian characteristics of $\hat{\mathbf{\Theta}}$ can also be seen in the visualization results obtained earlier in this section.

8.3 Comparison between the Simulated Data and the Real Data

In this section, the results from the simulated data will be compared to the previous results from the real data set.

First by comparing the visualizations of $\hat{\mathbf{X}}$ in figure 19 with figure 8, it is clear the political data have a similar structure to the DCSBM. In figure 8 the data drifts towards the origin similar to the DCSBM in figure 19.

The significant performance improvement observed in both GMM and spectral clustering, as discussed in Section 7.1, can be further explained by the similarity in data structure with DCSBM.

The similar data structure with DCSBM can be applied to explain the significant improvement of the performance in both GMM and spectral clustering in the real data mentioned in Section 7.1. In both simulated DCSBM and the real data, the performance of both algorithms improved after applying the spherical transformation (10) to the embedded matrix $\hat{\mathbf{X}}$. As concluded in the previous section $\hat{\mathbf{\Theta}}$ is more accurately modelled by mixed Gaussian distributions than the original $\hat{\mathbf{X}}$ matrix generated from a DCSBM as the number of nodes increases. The Gaussian nature of $\hat{\mathbf{\Theta}}$ can also explain why the improvement in GMM is more significant than that of spectral clustering in real data.

Despite the similarities in data structure between real data and simulated DCSBM data, significant differences still exist between the two. Specifically, when examining the clustering results based on the embedded matrix $\hat{\mathbf{X}}$, there are contrasting observations between DCSBM and real data. In the DCSBM case, as discussed in Section 8.2, the RI of GMM is significantly higher than that of spectral clustering. However, when analyzing real data, as described in Section 6.2 the RI of spectral clustering is significantly higher than that of GMM. The aforementioned difference suggests that the structure of the real data may exhibit less prominent Gaussian characteristics compared to the simulated data derived from DCSBM. This deviation in Gaussian characteristics could be attributed to

the presence of a strict party whip in most political parties, as described in Section 3. The strict enforcement of voting rules within each party, combined with the unknown division between 'forced' votes and 'free' votes reduces the random component in the distribution of real data, making the Gaussian model less suitable compared to the simulated data generated by DCSBM.

9 Adjusted Rand Index and Possible Bayesian Approaches

This section will discuss the potential methods that could be used to further cluster and analyse the data. All estimates and algorithms in this report have been assessed using the Rand index. As explained in (Section 5.3), this score reflects the pairwise similarities of each cluster in two different clusterings. There exists an adjusted Rand index (Vinh, Epps, and Bailey, 2009), which adjusts the Rand index for *chance* agreement by comparing the observed agreement between clusterings with the expected agreement under a random clustering model. This would require a random modelling of the real data under the SBM, which could be useful should prior data be known(such as the agreement of specific outliers, such as Rosie Winterton). Adjusted scores under these conditions provide a good start for future research.

As briefly outlined in the introduction, Bayesian approaches can be applied to estimate any of the parameters that have been chosen holistically in this report. For example, 13 and then 5 are chosen as the fixed number of clusters. However, this does not mean that better results are not possible. If K is the number of clusters, K can be modelled as its own random parameter, using maximum likelihood estimation, to find the best value. To reiterate, the data set was shrunk to 5 clusters specifically for the dataset and for valid reasons described previously in (Section 6.2). This means that for a different dataset, insight into the data would be needed to choose a new appropriate number of clusters, and also the reasoning for removing certain nodes might not be as rigorous as it is based on given information about the UK political parties (for example the removal of Sinn

Féin).

10 Conclusions

Both Gaussian mixture models and spectral clustering have been used, analysed and manipulated, with analyses performed via the Rand index. The voting records of UK Members of Parliament were used as a case study and the number of clusters greatly affected the subsequent results, with 13 clusters (the true number of political parties considered) producing an inaccurate result.

Using justifications in (Section 6.2) decreasing the number of clusters to 5 produced more promising results. Due to the strong similarities between the two parties, no algorithm was found that could separate them. Treating these two parties as one group and clustering into 4 produced the best results. It is important to note that in both clustering cases, spectral clustering outperformed GMM clustering. Hence, there is reason to believe the voting records do not follow a Gaussian distribution.

While both of these models have been considered under the SBM, using ideas stated in Section 7 the model could be degree corrected, namely by the DCSBM, which was shown visually with Figure 11. Intuitively, this was an improvement from Figure 10 (which was not done under the DCSBM) because it factored out the magnitudes of each of the latent vectors (representing each of the nodes) - stopping the clustering algorithms from clustering the vectors with the smallest Euclidean distance to the origin. Furthermore, the spectral clustering method under this degree-corrected model again performed better than GMM, although by a much smaller margin. Spectral clustering performed better on the SBM and degree correction clearly retains the necessary spectral properties of the graph since the lines that each cluster makes from the origin are evident. The RI of the spectral clustering method under the spherical transformation gave the largest score produced. This was the most effective algorithm that has been considered for sorting the dataset into 5 clusters representing different parties.

Supplementary Material

The supplementary material for this article contains two Jupyter notebooks with the use of the clustering algorithms and other libraries that led to the results, plots and simulation results detailed in this paper. This is publicly available at https://github.com/hamisr/M2R-project, for anyone to replicate these findings through these models.

References

Athreya, A et al. (2016). "A Limit Theorem for Scaled Eigenvectors of Random Dot Product Graphs". *Sankhya A* 78 (1), 1–18. DOI: 10.1007/s13171-015-0071-x.

Bishop, Christopher M. (2006). *Pattern Recognition and Machine Learning (Information Science and Statistics)*. Berlin, Heidelberg: Springer-Verlag.

Chung, F. (1997). "Spectral graph theory". *CBMS Regional Conference Seires in Mathematics* 92. DOI: 10.1090/cbms/092.

Fornberg, Bengt, Elisabeth Larsson, and Natasha Flyer (2011). "Stable Computations with Gaussian Radial Basis Functions". *SIAM Journal on Scientific Computing* 33.2, 869–892. DOI: 10.1137/09076756X.

Hegger, R., H. Kantz, and L. Matassini (2001). "Noise reduction for human speech signals by local projections in embedding spaces". *IEEE Transactions on Circuits and Systems I: Fundamental Theory and Applications* 48.12, 1454–1461. DOI: 10.1109/TCSI.2001.972852.

Higgs, Brandon W., Jennifer Weller, and Jeffrey L. Solka (2006). "Spectral embedding finds meaningful (relevant) structure in image and microarray data". *BMC Bioinformatics* 7 (1). DOI: 10.1186/1471-2105-7-74.

Hubert, Lawrence and Phipps Arabie (1985). "Comparing Partitions". *Journal of Classification* 2, 193–218. DOI: 10.1007/BF01908075.

Lee, Clement and Darren J. Wilkinson (2019). "A review of stochastic block models and extensions for graph clustering". *Applied Network Science* 4 (122). DOI: 10.1007/s41109-019-0232-2.

Rand, William M. (1971). "Objective Criteria for the Evaluation of Clustering Methods". *Journal of the American Statistical Association* 66 (336), 846–850. DOI: 10.1080/01621459.1971.10482356.

Rubin-Delanchy, Patrick et al. (2022). "A Statistical Interpretation of Spectral Embedding: The Generalised Random Dot Product Graph". *Journal of the Royal Statistical Society Series B: Statistical Methodology* 84 (1), 1446–1473. DOI: 10.1111/rssb.12509.

Sanna Passino, Francesco, Nicholas A. Heard, and Patrick Rubin-Delanchy (2022). "Spectral Clustering on Spherical Coordinates Under the Degree-Corrected Stochastic Blockmodel". *Technometrics* 64 (3), 346–357. DOI: 10.1080/00401706.2021.2008503.

Vinh, Nguyen Xuan, Julien Epps, and James Bailey (2009). "Information theoretic measures for clusterings comparison: is a correction for chance necessary?" *ICML '09: Proceedings of the 26th Annual International Conference on Machine Learning*, 1073–1080. DOI: 10.1145/1553374.1553511.

Xu, Dongkuan and Yingjie Tian (2015). "A Comprehensive Survey of Clustering Algorithms". *Annals of Data Science* 2 (1), 165–193. DOI: 10.1007/s40745-015-0040-1.