RADBOUD UNIVERSITY

MASTER'S DEGREE THESIS IN ARTIFICIAL INTELLIGENCE

Predicting connectomes using noisy and incomplete data

Author:
Annet Meijers[1]
annetmeijers@student.ru.nl
Student number: s4027957

Supervisors:
Rembrandt Bakker[1]
Paul Tiesinga[1]
Morten Mørup[2]
Marcel van Gerven[1]

[1]*Donders Institute for Brain, Cognition and Behaviour, Radboud University*
[2]*Section for Cognitive Systems, Technical University of Denmark*

Abstract

Tract tracing is a technique to show pathways of connections in the brain. The technique is invasive and therefore very costly and time consuming. Furthermore, not all results obtained with the technique are perfect. Therefore, a lot of tract tracing data is incomplete and/or noisy. By modelling the data with the latent space algorithm, we can predict unknown data and reduce the errors in the known data. The macaque tract tracing data of Markov et al. (2012) and Felleman & Van Essen (1991) are both incomplete. Furthermore, their claims about the density of the macaque's connectome are inconsistent with each other. By fitting a latent space model on the data, the unknown data can be estimated and some of the inconsistencies in the claims can be interpreted. The mouse tract tracing data of Zingg et al. (2014) consist of two different observations of the same connectome. The data, however, is the same for only 79% of the connections. By designing a new type of latent space algorithm, able to fit ordinal data, the two observations are modelled. By fitting the latent space to both datasets simultaneously, it is possible to merge the datasets and form a ground truth connectome. The algorithm used has yet to be perfected, since the found connectome shows some inconsistencies with the expected results. However, the likelihood and accuracy results of the fitted latent space model indicate the usefulness of this kind of models for the future.

Contents

1 Introduction

Connectomics is the field of research studying brain connectivity. In connectomics the connectivity is analyzed at multiple scales, from the synaptic connections between neurons (micro-scale) to the connections between various brain areas (macro-scale) (Behrens & Sporns, 2012) The ultimate goal of connectomics is to obtain a connectome; a dataset with the structural descriptions of the elements and connections in the brain (Sporns, Tononi, & Kötter, 2005). Having a connectome has a couple of advantages. First of all, to fully understand how a network (like the brain) works it is important to know all elements and connections of the network. A connectome will increase our understanding of how the functions of the brain arise from the structure. This can also help increase understanding of neurological and neuropsychiatric disorders (Bohland et al., 2009) and symptoms that arise from brain damage. Furthermore, a connectome can be used in neuroinformatics to inform a (large-scale) computational model of various brain functions (Sporns et al., 2005).

When talking about connections in the brain, there is a distinction between anatomical (or structural), functional and effective connectivity. In case of anatomical connectivity, the connections are physical connections between two elements, such as an axon connecting two neurons. Functional connectivity is concerned with the correlation between the activities of two elements. Take for example two brain areas that are both active during the same tasks. Lastly, effective connectivity defines the influence one element has on another element. For example, if a neuron is firing after being activated by another neuron, the second neuron has an influence on the behaviour of the first neuron (Friston, 1994).

One way to analyze the structural connectivity is to look at white-mater pathways. Tract tracing and diffusion imaging are two methods that are often used to find such pathways. Tract tracing is a technique to trace the connectivity of a neuron using fluorescent tracers (Markov et al., 2012). The tracers get injected in the brain and after a few days of survival the animal is sacrificed and its brain is removed. The path the tracers took in those days can then be analysed under a microscope. Tracers can be either anterograde (from the neuron's soma (the source) to its axon terminals (target)) or retrograde (the other way around; in this case the injection area will be called the target) (Zingg et al., 2014).

Diffusion imaging uses magnetic resonance imaging (MRI) to measure the diffusion of water in white matter tracks (Smith et al., 2006). Using diffusion imaging, large axon bundles can be shown.

Both diffusion imaging and tract tracing data have advantages and disadvantages. A big advantage of diffusion imaging is the fact that a complete connectome can be formed using only one brain, avoiding problems with combining data of different brains (Bakker, Wachtler, & Diesmann, 2012). Using tract tracing only the connections with one single brain area can be shown in one brain. Another disadvantage of tract tracing is its invasiveness, which makes it, in contrast to diffusion imaging, unsuitable for human experimentation.

However, there are also various major advantages to using tract tracing over diffusion imaging. Only a few examples are given here. First of all, it can be used to uncover long-range connections. Furthermore, in contrast to diffusion imaging, tract tracing studies can show the direction of a certain connection. Lastly, it allows for the estimation of connection strengths (Bakker et al., 2012).

Although tract tracing techniques have advantages over diffusion imaging, the technique is not perfect. Variants of the same experiments could reveal different pathways in the brain due to measurement errors and performing all possible injections in the brain is often too expensive and time consuming, leading to incomplete data.

The results of tract tracing data are used to make various statements about the connectivity in brains. However, due to the erroneous and missing data, these statements can vary

substantially across studies. One example is the density of connections in the brain. Both Felleman & Van Essen (1991) and Markov et al. (2012) made claims about the density in the macaque's cortex. The claims made are very different, though. Where Markov et al. report a density of 66%, Felleman and van Essen report a density of only 31%.

The aim of this research is to use generative models to decrease uncertainty in connectomes obtained using tract tracing experiments. The aim of such a generative model is to model how the observed variables arise from the underlying latent variables. Bayesian inversion of these models enables us to estimate these latent variables based on noisy data. These models can then be used to fill in missing data or to resolve conflicting data. By filling in missing data, more informed claims could be made about e.g. the graph-theoretical properties of the resulting connectomes. This can help to resolve discussions about, for example, connection density in the brain. With data fusion we can resolve the issue of conflicting data, making it possible to have a better estimation of the true connectome.

This thesis addresses the question whether and how we can use generative models to decrease uncertainty in connectomes due to incomplete or noisy data and how we can use these methods to make claims about the graph-theoretical properties of the true connectomes.

Part 1 Predicting connectomes using incomplete data

In the first part of this thesis the aim is to predict missing data in connectomes of macaques obtained with tract tracing methods. To predict this missing data the latent space model will be used.

2 Methods

2.1 Latent space model

A lot of research on networks is done in the field of social networks. Most social networks have four features in common. First of all, transitivity assumes that if individual A has a connection with B, and B a connection with C, it is more likely that A has a connection with C. Next there is homophily on observed attributes, which means that two individuals with similar properties are more likely to connect. Then, clustering assumes that individuals cluster in groups such that there is more connectivity within the groups than between groups. Lastly, there is degree heterogeneity, which refers to the trend that some individuals are more likely to receive or send connections than other individuals (Krivitsky, Handcock, Raftery, & Hoff, 2009). Since these four features are very common, link prediction algorithms typically incorporate these features when predicting new links.

Although the exact structure of a brain network is not known, there are empirical and theoretical reasons to assume that brain networks have some properties in common with social networks (Bassett & Bullmore, 2006). Using this knowledge it might be possible to use the techniques for link prediction in social networks for link prediction in brain networks. To strengthen this claim, some preliminary experiments have been performed. Results of these experiments can be found in Appendix A.

In the latent space model it is assumed that all individuals i have an unobserved location $\boldsymbol{z_i}$ in a D-dimensional Euclidean latent space. Connections between individuals depend only on their location in latent space. This means that connections are assumed to be independent of other connections, given the latent space positions (Hoff, Raftery, & Handcock, 2002). Hence,

$$P(\boldsymbol{Y}|\boldsymbol{Z}, \alpha) = \prod_{i \neq j} P(y_{i,j}|\boldsymbol{z_i}, \boldsymbol{z_j}, \alpha) \quad (1)$$

where the probability of link is given by the logit link function

$$P(y_{i,j}|\boldsymbol{z_i}, \boldsymbol{z_j}, \alpha) = \frac{1}{1 + \exp(-\eta_{i,j})} \quad (2)$$

with $\eta_{i,j} = \alpha - \|\boldsymbol{z_i} - \boldsymbol{z_j}\|$. Here, $\boldsymbol{Y}$ represents the connections, columns of $\boldsymbol{Z}$ are the locations of the individuals in latent space, α is a bias term and $\|\cdot\|$ is the Euclidean norm.

To estimate the locations and regression parameters, a Markov Chain Monte Carlo (MCMC) algorithm will be used (Hoff et al., 2002). This algorithm generates a chain of T possible latent spaces, where a new latent space $(t+1)$ in the chain depends on the previous latent space (t). With use of the `latentnet` package for R (Krivitsky & Handcock, 2008) a latent space model is fitted for every dataset (for exact settings see Appendix B). This package is based on various papers written about the latent space algorithm (Handcock, Raftery, & Tantrum, 2007; Hoff et al., 2002; Krivitsky et al., 2009).

After estimating a chain of latent spaces an expected connectome is calculated for every sample. In the connectome for sample t a connection is assumed to exist if $P(y_{i,j}^{(t)}|\boldsymbol{z_i^{(t)}}, \boldsymbol{z_j^{(t)}}, \alpha^{(t)}) \geq 0.5$. If this probability is less than 0.5, it is assumed the connection does not exist. Next to generating this chain of possible connectomes the marginal probabilities for all connections are computed:

$$P(y_{i,j}) = \frac{1}{T}\sum_{t=1}^{T} P\left(y_{i,j}^{(t)}|\boldsymbol{z_i^{(t)}}, \boldsymbol{z_j^{(t)}}, \alpha^{(t)}\right) \quad (3)$$

2.2 Estimation of graph-theoretical properties

For the recovered connectomes two graph-theoretical properties are analyzed. First, the density of the various connectomes with imputed connections is estimated. Second, brain areas are clustered using their expected connectivity.

2.2.1 Density of the network

In the literature there is a discussion about the density of macaque's brain networks. Densities go as low as 31% (Felleman & Van Essen, 1991) up to as high as 66% (Markov et al., 2012). Since there is no complete data of the connectivity in the brain this discussion is hard to solve. Analyzing the densities of the connectomes found by the algorithm can increase our understanding of the different results found in the literature and can help resolve this discussion.

The density is calculated for all connectomes in the chain. The density is defined as the fraction of all possible connections in the connectome that exist. Since self-links are not defined for tract tracing studies, these connections are excluded in the analysis. For the purpose of visualisation a Gaussian distribution is fitted over the densities found in the samples using the standard Matlab-function `normfit`.

2.2.2 Clustering

Brain areas will be clustered based on the marginal probabilities with which areas connect to one another. Before clustering, principal component analysis (PCA) (Wold, Esbensen, & Geladi, 1987) is performed on these marginal probabilities, using the standard Matlab-function `pca`. Only the first few principal components are used, such that the principal components together explain more than 95% of the variance.

The clustering itself is done with a Gaussian mixture model (GMM). The Matlab-function `fitgmdist` is used. This function fits a GMM using expectation maximization (Bishop, 2006). A GMM uses K multivariate Gaussian densities, one for each cluster, to describe the probability of a certain data point belonging to that cluster. The algorithm estimates the mean and standard deviation of the Gaussian densities by maximizing the likelihood:

$$L \equiv P(\boldsymbol{X}|\boldsymbol{\mu}, \boldsymbol{\Sigma}, \boldsymbol{\pi}) = \prod_{n=1}^{N} \left(\sum_{k=1}^{K} \pi_k \mathcal{N}(\boldsymbol{x_n}|\, \boldsymbol{\mu_k}, \boldsymbol{\Sigma_k}) \right) \quad (4)$$

where $\boldsymbol{X}$ is the set of N data points, so $\boldsymbol{x_n}$ are the principal component values of the nth brain area, $\boldsymbol{\mu}$ and $\boldsymbol{\Sigma}$ are the means and covariances of the multivariate Gaussian densities, and $\boldsymbol{\pi}$ are the mixing coefficients, i.e. the average responsibility taken by the Gaussian densities for explaining the data points.

A set of GMMs is fitted for the number of clusters K ranging from 1 to $N-1$. The fitting procedure assumed full covariance matrices $\boldsymbol{\Sigma}$ and a regularization factor $\lambda = 0.1$. Then for all fitted models the optimal number of clusters is chosen using the Akaike information criterion (AIC) (Bozdogan, 1987). The AIC is a measure of relative quality and can therefore be used to select the best model out of a set of models. The AIC is defined as follows:

$$\mathrm{AIC} = 2u - 2\ln(L) \quad (5)$$

where u is the number of free parameters in the model. By penalizing the number of free parameters the AIC favours simple models. The model with the minimal AIC is chosen and used to cluster the areas. The cluster c for data point n is given by:

$$c_n = \mathrm{argmax}_k(\pi_k \mathcal{N}(\boldsymbol{x_n}|\boldsymbol{\mu_k}, \boldsymbol{\Sigma_k}))\,. \quad (6)$$

2.3 Data

Various tract tracing studies with macaque monkeys as subjects have been performed (Bakker et al., 2012), but a full connectome of the macaque is not yet available. The missing parts in two partial connectomes described in the literature (Felleman & Van Essen, 1991; Markov et al., 2012) will be completed using the latent space model (see Figure 1).

The data by Felleman & Van Essen (1991) is given by an incomplete matrix of size 32×32 describing the connectivity of the visual system of the macaque. Every element $m_{i,j}$ denotes a connection between a source area i and a target area j. The data was generated by combining the results of 31 different studies. In this study, the binarized data of Felleman & Van Essen as found in the CoCoMac database (Bakker et al., 2012) was used.

Felleman & Van Essen report a density of 31%. However, they assumed that all unknown connections do not exist. Since around 33% of the connections in the matrix is unknown this drastically lowers the density estimations. When only taking into account the known connections a density of 44% is obtained.

The data by Markov et al. (2012) is a complete 29×91 matrix. In this matrix every column represents an injection in a target area and since they use retrograde tracers every row depicts a source area. The values in the matrix are FLNe (extrinsic fraction of labelled neurons) values. This is defined as $\mathrm{FLNe}_{i,j} = \frac{L_i}{L - L_j}$, where $\mathrm{FLNe}_{i,j}$ is the FLNe value of area i after injection in area j, L_i is the number of labelled neurons in area i and L is the total number of labelled neurons in the brain.

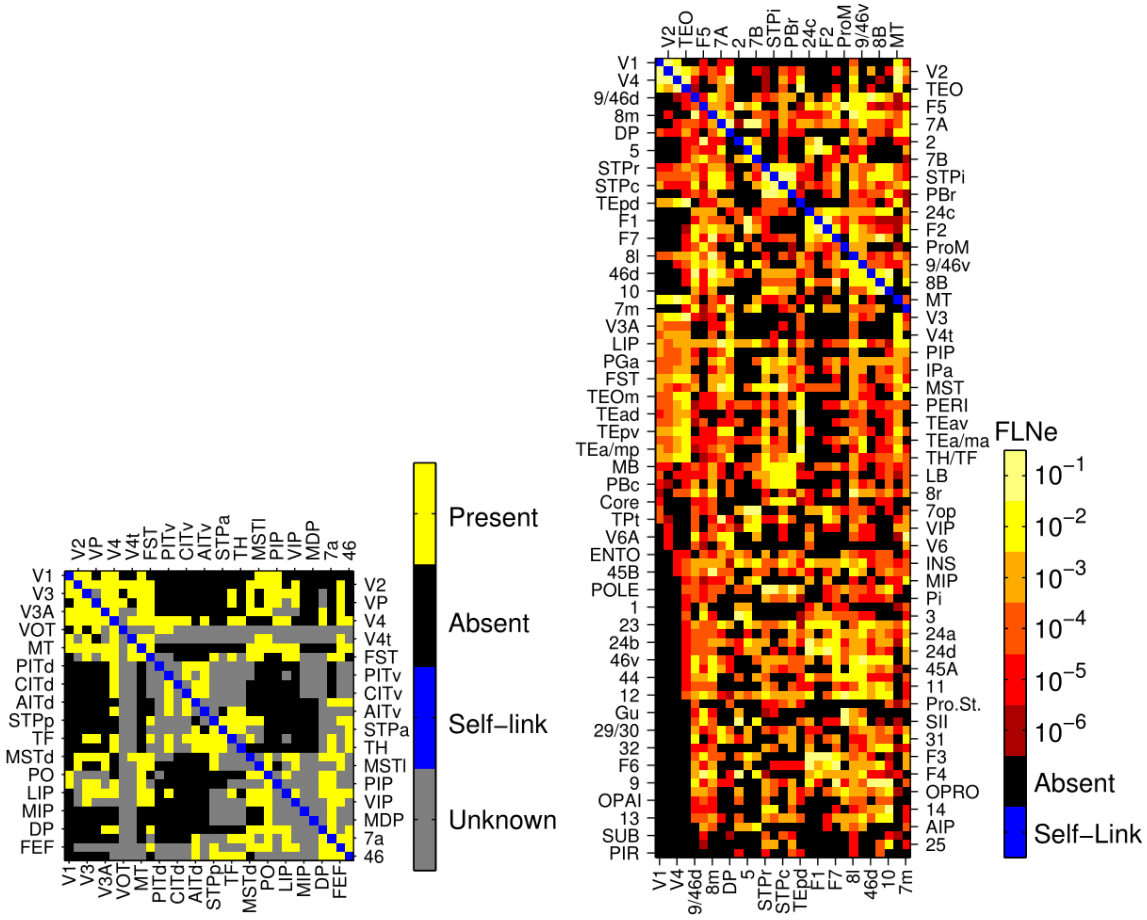

Figure 1: Data used in the experiments. On the left side data of Felleman & Van Essen (1991) and on the right side data of Markov, Ercsey-Ravasz, Gomes, et al. (2012).

Markov et al. define three levels of connections to interpret the data. They say every connection with a $\log_{10}(\text{FLNe}) > -2$ is a strong connection, connections with $-4 \leq \log_{10}(\text{FLNe}) \leq -2$ are moderate and $\log_{10}(\text{FLNe}) < -4$ indicate sparse connections. For their analysis Markov et al. include all projections, which leads to a density of 66%.

To form a complete connectome we need to find the connections between all 91 indentified brain regions, hence a 91×91 matrix. Since the data is a 29×91 matrix, there is still a big part of the data that has to be predicted by the latent space model.

The latent space model can only process and predict binary data. The data by Markov et al. is continuous and therefore it has to binarized. To do this, we put a threshold on the data so that all connections that have a higher FLNe value than this threshold are set to 1 (hence existing) and all connections with a lower FLNe value than this threshold are set to 0 (hence non-existing). Since choosing an appropriate threshold is not a well-defined task, three thresholds are defined, leading to three different datasets. First, all connections with a FLNe value higher than 0 are set to 1. Second, all connections for which holds that $\log_{10}(\text{FLNe}) \geq -4$ are set to 1, and lastly all connections with $\log_{10}(\text{FLNe}) \geq -2$ are set to 1. These thresholds are directly adopted from the Markov et al (2012) paper. Later in this thesis the thresholded data will be named Markov all connections, Markov moderate connections and Markov strong connections respectively.

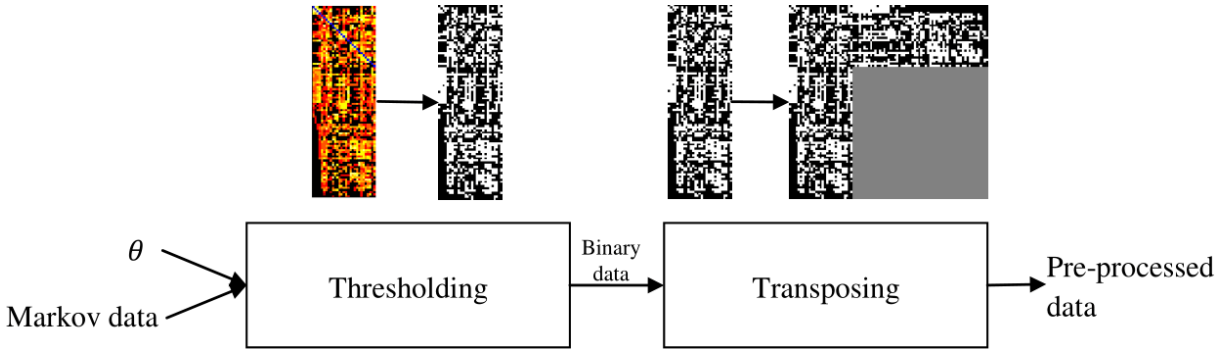

Figure 2: The pre-processing pipeline for the Markov data.

After the data is binarized another pre-processing step is performed, namely transposing the data. The data by Markov et al. is a 29×91 matrix, but the goal is to predict a 91×91 matrix. The matrix is therefore transposed to generate some extra data. As stated by Felleman & Van Essen (1991) the majority of connection are reciprocal, therefore we assume that a majority of the connections generated by transposing the matrix is correct. Note that the upper complete 29×29 matrix is kept fixed and is not transposed. In Figure 2 a visual representation of the pre-processing pipeline of the Markov data is shown.

2.4 Experiments

The latent space model is fitted for four different datasets. First the dataset of Felleman & Van Essen (1991) and then three datasets of Markov et al. (2012) using the different thresholds. For all experiments the same settings are used; ten chains of 24,000 samples are fitted, of which the first 20,000 are discarded as a burn-in. All latent spaces are fitted in two dimensions. All ten chains of 4,000 samples are used to calculate the Potential Scale Reduction Factor (PSRF). PSRF is a measure to determine the convergence of the chains (Brooks & Gelman, 2013). The PSRF is calculated over the distances between two nodes in the latent space as well as for the bias term α. To calculate the PSRF the function `psrf` of the `mcmcdiag` package by Simo Särkkä and Aki Vehtari[1] is used. If all distances and α are converged, the first chain is kept and the other nine chains are discarded in the further analysis.

In the first chain every sample in binarized in order to calculate the accuracy of the samples. Every connection is either 1 if the probability is larger than or equal to 0.5 or 0 otherwise. This results in 4,000 connectomes. For all these connectomes the accuracy is calculated. The accuracy is defined as the connections that have the same prediction as the value in the data divided by the number of connections that are known in the data. This accuracy is also calculated for the MAP connectome found by the latent space algorithm.

Next, the density is calculated for every binarized sample. Afterwards a Gaussian function is fitted for each of the density estimates (see Section 2.2.1).

With the unbinarized samples the marginal probability for every connection is calculated, by adding the probabilities of this connection in every sample and dividing it by the number of samples taken. With these marginal probabilities the different brain areas are clustered according to the GMM algorithm described in Section 2.2.2. The best clustering found is used to generate various figures. First, a visual representation of the marginal probabilities, such that areas in the same cluster appear together. Second, a plot of the MAP estimate of the latent space in which nodes in the same cluster have the same colour. Last, a colour reproduction of the macaque brain, where areas in the same cluster have the same colour.

[1] Downloaded at `http://becs.aalto.fi/en/research/bayes/mcmcdiag`.

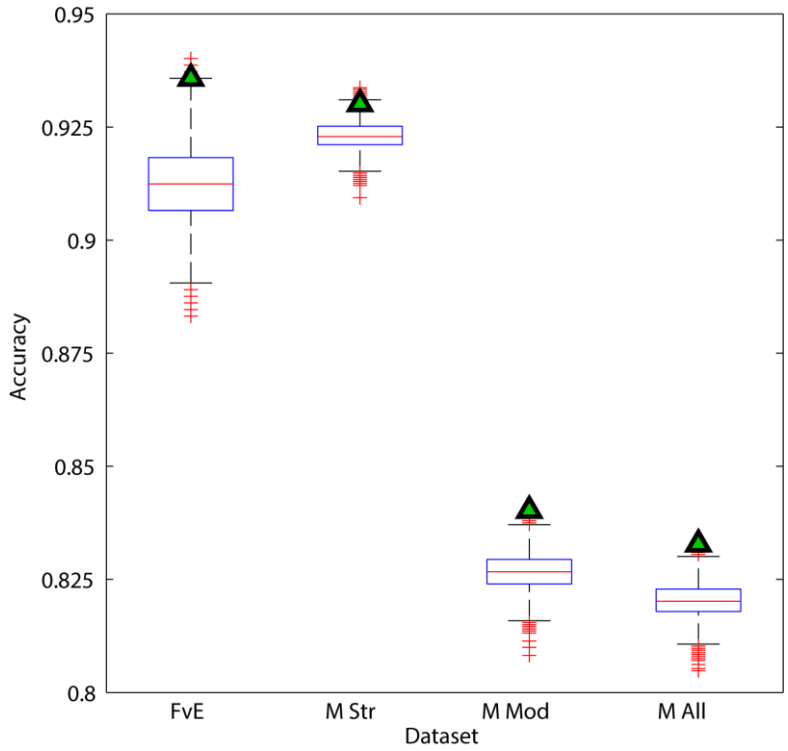

Figure 3: Accuracy of the algorithm on predicting known connections. The boxplots indicate the accuracies for the 4,000 samples. The green triangles indicate the accuracies of the MAP estimates.

3 Results

3.1 Density estimation

After training the model on all data (including the transposed data), the accuracy of the recovery of the known connections in calculated. In Figure 3 boxplots of the found accuracy of the latent space algorithm are shown. For all 4,000 samples the accuracy is calculated and a green triangle indicating the accuracy of the MAP estimate is added. The accuracies are relatively high (between 0.805 and 0.940 over all datasets), and for every dataset there is not a lot of difference between the accuracies for various samples. The accuracies of the MAP estimates are 0.936, 0.930, 0.840, and 0.833 for Felleman & Van Essen and Markov et al. (strong, moderate and all) respectively.

Using the various samples the densities of the true connectomes are estimated. In Figure 4 the Gaussian densities fitted over the various densities for the 4,000 samples per dataset are shown. Furthermore the density reported by Felleman & Van Essen (red dotted line, 0.31) and by Markov et al. (purple dotted line, 0.66) are shown. These densities vary from the true densities found in the used data. The densities of the data before using the latent space algorithm are plotted with the various dashed lines.

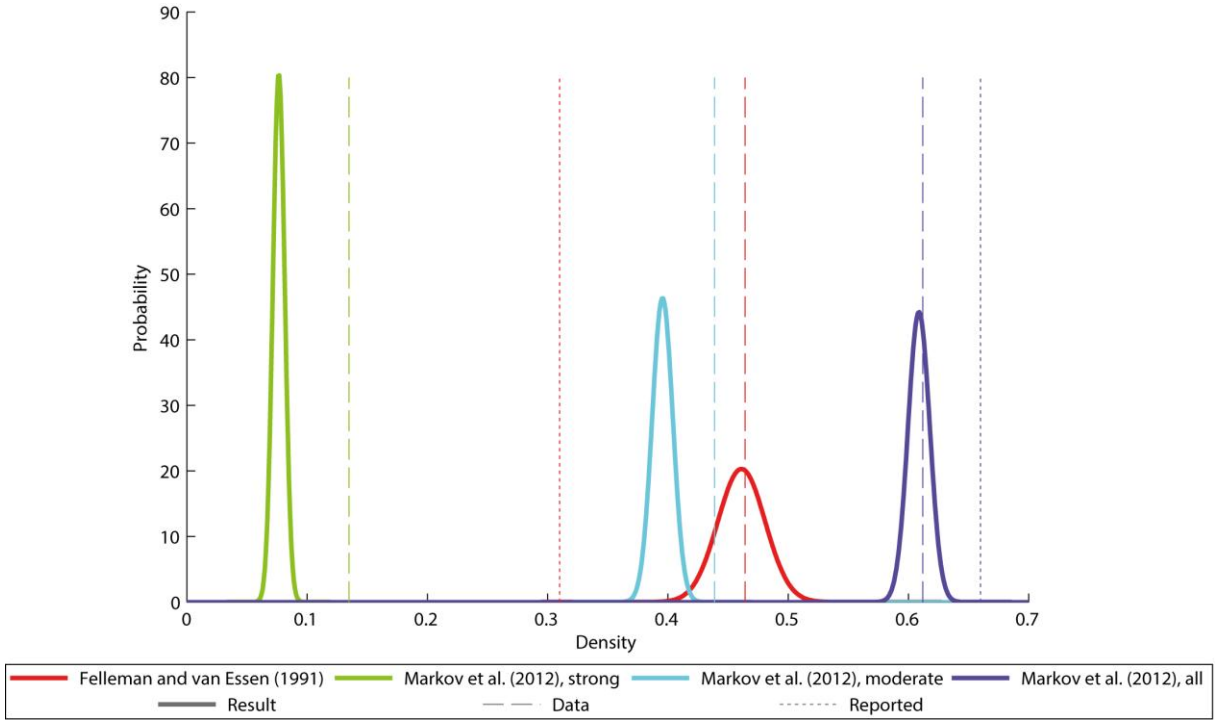

Figure 4: The distribution over densities for the different datasets with the reported density in the article (dotted lines) and the density of the data before using the algorithm (dashed lines).

3.2 Clustering

Figure 5 shows the marginal probabilities found by the algorithm on which the data is clustered. Using the AIC values the number of clusters for the different datasets is chosen. The number of clusters used are 7, 4,12, and 11 for Felleman & Van Essen and Markov et al. strong, moderate and all respectively. In Appendix C the AIC-curve is shown for the various datasets and various numbers of clusters (K). The black lines in Figure 5 show the boundaries between various clusters.

Figure 6 en Figure 7 give visualisations of the found clusters. First in Figure 6 it is shown how the clusters fit in the latent space of the MAP estimate. Figure 7 shows the found clusters superimposed on a drawing of the macaque brain. Areas in the same cluster have the same colour. This image can show whether areas in the same cluster are close to each other in the brain.

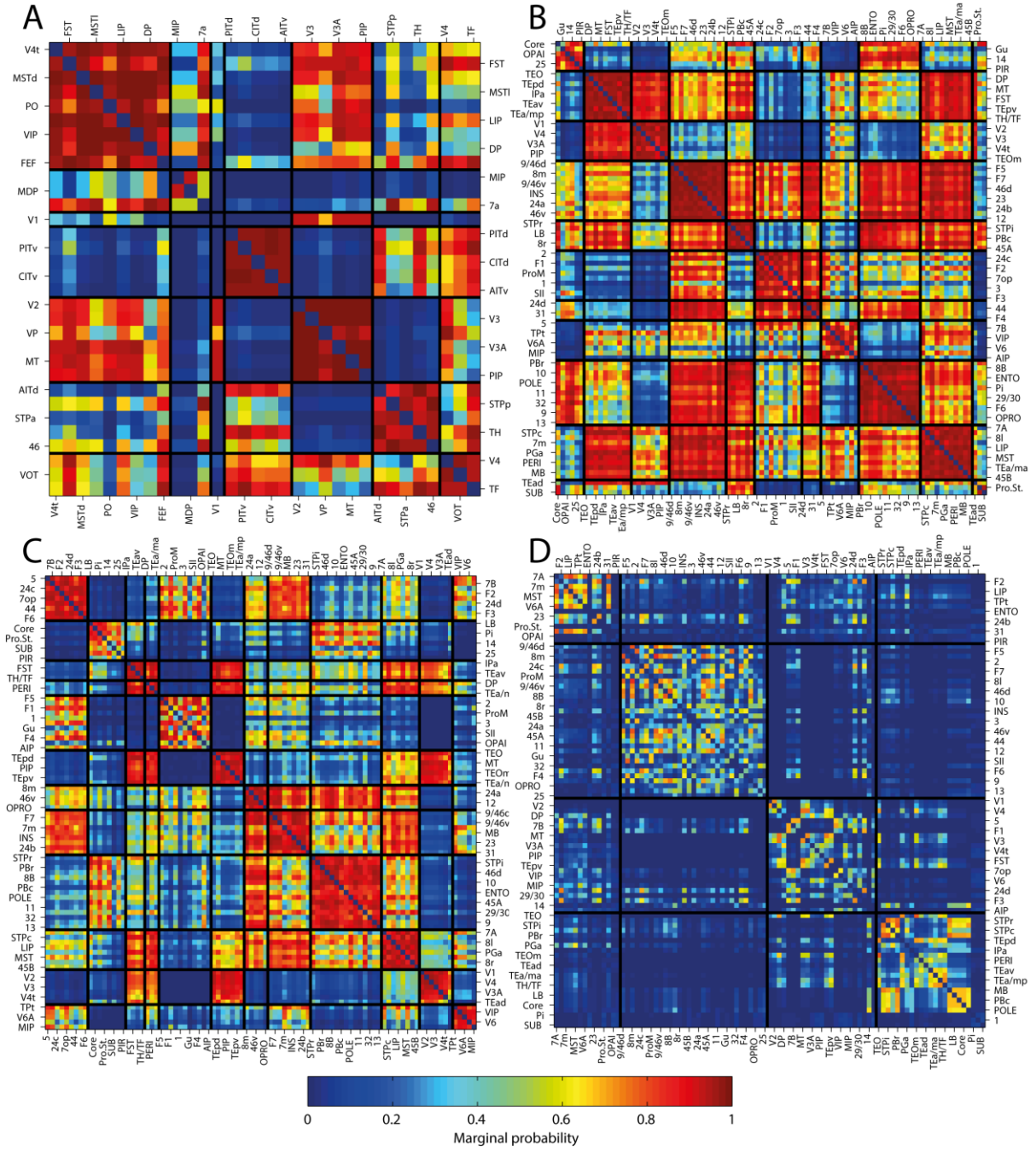

Figure 5: The marginal probabilities of connections found by the latent space algorithm for data of (A) Felleman & Van Essen (1991) and Markov et al. (2012) with (B) all connections, (C) moderate connections and (D) strong connections. The matrices are sorted such that areas in the same cluster are displayed directly next to each other. The black lines indicate the boundaries between two clusters.

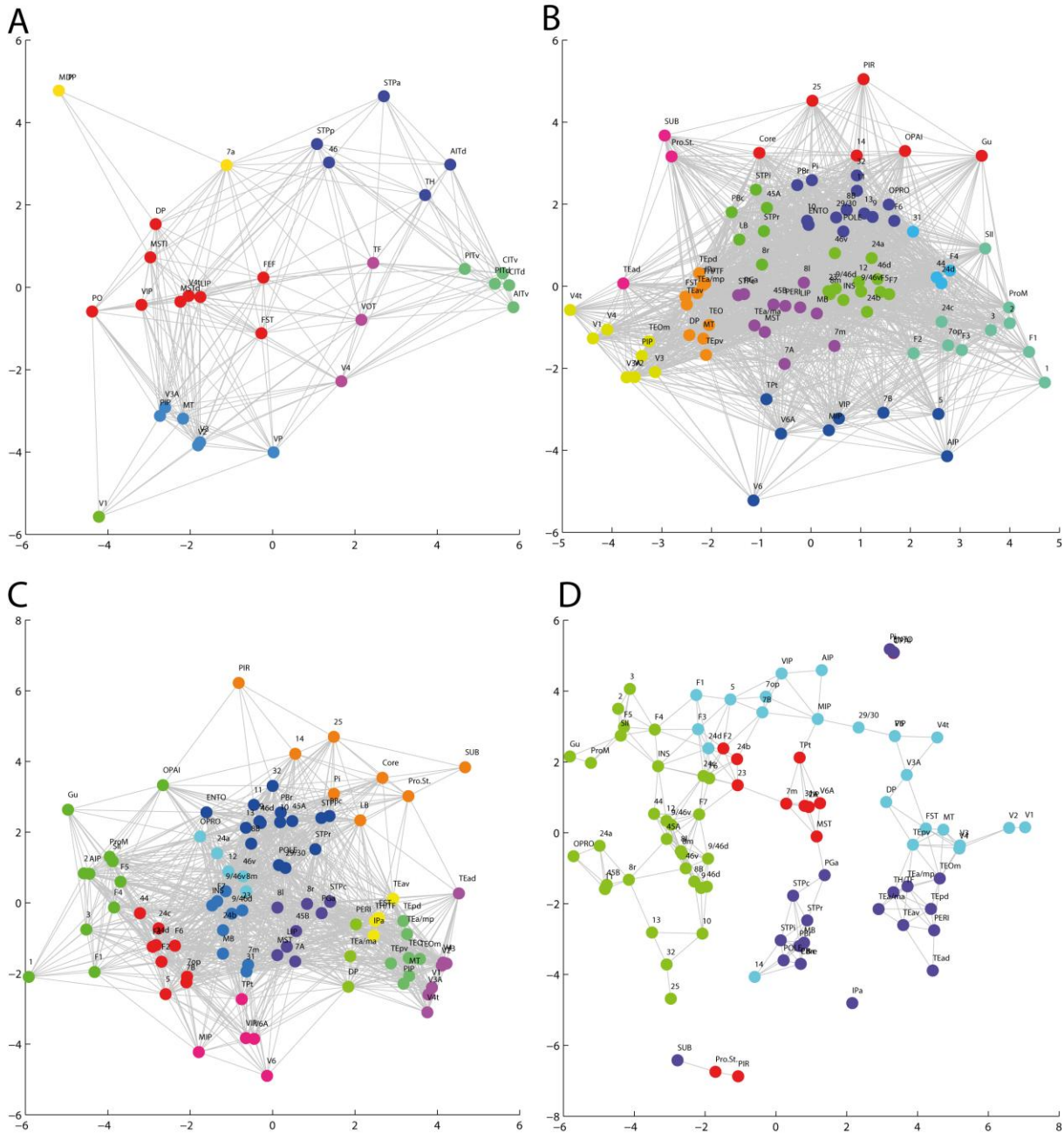

Figure 6: Plots of the 2D latent spaces of the MAP estimates found by the algorithm for data of (A) Felleman & Van Essen (1991) and Markov et al. (2012) with (B) all connections, (C) moderate connections and (D) strong connections. The different colours indicate which areas belong to the same cluster.

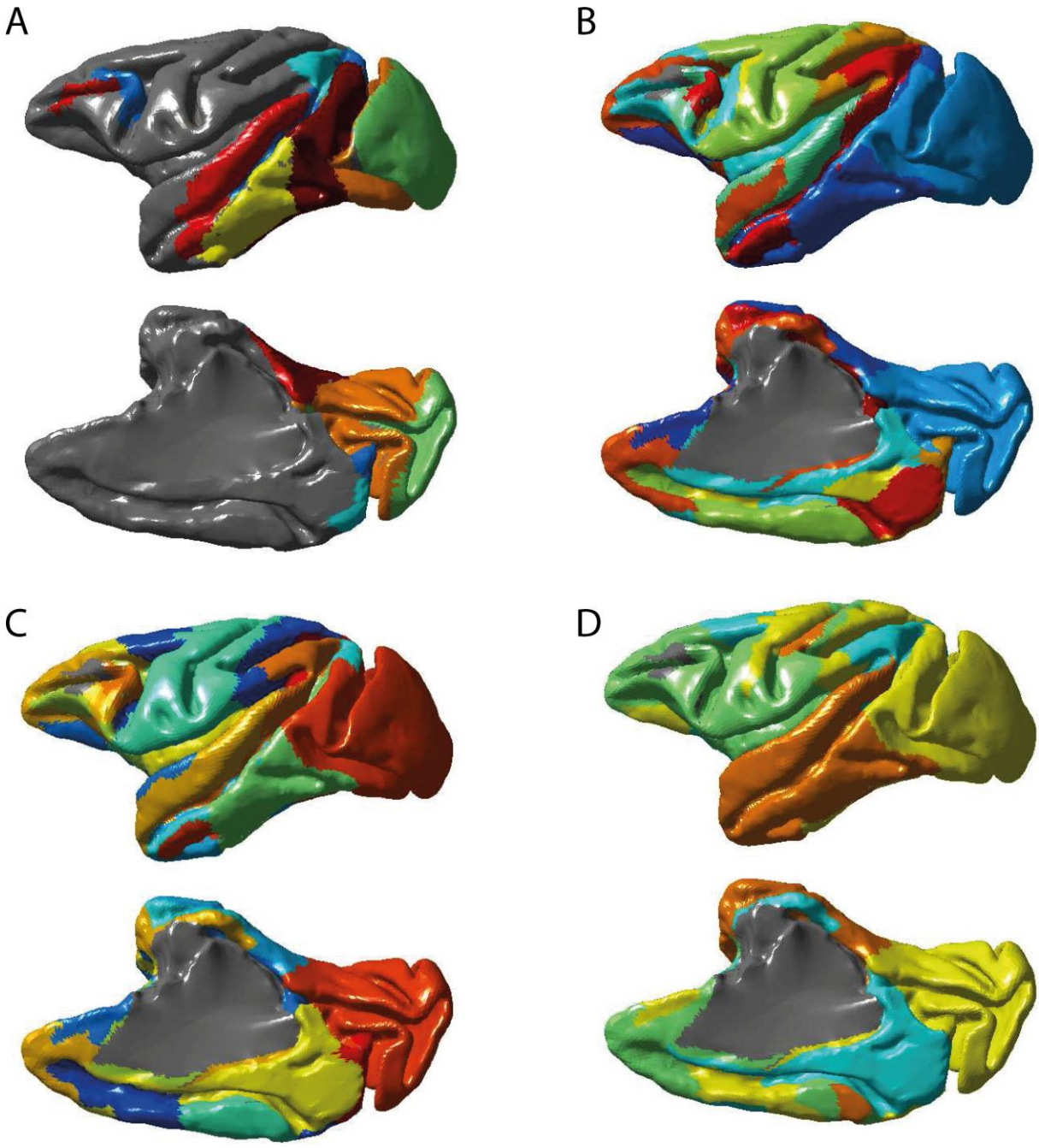

Figure 7: The brain of the macaque with the different brain areas coloured according to their cluster assignment for (A) Felleman & Van Essen (1991) and Markov et al. (2012) with (B) all connections, (C) moderate connections and (D) strong connections. Areas in the same cluster have the same colour. The gray areas are areas not present in the data.

4 Discussion

Looking at the estimated densities we can conclude that the difference in density between different experiments is not as high as reported in the various studies. The thresholding of the tract tracing results is very important. Since the latent space results of the strong connections of Markov et al. (2012) lead to a connectome that is not a connected graph (hence there is not a path from every node in the graph to every other node) we can assume that this threshold was too high. Therefore we can conclude that the density of the macaque's brain probably lies between the 40% and 60%. This is still

a large margin (though smaller than the one we started with). Without proper knowledge of the way different data sources are thresholded, it is hard to obtain a better estimate.

As can be seen in the raw data of Markov et al. (2012) (supplementary material) there is a relatively large group with very sparse connections (almost 22% of the found connections have 10 neurons or less in the source area that are connected to the target area). Using repeated experiments in the areas 'V1', 'V2', 'V4' and '10' Markov et al. (2012) also show that repeating the same experiment does not always yield the same results. For example, five injections in target area 'V1' only succeeded to show a connection with '7op' in one instance. In the resulting matrix, this connection was still labelled as existing. The question remains whether this is a valid conclusion, or whether the results found should be thresholded to get the true connectome. Looking at Figure 4 it can be seen that thresholding the Markov data to only contain the moderate and strong connections brings the distribution closer to the distribution found for the Felleman & van Essen data. Therefore, there will probably also be a threshold for which the found distribution of the Markov data is almost the same as the found distribution for the Felleman & van Essen data. Since the Felleman & van Essen data are the aggregated findings of a set of experiments it is hard to recover whether and, if so, how the data is thresholded.

A second main goal of this experiment was to cluster the areas according to their connectivity. By sorting the marginal probabilities on the found clusters (Figure 5) it is clear that there are more connections in clusters than between clusters. This is a property of small-world networks (Milgram, 1967). Another property of small-world networks is a high fraction of short-range connections and only a few long range connections. This combined with the high within cluster connectivity should result in clusters containing brain areas that are close to each other in the brain. Figure 7 shows that this is also the case. Stephan et al. (2000) claim that a brain network is indeed a small-world network. Assuming this claim is true, the clusters provide some visual evidence that the found marginal probabilities found by the algorithm are plausible. With the latent space model it is also possible to find clusters while sampling. By assuming cluster structures the results of the algorithm can improve. Our preliminary experiments have shown that, in this dataset, assuming cluster structures does not really change performance (see Appendix A). Therefore, it is chosen to find the clusters after using the latent space algorithm.

As can be seen in Figure 3 the algorithm does not always yield the results that are expected when predicting known data. This indicates that the algorithm does not make perfect predictions with respect to the connections between areas. Part of the errors, however, might also be explained by errors in the data. Tract tracing is a sensitive technique for finding tracts in the brain, but it is not perfect. By making general assumptions about structural brain organization (as done in the latent space model) these errors could be reduced.

Two of the assumptions that are made by the latent space model are high levels of transitivity and clustering in the network. These assumptions are both characteristics of a small-world network. Therefore, we could argue that, when the assumption that connectomes are small-world networks is correct, errors in data can be reduced by using latent space models. This does not imply however that the latent space model does not make errors itself.

Despite the fact that the origin of the errors is not certain, there are some extensions of the latent space model, that might make it a better fit with the data. Krivitsky et al. (2009) proposed an extension of the model so it would also incorporate homophily on observed attributes and degree heterogeneity. They propose to define a probability of a link $y_{i,j}$ as follows:

$$\text{logit}\left(P\left(y_{i,j}\middle|\boldsymbol{Z},\boldsymbol{X},\boldsymbol{\beta},\boldsymbol{\delta},\boldsymbol{\varepsilon}\right)\right) = \left(\sum_{k=1}^{p}\beta_k x_{k,i,j}\right) - \left\|\boldsymbol{z_i} - \boldsymbol{z_j}\right\| + \delta_i + \varepsilon_j \ . \tag{7}$$

Here $\mathbf{X}$ are p different $N \times N$ matrices that represent some covariate information of two individuals. Every individual has a probability to send a link ($\boldsymbol{\delta}$) and a probability to receive a link ($\boldsymbol{\varepsilon}$). These probabilities are calculated using the data. The regression parameters $\boldsymbol{\beta}$ have to be estimated alongside the locations $\boldsymbol{Z}$. Although it is not instantly clear how the homophily on observed attributes will be used in brain networks, degree heterogeneity could be a good extension. Hubs (nodes in a network with a high number of connections) are a property often associated with small-world networks. Therefore, even though the preliminary experiments indicated that adding the random effect δ and ε

did not improve accuracy of the model, it might still be beneficial to further investigate the effect random effects have on the found results.

Next to the obvious benefit of using imputed connectomes over incomplete connectomes, this probabilistic method can help to make better design choices for upcoming experiments in the form of sequential optimal design. The goal of optimal design is to find the stimulus that is expected to reduce the uncertainty about the posterior the most. In sequential optimal design a new stimulus is chosen, based on this expected uncertainty reduction, after every single experiment. Lewi, Butera, & Paninski (2009) define sequential optimal design for neurophysiology experiments. As neurophysiology experiments, the tract tracing experiments used here are very time-consuming and costly and therefore very suitable for using optimal design. To use optimal design a set of variables has to be defined in terms of the probabilistic model (e.g. the posterior and the expected response). Although most of these variables are quite easily defined in terms of the latent-space model, it remains to be seen how the expected reduction of the uncertainty can be determined due to the fact that there is no independence between the different connections.

To sum up, density estimation highly relies on thresholding of the data. Without proper knowledge about the thresholds in various data sources it is impossible to compare results about density. Still, the performed experiments narrowed down the probable margins in which the density of the connections in the macaque's brain lies to somewhere between 40% and 60%. The found clustering of the data shows that, with the found connection probabilities, the clusters support the theory that connectomes are small-world networks. Altogether, it seems that the latent space model is well suited for the use on brain networks.

Part 2 Predicting connectomes using noisy data

The aim of part two is to find a true connectome using two conflicting measurements of the connectome. To do this, a modified version of the latent space model will be used.

5 Methods

5.1 Data

Zingg et al. (2014) used both anterograde and retrograde tracing to find the connections between areas in the mouse neocortex. By manual inspection of the labelling of the brain after a certain injection, two separate connection matrices are formed. The first one with data from the anterograde labelling (matrix $\boldsymbol{A}$), the second one with data from the retrograde labelling (matrix $\boldsymbol{R}$). All connections are manually placed into one of four classes: absent, sparse, moderate, or dense. The two matrices can be found in Figure 8.

There is a discrepancy of approximately 21% between the two found matrices. The goal is to find a 'true' connectome overcoming these discrepancies.

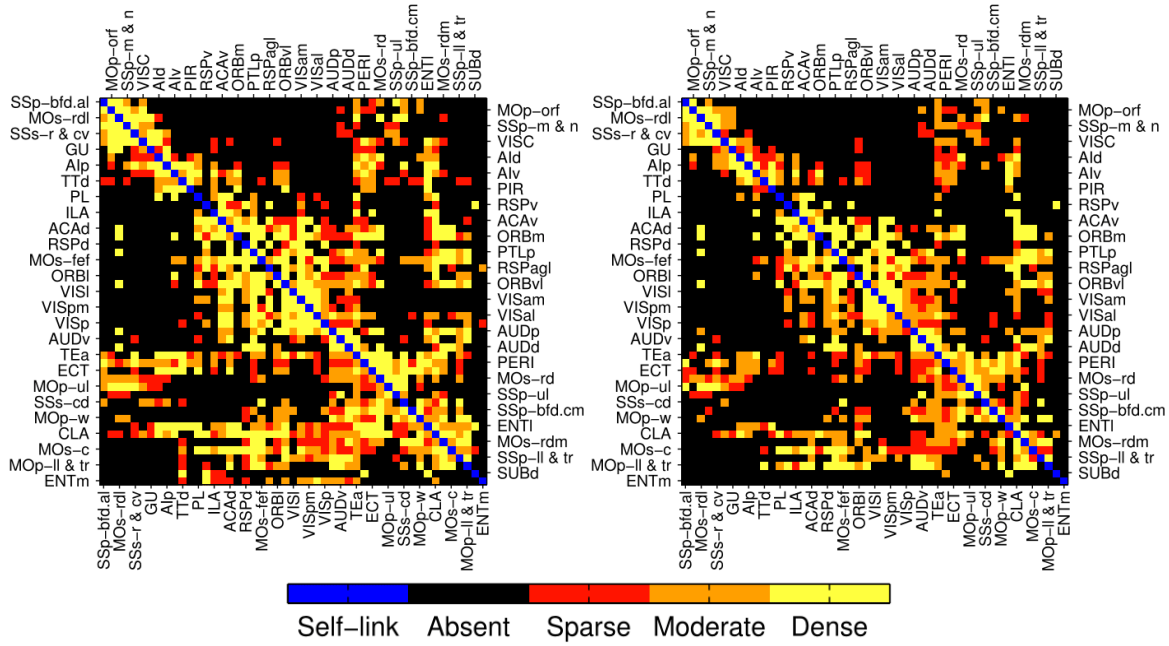

Figure 8: Data by Zingg et al. (2014). On the left side the data resulting from the experiments with anterograde tracers ($\boldsymbol{A}$), on the right side data resulting from experiments with retrograde tracers ($\boldsymbol{R}$). The regions are clustered to get high within cluster connectivity as is done in the paper.

5.2 Latent space adaptation

Since the data at hand is ordinal data a model needs to be defined that can fit this data. The first step is to transform the data. All connections fit in one of the four classes absent, sparse, moderate, or dense. To make the data more accessible we associate numeric values with these classes: absent will become 0, sparse will be 1, moderate 2, and dense 3. We say a connection i, j belongs to class k in dataset $\boldsymbol{C}$ if $c_{i,j} = k$ with $k \in \{0,1,2,3\}$.

Then we define a mixture model of K Cauchy distributions, one for each class in the data (in this case four). Cauchy distributions are heavy tailed and the probability for a certain class will therefore not easily go to zero. Since we expect the data to not be easily separable this is favourable over the normal distribution in which the probabilities go the zero fast. Each Cauchy distribution has two parameters. The first parameter is v_k, the location of the distribution. The second one is γ_k, the scale of the distribution. These parameters are unknown and have to be sampled. The location parameter v_k is sampled out of a normal distribution with parameters μ_k (mean) and σ_k (variance). By sampling from a normal distribution we can, next to the most probable locations of the Cauchy distributions, also see the uncertainty of these locations. Since we do not want to constrain the location v_k, μ_k has a completely flat prior between $-\infty$ and ∞. The variance of the normal distribution σ_k is sampled from an inverse gamma distribution with parameters $\text{shape} = 1$ and $\text{scale} = 3$. The inverse gamma distribution forces the variance to be positive. Furthermore, it favours low variances. By increasing the scale parameter to three, however, also higher variances are possible, which makes the prior more flexible. Both μ_k and σ_k are separate for all Cauchy distributions. Next to the normal distribution prior, the location parameters $\boldsymbol{v}$ are forced to be ordinal. Hence, $v_k < v_{k+1}$ for all $0 \leq k < K - 1$. The scale parameter γ_k is sampled out of a gamma distribution with $\text{shape} = 1$ and $\text{scale} = 1$, which forces the parameters to be positive, and favours low values for γ_k.

Combining this gives a set of K ordered one-dimensional Cauchy distributions, one for each of the classes in the data. Next we have to define a measure that determines the place of a data point in this one-dimensional space. Therefore, we place all the brain areas in a D-dimensional latent space. The distance between two points i and j in this latent space will then determine the place of the connection i, j in the one-dimensional Cauchy distribution space. To then determine the probability that connection i, j belongs to a certain class k you can calculate:

$$f_{i,j}^{(k)} \equiv P\left(c_{i,j} = k \mid \theta\right) = \frac{1}{U} \cdot \text{Cauchy}\left(\text{dist}_{i,j} \middle| v_k, \gamma_k\right). \tag{8}$$

where θ is the set of parameters and U is the normalization constant so that $\sum_k P\left(c_{i,j} = k \mid \theta\right) = 1$.

The distance between two points in latent space can be calculated in two ways. Either the Euclidean distance or the dot product distance could be used. When using the Euclidean distance $\text{dist}_{i,j} = -\left\|\boldsymbol{z_i} - \boldsymbol{z_j}\right\|$. Note that we take the negative of the Euclidean distance since we want higher distances to be less connected. By taking the negative the distance value gets lower when the distance is higher. Because the locations of the Cauchy distributions are ordered, so that the location of the distribution associated with the unconnected connections is the lowest, we want areas with a high distance to have a low distance value so that they fall in the unconnected class. A second distance measure that is used is the dot product distance: $\text{dist}_{i,j} = \boldsymbol{z_i} \boldsymbol{z_j}^T$. Since the dot product is a measure of similarity, this is already lower for points that are further away from each other.

Last thing to do is to determine the location of the areas in latent space. The position of $\boldsymbol{z_i}$ in the d-th dimension is sampled from a normal distribution with zero mean and variance ρ_d. By doing this we make sure the positions of $\mathbf{Z}$ stay centred around zero, avoiding that the algorithm will shift the whole latent space. The variance parameter ρ_d is also sampled according to an inverse gamma distribution with $\text{shape} = 1$ and $\text{scale} = 1$. This again forces the variance to be positive and favours small variances, making even surer that the locations of z will stay close to the origin.

Combining all the previously mentioned we have a complete generative model that will be the basis of all models we use in the experiments. In the next section some extensions of this model are discussed.

5.2.1 Extensions

To make the new latent space algorithm provide a good fit to the data, various additions are proposed and tested. The first one is the addition of mixing coefficients for the Cauchy mixture; $\boldsymbol{\pi}$. These mixing coefficients change the values of $f_{i,j}$ so that $f'^{(k)}_{i,j} \equiv P\left(c_{i,j} = k \mid \theta\right) = \frac{1}{U'} \cdot \pi_k \cdot \text{Cauchy}\left(\text{dist}_{i,j} \middle| v_k, \gamma_k\right)$. The values of $\boldsymbol{\pi}$ are sampled according to a Dirichlet distribution with concentration parameters $\boldsymbol{\alpha}$. Where α_k is the number of times class k is present in the data (this can either be $\boldsymbol{A}$, $\boldsymbol{R}$, or both), by doing this we implement the assumption that the various classes are represented in the data approximately as much as they will be in the true

connectome. U' is the changed normalization constant so that $\sum_k f'^{(k)}_{i,j} = 1$. Note that by setting α to the densities of the empirical data, the model is not completely generative anymore. Still, it is decided to do this to give the model some extra information about the expected densities of the true connectome. It is easy to change the model back to a purely generative model again by setting $\boldsymbol{\alpha}$ to fixed values.

The second addition is the use of random effects. Random effects are values used to favour connections with areas that have a lot of connections over connections with areas that are very sparsely connected. There are two ways to implement random effects. Either with general random effects (only $\boldsymbol{\delta}$) or with random sender/receiver effects (both $\boldsymbol{\delta}$ and $\boldsymbol{\varepsilon}$). To implement the random effects the distance measure gets slightly adjusted. To implement general random effects the new distance measure becomes $\text{dist}'_{i,j} = \text{dist}_{i,j} + \delta_i + \delta_j$. When using random sender/receiver effects the new distance measure becomes $\text{dist}'_{i,j} = \text{dist}_{i,j} + \delta_i + \varepsilon_j$. Both $\boldsymbol{\delta}$ and $\boldsymbol{\varepsilon}$ are sampled according to a normal distribution with zero mean and ρ_δ or ρ_ε respectively as variance. Note, there is only one ρ_δ and one ρ_ε out of which all $\boldsymbol{\delta}$ and $\boldsymbol{\varepsilon}$ are sampled. Both ρ_δ and ρ_ε are sampled according to an inverse gamma distribution with $\text{shape} = 1$ and $\text{scale} = 1$. This again favours low variances for the normal distribution, making sure the random effects stay small and will not become too important in the model.

The last addition is only used with the dot product distance measure. To encode the possibility of asymmetry the matrix $\boldsymbol{P}$ is introduced. The $\boldsymbol{P}$-matrix is $D \times D$ and sampled out of a completely flat distribution. To incorporate P in the sampler the dot product distance measure is changed to: $\text{dist}'_{i,j} = \boldsymbol{z_i P z_j}^T$. By using this $\boldsymbol{P}$ matrix the model can become directional, giving different roles to different dimensions of the latent space.

Combining all distributions and formulas given in the last two sections we obtain the most general form of the model used in the experiments:

$$
\begin{aligned}
\rho_d &\sim \text{inverse gamma}(1,1) \\
\rho_\delta &\sim \text{inverse gamma}(1,1) \\
\rho_\varepsilon &\sim \text{inverse gamma}(1,1) \\
z_{i,d} &\sim \text{normal}(0, \rho_d) && \text{(Locations in latent space)} \\
\delta_i &\sim \text{normal}(0, \rho_\delta) && \text{(Random sender effects)} \\
\varepsilon_i &\sim \text{normal}(0, \rho_\varepsilon) && \text{(Random receiver effects)} \\
n^{(k)}_{i,j} &= \begin{cases} 2, & a_{i,j} = k \wedge r_{i,j} = k \\ 1, & a_{i,j} = k \veebar r_{i,j} = k \\ 0, & otherwise \end{cases} \\
\alpha_k &= \textstyle\sum_{i \neq j} n^{(k)}_{i,j} \\
\boldsymbol{\pi} &\sim \text{Dirichlet}(\boldsymbol{\alpha}) && \text{(Mixing coefficients)} \\
\mu_k &\sim \text{flat}(-\infty, \infty) \\
\sigma_k &\sim \text{inverse gamma}(1,3) \\
\gamma_k &\sim \text{gamma}(1,1) && \text{(Scale of the Cauchy distribution)} \\
v_k &\sim \text{normal}(\mu_k, \sigma_k) && \text{(Location of the Cauchy distribution)} \\
\boldsymbol{P} &\sim \text{flat}(-\infty, \infty) \\
\text{dist}_{i,j} &= -\left\|\mathbf{z_i} - \mathbf{z_j}\right\| + \delta_i + \varepsilon_j \text{ OR} && \text{(Distance measures)} \\
\text{dist}_{i,j} &= \boldsymbol{z_i P z_j}^{\mathrm{T}} + \delta_i + \varepsilon_j \\
f^{(k)}_{i,j} &= \frac{1}{U} \cdot \pi_k \cdot \text{Cauchy}\left(\text{dist}_{i,j} \middle| v_k, \gamma_k\right) \\
a_{i,j} &\sim \text{categorical}\left(\boldsymbol{f_{i,j}}\right) && \text{(Dataset } A\text{)} \\
r_{i,j} &\sim \text{categorical}\left(\boldsymbol{f_{i,j}}\right) && \text{(Dataset } R\text{)}
\end{aligned}
\tag{9}
$$

Note $n^{(k)}_{i,j}$ will never become 2 if only one dataset is used to train the model. Figure 9 shows a visual representation of the model.

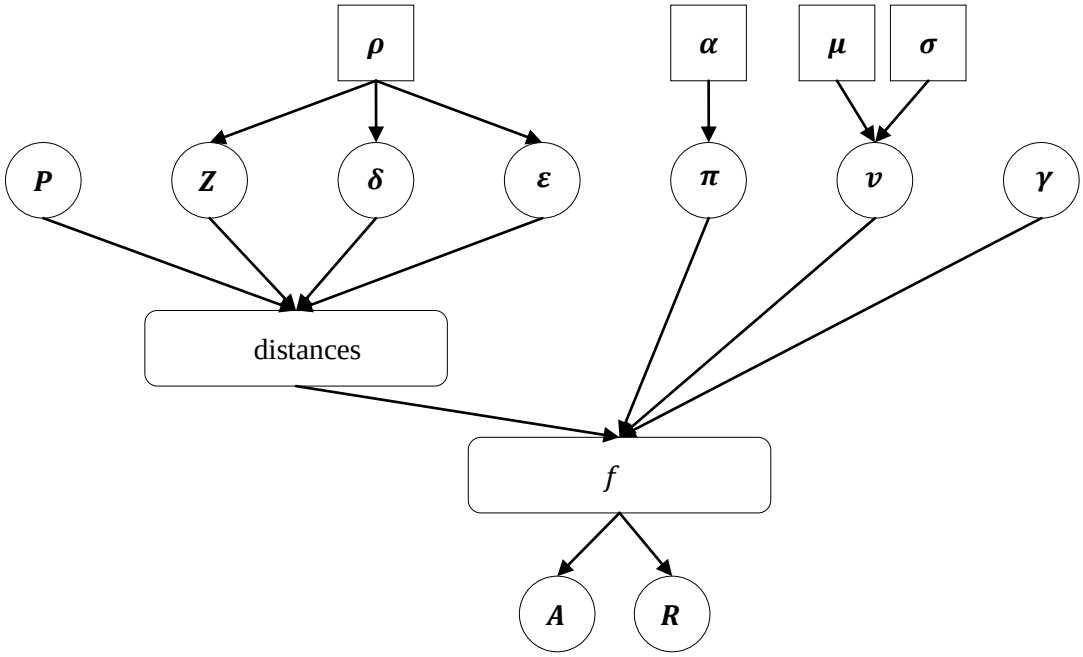

Figure 9: A visual representation of the complete model. The squares indicate hyperpriors, the circles indicate the variables of interest and the rounded rectangles indicate values that are calculated without using sampling.

5.2.2 Sampling

The sampling of the model is done by MatlabStan (Lau, 2014). Stan is a probabilistic programming language implementing full Bayesian statistical inference using No-U-Turn Sampling (NUTS) (Hoffman & Gelman, 2014), which is an extension of Hamiltonian Monte Carlo (HMC) sampling (Duane, Kennedy, Pendleton, & Roweth, 1987). In contrast to HMC, NUTS has no user-specified parameters, which makes it easier to use in automatic inference engines such as Stan.

5.3 Experiments

In the first step of the experiment, the aim is to find the variant of the algorithm that can best fit the data. Therefore the NUTS-sampler will run a set of algorithms to determine which settings result in the highest log-likelihood. For the first set of experiments four chains of 500 samples (of which 250 samples burn-in) were fitted on twelve different combinations of settings for both datasets. All experiments were done in a latent space with four dimensions. The settings varied the distance measure, the use of $\boldsymbol{\pi}$ and random effects, and for the dot-product distance the use of $\boldsymbol{P}$. Since the expectation that $\boldsymbol{\pi}$ is a good extension was confirmed early in the experiments most experiments without using $\boldsymbol{\pi}$ are omitted.

For all different settings the mean negative predictive log-likelihood (MNPLL) is calculated. The negative predictive log-likelihood (NPLL) of a sample is defined as follows:

$$\text{NPLL} = -\sum_{i \neq j} \ln f_{i,j}^{(a_{i,j})}. \tag{10}$$

Note, that in this case $a_{i,j}$ is element i,j of data matrix $\boldsymbol{A}$. When analyzing matrix $\boldsymbol{R}$, a can be replaced by r. To get the MNPLL, the NPLL is averaged over all chains and samples of one setting.

After analyzing the results the most promising settings are selected by looking at the minimum MNPLL. These settings are then selected for dimension selection. To select the proper dimension the selected settings are fixed and then four chains of 500 samples are fitted again for dimensions varying from 3 to 6. After which the MNPLL is calculated again to select the dimension that can best fit the data.

After selection of the best dimension another two chains are fitted for the three settings with the lowest MNPLL. Instead of 500 samples, these chains are fitted with 2000 samples (of which 1000

samples burn-in) to be more certain of convergence. Furthermore, two chains of 2000 samples are fitted on the datasets where 10% of the data is set to unknown. The accuracy on these 10% connections is used to check for the predictive qualities of the different models. Afterwards the best settings are selected and the rest of the settings are not used anymore.

Now the model that can best fit the data is selected, it is used to combine the two datasets into one dataset. Two chains of 2000 samples (of which 1000 burn-in) are fitted with shared variables for both datasets. This is done twice. The first time with all data, while the second time 10% of the data will be deleted to check the predictive performance of the algorithm. After fitting, the MNPLL and the mean accuracy are calculated over the first set of fits and the mean accuracy for the deleted connections is calculated for the second set of fits. To make it comparable with earlier results, both measures are calculated for the two datasets separately.

The results of these fits are used to form expectations of the true underlying connectome. For all samples the most probable connectome according to that sample is generated by assigning $c_{i,j} = \text{argmax}_k f_{i,j}^{(k)}$. These connectomes are used to make a confusion matrix and an estimate of the distribution of the connections over the four classes.

The marginal probabilities of each connection are calculated using the two chains fitted with all the data. The marginal probability of connection i, j in G chains of T samples is calculated as follows:

$$P(c_{i,j} = k) = \frac{1}{GT} \cdot \sum_{g=1}^{G} \sum_{t=1}^{T} P(c_{i,j}^{(g,t)} = k \mid \theta^{(g,t)}). \tag{11}$$

where (g, t) denotes the t-th sample of the g-th chain and θ is the set of parameters. These marginal probabilities are then also used to make a connectome, where each connection is assigned to the most probable class according to the marginal probabilities.

The last experiment that is conducted aims to identify certain biases in the data. Since anterograde and retrograde tracers label different parts of the brain cells it is possible that the researchers who labelled the data are biased to label data in a certain way. Therefore, either $\boldsymbol{\pi}$, the random effects, or both are assigned to be possible biases. This means that although the latent space is still shared between the two datasets, each dataset is believed to have its own $\boldsymbol{\pi}$ and random effects values. For these three assumptions another set of two times two chains of 2000 samples (1000 samples for burn-in) are constructed, in for which the first two chains are trained on all the data, and the last two chains use a 10% hold-out. Then the MNPLL and accuracies are calculated over the results to check whether assuming biases increases the fit of the model to the data.

6 Results

6.1 Latent space adaptation

In Figure 10 the MNPLL of the last 250 samples of four 500 sample chains is shown for different settings. In the distance row E stand for Euclidean distance and D stand for dot product distance. The red and blue bars represent respectively the MNPLL for dataset $\boldsymbol{A}$ and for dataset $\boldsymbol{R}$. The yellow/green bars show the average MNPLL between those two datasets. When the bar is green the MNPLL is one of the three lowest and therefore this setting is used in further experiments. These settings are; Euclidean distance with $\boldsymbol{\pi}$, and random sender/receiver effects; dot product distance with $\boldsymbol{\pi}$, and general random effects; and dot product distance with $\boldsymbol{\pi}$, and random sender/receiver effects.

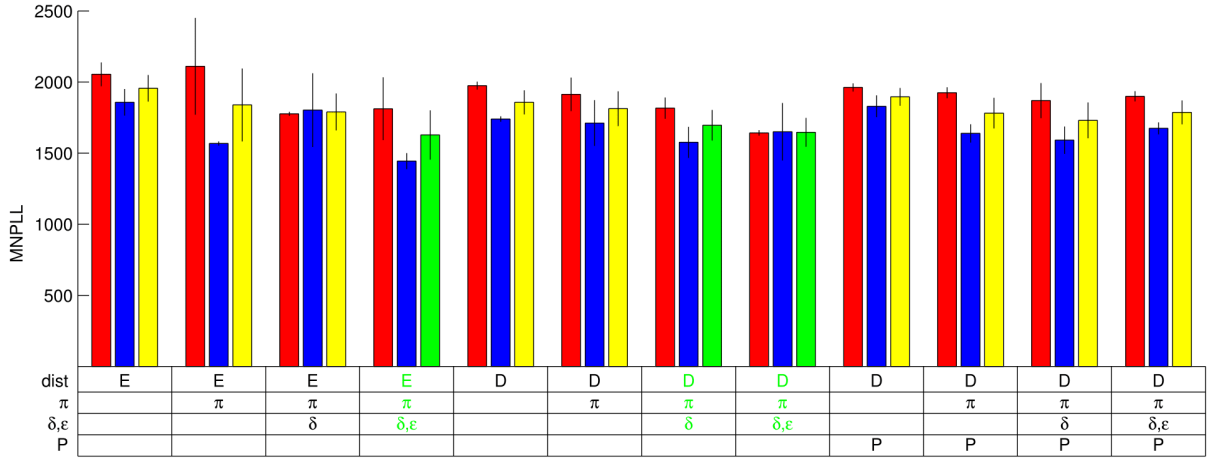

Figure 10: The MNPLL for the different settings. All MNPLL values are averaged over four chains of 250 samples. The red bar indicates the MNPLL for dataset *A*, the blue bar indicates the MNPLL for dataset *R*, and the yellow/green bar indicates the average of the MNPLL for both datasets. The vertical black lines show the mean plus/minus two times the standard deviation of the mean. The three green bars show the lowest MNPLL values. The settings associated with these bars (also in green) are used for further experimentation.

Figure 11 depicts the MNPLL for the three selected settings with dimensions varying from 3 to 6. Again the red and blue bars show the MNPLL for dataset ***A*** and dataset ***R*** respectively. The yellow and green bars show the average of these two datasets, where the green bars indicate the settings with the lowest MNPLL. These settings are; Euclidean distance with $\boldsymbol{\pi}$, and random sender/receiver effects in four dimensions; and dot product distance with $\boldsymbol{\pi}$, and random sender/receiver effects in five and in six dimensions.

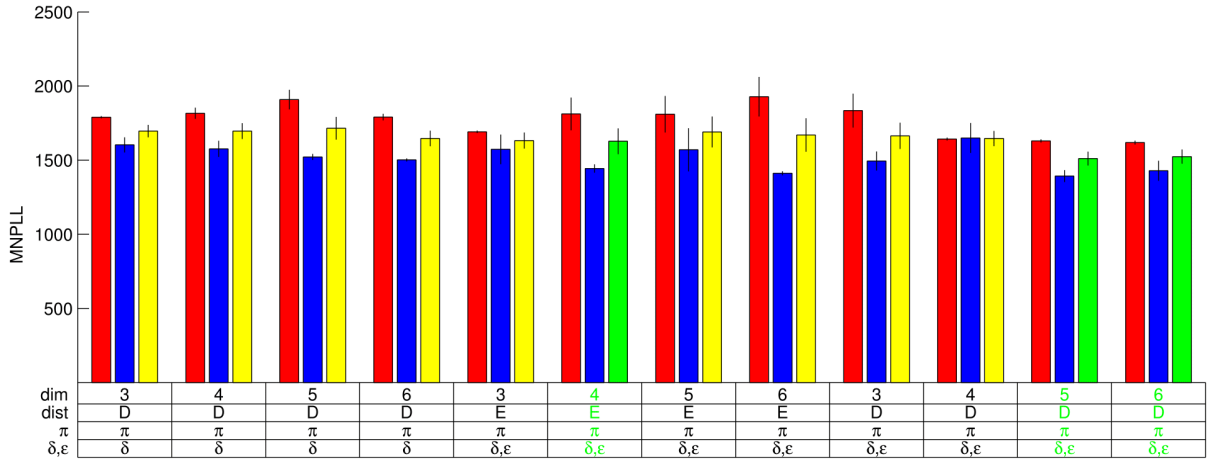

Figure 11: The MNPLL for the three top settings varied over dimensions three to six. All MNPLL values are averaged over four chains of 250 samples. The red bar indicates the MNPLL for dataset *A*, the blue bar indicates the MNPLL for dataset *R*, and the yellow/green bar indicates the average of the MNPLL for both datasets. The vertical black lines show the mean plus/minus two times the standard deviation of the mean. The three green bars show the lowest MNPLL values. The settings associated with these bars (also in green) are used for further experimentation.

Appendix D shows an example of the convergence of the algorithm after 500 chains. It can be seen that the algorithm has not yet completely converged. Therefore 2000 samples of the top three settings will be generated. An example of this convergence can also be found in the Appendix. Note that the algorithm is still not completely converged after 2000 samples in all cases. However, it is decided not to increase the number of samples a second time.

In Figure 12 the results are shown for the three chosen settings with two chains 2000 samples per setting. The left plot shows the MNPLL for the different settings. The lowest MNPLL is achieved by using six dimensions and the dot product distance. Note that the MNPLL went down in comparison to the MNPLL for 500 samples. The middle graph shows the accuracy of the prediction the latent space algorithm makes on the train set. The dashed lines in the background show the percentage of correct

prediction if everything is assigned to the majority class (in this case unconnected). Note that the predictions are better than chance. The right figure shows the accuracy of the latent space prediction on an unseen hold-out set. In all three figures the green bar shows the best performing setting. The optimal settings for predicting unseen data are different than the optimal settings to fit the known data. Since the goal of the thesis is to merge the two datasets and not predict unseen data further experimentation will be done with the settings that are most optimal to describe known data. Hence, six dimensions, dot product distance, $\boldsymbol{\pi}$, and random sender/receiver effects.

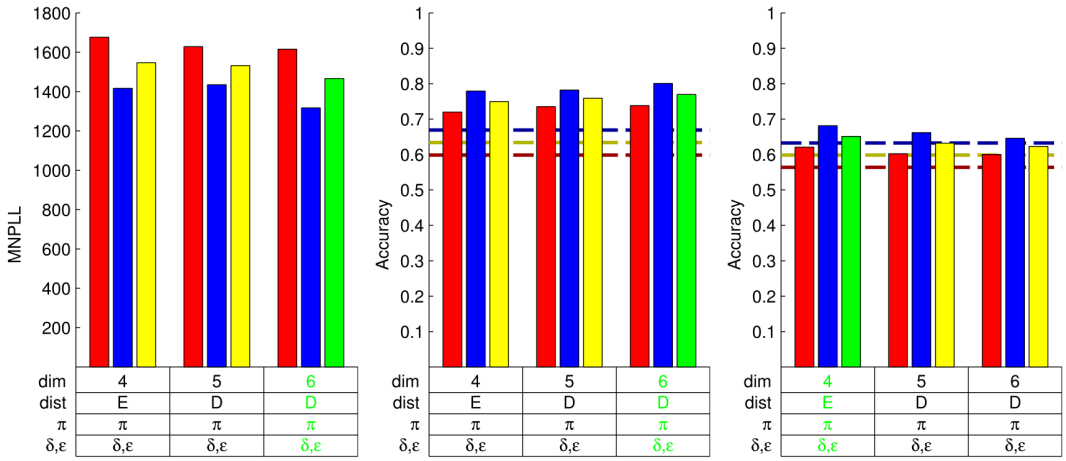

Figure 12: On the left side the plot of the MNPLL of the last 1000 samples of two 2000 sample chains. In the middle the accuracy of the same data. The dashed lines in the background indicate the percentage of the majority class in the data. On the right side the accuracy of the last 1000 samples of two 2000 sample chains on a hold-out set of the data. The red and blue bars denote the MNPLL/accuracy of dataset *A* and *R* respectively. The yellow/green bars show the average of the two. The best performing settings have a green colour.

Figure 13 shows the confusion matrices obtained when using the optimal settings. The values in the table represent $P(c_{i,j} = row \mid a_{i,j} = col)$ where row is the class number assigned to that row and col is the class number assigned to the column.

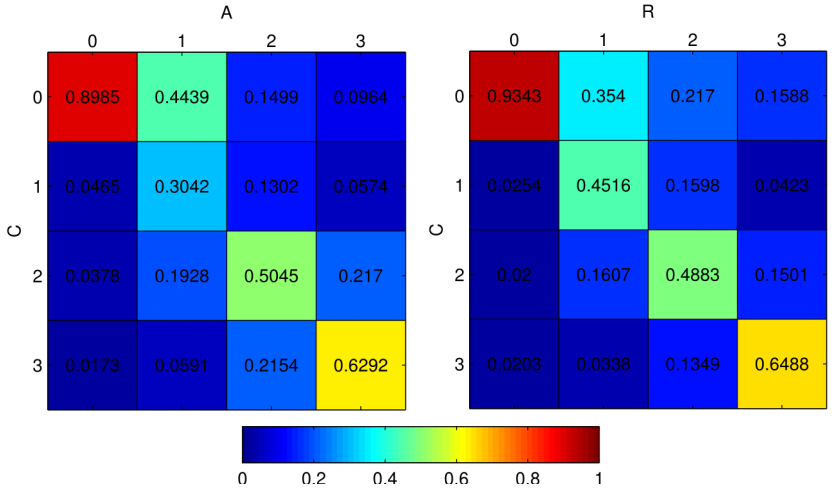

Figure 13: The confusion matrix for dataset *A* and *R* for the last 1000 samples of two chains of 2000 samples using $\boldsymbol{\pi}$, and random sender/receiver effects in six dimensions. The values represent $\boldsymbol{P(c_{i,j} = row \mid a_{i,j} = col)}$ or $\boldsymbol{P(c_{i,j} = row \mid r_{i,j} = col)}$.

6.2 Combining data

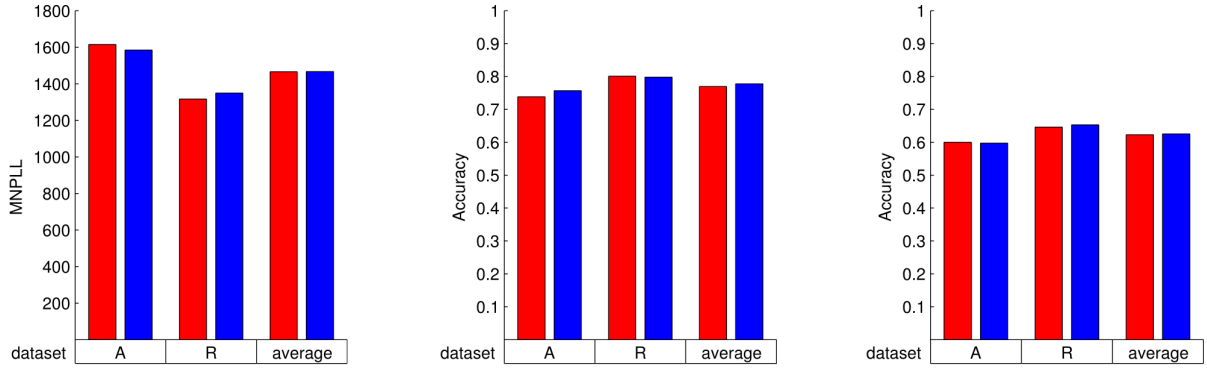

Figure 14: Comparison between the results when the model fits all variables for the two datasets separate (red) and when all variables in the model are shared between the two datasets (blue). On the left side MNPLL of the fit, in the middle the accuracy on the train data and on the right side the accuracy on unseen test data.

The settings found for the latent space adaptation will be used to combine the data. In Figure 14 a comparison is made between the results of the latent space algorithm where all variables are dataset dependent (red bars) and the latent space algorithm where all variables are shared between datasets (blue bars). The most left plot shows the change for the MNPLL, the middle plot shows the change for accuracy on the train data, and the right plot shows the accuracy on unseen test data.

For every sample in the two chains a connectivity matrix is constructed where the class of connection i, j is assigned to be the class k where $k = \text{argmax}_l \ f_{i,j}^{(l)}$. This leads to 2000 connectivity matrices. Figure 15 shows the confusion matrix of these 2000 samples. The values in the matrix show the fraction of times a connection in the constructed connectivity matrix has a value as given in de column, while that connection in $\boldsymbol{A}$ has the value as in the upper row, and in $\boldsymbol{R}$ as in the second row. Note that this means that all columns add to one.

	A	0				1				2				3			
	R	0	1	2	3	0	1	2	3	0	1	2	3	0	1	2	3
C	0	0.9576	0.7087	0.6577	0.707	0.7818	0.2849	0.2483	0.4952	0.5688	0.2404	0.1461	0.0779	0.6098	0.2234	0.1566	0.1441
	1	0.0177	0.2412	0.1712	0.0793	0.1533	0.562	0.4278	0.3026	0.1744	0.4003	0.1427	0.0586	0.0854	0.3059	0.0881	0.0368
	2	0.0091	0.0419	0.1365	0.079	0.038	0.1203	0.2863	0.1478	0.165	0.3069	0.608	0.2139	0.0685	0.2147	0.3507	0.0665
	3	0.0157	0.0082	0.0346	0.1347	0.0269	0.0328	0.0376	0.0544	0.0918	0.0524	0.1032	0.6496	0.2363	0.256	0.4046	0.7526

0 0.1 0.2 0.3 0.4 0.5 0.6 0.7 0.8 0.9 1

Figure 15: The confusion matrix for combining dataset $\boldsymbol{A}$ and $\boldsymbol{R}$ for the last 1000 samples of two chains of 2000 samples using $\boldsymbol{\pi}$, and random sender/receiver effects in six dimensions. The values represent $\boldsymbol{P(c_{i,j} = row \mid a_{i,j} = col_a, r_{i,j} = col_r)}$.

For all constructed connectivity matrices the fraction of each class is calculated. Afterwards there is a normal distribution fitted over these fraction of all 2000 connectomes. These normal distribution can be found in Figure 16. The dashed lines in this figure indicate the fraction of the class in dataset $\boldsymbol{A}$, the dotted lines the fraction of the class in dataset $\boldsymbol{R}$.

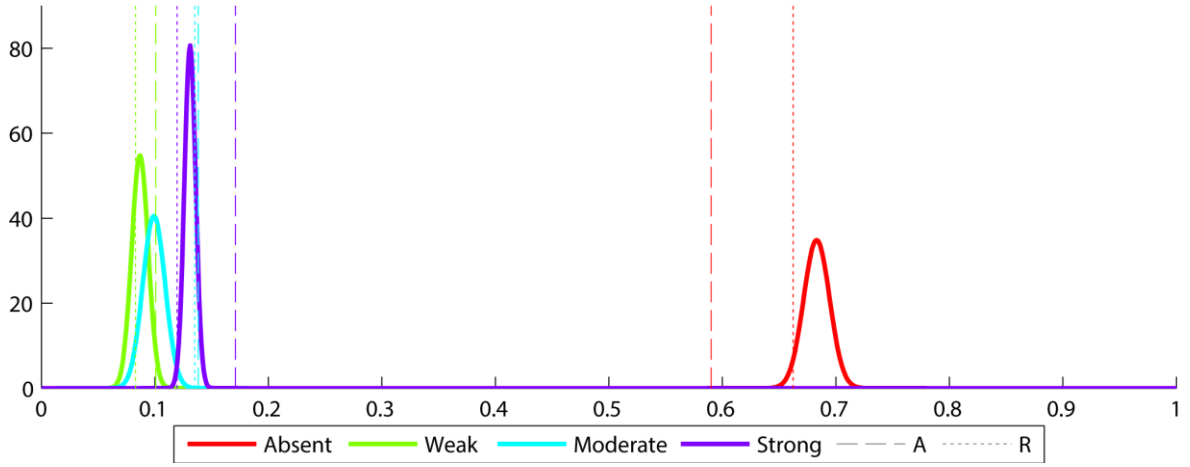

Figure 16: The normal distribution fit for the fraction of every class in the maximum likelihood networks of the 1000 samples in the two chains combining the two datasets. The dashed lines show the fraction of that class in the anterograde data. The dotted lines show the fraction of that class in the retrograde data.

Now the marginal probabilities of all connections are calculated. Afterwards, a connectome is constructed using these probabilities. The connectome is clustered by using the same algorithm as in part 1 (section 2.2.2). To cluster only the probabilities of class 0 are used (hence the probabilities of not connecting with other areas). Using the AIC measure, the number of clusters is chosen. The number of clusters used is four. The top left side of Figure 17 shows the connectome obtained sorted by cluster. The white lines indicate the boundaries between two clusters. The top right figure shows the probability of the class the connection is assigned to, hence, the certainty of the prediction in the connectome.

Below the results the connectomes as found in the data are shown, with on the left side dataset ***A*** and on the right side dataset ***R***. These connectomes are also sorted according to the clusters so that comparison with the result is easier.

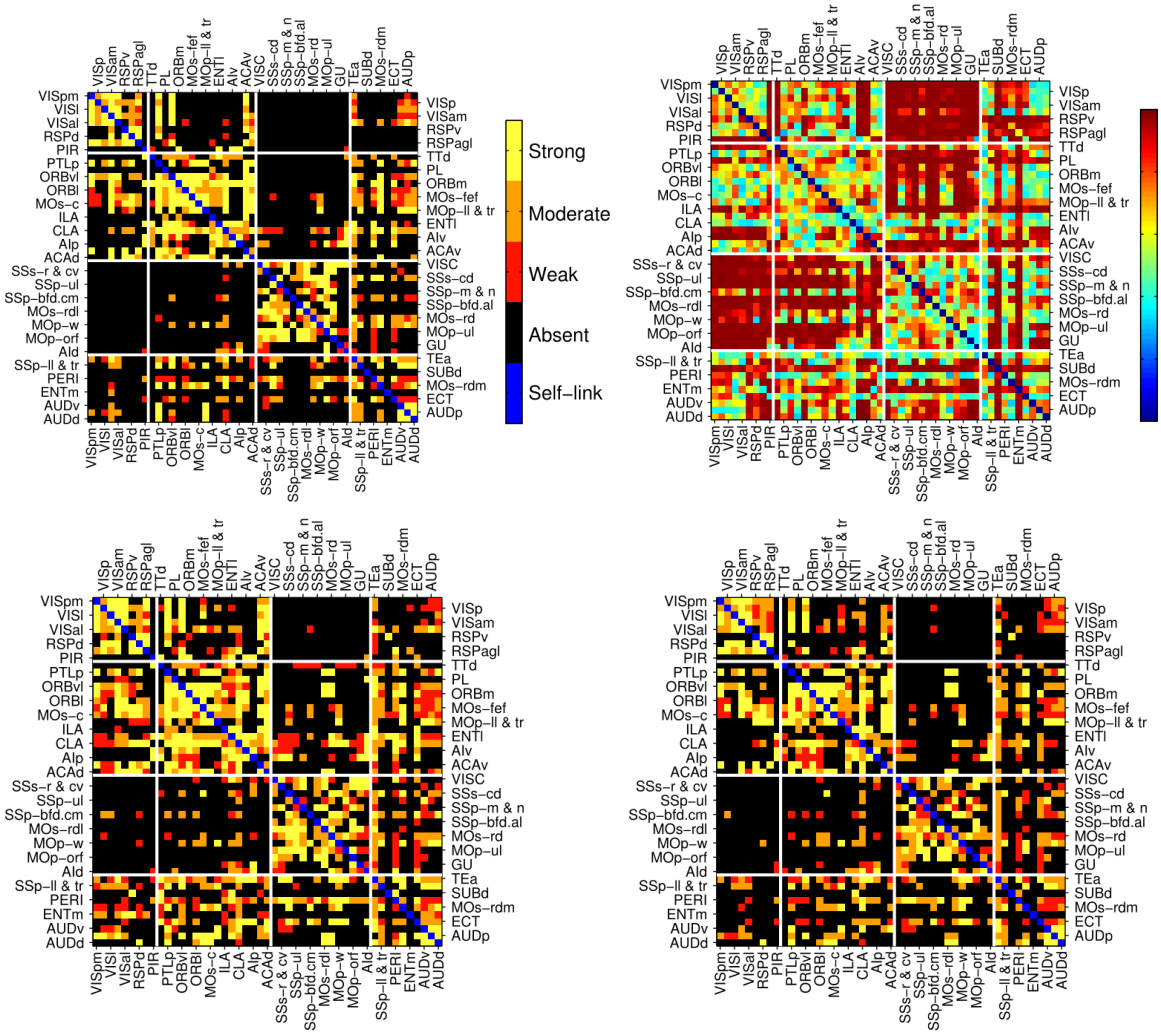

Figure 17: On top the results of combining the two datasets using the latent space algorithm. On the left side the classes of all the connections. These results are obtained by first calculating the marginal probabilities over the 1000 samples and then taking the class with the highest probability for every individual connection. On the right side the marginal probabilities of these classes. The areas are clustered using Gaussian mixture models (section 2.2.2) on the probabilities they have to not connect with the other areas (hence the probability of class zero). The white lines indicate the boundaries between the different clusters. Below the results you find the plots of the datasets (*A* on the left side and *R* on the right side) sorted the same way as the results to make comparison easier.

The best sample out of the two chains is selected. The selection criterion is the log probability given by Stan. In Figure 18 on the left side the Cauchy mixture of this sample is shown. The right side shows how this mixture translates to probabilities for the various classes as a function of the distance between two areas in the latent space.

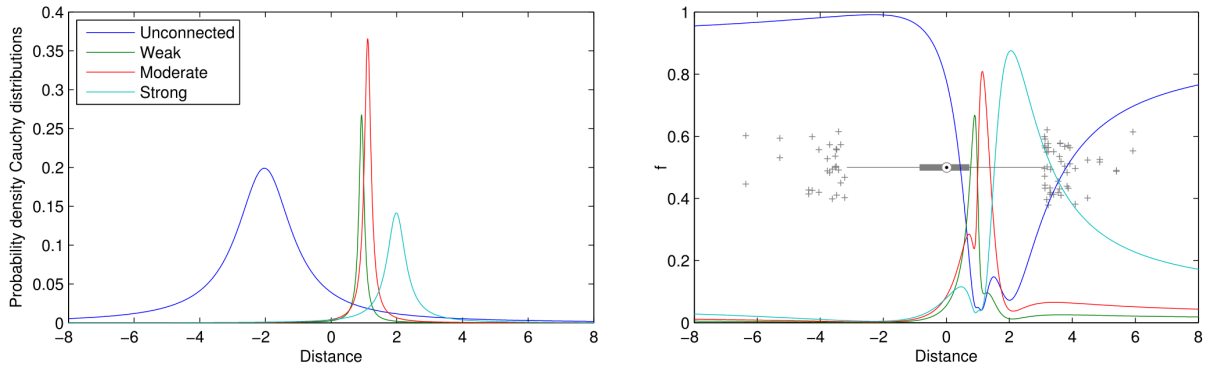

Figure 18: Left the Cauchy mixture as in the sample with the maximum log probability. Right the Cauchy mixture normalized to get f. On the x-axis the input of the Cauchy mixture, which is in this case the distance between two areas in the latent space. In the right plot a boxplot is added for the dot product distances found between areas in the latentspace. The dot indicates the median, the sides of the rectangle shows the 25th and 75th percentile. The crosses are outliers.

6.3 Bias identification

Figure 19 shows the results of the algorithm with certain parameters identified as biases. The bias parameter is then dependent on the dataset. The figure shows the MNPLL, the accuracy on train data and the accuracy on unseen test data for no bias, $\boldsymbol{\pi}$ as bias, the random effects as bias, or $\boldsymbol{\pi}$ and the random effects as biases. All runs are two chains of 2000 samples, of which the first 1000 are discarded as burn-in. The settings are the same for all four experiments; six dimensions, $\boldsymbol{\pi}$, random sender/receiver effects and dot product distance.

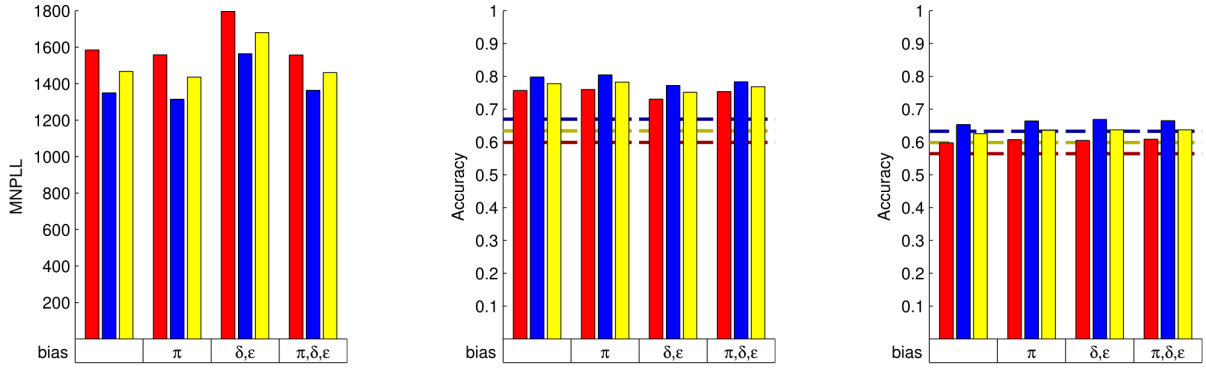

Figure 19: Left the MNPLL for different two chains of 1000 samples with various biases. In the middle the mean accuracy of these samples on train data, and right the accuracy of these samples on unseen test data. The red bar indicates the score on dataset *A*, the blue bar on dataset *R* and the yellow bar is the average between the two. The lines in the background shows the accuracy when all connections are assigned to the majority class.

Figure 20 shows how the $\boldsymbol{\pi}$-value changes when you use it as a bias value. The four figures each represent one of the four $\boldsymbol{\pi}$-values (for the four different classes). The three boxplots in every figure show from left to right the $\boldsymbol{\pi}$-values when they are shared between the two datasets, when they are separate for dataset $\boldsymbol{A}$, and when they are separate for dataset $\boldsymbol{R}$. All (non burn-in) samples out of the two chains are used to construct the figures.

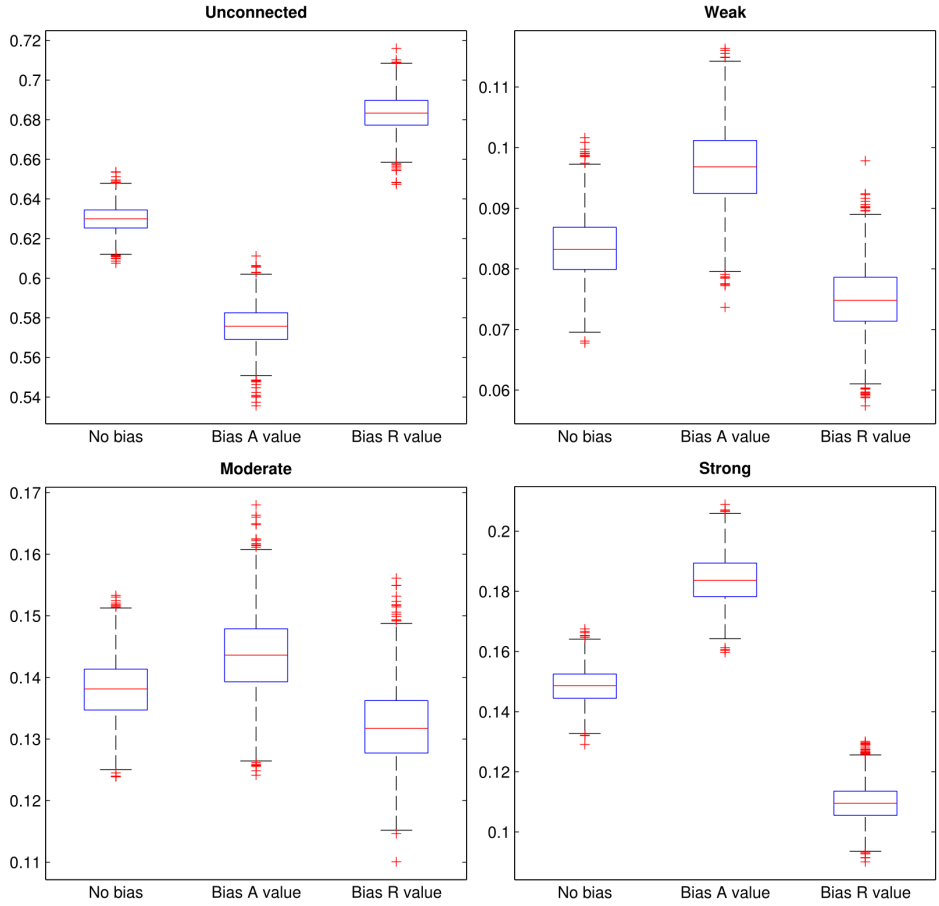

Figure 20: The change in four different π-values when you bias them to only represent one dataset. The no bias boxplot shows the values of π when it is shared between A and R. The other two show the values of π when it is separate for A and R respectively. The values used are all values in the two chains of 1000 samples.

7 Discussion

The data of the two experiments are quite different. About 21% of the connections in the datasets are different. By modelling the data we can merge the two datasets to find knowledge about the true underlying connectome. The found connectomes have a lower density than both of the datasets: mean 31.70% (dataset $\boldsymbol{A}$: 40.15%; dataset $\boldsymbol{R}$: 33.07%). This is unexpected, as one would expect the true density to be somewhere between the two found densities. It is possible that the algorithm has a strong bias to assign connections to the majority class, unconnected, which would explain the low density of the found connectomes. In Figure 18 you can also see that for the maximum probability sample π_0 is so large compared to the other $\boldsymbol{\pi}$ values that there are quite a lot of connections that would be assigned to be unconnected class because their similarity is big (hence a big dot product distance).

One of the main goals of the experiments was to find which settings and extensions of the standard model were needed to model the data. One point that can be made on this, is that there is not a large difference between the two distance measures. Although, in the end, the dot product distance had the best performance, the results show that there is not a big improvement of this measure in comparison with the Euclidean distance, when keeping all other things the same. It could even be useful to use the Euclidean distance over the dot product distance, because it is more intuitive. It was chosen to go on with the dot product distance though, since the choices made were mainly result driven.

The main reason for adding the dot product distance as a possibility in the first place was such that the matrix $\boldsymbol{P}$ could be introduced. Roughly 38% of the data is not symmetrical (hence connection i,j belongs to an other class than connection j,i) and introducing matrix $\boldsymbol{P}$ was expected to help in encoding this asymmetrical properties of the data. However, this was not the case. Adding $\boldsymbol{P}$ would worsen the results. The only way to encode asymmetry used besides this $\boldsymbol{P}$ matrix is to add random

sender/receiver effects. Adding these effects does contribute to a better fit of the data, also over using general random effects. This shows us that the data is not asymmetrical in its structure, but has merely asymmetric node biases.

The choice for the Cauchy distribution was carefully made. However, a mixture of, for example, normal distributions could be more suited to fit the data at hand because the tails are less heavy. A problem with the normal distribution is that probabilities will go to zero fast, which will increase the MNPLL a lot when there are connections that do not follow the regularity of the data. Another distribution that was considered is the Student's t-distribution. The Student's t-distribution has a parameter that controls the heaviness of the tails. However, the variance of the Student's t-distribution cannot be controlled, which explains why it was chosen to use a Cauchy distribution instead. Although, there is enough argumentation for using the Cauchy distribution, and it is not expected that the other distributions perform significantly better, the algorithm is not trivial enough to oversee the impact other distributions would have. The same holds for the hyperpriors. All chosen hyperpriors are well thought out, however, this does not necessarily mean that these settings are the most optimal for the problem. More experimentation can show whether other distribution and hyperpriors are better suited.

Even though it seems that too many connections are placed in the unconnected class, the overall behaviour of the algorithm is quite good. Looking at Figure 15 you can see that most connections are placed in the class in which you would expect them. Hence if $a_{i,j} = r_{i,j}$ almost all connections $c_{i,j}$ are placed in the same class. Also if $a_{i,j} = r_{i,j} \pm 1$ most connections are placed in either the class corresponding to $a_{i,j}$ or the class corresponding to $r_{i,j}$. However, if there is a bigger difference between the two class numbers than one, the algorithm does not do the most expected thing. Usually in this case it still chooses either the class of $a_{i,j}$ or the class of $r_{i,j}$, while you would expect that most of the time it would opt for a class between the two.

The used algorithm can also be used to predict unknown connections and can therefore also be used with other (incomplete) datasets. Furthermore, it is flexible in the number of classes and is therefore not restricted to the four classes used in this problem. It is even possible to use the designed model with binary data. Of course, the new model has more parameters then the standard latent space algorithm and it remains to be seen whether it works better.

Although it is possible to use the model for predicting unknown data, the process of model selection should change when doing so. In Figure 12 it shows that the model that best describes the data does not necessarily also make the best predictions on unseen data, most probably due to overfitting. Since the goal of this experiment was to merge data, and not to predict data, it was chosen to go for the model that best describes the complete data. However, by using cross-validation it should not be difficult to select a model for prediction. Even though accuracy on unseen data is not at all used in the first steps of model selection, the algorithm is still possible to predict unseen data above chance level. This is very promising for using these kind of algorithms in future prediction problems.

After model selection, the datasets were combined. Combining the datasets did not immensely change either the MNPLL or accuracy of the algorithm. This is a promising result. There is indeed more data to train the model, however, the data is not completely consistent with each other. Knowing that the conflicts in the data do not lower the performance of the algorithm shows us that the dataset indeed have the same underlying structure, which makes it viable to combine the two datasets to form a ground truth.

The data used in the experiment is labelled by experimenters, which makes it more susceptible for biases, especially since both tracers label a different part of a neuron. Figure 19 shows that this data indeed has a bias, namely $\boldsymbol{\pi}$. Setting $\boldsymbol{\pi}$ as a bias improves not only MNPLL, but also accuracy on both train data and unseen test data. Of course, having a different $\boldsymbol{\pi}$ for both datasets gives some problems with combining the data, since you would have to know the $\boldsymbol{\pi}$ values for the true connectome. The simplest solution would be to average the two values for $\boldsymbol{\pi}$ for the two datasets, but this would lead to similar $\boldsymbol{\pi}$ values as you would get by just sampling one $\boldsymbol{\pi}$ for both datasets (as can be seen in Figure 20). It is also possible to take a weighted average of $\boldsymbol{\pi}$ to express your confidence in a certain dataset. Say, for example, that dataset $\boldsymbol{A}$ is more likely to have accurate data, it is possible to choose values for $\boldsymbol{\pi}$ closer to the values of $\boldsymbol{\pi}$ as associated with dataset $\boldsymbol{A}$ to form your conclusions about the true connectome.

All in all the latent space algorithm can be adapted to not only fit but also predict unseen ordinal data. There is still a lot flexibility left in the model and making small changes in distributions and hyperpriors can have an impact on the performance of the algorithm.

8 Conclusion

Tract tracing data is a great way to show pathways in the brains of various animals. However, there are various problems with this technique that are not easy to overcome. Modelling the data can help to predict the unknown connection and reduce errors. By doing this we are able to give answers to various questions found in the literature as density and clustering of the brain and resolve problems with conflicting data. This thesis has shown that latent space algorithms are suitable to model the data obtained by various forms of tract tracing experiments and can help analyzing connectivity of the brain in various animals.

References

Bakker, R., Wachtler, T., & Diesmann, M. (2012). CoCoMac 2.0 and the future of tract-tracing databases. *Frontiers in Neuroinformatics*, *6*(December). doi:10.3389/fninf.2012.00030

Bassett, D. S., & Bullmore, E. (2006). Small-world brain networks. *The Neuroscientist*, *12*(6), 512–523. doi:10.1177/1073858406293182

Behrens, T. E. J., & Sporns, O. (2012). Human connectomics. *Current Opinion in Neurobiology*, *22*(1), 144–153. doi:10.1016/j.conb.2011.08.005

Bishop, C. M. (2006). *Pattern recognition and machine learning*. Springer.

Bohland, J. W., Wu, C., Barbas, H., Bokil, H., Bota, M., Breiter, H. C., … Mitra, P. P. (2009). A proposal for a coordinated effort for the determination of brainwide neuroanatomical connectivity in model organisms at a mesoscopic scale. *PLoS Computational Biology*, *5*(3), e1000334. doi:10.1371/journal.pcbi.1000334

Bozdogan, H. (1987). Model selection and Akaike's Information Criterion (AIC): The general theory and its analytical extensions. *Psychometrika*, *52*(3), 345–370. doi:10.1007/BF02294361

Brooks, S. P., & Gelman, A. (2013). General methods for monitoring convergence of iterative simulations. *Journal of Computational and Graphical Statistics*, *7*(4), 434–455. doi:10.2307/1390675

Duane, S., Kennedy, A. D., Pendleton, B. J., & Roweth, D. (1987). Hybrid Monte Carlo. *Physics Letters B*, *195*(2), 216–222. doi:10.1016/0370-2693(87)91197-X

Felleman, D. J., & Van Essen, D. C. (1991). Distributed Hierarchical Processing in the Primate Cerebral Cortex. *Cerebral Cortex*, *1*(1), 1–47. doi:10.1093/cercor/1.1.1

Friston, K. J. (1994). Functional and effective connectivity in neuroimaging: A synthesis. *Human Brain Mapping*, *2*(1-2), 56–78. doi:10.1002/hbm.460020107

Handcock, M. S., Raftery, A. E., & Tantrum, J. M. (2007). Model-based clustering for social networks. *Journal of the Royal Statistical Society: Series A (Statistics in Society)*, *170*(2), 301–354. doi:10.1111/j.1467-985X.2007.00471.x

Hoff, P. D., Raftery, A. E., & Handcock, M. S. (2002). Latent Space Approaches to Social Network Analysis. *Journal of the American Statistical Association*, *97*(460), 1090–1098. doi:10.1198/016214502388618906

Hoffman, M. D., & Gelman, A. (2014). The No-U-Turn Sampler: Adaptively Setting Path Lengths in Hamiltonian Monte Carlo. *Journal of Machine Learning Research*, *15*(1), 1593–1623.

Krivitsky, P. N., & Handcock, M. S. (2008). Fitting Position Latent Cluster Models for Social Networks with latentnet. *Journal of Statistical Software*, *24*(5), 1–23.

Krivitsky, P. N., Handcock, M. S., Raftery, A. E., & Hoff, P. D. (2009). Representing degree distributions, clustering, and homophily in social networks with latent cluster random effects models. *Social Networks*, *31*(3), 204–213. doi:10.1016/j.socnet.2009.04.001

Kruijswijk, J., Mørup, M., Bakker, R., & van Gerven, M. (2014). *Link prediction applied to tract-tracing data*.

Lau, B. (2014). Matlabstan: the matlab interface to stan. Retrieved from htp://mc-stan.org/matlab-stan.html

Lewi, J., Butera, R., & Paninski, L. (2009). Sequential optimal design of neurophysiology experiments. *Neural Computation*, *21*(3), 619–687. doi:10.1162/neco.2008.08-07-594

Markov, N. T., Ercsey-Ravasz, M. M., Ribeiro Gomes, A. R., Lamy, C., Magrou, L., Vezoli, J., … Kennedy, H. (2012). A weighted and directed interareal connectivity matrix for macaque cerebral cortex. *Cerebral Cortex*, *24*(1), 17–36. doi:10.1093/cercor/bhs270

Milgram, S. (1967). The small world problem. *Psychology Today*, *2*(1), 60–67.

Smith, S. M., Jenkinson, M., Johansen-Berg, H., Rueckert, D., Nichols, T. E., Mackay, C. E., … Behrens, T. E. J. (2006). Tract-based spatial statistics: voxelwise analysis of multi-subject diffusion data. *NeuroImage*, *31*(4), 1487–1505. doi:10.1016/j.neuroimage.2006.02.024

Sporns, O., Tononi, G., & Kötter, R. (2005). The human connectome: A structural description of the human brain. *PLoS Computational Biology*, *1*(4), 245–251. doi:10.1371/journal.pcbi.0010042

Stephan, K. E., Hilgetag, C. C., Burns, G. A., O'Neill, M. A., Young, M. P., & Kötter, R. (2000). Computational analysis of functional connectivity between areas of primate cerebral cortex. *Philosophical Transactions of the Royal Society of London. Series B*, *355*, 111–126. doi:10.1098/rstb.2000.0552

Wold, S., Esbensen, K., & Geladi, P. (1987). Principal component analysis. *Chemometrics and Intelligent Laboratory Systems*, *2*(1-3), 37–52. doi:10.1016/0169-7439(87)80084-9

Zingg, B., Hintiryan, H., Gou, L., Song, M. Y., Bay, M., Bienkowski, M. S., … Dong, H. W. (2014). Neural networks of the mouse neocortex. *Cell*, *156*(5), 1096–1111. doi:10.1016/j.cell.2014.02.023

A Preliminary experiments

To determine whether the latent space algorithm is a valid algorithm to predict connections between brain areas, and to find the settings for this algorithm some preliminary experiments are done similar to Kruijswijk et al. (2014).

For these experiments the fully observed part of the data by Markov et al. (2012) is used. This matrix is the top part of the matrix shown in Figure 1, consisting of 29 injection areas. For these experiments 50 of the known connections are deleted. Then a latent space model is trained on the remaining 791. Afterwards the model is used to predict the 50 deleted connections. All the experiments are done on the dataset with the moderate connections and in all experiments the same 50 (randomly chosen) connections are deleted. For the experiments three variables were varied to test for the optimal settings. First, the dimension of the latent space, then the number of clusters used by the latent space algorithm and last whether or not to use random receiver and sender effects (these effects favour connections with areas that already have a lot of connections). For all these parameter variations an MCMC-chain is trained and afterwards the accuracy is calculated for all samples in the chain. In Figure 21 to Figure 23 you can find boxplots of the results. The blue line indicates the accuracy when assigning the connections randomly. After these experiments it is chosen to do the rest of the study with two dimensions, no clustering and no receiver effects.

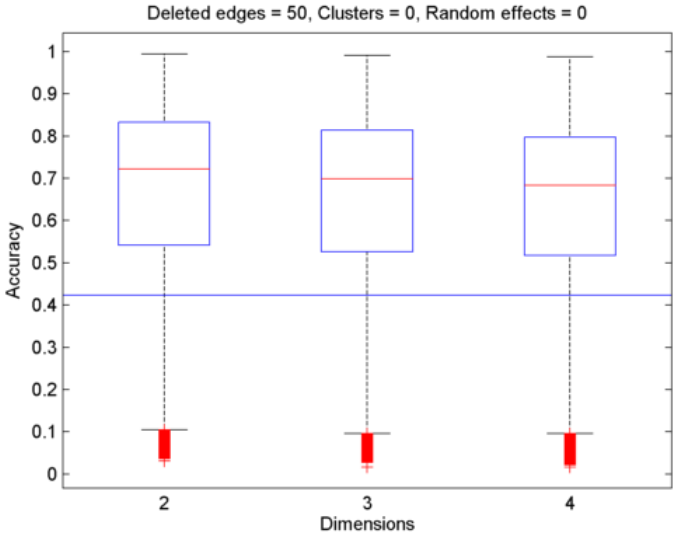

Figure 21: The accuracy of the latent space algorithm using different dimensionality.

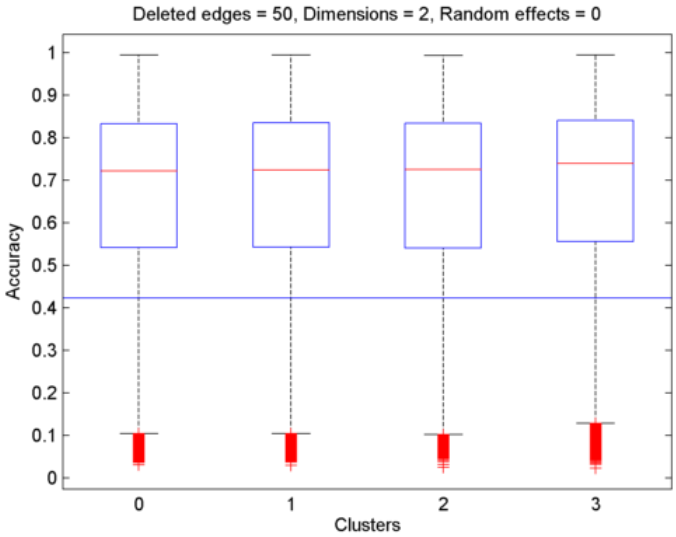

Figure 22: The accuracy of the latent space algorithm using different numbers of clusters.

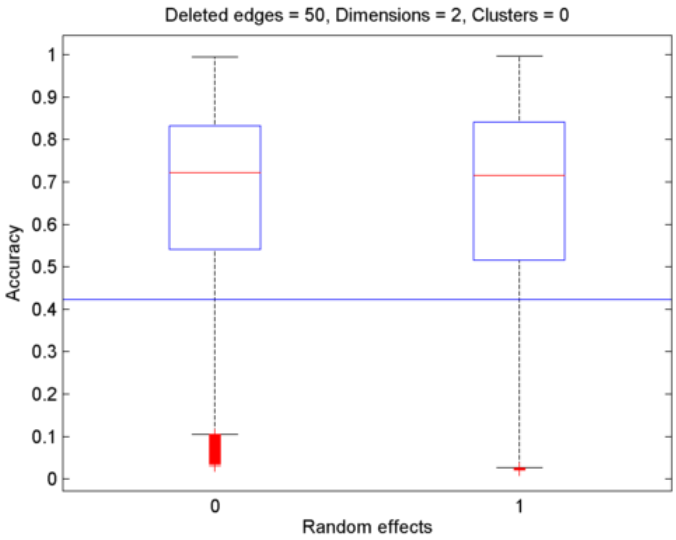

Figure 23: The accuracy of the latent space model with and without using random effects.

B Latent-space algorithm

To fit the latent space model the function `ergmm` of the `latentnet`-package (Krivitsky & Handcock, 2008) is used. A $N \times N$ matrix which is filled with ones, zeros and empty entries is the most important input for this function. In this matrix one stands for a found connection, zero for a connection that does not exist and the empty entry represents unknown connections. The most parameters are kept on the default with the exception of the following list:

Name	**Default value**	**Used value**
Dimensions	-	2
Burn-in	10,000	20,000
To fit	`mcmc, mkl, mkl.mbc, procrustes, klswitch`	`mcmc, mkl, mkl.mbc, procrustes, klswitch, pmode`

This also means that, although the package had the option, the algorithm is not set to use random effects as described earlier. The reason for this lies in accuracy testing done before starting the project (see Appendix A). The accuracy decreased using random sender and receiver effects. Furthermore, no properties of the areas are added to increase performance.

After fitting the latent space model, various results are saved for later use. First, for the fitted samples the locations of the nodes in latent space (Z) and the regression parameter (β) are saved. Furthermore, both the probability matrix and the locations of the nodes (Z) of the MAP are saved.

C AIC

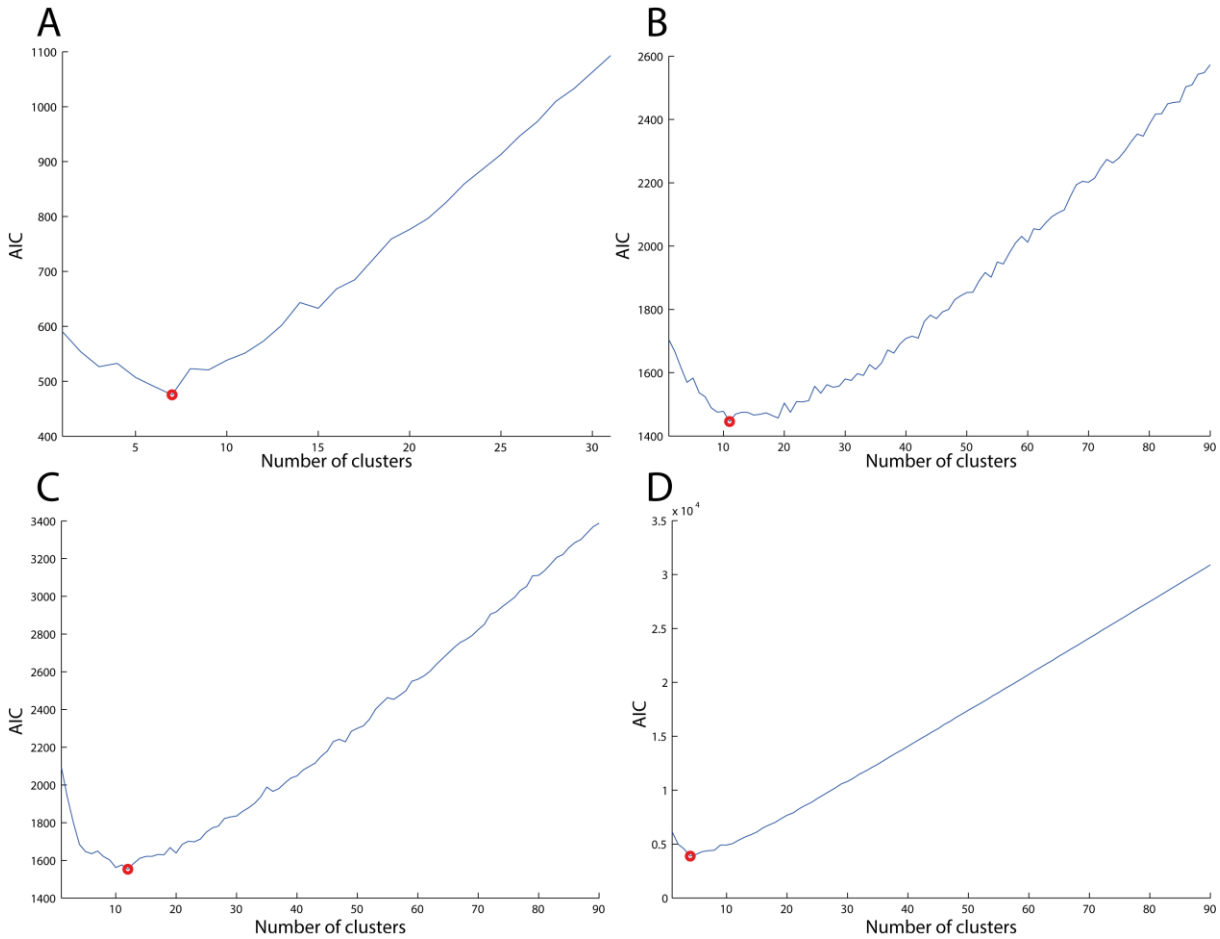

Figure 24: The AIC value of the Gaussian Mixture Models with the different number of Gaussian functions for (A) Felleman & Van Essen (1991) and Markov et al. (2012) with (B) all connections, (C) moderate connections and (D) strong connections. The red circle indicates the minimum of the AIC-curve and therefore the number of clusters used to cluster the data.

D Convergence of the algorithm

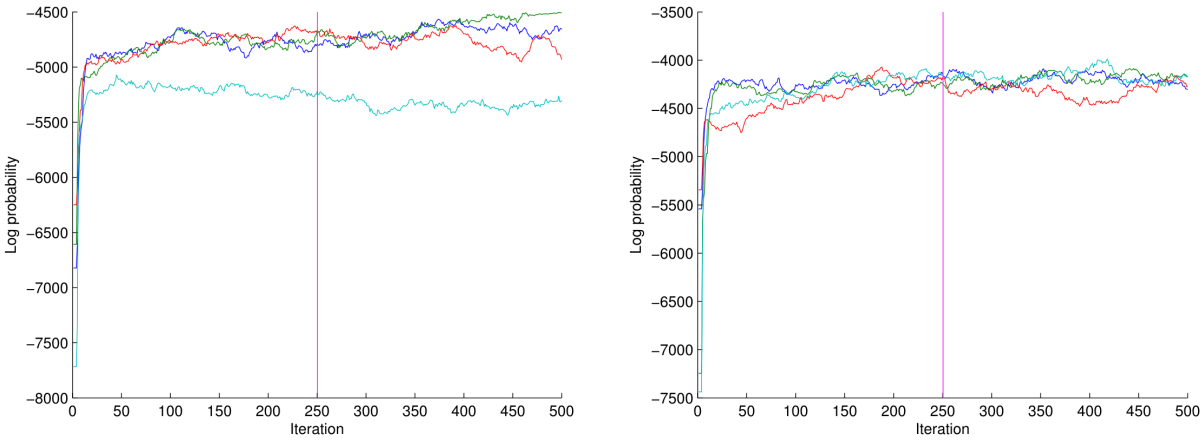

Figure 25: The convergence of the log probability of four chains of 500 samples. The settings used for these chains are four dimensions, Euclidean distance with π and random sender/receiver effects. The left figure shows the convergence on dataset *A* and the right figure on dataset *R*. Everything left from the pink line is discarded as burn-in in the results.

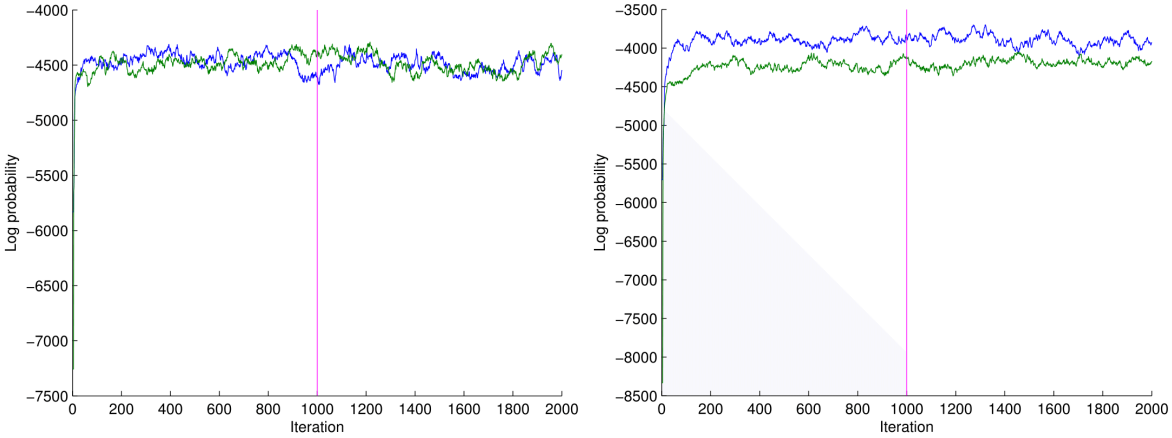

Figure 26: The convergence of the log probability of two chains of 2000 samples. The settings for these chains are five dimensions, dot product distance with π and random sender/receiver effects. The left figure shows the convergence on dataset *A* and the right figure on dataset *R*. Everything left from the pink line is discarded as burn-in in the results. Note, that of the six runs done with 2000 samples, this example is the least converged of all.