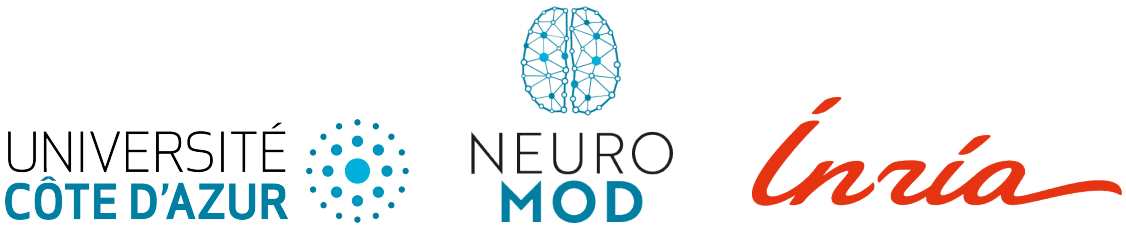
A REVISITING OF STRUCTURE-FUNCTION MAPPING USING GRAPH CONVOLUTIONAL NETWORKS

Sarah Mouffok

August 22, 2022

Supervisor: Samuel Deslauriers-Gauthier
Location: Inria Centre at Université Côte d'Azur,
Project-Team Athena
Project period: 01/03/2022 - 30/08/2022



Internship report
MSc 2 - Modeling for Neural and Cognitive Systems
Université Côte d'Azur, France

Acknowledgements

Data were provided [in part] by the Human Connectome Project, WU-Minn Consortium (Principal Investigators: David Van Essen and Kamil Ugurbil; 1U54MH091657) funded by the 16 NIH Institutes and Centers that support the NIH Blueprint for Neuroscience Research; and by the McDonnell Center for Systems Neuroscience at Washington University.

The authors are grateful to the OPAL infrastructure from Université Côte d'Azur for providing resources and support.

I am thankful to Yang Ji for helping to clear any questions I had when I picked up on this project that extends her previous work. I would also like to thank every member of the Athena Project-Team at INRIA for making my days of working on this project more enjoyable. Given the interdisciplinary nature of this project, I gained a lot from team members that had a great deal of expertise in the disciplines that make up this project. Independently from the work, I am also thankful for their friendship and the very interesting lunch breaks and coffee breaks that came with it.

I am also grateful to the Neuromod institute for Modeling in Neuroscience and Cognition, for allowing me to present my work at the 2022 Neuromod conference, in the form of a poster.

Last but not least, this project could not have taken this form without my supervisor Samuel Deslauriers-Gauthier. The iterative steps of the research process were animated by his expertise in the domain, and he kept on proposing extensions and leads to investigate in order to understand the problem at hand a bit more each time. Not only did he guide the process, but at each step, remained available should help be needed. He has been a great teacher and managed to explain complex concepts simply and with patience, which enabled me to learn an immense quantity of information during this internship.

Contents

1 Introduction

Being able to infer the function of a brain given the knowledge of its structure is valuable in order to understand the impact of structural alterations caused by injuries and/or diseases on the function of the brain. Indeed, devising a mapping from brain structural connectivity (SC) to brain functional connectivity (FC) is motivated by the thought that structure is the physical support on which function operates.

Consequently, using supervised learning, we attempt to predict a subject's FC matrix from their SC matrix. This work extends [12]'s, in which an auto-encoder architecture is used to perform the aforementioned task. The SC matrix is used inside the encoder to encode an initial activation state of the brain into a latent space, and the decoder outputs the FC matrix, that we assume to be a function of the SC matrix.
More particularly, its encoder is a Graph Convolutional Network (GCN). Its architecture is chosen due to the fact that these connectivity matrices can be interpreted as the adjacency matrices of graphs, with each node being a brain region, and an entry in the matrix holding the weight of the edge between the two brain regions corresponding to the indexes of the entry. Therefore, if the brain is initialized with an initial state, and the graph convolution operation is performed, exchange of information between regions occurs following the weight of their connections, as described by the SC matrix.

The convolution operation has filters applied to the SC matrix's Laplacian following a Chebyshev polynomial, as defined by [14], and applied by [12]. In tbis work, we revisit approximations made on this polynomial in order to explore the influence of individual weight sets per polynomial members, and higher polynomial orders.
Additionally, we investigate the usage of different distance metric in order to drive the network's learning. Currently, the distance metric used is the Mean Squared Error (MSE) which merely considers each matrix as a set of scalar values. FC matrices are symmetric positive definite matrices, leading us to use the Affine-Invariant Riemannian distance metric on SPD matrices as the network loss function.

Consequently, the question we attempt to answer is:
Can we improve the prediction of FC of brain cortical areas from an estimation of brain SC by 1. removing assumptions found in Yang. Ji's GCN model [12] and 2. considering a different distance metric between FC matrices?

Performances are compared to a reference estimator, the Fréchet Mean of FC matrices. The comparison criteria is the metric used as the network loss function and the Pearson Correlation.

The dataset is provided by the Human Connectome Project (HCP) and holds 1050 healthy subjects' diffusion Magnetic Resonance Imaging (dMRI) scans, their rest-state functional Magnetic Resonance Imaging (fMRI) scans, and their structural Magnetic Resonance Imaging scans. Standard processing pipelines are used to generate SC matrices and FC matrices. The brain is divided according to two different atlases ($N = 68$ regions following the Desikan-Killiany cortical atlas, $N = 200$ following the Schaefer cortical atlas), leading to squares matrices with the value at cell (i, j) corresponding to a measure of connectivity between the area indexed by i and the area indexed by j.

This report will start off with a review of related works in structural connectivity, functional connectivity, Graph Convolutional Networks and Structure Function Mapping. Then, the details of the processing steps that led to the two datasets being used will be developed. Following that, the autoencoder architecture performing the FC estimation will be described. Afterwards, the results of the estimations will be described. Finally, we will reflect on our results and point out limitations of the design choices that were made and possible future improvements.

2 Related work

The neuron is the elementary processing unit of the brain. A neuron receives information from other neurons through its dendritic projections. Aggregation and computation is made from this information in the neuron's cell body, and the result is then propagated to other neurons through a long fiber, called an axon. Observing a structural MRI of the brain, two tissue types can be visually told apart: "White Matter" (WM) and "Grey Matter" (GM). GM is found at the external surface of the brain (cortical grey matter), as well as deep structures (sub-cortical grey matter). This tissue type has a high density in cell bodies. WM is made of densely packed myelinated axons or fiber bundles, as well as glial cells and blood vessels. The majority of WM is made up of commissural fibers and association fibers: they transfer information between intra and inter-hemispheric brain regions [13].

Structural connectivity corresponds to a quantification of these WM fibers that link brain cortical regions. Structural information can be inferred from diffusion-weighted MRI data: dMRI holds information on the diffusion of water molecules in the brain.

- Grey Matter (GM) tissue can often be seen as generally unorganized and hinders diffusion equally in all directions, so diffusion is modeled and measured as equal in all directions.
- In Cerebro-spinal fluid (CSF), water molecules also diffuse equally in all directions.
- In White Matter (WM), and more precisely, within axon fibers, water molecule diffusion direction tends to follow the orientation of the axons they are in: dMRI is therefore particularly interesting in order to study WM tracts.

Inferring structural connectivity from dMRI is in itself a field of research, often based on tractography methods that attempts to reconstruct WM tracts [19, 32]. In this work, structural connectivity is represented by a square matrix. At the cell in position i, j, there will be a quantification of the connectivity between cortical regions indexed by i and the region indexed by j. The brain cortex is modeled as a network of cortical regions: this network can be seen as a graph, with the structural connectivity matrix as its adjacency matrix. Information held in cortical regions is modeled as node states, and this information propagates following the strength of the connections to other brain regions, properties of the edges of the graph.

Functional MRI (fMRI) is a brain imaging method that indirectly captures activity in the brain through time. This imaging method measures a signal dependent on variations of blood oxygenation levels in the brain, that is referred to as the BOLD signal (Blood Oxygenation Level-Dependent). Resting state functional connectivity is defined by the similarity between temporal activation patterns of cortical brain regions, while a subject is at rest. This information can be computed from resting-state fMRI data following different methodology [29]. In this work, it is also held in a square matrix, with each cell indexed by i, j containing the similarity of the activation patterns of brain regions indexed by i and j.

Structure-function mapping has been studied a variety of ways. [6] groups under a general formulation multiples structure-function mappings based on the eigenmodes of the SC or its Laplacian. Later on, they extend this work in order to use Riemannian metrics to the problem [5]. Both of these consider analytical models of structure-function mapping. [12] approaches the problem by using artificial intelligence to perform this mapping, more particularly Graph Convolutional Networks (GCN) and Graph Transformer Networks.

As this work uses the GCN architecture of [12] as a starting point and investigates improvements in their model,

it makes sense to look at the work leading to this GCN formulation. In [2], a GCN architecture based on spectral filtering theory was developed, with the added properties of being fast, with a reduced number of parameters to estimate and a recursive form, as well as localized. The details of this will be developed in Section 4.6.2. Similarly, [14] used GCNs in the context of semi-supervised classification. Their GCN fits into the general formula proposed in [2], under some parameter choices and approximations. It is this GCN that has been used in [12], the work at the basis on this current project.

3 Data and preprocessing

3.1 HCP database

Data was obtained from the WU-Minn Human Connectome Project's database [30]. It contains different imaging modalities for 1206 healthy young adult participants, including:

- 3T Diffusion MRI data, pre-processed with the HCP diffusion pipeline [18, 7, 21, 31, 23]. 1065 out of 1206 subjects have such a package.
- 3T Resting State Functional MRI data pre-processed with the HCP functional pipeline [18, 7, 21, 31], from the REST1 session. 1096 out of 1206 have such a package.
- 3T Structural MRI, pre-processed with the HCP structural pipeline [15]. 1113 out of 1200 have such a package.

The number of subjects that have all packages available as well as all necessary files within each package is 1050 for $N = 68$ and 1042 for $N = 200$.

3.2 Processing pipelines to connectivity matrices

3.2.1 Atlases

An atlas is a brain volume that attributes to each voxel a label that indicates the brain structure this voxel belongs to. They are usually registered to a known template space, enabling its user to perform the appropriate registration steps to use it. Atlases usually have different focuses (cortex, thalamus, STN, white matter, etc.), making them more or less relevant depending on the intended usage. We are looking for an atlas that describes the division of the brain cortex into sub-regions, so cortical atlases will be used. The cortex can be divided into sub-regions following different criteria. This division is itself a field of research. The two atlases used are examples of this varying methodology:

Desikan-Kiliany [4] This atlas was devised by considering anatomical properties of the brain, particularly the brain curvature. Inflated versions of brain structural MRI scans (in which the brain is visualized with its folds flattened, but curvature information is preserved). These inflated brains are manually labeled with 34 regions of interests (ROIs) per hemisphere. Then, the atlas is generated using a registration procedure that aligns the cortical folding patterns (from the curvature information) [10] and a probabilistic labeling algorithm assigns a region to every point on the non-flattened cortical surface. In total, 68 cortical regions are described by this atlas.

Schaefer [20] This atlas uses rest-state fMRI to compute cortical regions: its model combines two approaches: a local gradient that detects differences in functional connectivity patterns, relevant to identify cortical area boundaries, and a global similarity approach that clusters functional connectivity patterns, without considering matters of spatial proximity. Multiple resolutions of this atlas are available (from 100 parcels to 1000 parcels). Here, we chose the parcellation scheme with 200 cortical regions.

3.2.2 fMRI to FC matrix

In this work, functional connectivity is defined as the correlation of the time series of activity between brain cortical regions.
Four files are used:

- rfMRI_REST1_LR_hp2000_clean.nii.gz - from the HCP database
- Movement_Regressors.txt - from the HCP database
- Movement_Regressors_dt.txt - from the HCP database
- The cortical atlas parcellation registered to the fMRI data

REST1 refers to the acquisition session name, Left to Right (LR) phase encoding was performed during the acquisition, and highpass filtering is applied with a "cutoff" of 2000s (hp2000).

An fMRI acquisition corresponds to a time series of 3D brain volumes. Each volume is made up of voxels, which are its elementary units. A voxel is a small cube defined by its size (determining the spatial resolution of the fMRI) and its value (representing the intensity of the signal, which is a quantification of brain activity). As an example, Subject 100307's fMRI volume has the following dimensions:

$$(X, Y, Z, T) = (91, 109, 91, 1200)$$

Subject head movement during the scan leads to motion artefacts. These have been corrected with FLIRT, an FSL-based tool performing an 12-parameter affine-body registration onto a reference image (the "Single-Band Reference" image) acquired in a moment closely preceding the fMRI volume sequence acquisition. The transformations have already been applied to the fMRI volumes, but their output is available in the two last files mentioned.
An affine transformation has the following format:

$$\begin{bmatrix} m_{00} & m_{01} & m_{02} & t_0 \\ m_{10} & m_{11} & m_{12} & t_1 \\ m_{20} & m_{21} & m_{22} & t_2 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

It contains 12 parameters, describing operations of rotation, scaling, translation and shearing. There are as many of these matrices as time points. The nature of the transformation picked (affine) means that the transformation for all voxels of the volume is the same. Each movement regressor (component of the regression matrix) has a time evolution, explaining this regressor's contribution in the movement that occurred. To each voxel can be associated a timeseries. This timeseries can be partly described by the movement that occurred, and whichever way the movement influenced the voxel's activation time series needs to be removed.

Movement regressors and their derivatives over time are available in two text files.

Removing movement can be described as a minimisation problem: we would like to minimize $\bar{y}$, the amount of data that cannot be explained by movement:

$$\underset{\alpha}{\operatorname{argmin}} \|\bar{y}\|^2 = \|y - y_{mov}\|^2 = \|y - R\alpha\|^2$$

The solution to this minimisation is:

$$\alpha = R_L^{-1} y = (R^T R)^{-1} R^T y$$

At each voxel:

- y is a (1200×1) matrix

- R is a (1200×24) matrix
- α is a (24×1) matrix of estimated coefficients.

Then, a band-pass filter of frequency range [0.01, 0.08] Hz is applied onto the signal, following findings that the major frequency contributions to resting-state functional connectivity come from fluctuations less than 0.1 Hz [1].

Following that, using the information found in the atlas volume, for each brain region, all voxels belonging to this region have their timeseries of activation averaged together, resulting in the timeseries of activation of the brain region. Finally, the Pearson Correlation between pairs of regions is computed and organized into a square matrix. The Pearson correlation can be written the following way:

$$f(x_i) = \frac{x_i - \bar{x}}{\sqrt{\sum_{x_i}(x_i - \bar{x})^2}} \qquad r = f(X_{i,j})f(X_{i,j})^T \tag{1}$$

With X the matrix of dimensions (N, T), which holds the fMRI signal associated to N cortical regions in a time window of size T.
r is a symmetric positive semi-definite matrix, with each of the entries of the matrix in [-1, 1]. It also has a diagonal of 1s, as signals are fully correlated with themselves.

While it is ensured to be symmetric positive semi-definite by its construction, in practice, this FC matrix has no zero eigenvalues and is treated as symmetric positive definite. A zero eigenvalue is an indicator that there is colinearity between timeseries. With $T >> N$, there are much fewer timeseries than number of values in each timeseries: as a consequence, it is very unlikely that a cortical region would have its timeseries colinear to another cortical region's. This leads the matrix of timeseries' Pearson Correlation to have no 0 eigenvalue.

It is worth noting that unlike the data in [12], the FC matrix's negative correlations are not systematically clipped to 0 in order to preserve the SPD properties of the FC matrices.

(a) $N = 68$ (b) $N = 200$

Figure 1: Subject 100307's FC matrices, with the cortex divided into N regions

3.2.3 dMRI to SC matrix

Theory behind dMRI processing In order to trigger the displacement of water molecules in the brain, magnetic gradients are applied by the dMRI machine. These gradients can be described by two values: the b-value and the b-vector. The b-value holds information on the strength and timing of the gradients used to generate the dMRI image, expressed in $s \cdot mm^{-2}$. The b-vector describes the diffusion direction.
Additionally, some brain volumes are acquired for the b-value $b_0 = 0\ s \cdot mm^{-2}$, where no diffusion-inducing gradient is applied on the brain. These volumes are used as references.

The number of shells associated to an acquisition refer to how many different b-values different from 0 were used. A single-shell acquisition uses a single b-value, while a multi-shell acquisition uses multiple.

A dMRI image is 4-dimensional. The first 3 dimensions are spatial dimensions describing a 3D brain, and the 4th dimension indicates which (b-values, b-vectors) pair describes the magnetic gradient applied.

A single voxel often contains multiple fiber orientations. Diffusion Tensor Imaging (DTI) methods are lacking in order to capture this anatomical fact, as they only estimate a single fiber orientation per voxel. Rather, it makes more sense to model these orientations as a distribution. Constrained Spherical Deconvolution (CSD) [25] is a method that was created in order to address the aforementioned limitations of DTI. Its principles will be explained superficially, in order to understand the processing pipeline.

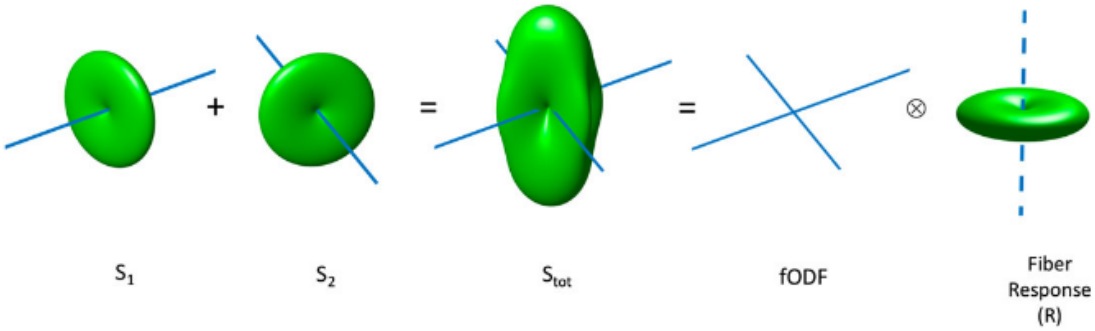


Figure 2: An illustration of the principles behind spherical deconvolution: the goal is to estimate the $fODF$. Deconvoluting the signal s_{tot} by the fiber response function R results in the $fODF$. s_1 and s_2 can be seen as the decomposition of the signal by the individual contributions of coherently oriented fibers going in 2 different directions. Image source: [3]

CSD performs a task that is referred to as the estimation of a fiber Orientation Distribution Function (fODF). At each voxel, this signal (s_{tot} on Figure 2) is said to live on a sphere, because the b-vectors that describe the acquisition of the signal are sampled in order to cover a sphere.

If the signal is seen as the summation of the signals resulting from coherently-oriented fibers going in different direction, the signal resulting from a single bundle of axons with a coherent orientation can be used as a deconvolution kernel. This deconvolution of the signal by the response function can be seen as the division of the Fourier transform of the signal by the Fourier transform of the estimated response function.

Projecting the signal in Fourier domain, it is expressed according to a basis of spherical harmonics. Some properties of the signal and the estimated response function limit the spherical harmonics that will be used in Fourier space:

- The dMRI signal is antipodally symmetric (the signals associated to two colinear b-vectors are equal), which allows for the signal to be described according to spherical harmonics of even degree l (those of odd degrees are not antipodally symmetric) [24]
- The response function is axially symmetric (it represents the signal for a coherently aligned bundle of fibres aligned with the z axis), meaning only spherical harmonics with order $m = 0$ are used. These can be referred to as zonal spherical harmonics [24]

dMRI processing Relevant files retrieved for future processing of dMRI data are:

- From HCP Diffusion data:
  - data.nii.gz, a compressed NIfTI file holding the dMRI data
  - bvals, a file containing the b-values
  - bvecs, a text containing the b-vectors
- From HCP Structural data:
  - *atlas_name*.nii.gz, a compressed NIfTI file, attributing to each voxel of the brain volume a label describing the structure it belongs to (for the whole brain, including cortical GM, subcortical GM, WM and CSF). This is a parcellation image generated by Freesurfer [9].
  - 100307.L.white_MSMAll.32k_fs_LR.surf.gii, a GiFti file created with Freesurfer tools, holding a list of vertices on the surface of the left hemisphere.
  - 100307.R.white_MSMAll.32k_fs_LR.surf.gii, a GiFti file created with Freesurfer tools, holding a list of vertices on the surface the right hemisphere.
  - *atlas_name*.label.gii, a GiFti file attributing for each vertex of the two previous files, a label according to a parcellation scheme of the cortex.

Subject 100307's dMRI acquisiton has the following dimensions:

$$(X, Y, Z, K) = (145, 174, 145, 288)$$

The scanner can apply gradients of different strengths and timing (b-value), and directions (b-vector), and a 3D volume is produced for all (b-value, b-vector) pairs. The value of these pairs can be extracted from the associated text files:

- b-values.txt described the b-values $\{0, 1000, 2000, 3000\}$. $b_0 = 0$ is used as a reference.
- b-vectors.txt contains 288 b-vectors. 18 vectors are associated to b_0, and for the three remaining b-values, there are 90 vectors each, leading to $K = 18 + (90 \times 3) = 288$

Processing of dMRI data is done using MRTrix 3.0 tools [26].

Tissue segmentation Using the MRTrix's **5ttgen** tool, a 5 tissue-type (5tt) image is generated, by "splitting a voxel into 5", leading to 5 3D volumes, describing the individual contribution of 5 tissues types: cortical GM, Subcortical GM, WM, CSF, Pathological tissue. The option defining the algorithm to choose is *freesurfer* due to the fact that we provide as input the Freesurfer parcellation image: *atlas_name.nii.gz*. A default parcellation scheme is considered for this algorithm, described in a table that can be found on Freesurfer's website, under the name *FreeSurferColorLUT.txt*. Should the input parcellation file be done according to a different parcellation scheme, a look up table would need to be provided. The output file is named under *5tt.mif*.

WM-GM interface extraction The voxels at the interface between WM-GM are extracted into a file called *gmwmi.mif*. This is done using **5tt2gmwmi**.

Converting the data into MRTrix's format The data is packaged into MRTrix's .mif format, which includes the diffusion data, the b-values and the b-vectors. This is done using **mrconvert**. The bvalues and bvectors are specified using the *–fslgrad* option.

Response function estimation Only zonal spherical harmonics of even degree are used to project the signal in Fourier domain. Additionally, the maximum order of these zonal spherical harmonics is a function of the number of b-vectors per b-value. The total number of spherical harmonics (including non-zonal) of all considered orders should be inferior to the number of b-vectors associated to non-zero b-values. Moreover, there are $2 \times (l+1)$ harmonics per order l.

For a maximum order $l_{max} = 8$, the orders considered are $S = \{0, 2, 4, 6, 8\}$.

$$\sum_{l \in S} 2 \times (l+1) = 46 \qquad \sum_{l \in S \cup \{10\}} 2 \times (l+1) = 67 \qquad \sum_{l \in S \cup \{10,12\}} 2 \times (l+1) = 95$$

While there are enough b-vectors to consider spherical harmonics up to order $l_{max} = 10$, in [27], it has been found that in practice, spherical harmonics terms above $l = 8$ are negligible and noisy. The final maximum order chosen is $l_{max} = 8$.

The MRTrix's command to perform this estimation is **dwi2response**. The *msmt_tt* [11] algorithm is chosen. This algorithm is appropriate for multi-shell (multiple b-values) and multiple tissue response function estimation. It takes as input both the data in .mif format, and the division of the parcellation in 5 tissue types. The maximum harmonics degree is also specified for each b-value.

As a result, the estimation of the response function outputs files of 6 values each, corresponding to the coefficients of the 6 spherical harmonics of the estimated response functions in Fourier space. There are 3 files output: the estimated response function for WM, GM and CSF.

Fiber orientation distribution function estimation MRTrix's **dwi2fod** is used for fODF estimation, with the algorithm *msmt_csd*, based on the same paper as the one describing the estimation of the response function [11]. It takes as input the data packaged in .mif format, and the 3 estimated response functions. It outputs the fODF, as well as volume fractions of CSF and GM.

Generation of streamlines with tractography Anatomically-Constrainted Tractography (ACT) is performed using MRTrix's **tckgen** command with the *iFOD2* algorithm [28]. This anatomical constraint is possible by providing the 5tt segmentation as parameter. Aditionally, the GM-WM seeding constraint is applied, which reduces the start of streamlines at the interface between the GM and WM, following known anatomical priors. 5 millions streamlines are generated, with constraints on their minimal and maximal size (sizes are included in [30, 400]). The backtrack option is specified, allowing tracks to be truncated and re-tracked if the streamline termination is not satisfactory (for example, the streamline ends in WM when the anatomical priors indicate it should end at the WM-GM interface). Additionally, the crop_at_gmwmi option is used, to crop streamline endpoints more precisely when they cross the GM-WM interface.

Compiling the connectivity matrix A brain hemisphere can be describes by vertices: 3D points belonging to the hemisphere surface. In the files 100307.L.white_MSMAll.32k_fs_LR.surf.gii and 100307.R.white_MSMAll.32k_fs_LR.surf.gii, the coordinates of these vertices are available, for the left and the right hemisphere respectively. In the file *atlas_name*.label.gii, the label attributed to that point according to a parcellation scheme is available.

As a first step, the vertices of the brain are organized in a k-dimensional tree, in order to perform nearest-neighbors search efficiently.

A streamline resulting from tractography is a list of vertices on the path of the constructed fiber it represents. For each streamline, the starting point (first in the list of points on its path) is used as a parameter is the search for a nearest neighbor in the k-dimensional tree: this looks for the vertex of the brain that is closest to the streamline seed. Matches of nearest neighbors are pruned in order to be within a maximal distance of 2.0 mm. The same search

is done with the streamline final points. The result of this search is the index of the vertex in the brain that is the closest to the streamline start, and the index of the vertex in the brain that is closest to the streamline end.

Then, for these two identified points, the label of the brain region they belong to is retrieved. This nearest-neighbor information is then reorganized into a mapping that registers each streamline to the right pair (starting region label, ending region label).

Finally, for each pair of labels, the list of streamlines that have been registered to them is retrieved, and a function is applied to that list, depending on what information is relevant. In our case, the only information extracted is the number of streamlines. Another example of information that could be extracted is the average streamline length, as investigated in [12].

(a) $N = 68$ (b) $N = 200$

Figure 3: Subject 100307's SC matrices, with the cortex divided into N regions (after applying $x : \log(x + 1)$ due to big differences in streamlines count values between region pairs.)

4 Methods

4.1 Tools and Libraries

The neural network was implemented in Python 3.9, with the Pytorch 1.11.0 library. Riemannian geometry tools are included in the geomstats 2.5.0 library [16, 17]. Data processing and neural network training are ran on the NEF Cluster of INRIA, using GPU resources for the neural network training.

4.2 Structure Function Mapping

The goal of structure-function mapping is find a function f of a subject's SC matrix that computes an approximation of their corresponding FC matrix.

$$\underset{f}{\operatorname{argmin}} \sum_{n=0}^{N} d^2(FC, f(SC)) \tag{2}$$

4.3 Metrics

Metrics have been chosen in order to describe the distance between real FC matrices and predicted FC matrices.

4.3.1 Mean Squared Error

A simple distance function between two matrices is the mean of any chosen distance between two scalars, applied on all of their cells. Here, the distance separating two cells is their squared distance.

The mean squared error (MSE) is defined as:

$$MSE(A, B) = \frac{1}{N^\star} \sum_{i=1}^{N} \sum_{j=1}^{N} (a_{i,j} - b_{i,j})^2 \tag{3}$$

The matrix to predict is symmetric and its diagonal is full of ones, as it holds cortical regions signals' correlation with themselves. The prediction of the FC matrix is reduced to the prediction of its lower triangular half.

Since the matrix is symmetric, computationally, this is done setting to zero all values on the diagonal or above, and dividing the sum of squared differences by $N^\star = \frac{N \times (N-1)}{2}$. When using the MSE as the loss function of a neural network, the network optimizes its weights only in order to reduce the difference on the cells of the lower diagonal only, because the loss between the cells on the diagonal and upper diagonal triangle is always 0. Even if the network outputs a square matrix, the values on the diagonal and above are never taken in consideration and are fully ignored, as they are cleared.

4.3.2 Affine-Invariant Riemannian metric on Symmetric Positive Definite Matrices

The FC matrices being SPD matrices, more appropriate distance metrics may be applied to it, that take into account the matrix and its properties as a whole.

Granted we can ensure the network also outputs a SPD matrix, the Affine-Invariant Riemannian metric (AIRM) on SPD matrix is used as a loss function. It is implemented with geomstats with a pytorch backend, enabling automatic differentation and allowing it to be integrated in the network seamlessly. This metric is defined below:

$$d^2(A, B) = \sqrt{Tr(\log^2(A^{-1/2} B A^{-1/2}))} \tag{4}$$

4.3.3 Pearson Correlation

The Pearson Correlation is used as a neutral metric. The network does not aim to optimize this metric, and it is interesting to see how correlated matrices are when the goal is to optimize the loss function. Again, this metric is applied on the lower triangular part of the matrices. Computationally, it is done according to the following formula:

$$r_{AB} = \frac{N^\star \sum a_{i,j} b_{i,j} - \sum a_i \sum b_{i,j}}{\sqrt{N^\star \sum a_{i,j}^2 - \left(\sum a_{i,j}\right)^2} \sqrt{N^\star \sum b_{i,j}^2 - \left(\sum b_{i,j}\right)^2}} \tag{5}$$

With $N^\star = \frac{N \times (N-1)}{2}$.

4.4 Reference estimator

The performances of the network need to be compared to a reference. Following [12, 5], given a dataset of healthy subjects, a sensible estimation of one subject's FC is chosen to be the mean of all existing subjects'.

The definition of a mean is dependent on the distance metric that separates two FC matrices. The Frechet Mean captures this fact: given a set of values $y_1, \ldots, y_N \in \mathbb{Y}$, the Frechet mean is the point in the set that minimizes the sum of its squared distance with all points [8].

$$\mu = \underset{y \in \mathbb{Y}}{\operatorname{argmin}} \frac{1}{N} \sum_{n=1}^{N} d^2(y, y_n) \tag{6}$$

With d, a distance function in the space $\mathbb{Y}$. While there may be multiple solutions to this minimisation problem, we assume that the computed solution is unique.

4.4.1 Frechet Mean - Riemannian distance

The need for a general definition of the mean is guided by the usage of the AIRM on SPDs. When training the network with the AIRM on SPDs as a loss function, the computation of the Frechet Mean is done by specifying this very distance function into geomstats' Frechet Mean estimator, which uses gradient descent to perform such an estimation.

4.4.2 Frechet Mean - Euclidean distance

It is worth noting that the arithmetic mean is a special case of Frechet Mean, with the squared distance metric taken as the element-wise Euclidean distance $d_E(A, B) = A_{i,j} - B_{i,j}$.

$$\mu = \underset{y \in \mathbb{R}^{n \times n}}{\operatorname{argmin}} \frac{1}{K} \sum_{k=1}^{K} d_E^2(y, F^k) = \underset{y \in \mathbb{R}^{n \times n}}{\operatorname{argmin}} \frac{1}{K} \sum_{k=1}^{K} (y_{i,j} - F_{i,j}^k)^2 = \frac{1}{K} \sum_{k=1}^{K} F_{i,j}^k \tag{7}$$

This mean is taken as the reference estimator to compare to the network trained with the MSE as a loss function.

4.5 Autoencoder

An autoencoder is a neural network architecture that attempts to represent its input in a lower dimensional space, and eventually reconstruct it back from that "compressed" space, minimizing the error between the reconstruction and the original input. This lower dimensional space, called latent space, is intended to be an extraction of essential features (latent variables) of the input, that can describe it as completely as possible.

An alternative usage of such an architecture is to extract meaningful features of the input as intended by the encoder, but to produce a network output that is a function of the input, rather that the input itself.

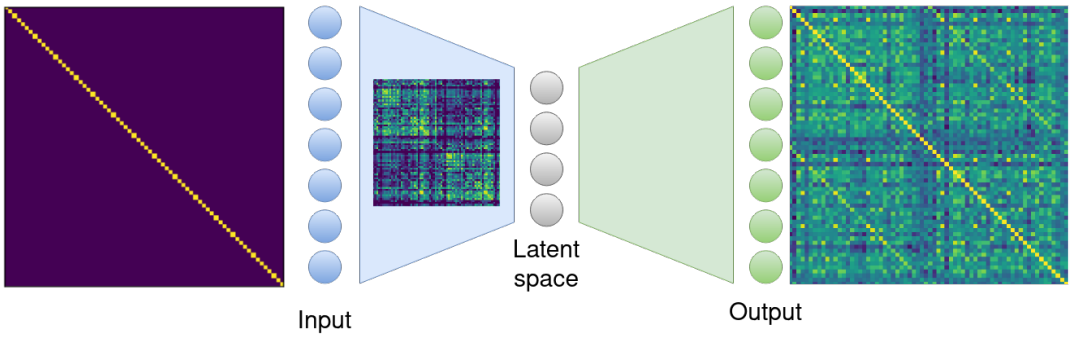


Figure 4: An illustration of the autoencoder architecture

4.6 Encoder: Graph Convolutional Network

The choice of architecture is motivated by the following macroscopic view of brain function: when a task is being performed by the brain, cortical areas exchange information. Following this, the resulting state of the brain cortical areas depends on:

- the initial state of cortical areas, before propagation
- the links between cortical areas
- the length in time of information propagation

Initially, the information we are working with is structural connectivity information: as described in Section 3.2.3, it is a quantification of the physical pathways that link brain areas. In an attempt to model the brain as a graph, this matrix can be interpreted as the adjacency matrix of a graph.

Depending on the atlas being used, this graph has $N = 68$ or $N = 200$ nodes, each corresponding to a cortical area. A value in the SC matrix at index (i, j) corresponds to the weight of the edge between the cortical area indexed by i and the one indexed by j.

4.6.1 Input

The input to the network is interpreted as the initial state of brain cortical areas, before this state propagates through the fibers that link them. There is no clear way to determine which input is most appropriate. Following [12], the input to the network is chosen to be the identity matrix $\mathbb{I}_N$. We interpret it to represent N graph states, each a vector of size N. For each graph state, a single node has a non-zero state, and therefore, the influence of a single node on the whole graph is being observed. In reality, the network can accept any matrix of size $(\star, N)$ as input. Finding a sensible input matrix may be worth investigating in the future.

4.6.2 Why convolutional?

The most common convolutional networks perform a discrete convolution operation between two elements in 2D Euclidean space. The convolution filter is being convoluted to the input of the network, often an image. This convolution operation is often illustrated with a sliding filter traveling on the input image, which is dependent on a translation operation. On a 2D grid, a translation is a straight-forward operation: the space is defined by two axes, and translating according to either is equivalent to an addition or substraction along one of these two axes. On a graph, there is no such clear translation operation as there is no notion of spatial proximity.

A property of the convolution operation is that is its equivalence to a product in Fourier space.

$$y = x \star f = \mathcal{F}^{-1}\{X \cdot F\} \tag{8}$$

Therefore, the input signal on the graph X can be convoluted to the filter F if it can be projected into Fourier space: the Fourier transform of a function living on a graph is its expansion in terms of the eigenfunctions of the Laplacian of the graph [22].

Given an undirected weighted graph of N nodes, and an adjacency matrix A, we define the normalized Laplacian of this graph to be:

$$L = I - D^{-1/2} A D^{-1/2} \tag{9}$$

With D is diagonal matrix holding the degree of each node, that is, the sum of the weights of all edges connected to it: $D_{ii} = \sum_j A_{ij}$

The Laplacian is a real symmetric matrix, with a set of orthonormal eigenvectors $U = \{\chi_l\}_{l=0,1,\ldots,N-1}$ and non-negative eigenvalues $\Lambda = \{0 = \lambda_0 < \lambda_1 \leq \lambda_2 \leq \cdots \leq \lambda_{N_1} = \lambda_{max}\}$.

A one-dimensional signal x on a graph is a set of values, each of them associated to a node of the graph. This signal refers to the function that lives on the graph that can be projected onto Fourier space. The following formula describes the lth frequency of the Fourier transform of the signal x:

$$\hat{x}(\lambda_l) = \chi_l^T x \qquad \mathcal{F}\{X\} = \chi^T X \tag{10}$$

In frequency space, the signal is a function of the eigenvalues: the signal goes from having N values each attributed to a node, to having N values, each attributed to an eigenmode. Each of these values is expressed as a function of the eigenvalues of the graph.

The inverse Fourier transform is defined as:

$$x = \sum_{l=0}^{N-1} \chi_l \hat{x}(\lambda_l) \qquad \mathcal{F}^{-1}\{X\} = \chi \hat{X} \tag{11}$$

Convolution, being a product in Fourier space is:

$$\mathcal{F}\{(x * f)\} = \sum_{l-1}^{N-1} \hat{x}(\lambda_l)\hat{f}(\lambda_l) \qquad \mathcal{F}\{X \star F\} = \hat{X} \times \hat{F} \tag{12}$$

Going back into vertex space, this leads to the following:

$$(x * f) = \sum_{l-1}^{N-1} \chi_l \hat{x}(\lambda_l)\hat{f}(\lambda_l) = \sum_{l-1}^{N-1} (\chi_l \hat{f}(\lambda_l)\chi_l^T)x \qquad X \star F = \chi(\hat{X} \times \hat{F}) \tag{13}$$

In a convolutional network, the optimal choice of filters being convoluted to the signal is what enables the estimation of the output. The learning that the convolutional network performs is the learning of appropriate filters in order for their convolution to the signal to lead to the desired output. Additionally, when working with graph convolutions, in frequency space, the filter is a function of the eigenvalues. Therefore, a graph convolutional network has the goal to learn an appropriate function of the graph eigenvalues, in order to produce the desired output after convolution with the signal.

In matrix format:

$$y = (x * f) = (U f(\text{diag}(\Lambda))U^T)x \tag{14}$$

A function applied on the Laplacian's eigenvalues can be expressed as a function applied on the Laplacian itself. Consequently, graph filtering is also possible in vertex space, without the need for eigendecomposition.

$$y = (x * f) = f(U\ \text{diag}(\Lambda)\ U^T)\ x = f(L)\ x \tag{15}$$

In conclusion, the convolution of a signal with filters on the graph can be expressed as the product of the signal with a function of the graph's Laplacian. The filters are defined by the function applied on the graph Laplacian and it is the Graph Convolutional Network's goal to approximate that function as well as possible in order to produce the intended outputs.

4.6.3 A toy example

The usage of the graph's adjacency information in the propagation of node states throughout the graph is a matter of design. The following example uses a toy graph and character states to illustrate the propagation of node states:

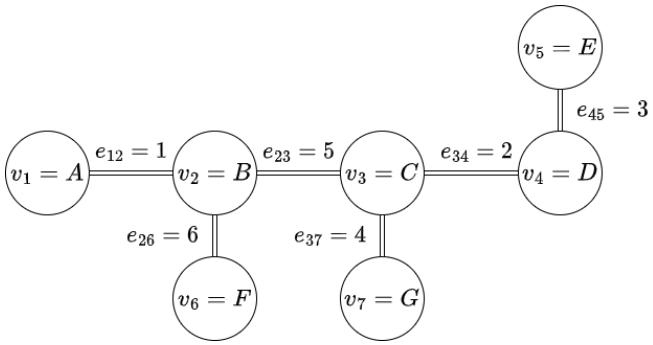


Figure 5: The toy graph considered

The Laplacian of a graph's adjacency matrix applied on a graph state propagates the information held in each node to its neighbors following the weights of the edges linking them (due to the adjacency matrix), while keeping a weighted version of its previous state (due to the degree matrix). The un-normalized Laplacian is defined as:

$$L = D - A \qquad \text{with} \qquad D_{ii} = \sum_j A_{ij} \tag{16}$$

In this example, the un-normalized version of the Laplacian is used for simplicity. We define X_0 as the initial graph state, and x^i as the state of node i.

Applied to the toy graph:

$$A = \begin{pmatrix} 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 1 & 0 & 5 & 0 & 0 & 0 & 0 \\ 0 & 5 & 0 & 2 & 0 & 6 & 4 \\ 0 & 0 & 2 & 0 & 3 & 0 & 0 \\ 0 & 0 & 0 & 3 & 0 & 0 & 0 \\ 0 & 0 & 6 & 0 & 0 & 0 & 0 \\ 0 & 0 & 4 & 0 & 0 & 0 & 0 \end{pmatrix} \qquad L = \begin{pmatrix} 1 & -1 & 0 & 0 & 0 & 0 & 0 \\ -1 & 12 & -5 & 0 & 0 & 0 & 0 \\ 0 & -5 & 11 & -2 & 0 & -6 & -4 \\ 0 & 0 & -2 & 5 & -3 & 0 & 0 \\ 0 & 0 & 0 & -3 & 3 & 0 & 0 \\ 0 & 0 & -6 & 0 & 0 & 6 & 0 \\ 0 & 0 & -4 & 0 & 0 & 0 & 4 \end{pmatrix}$$

$$X_0 = \begin{pmatrix} A \\ B \\ C \\ D \\ E \\ F \\ G \end{pmatrix} \qquad X_1 = LX_0 = \begin{pmatrix} A - B \\ 12B - A - 5C - 6F \\ 11C - 5B - 2D - 4G \\ 5D - 2C - 3E \\ 3E - 3D \\ 6F - 6B \\ 4G - 4C \end{pmatrix} \qquad X_2 = LX_1$$

$$X_2 = \begin{pmatrix} (A - B) - (12B - A - 5C - 6F) \\ 12(12B - A - 5C - 6F) - (A - B) - 5(11C - 5B - 2D - 4G) - 6(6F - 6B) \\ 11(11C - 5B - 2D - 4G) - 5(12B - A - 5C - 6F) - 2(5D - 2C - 3E) - 4(4G - 4C) \\ 5(5D - 2C - 3E) - 2(12B - A - 5C - 6F) - 3(3E - 3D) \\ 3(3E - 3D) - 3(5D - 2C - 3E) \\ 6(6F - 6B) - 6(12B - A - 5C - 6F) \\ 4(4G - 4C) - 4(11C - 5B - 2D - 4G) \end{pmatrix}$$

When applying the Laplacian to the power 1 on the initial state X_0, each node's state becomes a function of its old state and its direct neighbor's old states.

When applying the Laplacian to the power 2 on the initial state X_0, each node's state becomes a function of their state and their neighbors', after its own state has been altered by its neighbors and their neighbors by their own. For example, v_1 has no direct link to v_6, nor v_7. Yet after applying the Laplacian to the power 2, the characters characteristic of node v_6 and v_7 (respectively C and F) have propagated to v_1's state. They are both 2 hops away from v_1, after going over its direct neighbor v_2.

Additionally, it can also be seen that a node itself is two hops away from itself. When v_1 gets influenced by v_2, a step after v_2 has been influenced by its own neighbors, v_1's influence on v_2 travels back to v_1.

Should it be decided that the graph Convolutional network is a polynomial of the Laplacian, its output state would be a linear combination of the impact of the direct neighbors, neighbors of distance 2, neighbors of distance 3, up until the distance l_{max}.

$$Y = \sum_{l=1}^{l_{max}} L^l X_0 \tag{17}$$

Finally, the information in the Laplacian can be modulated by the usage of the filters defined in Section 4.6.2, that can not only perform structural alterations in each Laplacian to the power l, but influence the magnitude of the influence of each polynomial member, and as a result, give variable importance to neighbors l hops away.

4.6.4 Related work in learning graph convolution filters

After having explained the concepts of graph convolution and filtering mathematically and through a practical example, the chosen Graph Convolution formulation will be explained with respect to previous work in this field. As this work is an extension of the GCN architecture of [12], we remind the form of their architecture.

In [12], the encoder performs the following graph convolution operation:

$$X^{(l)} \approx \theta \tilde{D}^{-\frac{1}{2}} \tilde{A} \tilde{D}^{-\frac{1}{2}} X^{(l-1)} \tag{18}$$

This formulation is based on [14]. In this work, they note their GCN architecture fits into [2]'s formulation of localized spectral filters on graphs, under some approximations.

Indeed, in [2], they start with the consideration of the spectral filtering Formula 14. They state that the filtering of the Laplacian's eigenvectors can be can be done through a non-parametric filter, a set of N weights organized in a diagonal matrix: $f(\text{diag}(\Lambda)) = \text{diag}(W)$, with $W \in \mathbb{R}^N$. The disadvantages of such an approach is that there is no notion of localization in space, and there are as many weights to learn as the dimensionality of the data.

The polynomial formulation is introduced, which allows for localization in space. This localization refers to the consideration of nodes k hops away at each polynomial member, as seen in the practical example. Additionally, there is a reduction of the number of weights to learn: the number of weights to learn is now the order of the polynomial K ($W \in \mathbb{R}^K$). This means all eigenvalues share the same weight. Moreover, the Chebyshev Polynomial is used, due to its recursive formulation.

$$f(\text{diag}(\Lambda)) = \sum_{k=0}^{K} W_k T_k(diag(\Lambda)) \tag{19}$$

With $T_0(x) = 1 \qquad T_1(x) = x \qquad T_k(x) = 2xT_{k-1}(x) - T_{k-2}(x)$.

Then, in order to avoid eigen-decomposition, the filtering is moved to vertex space similarly to Formula 15.

$$y = f_W(L)x = \sum_{k=1}^{K-1} W_k T_k(\tilde{L})x \tag{20}$$

Additionally, in order to scale the eigenvalues to be in [-1, 1], the following rescaling is applied:

$$\tilde{L} = \frac{2}{\lambda_{max}} L - I_N \tag{21}$$

Formula 18 fits into Formula 20 under the following design choices:

1. $K = 2$
2. $\theta = W_0 = -W_1$
3. The scaling of the eigenvalues is done with the approximation $\lambda_{max} \approx 2$
4. The identity matrix is added to the adjacency matrix of the graph: $\tilde{A} = A + \mathbb{I}_N$, and $\tilde{D}$ is constructed from this new adjacency matrix.
5. This formula is generalized for a C-dimensional signal at each node, and for the learning of F filters, leading to $\theta \in \mathbb{R}^{CxF}$, instead of $\mathbb{R}$. The dimensions of the signals are in fact the number of nodes in the graph $C = N$.

In this work, the following design choices are made:

- We investigate the influence of the Chebyshev polynomial order on the encoder's performances. Design choices 1 and 2 of [14] are therefore dropped.
- No scaling as done in Formula 21 is performed.
- The generalization for C dimensional signal at each node, and F filters to be learned is kept. However, since we are still in the context of a polynomial, this results to having $\{W_k \in \mathbb{R}^{NxF}\}_{k=0,1,...,K-1}$ weights to learn.

In conclusion, the final formulation of the encoding layer is:

$$X^{(l)} = \text{ReLU}(\sum_{k=0}^{K} W_k^{(l)} T_k(L) X^{(l-1)}) \tag{22}$$

A Rectified Linear Unit (ReLU) activation function is added following Graph Convolution. The encoder having only a single layer, $X^{(l-1)}$ is the input to the network and $X^{(l)}$ is the latent space.

4.7 Decoder

The decoder decompressed the latent space in order to produce the estimation of the FC matrix. This latent space is of size $(N, 32)$, which can be interpreted as 32 graph states.

As a first step, a trainable $(32, 32)$ layer is applied onto the latent space.

$$X^{(l+1)} = X^{(l)} W_{(l+1)} \tag{23}$$

Afterwards, we remind that, as described in Formula 1, the FC matrix is the result of the Pearson Correlation of a $(N, 1200)$ matrix of timeseries. In order to produce a square (N, N) matrix that is symmetric positive semi-definite, the layer performs the multiplication of the latent space with its transpose.

$$Y = X^{(l+1)} (X^{(l+1)})^T \tag{24}$$

Here, it is not expected to be SPD because $(N, 32)$ matrices may have colinear columns. However, while we expect it to be solely symmetric positive semi-definite, due to problems of computational precision, the output FC matrix may have negative eigenvalues very close to 0, instead of 0. To address this issue, when using this loss (this doesn't apply to networks trained with the MSE as a loss), the identity matrix is added to the output, leading to SPD matrices (with the minimal eigenvalue slightly smaller than 1), enabling the usage of the AIRM on SPDs.

Compared to the work of [12], a layer of learning in the decoder followed by a *tanh* activation function has been removed. More importantly, the final *ReLU* has been removed in order to obtain a Symmetric Positive Semi-Definite matrix.

5 Results

5.1 Values describing the dataset

As a first step in understanding the data being used, we wanted to investigate the distances that separate the subjects from each other. In order to do so the average Pearson Correlation of SC matrices between pairs of subjects is computed. The same is done with their FC matrices.

As a reminder, in order to be able to use the AIRM on SPD matrices, the FC matrices were not clipped when using this loss. However, when using the MSE, both the clipped FC (cFC) and unclipped FC (uFC) matrix were used.

For $N = 68$:

- Average inter-subject correlation of SC matrices is **0.913**
- Average inter-subject correlation of uFC matrices is **0.517**
- Average inter-subject correlation of cFC matrices is **0.528**

For $N = 200$:

- Average inter-subject correlation of SC matrices is **0.803**
- Average inter-subject correlation of uFC matrices is **0.375**
- Average inter-subject correlation of cFC matrices is **0.388**

Clipped FC matrices are only slightly more correlated than unclipped FC matrices. However, the importance behind these values is understanding that we are starting from very correlated data, and trying to construct outputs that are much less correlated. Small differences in structure should somehow amplify and create big functional differences.

For $N = 68$:

- Average inter-subject MSE of cFC matrices is **0.0775**
- Average inter-subject MSE of uFC matrices is **0.0931**
- Average inter-subject AIRM on SPDs of uFC matrices is **16.0310**

For $N = 200$:

- Average inter-subject MSE of cFC matrices is **0.0581**
- Average inter-subject MSE of uFC matrices is **0.0694**
- Average inter-subject AIRM on SPDs of uFC matrices is **31.6064**

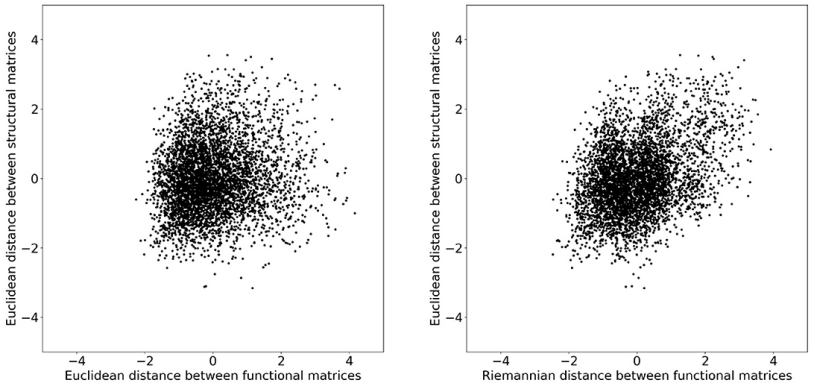
5.2 Reference estimator

The results of the reference estimators are presented in the table below.

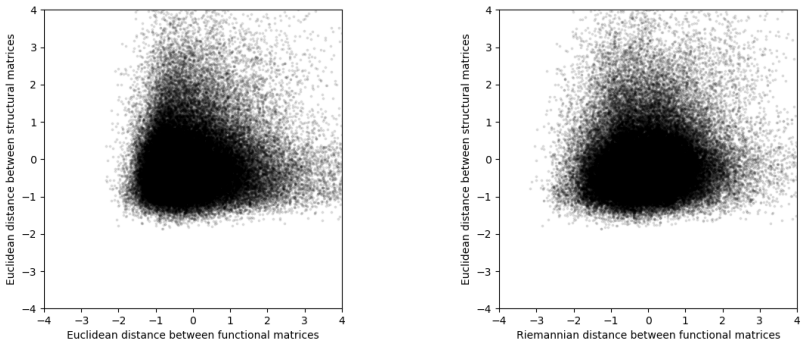
	Metric	N = 68		N = 200	
		uFC	cFC	uFC	cFC
Frechet Mean with Euclidean distance	MSE	0.0465	0.0387	0.0347	0.029
	Pearson Correlation	0.7194	0.7272	0.6125	0.623
Frechet Mean with AIRM on SPDs	AIRM on SPDs	10.80	-	21.08	-
	Pearson Correlation	0.7062	-	0.571	-

5.3 Riemannian distance, initial motivation

In [5], they investigated the possible relationship between the distance between two subjects' SC, and the distance between these subjects' FC.



(a) For 100 subjects



(b) For 383 subjects

Figure 6: Figure (a) results from an analysis in [5] that aims to investigate whether there is a link between the structural proximity of a pair of subjects and their functional proximity. Figure (b) is a reproduction of this analysis for a bigger number of subjects.
On subfigures on the left, the Euclidean distance between a subject pair's SCs is plotted against the Euclidean distance between their FCs.
On subfigures on the right, the Euclidean distance between a subject pair's SCs is plotted against the Riemannian distance between their FCs.

The findings of [5] are that subjects that have similar structural matrices also have similar functional matrices, but only when FC distances are measured using the Riemannian metric: this statement was motivated by the computation of the coefficient of determination R^2 (Pearson Correlation squared) between x and y values of both graphs. It was found that the coefficient of determination was higher when using the Riemannian metric between pairs of subjects ($R^2 = 0.016$ for the FC Euclidean distance vs $R^2 = 0.135$ for the FC Riemannian distance). On 383 subjects, this difference is much less noticeable: $R^2 = 0.001$ for the FC Euclidean distance and $R^2 = 0.003$ for the FC Riemannian distance. This invalidates the findings that supported a link between structural proximity in Euclidean terms leading to functional proximity in Riemannian terms.

5.4 Chebyshev Polynomial order

Each GCN was trained for 200 epochs using cross-validation over 10 folds. The means and standard deviations visible on the figures are computed across these 10 folds. This small number of epochs has been chosen after observation of the evolution of the loss for higher epochs, and the realisation that after this point, it stayed rather constant. Chebyshev polynomial orders ranging from $K = 1$ to $K = 7$ were tested for.

5.4.1 MSE - Clipped FC

Looking at Figure 7, for both $N = 68$ and $N = 200$, similar tendencies can be observed. First, increasing the order of the polynomial does not improve the estimation of the FC matrix. Second, given the units of the y axis, this approximation is very close to the approximation made by the reference estimator in terms of performance. This means the prediction of the FC matrix is approximately as close as the prediction of the reference estimator. This opened up the question of whether the network was learning the same minimizing FC matrix as the reference estimator.

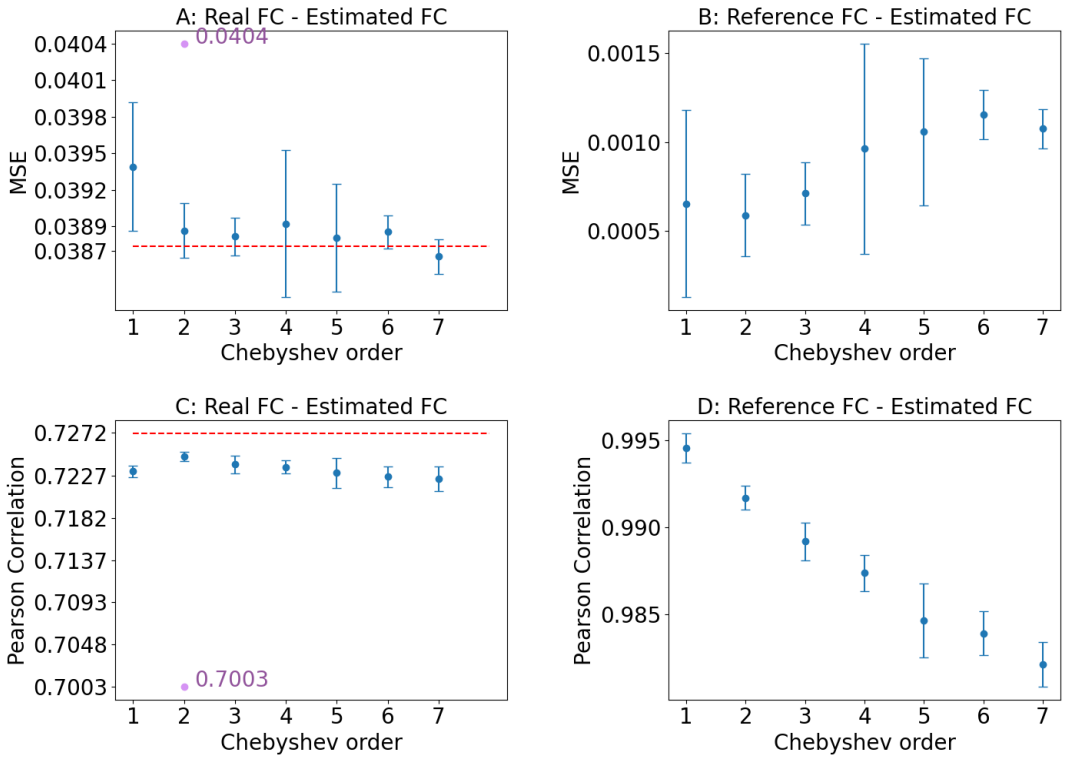
Subfigure B compares the estimations of the network (for all points in the dataset) to the reference estimator. MSEs range in [0, 0.0015], leading us to conclude the network is indeed learning a FC matrix very close to the one computed by the reference estimator.

The estimation of FC matrices done by this network yields MSEs in the range [0.0386, 0.0394], which is a slight improvement in comparison to [12].

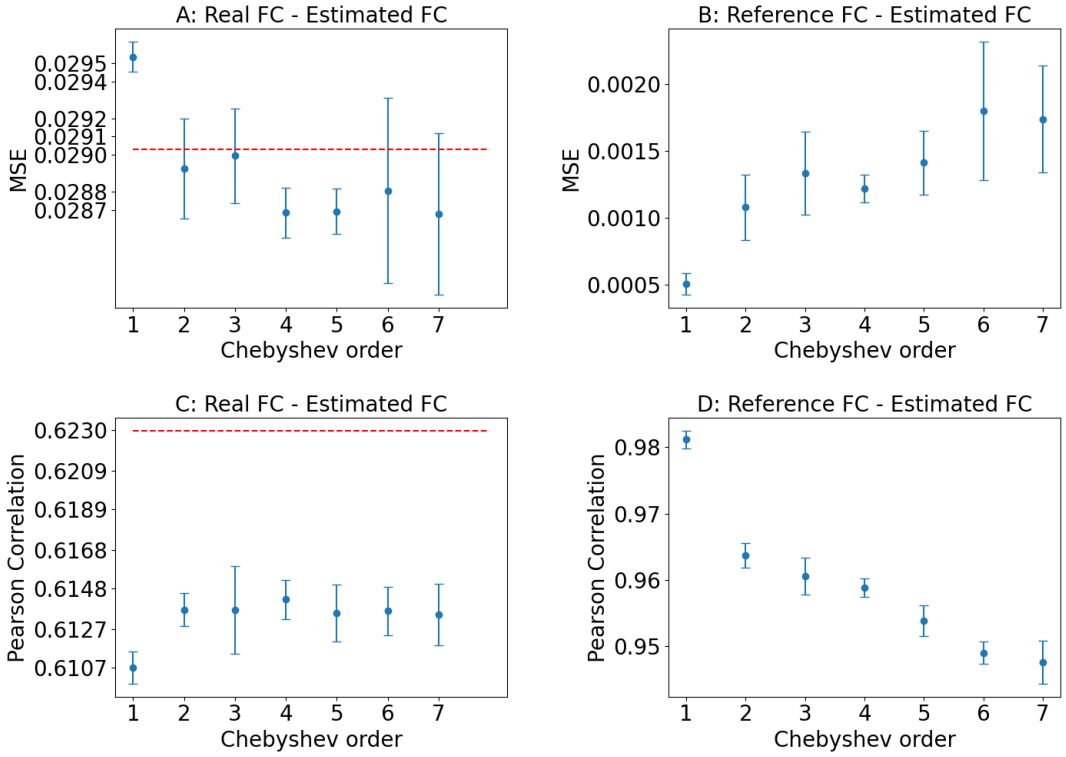
Pearson Correlation follows patterns similar to the MSE. The network estimates FC matrices with Pearson Correlation nearing those with the reference estimator. The improvement with respect to [12] is more noticeable when observing the Pearson Correlation.

Moreover, order 1 of the Chebyshev polynomial does not use the SC's Laplacian in any way, since it is raised to the power 0.

For $N = 200$, looking at the performances of the first order, in terms of MSE, it looks as if it is exactly the reference estimator's estimation that the network yields. On figure b. D, the average of MSEs between the reference estimator and network predictions is 0 for $N = 200$ and almost 0 for $N = 68$. Additionally, their Pearson Correlation reaches 1. If admitting that the network is learning the FC mean, it is not a surprising result, as the only member of the Chebyshev order is the Laplacian to the power 0, the input is the identity matrix, and a single weight set and biases need to be learned in the encoder: the network has full freedom in order to adjust weights and biases to create the latent space that would lead to the FC mean. For higher orders, the weights need to compensate for the Laplacian's structure in order to produce the latent space that will lead to FC mean.



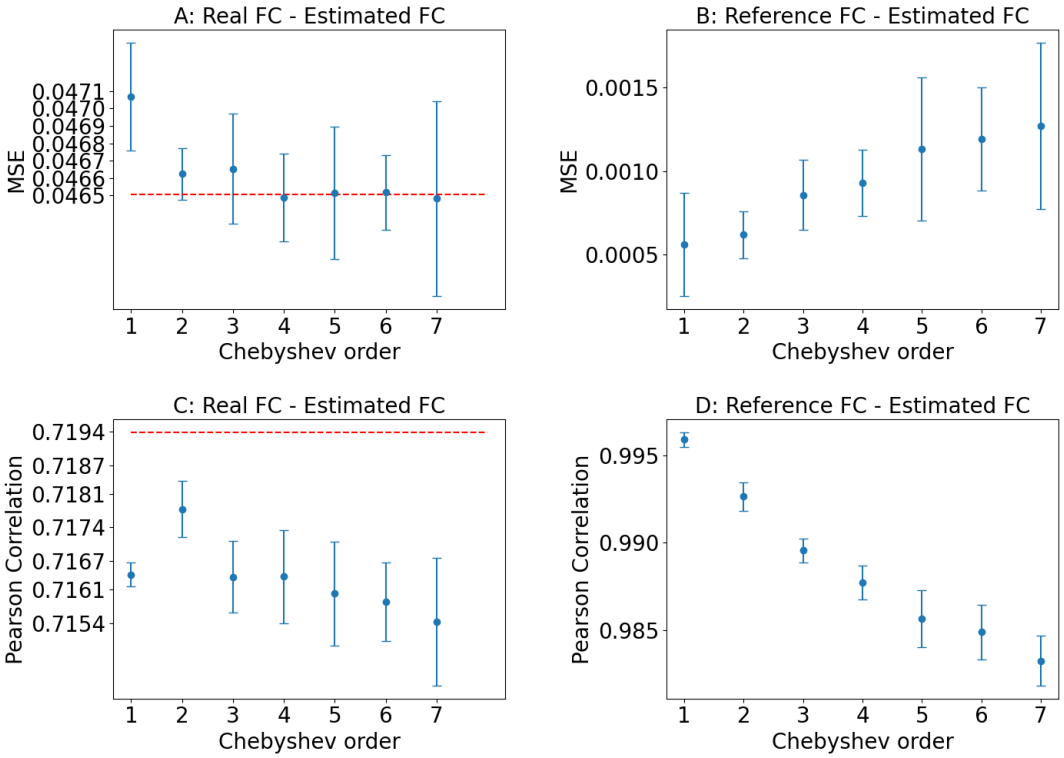
(a) $N = 68$



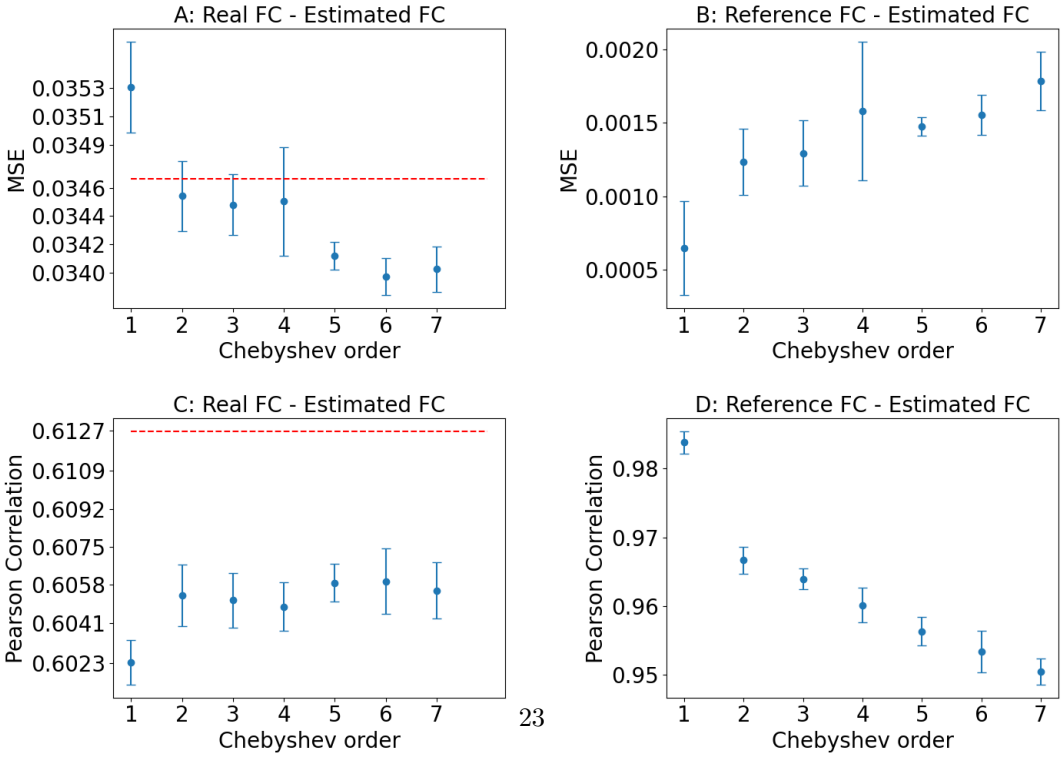
(b) $N = 200$

Figure 7:
(A) Mean (+ standard deviation) MSE between the predicted FC matrices and the real FC matrices. The red line corresponds to the reference estimator. In purple, we can see the results of [12].
(B) Mean (+ standard deviation) MSE between the predicted FC matrices and the reference estimator.
Subfigures (C) and (D) are the same as (A) and (B) but for the Pearson Correlation.

5.4.2 MSE - Unclipped FC



(a) $N = 68$



(b) $N = 200$

Figure 8:
(A) Mean (+ standard deviation) MSE between the predicted FC matrices and the real FC matrices. The red line corresponds to the reference estimator. In purple, we can see the results of [12].
(B) Mean (+ standard deviation) MSE between the predicted FC matrices and the reference estimator.
Subfigures (C) and (D) are the same as (A) and (B) but for the Pearson Correlation.

The usage of the clipped FC enabled comparison to the work of [12]. However, in order to compare to the AIRM on SPDs loss, the same pre-processing of data needs to be applied, and that includes bypassing the clipping step.
Not clipping the FC matrices yields results with the same tendencies as with the clipped FCs. For example, looking at Figure 8.a for $N = 68$, observing the Pearson Correlation (subfigures C and D):

- On figure C, the reference estimator's performance is displayed at 0.7194.
- On figure C, for the first order Chebyshev polynomial, Pearson Correlation between predicted FCs and real FCs is 0.7160
- On figure C, for the first order Chebyshev polynomial, Pearson Correlation between predicted FCs and the reference estimator is 0.9960: the reference's estimation and the network's estimation are very similarly correlated to the real FC matrices, and almost fully correlated to each other.

5.4.3 AIRM on SPDs - Unclipped FC

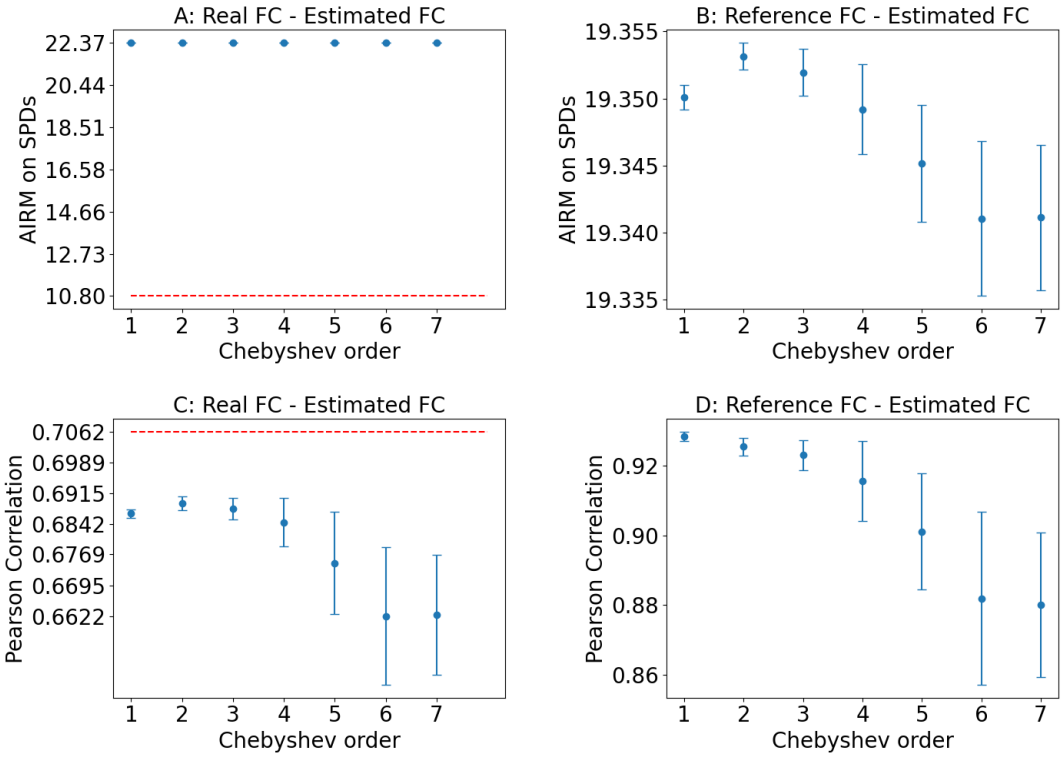


Figure 9: For $N = 68$
(A) Mean (+ standard deviation) AIRM on SPDs between the predicted FC matrices and the real FC matrices.
(B) Mean (+ standard deviation) AIRM on SPDs between the predicted FC matrices and the reference estimator.
Subfigures (C) and (D) are the same as (A) and (B) but for the Pearson Correlation.

At the time of this report, the corresponding figure for $N = 200$ could not be produced.

When using the AIRM on SPDs, there does not seem to be a phenomenon of mean estimation by the network: the distance between real FC matrices and predictions is bigger (around 22.53 for all orders) than the distance between real FC matrices and the reference estimator (10.80).

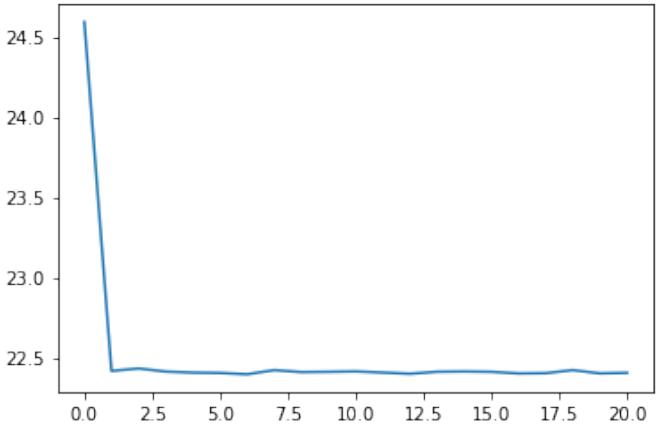


Figure 10: Evolution of the loss over all epochs of training for the first fold of the GCN trained with the AIRM on SPDs as loss, for Chebyshev polynomial order 4, and dataset $N = 68$

When observing more in detail the evolution of loss for any example of network driven by the AIRM on SPD, we can observe that the loss stays the same, after an initial drop in value. This could be indicative of a local minimum being reached.

Nevertheless, the estimations of the network may be far from the real FC matrices, they seem to have the correlations near those of the reference estimator (above 0.67 for the first 5 orders), and these predictions seem to be very correlated to the reference estimator (above 0.91 for the first 5 orders).

6 Discussion

We have extended the work of [12] and altered the GCN architecture in multiple ways. First, using separate weight sets for the second-order Chebyshev polynomial leads to small improvements in performances as quantified by the MSE, and a non-negligible increase in Pearson correlation. Second, this separation of weight sets enabled the usage of higher Chebyshev polynomial orders. Increasing the order of the Chebyshev polynomial does not increase performances. However, through the observation of the first order Chebyshev Polynomial that makes no use of SC information and still yields good results, we found out that the estimations of the network were in fact very close to the FC matrix estimated by the reference estimator, both in MSE and Pearson Correlation, leading to the conclusion that the network is converging towards the mean. Moreover, using the Affine Invariant Riemannian Metric on Symmetric Positive Definite matrices does not lead to an estimation of the Frechet Mean, but it does output estimations that are correlated to the real functional connectivity almost as much as the mean's correlation to real functional connectivity. The current architecture does not seem to be appropriate for this loss, as the loss seems to reach a local minimal value early on in the learning. Finally, the findings of [5] stating there is a link between the structural proximity of two subjects in Euclidean terms and their functional proximity in Riemannian terms has been invalidated when performing their same analysis on a bigger dataset.

Following this, we will reflect on the design choices that were made and propose future improvements.

Neural network input The relevance of Graph Neural Networks lies in the fact that node states change as some function of its neighbors. However, this actually implies that a node should have an initial state, be it a meaningful activation level, or a set of attributes. While the identity matrix can be interpreted as a set of networks with a single node activated, it has no physiological basis. We wonder what initial state should be given to the brain network. As we are hoping to predict correlations of activity at a rest state, perhaps cortical network modes can be identified across all subjects.

Processing pipelines Although, this work is focused on improving the mapping between structure and function, it is limited by the input and outputs it works with. Entire fields of research are dedicated to dMRI and fMRI processing and the pipelines of connectivity matrices aim to follow the best practices established in these fields. However, considering the many steps that separate the original MRI acquisitions to the I/O of the network, we have no way of validating that these matrices hold inter-subject differences that are consistent with the original MRI acquisitions. Are these differences a result of the variability between subjects, or artefacts of the processing pipelines?

Latent space size The latent space size was defined at an early stage of the project, using the original architecture of [12] and finding a trade-off between performances and size. It was found that for sizes above 32, the increase in performance was negligible compared to the added computational complexity. The latent size is therefore motivated by purely computational concerns and doesn't have any physiological motivation. Is it possible to have a basis of graph states that can help to decompose known network modes?

Bias towards the mean Since we found that the network is biased towards the mean, future work may be oriented towards the current task at hand, with the input and output demeaned. However, demeaning an adjacency matrix seems to be non-straightforward task, as it may lead to negative adjacency and would alter the dynamics of signal propagation on the graph.

Frequency analysis While graph filtering originally occurs in frequency space, this operation was moved to vertex space, due to computational concerns. Given the size of the networks we work with, this is not a concern anymore. Perhaps, it could be interesting to compare the eigenmodes of the subjects' SC matrices.

Additional thoughts It seems that the project in its current state does not harness the power of GCNs because no interpretable information is travelling through cortical areas and modulating the propagation of information with increasing Chebyshev orders may only be investigated once this fact has been addressed. For example, considering a GCN on its own, outside of the autoencoder architecture, in order to compute a new graph state from a sensible initial graph state might be of interest. Of course, this may not lead to an application to structure-function mapping, or it may, if this first GCN output is an intermediate step in the functional connectivity estimation.

References

[1] D Cordes et al. "Frequencies contributing to functional connectivity in the cerebral cortex in "resting-state" data". en. In: *AJNR Am J Neuroradiol* 22.7 (Aug. 2001), pp. 1326–1333.

[2] Michaël Defferrard, Xavier Bresson, and Pierre Vandergheynst. "Convolutional Neural Networks on Graphs with Fast Localized Spectral Filtering". In: *Advances in Neural Information Processing Systems 29 (2016)* (2016). DOI: 10.48550/ARXIV.1606.09375. URL: https://arxiv.org/abs/1606.09375.

[3] Flavio Dell'Acqua and J-Donald Tournier. "Modelling white matter with spherical deconvolution: How and why?" en. In: *NMR Biomed* 32.4 (Aug. 2018), e3945.

[4] Rahul S. Desikan et al. "An automated labeling system for subdividing the human cerebral cortex on MRI scans into gyral based regions of interest". In: *NeuroImage* 31.3 (2006), pp. 968–980. ISSN: 1053-8119. DOI: https://doi.org/10.1016/j.neuroimage.2006.01.021. URL: https://www.sciencedirect.com/science/article/pii/S1053811906000437.

[5] Samuel Deslauriers-Gauthier et al. "A Riemannian Revisiting of Structure–Function Mapping Based on Eigenmodes". In: *Frontiers in Neuroimaging* 1 (2022). ISSN: 2813-1193. DOI: 10.3389/fnimg.2022.850266. URL: https://www.frontiersin.org/articles/10.3389/fnimg.2022.850266.

[6] Samuel Deslauriers-Gauthier et al. "A unified framework for multimodal structure–function mapping based on eigenmodes". In: *Medical Image Analysis* 66 (2020), p. 101799. ISSN: 1361-8415. DOI: https://doi.org/10.1016/j.media.2020.101799. URL: https://www.sciencedirect.com/science/article/pii/S1361841520301638.

[7] David A. Feinberg et al. "Multiplexed Echo Planar Imaging for Sub-Second Whole Brain FMRI and Fast Diffusion Imaging". In: *PLOS ONE* 5.12 (Dec. 2010), pp. 1–11. DOI: 10.1371/journal.pone.0015710. URL: https://doi.org/10.1371/journal.pone.0015710.

[8] Simone Fiori. "Learning the Fréchet Mean over the Manifold of Symmetric Positive-Definite Matrices". In: *Cognitive Computation* 1.4 (Oct. 2009), p. 279. ISSN: 1866-9964. DOI: 10.1007/s12559-009-9026-7. URL: https://doi.org/10.1007/s12559-009-9026-7.

[9] Bruce Fischl. "FreeSurfer". en. In: *Neuroimage* 62.2 (Jan. 2012), pp. 774–781.

[10] Bruce Fischl, Martin I. Sereno, and Anders M. Dale. "Cortical Surface-Based Analysis: II: Inflation, Flattening, and a Surface-Based Coordinate System". In: *NeuroImage* 9.2 (1999), pp. 195–207. ISSN: 1053-8119. DOI: https://doi.org/10.1006/nimg.1998.0396. URL: https://www.sciencedirect.com/science/article/pii/S1053811998903962.

[11] Ben Jeurissen et al. "Multi-tissue constrained spherical deconvolution for improved analysis of multi-shell diffusion MRI data". en. In: *Neuroimage* 103 (Aug. 2014), pp. 411–426.

[12] Yang Ji, Samuel Deslauriers-Gauthier, and Rachid Deriche. "Structure-Function Mapping via Graph Neural Networks". In: *Machine Learning in Clinical Neuroimaging. 4th International Workshop, MLCN 2021, Held in Conjunction with MICCAI 2021, Strasbourg, France, September 27, 2021, Proceedings*. Ed. by Ahmed Abdulkadir et al. Vol. 13001. Lecture Notes in Computer Science. Springer, 2021, pp. 135–144. DOI: 10.1007/978-3-030-87586-2_14. URL: https://hal.archives-ouvertes.fr/hal-03352550.

[13] D.K. Jones. *Diffusion MRI*. Oxford University Press, 2010. ISBN: 9780199708703. URL: https://books.google.ch/books?id=dbZCMePD52AC.

[14] Thomas N. Kipf and Max Welling. *Semi-Supervised Classification with Graph Convolutional Networks*. 2017. arXiv: 1609.02907 [cs.LG].

[15] Mikhail Milchenko and Daniel Marcus. "Obscuring Surface Anatomy in Volumetric Imaging Data". In: *Neuroinformatics* 11.1 (Jan. 2013), pp. 65–75.

[16] Nina Miolane et al. “Geomstats: A Python Package for Riemannian Geometry in Machine Learning”. In: *Journal of Machine Learning Research* 21.223 (2020), pp. 1–9. URL: http://jmlr.org/papers/v21/19-027.html.

[17] Nina Miolane et al. *geomstats/geomstats: Geomstats v2.5.0*. 2022. DOI: 10.5281/ZENODO.6478729. URL: https://zenodo.org/record/6478729.

[18] Steen Moeller et al. “Multiband multislice GE-EPI at 7 tesla, with 16-fold acceleration using partial parallel imaging with application to high spatial and temporal whole-brain fMRI”. In: *Magnetic Resonance in Medicine* 63.5 (2010), pp. 1144–1153. DOI: https://doi.org/10.1002/mrm.22361. eprint: https://onlinelibrary.wiley.com/doi/pdf/10.1002/mrm.22361. URL: https://onlinelibrary.wiley.com/doi/abs/10.1002/mrm.22361.

[19] Tabinda Sarwar, Kotagiri Ramamohanarao, and Andrew Zalesky. “Mapping connectomes with diffusion MRI: deterministic or probabilistic tractography?” In: *Magnetic Resonance in Medicine* 81.2 (2019), pp. 1368–1384. DOI: https://doi.org/10.1002/mrm.27471. eprint: https://onlinelibrary.wiley.com/doi/pdf/10.1002/mrm.27471. URL: https://onlinelibrary.wiley.com/doi/abs/10.1002/mrm.27471.

[20] Alexander Schaefer et al. “Local-Global Parcellation of the Human Cerebral Cortex from Intrinsic Functional Connectivity MRI”. en. In: *Cereb Cortex* 28.9 (Sept. 2018), pp. 3095–3114.

[21] Kawin Setsompop et al. “Blipped-controlled aliasing in parallel imaging for simultaneous multislice echo planar imaging with reduced g-factor penalty”. In: *Magnetic Resonance in Medicine* 67.5 (2012), pp. 1210–1224. DOI: https://doi.org/10.1002/mrm.23097. eprint: https://onlinelibrary.wiley.com/doi/pdf/10.1002/mrm.23097. URL: https://onlinelibrary.wiley.com/doi/abs/10.1002/mrm.23097.

[22] David I Shuman, Benjamin Ricaud, and Pierre Vandergheynst. “Vertex-frequency analysis on graphs”. In: *Applied and Computational Harmonic Analysis* 40.2 (2016), pp. 260–291. ISSN: 1063-5203. DOI: https://doi.org/10.1016/j.acha.2015.02.005. URL: https://www.sciencedirect.com/science/article/pii/S1063520315000214.

[23] S. N. Sotiropoulos et al. “Effects of image reconstruction on fiber orientation mapping from multichannel diffusion MRI: Reducing the noise floor using SENSE”. In: *Magnetic Resonance in Medicine* 70.6 (2013), pp. 1682–1689. DOI: https://doi.org/10.1002/mrm.24623. eprint: https://onlinelibrary.wiley.com/doi/pdf/10.1002/mrm.24623. URL: https://onlinelibrary.wiley.com/doi/abs/10.1002/mrm.24623.

[24] J-Donald Tournier, Fernando Calamante, and Alan Connelly. “Robust determination of the fibre orientation distribution in diffusion MRI: Non-negativity constrained super-resolved spherical deconvolution”. In: *NeuroImage* 35.4 (2007), pp. 1459–1472. ISSN: 1053-8119. DOI: https://doi.org/10.1016/j.neuroimage.2007.02.016. URL: https://www.sciencedirect.com/science/article/pii/S1053811907001243.

[25] J-Donald Tournier, Fernando Calamante, and Alan Connelly. “Robust determination of the fibre orientation distribution in diffusion MRI: non-negativity constrained super-resolved spherical deconvolution”. en. In: *Neuroimage* 35.4 (Feb. 2007), pp. 1459–1472.

[26] J-Donald Tournier et al. “MRtrix3: A fast, flexible and open software framework for medical image processing and visualisation”. In: *NeuroImage* 202 (2019), p. 116137. ISSN: 1053-8119. DOI: https://doi.org/10.1016/j.neuroimage.2019.116137. URL: https://www.sciencedirect.com/science/article/pii/S1053811919307281.

[27] J.-Donald Tournier, Fernando Calamante, and Alan Connelly. “Determination of the appropriate b value and number of gradient directions for high-angular-resolution diffusion-weighted imaging”. In: *NMR in Biomedicine* 26.12 (2013), pp. 1775–1786. DOI: https://doi.org/10.1002/nbm.3017. eprint: https://analyticalsciencejournals.onlinelibrary.wiley.com/doi/pdf/10.1002/nbm.3017. URL: https://analyticalsciencejournals.onlinelibrary.wiley.com/doi/abs/10.1002/nbm.3017.

[28] Jacques-Donald Tournier, F. Calamante, and Alan Connelly. "Improved probabilistic streamlines tractography by 2nd order integration over fibre orientation distributions". In: *Proc. Intl. Soc. Mag. Reson. Med. (ISMRM)* 18 (Jan. 2010).

[29] Martijn P. van den Heuvel and Hilleke E. Hulshoff Pol. "Exploring the brain network: A review on resting-state fMRI functional connectivity". In: *European Neuropsychopharmacology* 20.8 (2010), pp. 519–534. ISSN: 0924-977X. DOI: https://doi.org/10.1016/j.euroneuro.2010.03.008. URL: https://www.sciencedirect.com/science/article/pii/S0924977X10000684.

[30] D C Van Essen et al. "The Human Connectome Project: a data acquisition perspective". en. In: *Neuroimage* 62.4 (Feb. 2012), pp. 2222–2231.

[31] Junqian Xu et al. "Highly accelerated whole brain imaging using aligned-blipped-controlled-aliasing multiband EPI". In: *Proceedings of the 20th Annual Meeting of ISMRM*. Vol. 2306. 2012.

[32] Chun-Hung Yeh et al. "Mapping Structural Connectivity Using Diffusion MRI: Challenges and Opportunities". In: *Journal of Magnetic Resonance Imaging* 53.6 (2021), pp. 1666–1682. DOI: https://doi.org/10.1002/jmri.27188. eprint: https://onlinelibrary.wiley.com/doi/pdf/10.1002/jmri.27188. URL: https://onlinelibrary.wiley.com/doi/abs/10.1002/jmri.27188.