Danmarks Tekniske Universitet

(02466) Project work - Bachelor of Artificial Intelligence and Data

Weighted Grassmann clustering for assessing dynamic functional coherence

Alexander Wittrup
s224196

Bertram Nyvold Larsen
s224194

Simon Kjølbye Kristensen
s224203

23. september 2024

1 Abstract

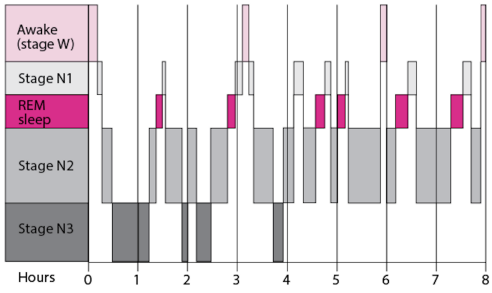
This project investigates the use of Grassmann Clustering (GC) and Weighted Grassmann Clustering (WGC), to analyzing sleep data from fMRI measurements. For both methods, we look at models with different number of clusters. An extended version of Leading eigenvector dynamics analysis (LEiDA) is used to extract the eigenvectors and eigenvalues from the raw fMRI data. GC demonstrated a better performance over traditional methods like Euclidean K-Means, particularly in high dimensional synthetic data scenarios, achieving a higher Normalized Mutual Information (NMI) scores. From GC on real fMRI data, it was found, that although the NMI scores increases with the number of cluster, there were some local spikes in NMI present for specific number of clusters, namely for models with $K = 3$ and $K = 6$. In the report we further provide a detailed description of the found centroids to explain some of the spikes in NMI score. The WGC algorithm generally resulted in lower NMI scores compared to the GC algorithm, which was disappointing given the expected advantages of WGC. This discrepancy could potentially be attributed to the highly subjectively labeled data.

Contents

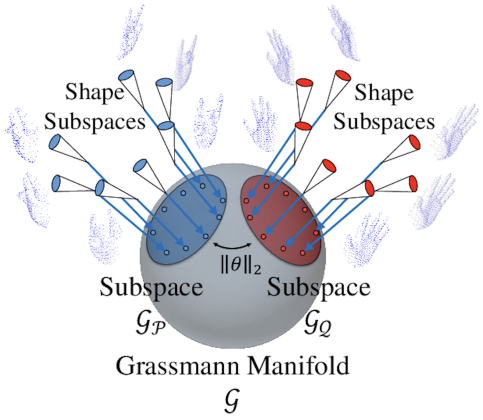
2 Introduction

The complicated dynamics of brain connectivity play a crucial role in shaping cognitive functions and understanding neurological disorders. Conventional approaches often rely on assessing Pearson correlation coefficients within fixed time-windows to capture these dynamics. However, recent advancements, such as the Leading Eigenvector Dynamics Analysis (LEiDA) model, have introduced innovative methods based on frame-wise phase coherence maps derived from the Hilbert transform.
In the field of neuroscience, sleep is an incredibly interesting area to study. Until recent times, sleep has remained very poorly understood, even though most people spend around a third of their lives sleeping. Sleep is a fundamental physiological process characterized by distinct stages marked by varying levels of neural activity and consciousness. Sleep has been shown to be of great importance for human health and functioning. Accurate classification of these stages is imperative for understanding sleep patterns, diagnosing sleep disorders, and unraveling the underlying mechanisms governing sleep-wake transitions.

Motivated by the limitations of current practices, particularly the tendency to only consider the leading eigenvector for clustering, this research project seeks to explore a more comprehensive approach. By leveraging Grassmann clustering, a robust mathematical technique capable of clustering subspaces, we aim to enhance our understanding of time-varying brain connectivity. Here, the subspaces are constructed from the two leading eigenvectors extracted from the data matrix. Furthermore, the incorporation of eigenvalue weighting allows us to capture the varying importance of eigenvectors. This is proposed to provide a more nuanced representation of brain dynamics.



(a) Sleep stages Gennaro [2023]



(b) Grassmann Manifold Yataka and Fukui [2017]

The purpose of this research project is twofold. Firstly, we aim to apply Grassmann clustering, to analyze neural phase sleep data comprehensively. By clustering subspaces rather than individual data points, we aim to reveal distinct connectivity patterns within the brain. Secondly, we seek to extend this model by introducing eigenvalue weighting, which hopefully enhances the method by considering the relative significance of each eigenvector. Through this research, we aspire to contribute to the advancement of neuroimaging analysis techniques. By providing a more nuanced understanding of time-varying brain connectivity, our findings may have implications for various fields, including neuroscience research and clinical diagnostics.

Introduced by Gruber and Theis in 2006, Grassmann clustering is a mathematically advanced approach capable of clustering subspaces. Unlike traditional clustering methods that operate on individual data points, Grassmann clustering considers entire subspaces, making it particularly well-suited for analyzing high-dimensional data with more than one component and underlying structural dependencies. Grassmann clustering has found applications across various domains, including computer vision, robotics, and signal processing. Its ability to capture the structure of data has made it well suited for researchers seeking to uncover underlying patterns in complex datasets. By the time of writing, the original article on Grassman clustering, authored by Peter Gruber and Fabian J. Theis, have received 37 citations.
In the context of neuroscience, some researchers have already turned to Grassmann clustering as a promising tool for unraveling the complexities of brain connectivity dynamics. By employing this method to cluster subspaces derived from neural phase data, researchers have improved the ability to successfully identify distinct patterns of connectivity within the brain and enhance our understanding of both normal brain function and dysfunction.

2.1 Research questions

Below, we outline the research questions for this project, along with the proposed course of action to address them.

How does Grassmann Clustering compare to traditional methods? (1)
Is there an optimal number of clusters, and how does this relate to dataset characteristics? (2)
What impact do eigenvalues have on clustering results when used as weights? (3)

Firstly (1), we will comprehensively understand and implement the Grassmann Clustering method. This includes its theoretical foundations and algorithmic structure, enabling us to effectively apply it in our analysis. Subsequently, we will apply the Grassmann Clustering technique to both synthetic and real datasets, conducting a thorough evaluation of the obtained results. This evaluation will serve as a benchmark for assessing its performance. Next, we will compare the outcomes derived from Grassmann Clustering with those obtained from traditional clustering methods. This comparative analysis will shed light on the relative strengths and limitations of Grassmann Clustering.
Moving forward (2), we will locate the optimal number of clusters by experimenting with different cluster configurations. This will help us understand how the number of clusters relates to dataset characteristics and facilitate the selection of an optimal clustering configuration.
Finally (3), we will introduce the weighting of eigenvectors in the Grassmann Clustering model, evaluating how this weighting strategy influence clustering outcomes and the overall quality of results.

3 Methods

Functional magnetic resonance imaging time-series data (fMRI) was gathered from brain activity during sleep which is later described in section 4. But briefly, the data consists of 3D NIfTI (.nii) images taken over time. Due to the nature of how these scans were created, they include a lot of data with no information, e.g. that a brain isn't cube-shaped, and a lot of the data consists of the air around the scanned person's head, for this reason, an atlas is used to partition these scans into specific brain regions, also called parcellation. The real data consists of 286 images with 116 regions or dimensions p each. Each region therefore contains a time-varying periodic wave which is hereafter referred to as $s_{\text{raw}}(t)$. Although this data could be analyzed, clustering wouldn't yield good results, since the mean strength of the signals can vary wildly from region to region, and how the correlation between areas of the brain can spike when one area is being activated, which is avoided when using phase coherence. It is something entirely separate to analyze areas of the brain by themselves, and how they change over time, and the interest of our report is instead focused on how the areas covariate i.e. the connectivity, which can be estimated with clustering.

3.1 Leading Eigenvector Dynamics Analysis

Leading eigenvector dynamics analysis (LEiDA) computes the leading eigenvector from the instantaneous interregional phase coherence pattern, also called the coherence map. This phase series can be estimated by calculating the phase using the Hilbert transform. However, later for eq.(4) to work as intended it is necessary that the signal received is a narrowband signal. For that reason, a 2nd-order Butterworth filter is defined from the following lower and upper bounds:

$$f_s = \frac{1}{2.1}, \quad f_{Nyquist} = \frac{f_s}{2} \tag{1}$$

$$\text{low} = \frac{0.008}{f_{Nyquist}}, \quad \text{high} = \frac{0.09}{f_{Nyquist}} \tag{2}$$

Where f_s is the sampling rate of the fMRI data, the new signal filtered with these bounds will be referred to as $s(t)$. Going back to the Hilbert transform, it has a rather simple interpretation: rotating the coefficients of the Fourier transform of the original signal by 90° or $\pi/2$ radians and then taking the inverse Fourier transform. The Hilbert transform then generates the analytic signal:

$$z(t) = s(t) + iH\{s(t)\} \tag{3}$$

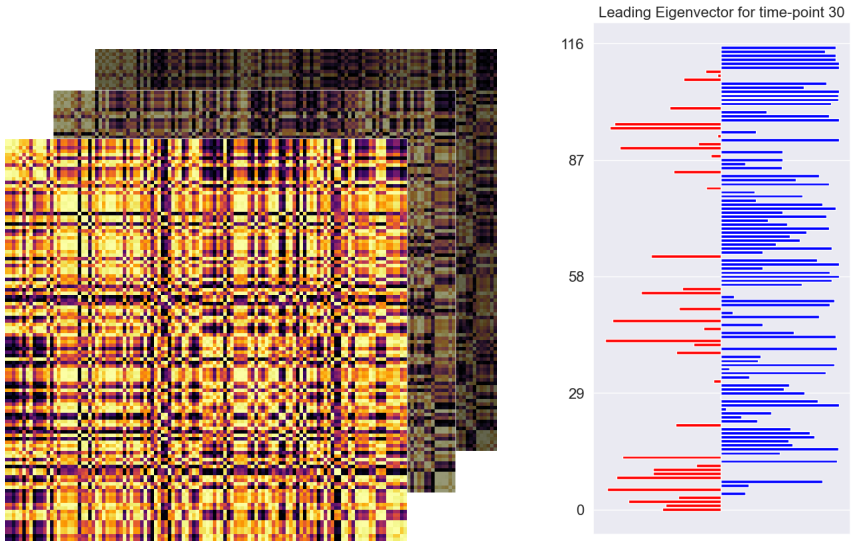
This signal contains both the original signal $s(t)$ and the phase-shifted (complex) Hilbert signal. Finally the instantaneous phase of the signal $\phi(t)$ is equivalent to the angle (argument) of the signal $z(t)$:

$$\phi(t) = \arctan\left(\frac{\Im\{z(t)\}}{\Re\{z(t)\}}\right) \tag{4}$$

The next step involves the construction of the instantaneous coherence matrices $\mathbf{A}_t$ for each time point t, where $\mathbf{A}_{t,j,k} = \cos(\phi_{t,j} - \phi_{t,k})$. Or expressed as the angle difference identity:

$$\mathbf{c} = \cos\phi(t), \quad \mathbf{s} = \sin\phi(t), \quad \mathbf{A} = \mathbf{c}\mathbf{c}^{\mathsf{T}} + \mathbf{s}\mathbf{s}^{\mathsf{T}} \tag{5}$$

These $\mathbf{A}_t$ matrices represent the connectivity dynamics across the different brain regions. Using the angle difference identity, a matrix with cosine elements is shown to have a rank of 2, which means that only two eigenvectors, with non-zero eigenvalues, are extracted from the relation $\mathbf{A}_t\mathbf{x}_t = \mathbf{\Lambda}_t\mathbf{x}_t$. And of these two eigenvectors, only the eigenvector with the highest eigenvalue (the leading eigenvector), is used in LEiDA, which most often leads to some loss of information. Coherence maps for time-points 30-32 and a visualization of the eigenvector for time-point 30 can be seen on Figure 2 (Olsen et al. [2022]).



(a) Coherence Maps (b) Eigenvector Bar plot

Figure 2: sub-figure (a) contains the instantaneous coherence maps for time-points 30-32. Sub-figure (b) is a bar plot representation of the leading eigenvector resulting from the eigendecomposition of the coherence map for time-point 30.

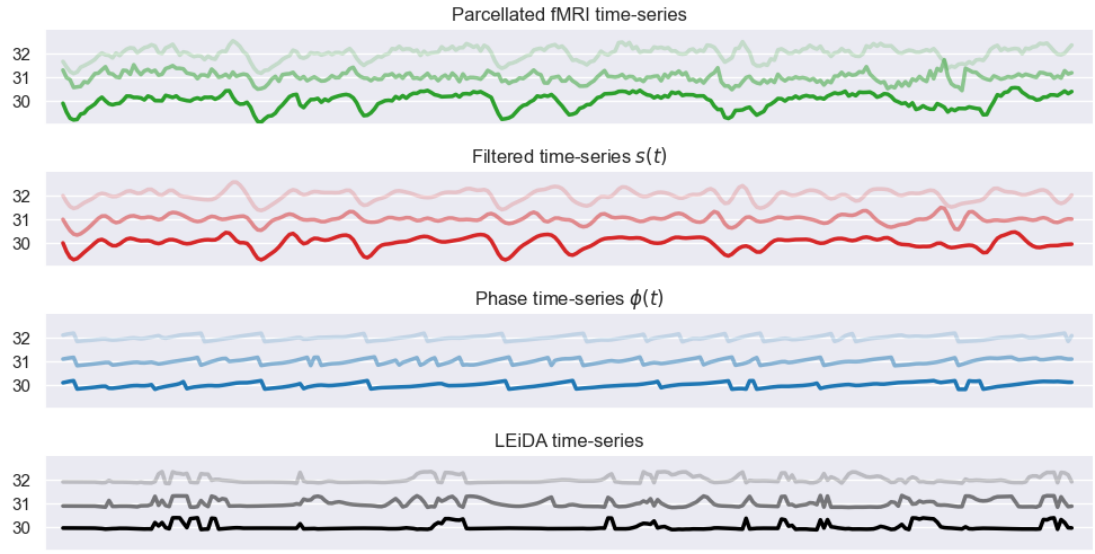


Figure 3: LEiDA time-series visualized on regions, 30-32, with time, t along the x-axis. The raw fMRI signal can be seen in green, and the filtering described in eq.(1) is visualized in red. In eq.(4) the angle of $z(t)$ creates the signal $\phi(t)$ which is visualized in blue. Lastly, the black lines represent the regions of the leading eigenvectors created from the coherence maps defined in eq.(5).

3.2 K-Means Clustering

As briefly mentioned in the introduction, the goal of our research questions is ultimately to analyze sleep patterns by exploring the possibility of there being substages to the currently known stages of sleep. With the extension of the LEiDA procedure, where both eigenvectors and their eigenvalues are included, it is possible to take the fMRI sleep data and output a representation of the parcellated brain regions, on which known clustering algorithms can be applied. In this section, the following K-Means algorithms, in order of effectiveness, will be defined: K-Means using Euclidean distance, and K-Means using Cosine distance, Diametrical Clustering, Grassmann Clustering, and a new weighted variant of Grassmann clustering.

3.2.1 Euclidean K-Means Clustering

Euclidean K-Means Clustering, or EC, being the simplest of the methods applied, is defined as per usual with an expectation step (E-step) commonly referred to as similarity (when maximizing), or distance (when minimizing), and then a maximization step for updating the centroids (M-step) also referred to as the mean function. After the E-step, the points have been assigned to a centroid and partitioned into sets $\mathcal{S}$ for each centroid. Where $\mathbf{X}$ represents both the eigenvectors from the LEiDA procedure, which is $(p \times 2)$-dimensional data and $\mathbf{x}$ represents only the leading eigenvector. $(p \times 1)$-dimensional data.

Step	Function
E-step (Expectation step)	$$D_{i,j} = \left\|\mathbf{x}_i - \boldsymbol{\mu}_j\right\|_2 \quad (6)$$ Where, $\mathbf{x}_i$ is a single data point and $\boldsymbol{\mu}_j$ is a single centroid. And the 2-norm is equivalent to the euclidean distance.
M-step (Maximization step)	$$\boldsymbol{\mu}_j = \frac{1}{\lvert\mathcal{S}_j\rvert} \sum_{\mathbf{x} \in \mathcal{S}_j} \mathbf{x} \quad (7)$$ Where $\mathcal{S}_j$ is the centroid of cluster j, $\mathcal{S}_j$ is the set of all points currently assigned to centroid $\boldsymbol{\mu}_j$, and $\lvert\mathcal{S}_j\rvert$ denotes the cardinality of $\mathcal{S}_j$, which is the number of points in $\mathcal{S}_j$. This equation represents the mean of all points $\mathbf{x}$ in the set $\mathcal{S}_j$

Table 1: E-step and M-step functions of the k-means algorithm

3.2.2 Cosine K-Means Clustering

From the LEiDA procedure, it can be derived that the resulting eigenvectors lie on a multi-dimensional unit sphere, since the data is normalized it means that the Euclidean distance is directly related to the cosine distance. This means that the EC algorithm previously described already performs as if it uses cosine distance. But in any case, since the M-step mean function for EC is unchanged it will result in centroids that are placed in the middle of the sphere. This can be improved by using a slightly different mean function, which would be the point of using Cosine K-Means Clustering, or CC (Olsen et al. [2022]).

Step	Function
E-step (Expectation step)	$$D_{i,j} = -\frac{\mathbf{x}_i^{\mathsf{T}} \cdot \boldsymbol{\mu}_j}{\left\|\mathbf{x}_i\right\| \left\|\boldsymbol{\mu}_j\right\|} \quad (8)$$ Notably, because both the data points $\mathbf{x}_i$, and the centroids $\boldsymbol{\mu}_j$ are normalized, the denominator in eq.(8) equals 1.
M-step (Maximization step)	$$\hat{\boldsymbol{\mu}}_j = \frac{\boldsymbol{\mu}_j}{\left\|\boldsymbol{\mu}_j\right\|}, \quad \boldsymbol{\mu}_j = \frac{1}{\lvert\mathcal{S}_j\rvert} \sum_{\mathbf{x} \in \mathcal{S}_j} \mathbf{x} \quad (9)$$ Where, $\boldsymbol{\mu}_j$ is the same mean function as in EC but it is also normalized, $\hat{\boldsymbol{\mu}}_j$, by dividing with the magnitude.

Table 2: E-step and M-step functions of the cosine k-means algorithm

3.2.3 Diametrical clustering

An integral part of the eigenvectors derived from the LEiDA procedure is that they are sign invariant. This means that all clusters have a diametrically opposed counterpart. For this reason, diametrical clustering, or DC, is applied, which has a sign invariant distance function

and a mean function that can effectively converge on good centroids on such a type of data.

Step	Function
E-step (Expectation step)	$$D_{i,j} = -\left(\frac{\mathbf{x}_i^\mathsf{T} \cdot \boldsymbol{\mu}_j}{\|\mathbf{x}_i\| \left\|\boldsymbol{\mu}_j\right\|}\right)^2 \quad (10)$$ The cosine distance, defined in eq.(8), is almost the same as this distance function, where the only change is squaring it, which makes it sign invariant.
M-step (Maximization step)	$$\mathbf{A} = \sum_{\mathbf{x} \in \mathcal{S}_j} \mathbf{x}\mathbf{x}^\mathsf{T} \quad (11)$$ $$\boldsymbol{\mu}_j = \mathbf{A}\boldsymbol{\mu}_j \quad (12)$$ $$\hat{\boldsymbol{\mu}}_j = \frac{\boldsymbol{\mu}_j}{\left\|\boldsymbol{\mu}_j\right\|} \quad (13)$$ Where, $\mathbf{x}$ now represents the partition of itself according to label assignments in $\mathcal{S}_j$. And $\mathbf{A}$ represents a shift in direction towards a better "mean"/centroid. And as with the M-step for CC in table 2 eq.(9) the centroids are normalized.

Table 3: E-step and M-step functions of the diametrical clustering algorithm

Intuitively the M-step for DC works by pointing the previous iteration of centroids in the direction $\mathbf{A}$, hence why eq.(12) also includes, $\boldsymbol{\mu}_j$. This method works because $\mathbf{A}$ represents a sort of pseudo covariance matrix, the difference being that it isn't normalized. And since $\mathbf{A}$ can skew the centroids outside the unit sphere the centroids are normalized.

Visualization of Clustering Methods Introduced Thus Far

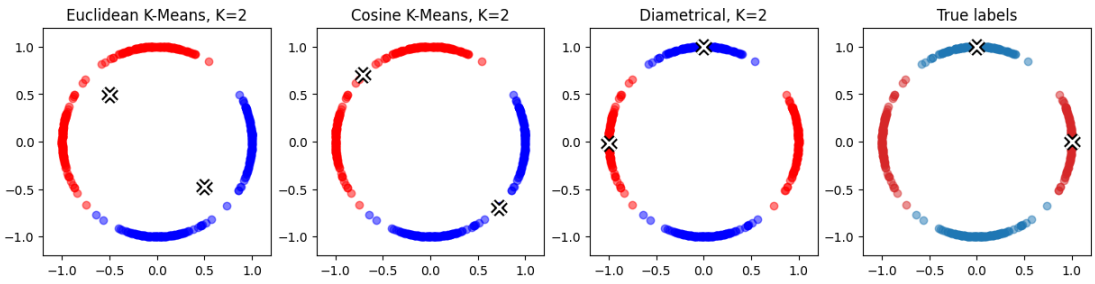


Figure 4: Generated 2D data scatter plot colored according to $\mathbf{x}_{\text{part}}$ results after clustering for respectively, Euclidean K-Means, Cosine Distance K-Means, and Diametrical Clustering.

As briefly mentioned in the sections about the three different clustering methods visible in Figure 4, they get progressively better, neither EC nor CC is sign invariant, and therefore naively splits the data into halves, whereas DC manages to correctly identify the diametrically

opposed clusters. Then from EC to CC, the main difference is the mean function, which for CC has been normalized, meaning that it more accurately puts the centroids on the unit sphere.

3.3 Riemannian K-Means clustering

Previously, only an understanding of the Euclidean space was needed to apply the various K-Means algorithms, but as is integral to LEiDA it has the downside of utilizing only the leading Eigenvector, which meant that only the Euclidean space was needed to analyze these $(p \times 1)$-dimensional spaces, which are these multidimensional unit spheres mentioned previously, and visualized in 2D in Figure 4. Riemannian manifolds are spaces that are curved in a way that is impossible to represent in an Euclidean space. Furthermore, Grassmannian manifolds are a subcategory of Riemannian manifolds that allow the representation of subspaces, in this case, both of the two eigenvectors resulting from the LEiDA procedure, resulting in a $(p \times 2)$-dimensional space. Going forward, q will be used to refer to the subspace dimensionality, i.e. the 2 eigenvectors.

3.3.1 Grassmann Clustering

To better understand clustering on the Grassmann Manifold, some properties of the orthonormal bases are going to be covered:

1. Each vector is a unit vector, i.e., $\mathbf{u}_l^\mathsf{T}\mathbf{u}_l = 1$ and thus located on the surface of the unit sphere.
2. Each vector represents an axis and is therefore sign invariant. The sign of the vector can be flipped without changing the axis and the representation
3. The vectors are pairwise orthogonal, i.e. $\mathbf{u}_l^\mathsf{T}\mathbf{u}_l = 0$.

These properties define the Grassmann manifold. The above three requirements can be summarized as the requirement $\mathbf{X}^\mathsf{T}\mathbf{X} = \mathbf{I}_q$ and we can therefore define the Grassmann manifold as:

$$\mathrm{Gr}(p, q) = \{\mathbf{X} | \mathbf{X}^\mathsf{T}\mathbf{X} = \mathbf{I}_q\} \tag{14}$$

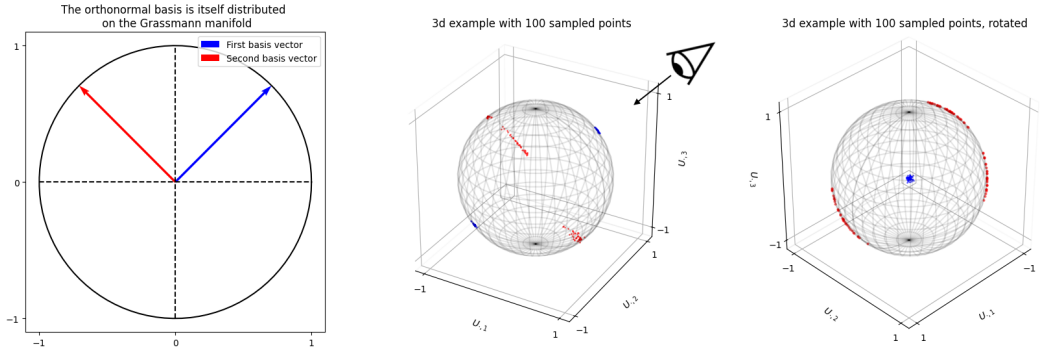


Figure 5: Orthonormal basis on Grassmann manifold (left), example distribution of pairwise orthonormal eigenvectors on a sphere (middle), and a version rotated (right) to the perspective of the eye from the middle plot

In the right-most plot in Figure 5 100 points have been sampled from the Grassmannian $\mathrm{Gr}(3, 2)$ by adding a random value to each element of a unitary phase vector $\boldsymbol{\theta} = [1, 1, 1]$, then computing the phase coherence matrix and the eigendecomposition, (which is reminiscent of the eigenvector

sampling technique described in section 4. The reason for showing this one in 3d is that $\mathrm{Gr}(2,2)$ is highly limited - points can only exist at $[1,0]$, $[0,1]$, $[0.7,0.7]$ and their sign-combinations. Note that variance was added to the red points but not really the blue ones. This happens due the nature of the eigendecomposition of A, which yields only one non-zero eigenvalue, thus only the blue point matters - the red point could be anywhere, as long as it's orthogonal to the blue point. This is best visualized on the right-most plot in Figure 5 where the perspective has been rotated to show that all the red points are placed orthogonally to the blue ones, forming a circle of possible positions in the form of a circle (the outline of the sphere).

The Grassmann manifold does not treat eigenvalues at all, and thus, both eigenvectors carry equal weight, even if their eigenvalues are highly different. In real human neuroimaging data, the eigenvalue for the leading eigenvector can explain anywhere between 50% and 100% of the variance, and thus, one solution for modeling phase coherence data is to just discard the trailing eigenvector completely and cluster such data-based on statistical distributions on the Watson manifold (sign-symmetric unit hypersphere). Making the simplest solution to use diametrical clustering. If we want to model both eigenvectors, albeit without the eigenvalues, we can do clustering on the Grassmann manifold. To do this we need a notion of distance on the Grassmann manifold and a way to update centroids in a K-means framework Wittrup et al. (GitHub, Weighted Grassmann Clustering Walkthrough).

E-step - Distance function

The distance may then be defined in a multitude of ways given the principal angles (Lim et al. [2016]). The most common solution is the norm of the principal angles, also called the geodesic distance, but we will instead focus only on the chordal distance which is the one used in the Grassmann clustering algorithm by Gruber and Theis [2006], which also works for cases where $\mathbf{U}_1$ and $\mathbf{U}_2$ are not of the same subspace dimensionality - say, $q_1 \neq q_2$.

After expanding and refactoring eq.(16) the closed-form expression in eq.(18) is reached, where $\mathrm{tr}(\cdot)$ represents the trace of a matrix, which is the sum of its diagonal elements:

$$d_{Gr(p,q)}(\mathbf{U}_1, \mathbf{U}_2) = \frac{1}{\sqrt{2}} ||\mathbf{U}_1\mathbf{U}_1^\top - \mathbf{U}_2\mathbf{U}_2^\top||_F^2 \tag{15}$$

$$= \frac{1}{\sqrt{2}} \mathrm{tr}\left(\mathbf{U}_1\mathbf{U}_1^\top - \mathbf{U}_2\mathbf{U}_2^\top\right)^\top \left(\mathbf{U}_1\mathbf{U}_1^\top - \mathbf{U}_2\mathbf{U}_2^\top\right) \tag{16}$$

$$= \frac{1}{\sqrt{2}} \left[\mathrm{tr}\left(\mathbf{U}_1\mathbf{U}_1^\top\right) + \mathrm{tr}\left(\mathbf{U}_2\mathbf{U}_2^\top\right) - 2\mathrm{tr}\left(\mathbf{U}_1\mathbf{U}_1^\top\mathbf{U}_2\mathbf{U}_2^\top\right)\right] \tag{17}$$

$$= \frac{1}{\sqrt{2}} \left[2q - 2||\mathbf{U}_1^\top\mathbf{U}_2||_F^2\right] \tag{18}$$

Note also that the extra derivations from the chordal distance function are necessary to avoid computing $\mathbf{U}_1\mathbf{U}_1^\top$ and $\mathbf{U}_2\mathbf{U}_2^\top$ since, if p is high, this can be prohibitively memory-expensive.

M-step - Mean function

The averaging step is not as simple, however. As explained in the paper "Finding a subspace mean or median to fit your need" by Marrinan et al. [2014], there are a bunch of different ways to define an average subspace, once again also dependent on whether the points are of the same subspace dimensionality q. The one that is the closest comparison to a euclidean mean, i.e., a "Riemannian" center of mass, is the Karcher mean (Karcher [2014]). The Karcher mean is the solution to the minimization of the (squared) geodesic distance, when summed over data points. However, as explained by Marrinan and colleagues, the Karcher mean does not have a closed-form solution and thus requires iterative optimization, which was part of the reason why Geodesic distance wasn't chosen as the distance function.

Instead, we looked at two other methods. The extrinsic manifold is the solution to the minimization of the sum of the (squared) chordal distances between the mean and the data points when the subspace dimensionality of these are equal. Whereas the flag mean is the generalization of the extrinsic manifold mean to instances, where the subspace dimensionality differs over data points. The derivations may be found in the (Marrinan et al. [2014]), but the solution to the flag mean is rather simple: Stack all of the points horizontally:

$$[\mathbf{U}] = [\mathbf{U}_1, \mathbf{U}_2, \ldots, \mathbf{U}_N] \tag{19}$$

Note that each point is a $(p \times q)$ orthonormal matrix. Then compute the SVD of this matrix:

$$[\mathbf{U}] = \mathbf{U}_{flag}\mathbf{\Sigma}_{flag}\mathbf{V}_{flag}^{\top}$$

The centroid is then the first q left singular vectors corresponding to the highest singular values. The SVD is super cheap to evaluate, even for high dimensions, making the flag mean very attractive. When the subspace dimensionality does not change, the flag mean reduces to the extrinsic manifold mean, i.e., the SVD may also be used in the same way to calculate the extrinsic manifold mean.

Step	**Function**
E-step (Expectation step)	$$D_{i,j} = \frac{1}{\sqrt{2}}\left(2q - 2\left\|\mathbf{X}_i^{\mathsf{T}}\mathbf{C}_j\right\|_F^2\right) \quad (20)$$ Where, eq.(20) is the close-form expression for the Chordal distance, $\mathbf{C}_j$ is the centroid with dimensions $(p \times q)$ for cluster j, and q is the subspace dimensionality, and $\|\cdot\|_F$ is the Frobenius norm.
M-step (Maximization step)	$$[\mathbf{X}]_j = [\mathbf{X}_1, \mathbf{X}_2, \ldots, \mathbf{X}_{N,j}] \quad \text{where} \quad \mathbf{X} \in \mathcal{S}_j \quad (21)$$ $$[\mathbf{X}] = \mathbf{U}_{flag}\mathbf{\Sigma}_{flag}\mathbf{V}_{flag}^{\mathsf{T}}, \quad \mathbf{C}_j = \mathbf{U}_{flag} \quad (22)$$ Where $[\mathbf{X}]$ represents a new matrix where all points in $\mathbf{X}$ are stacked, and N, j represents the last index in $\mathcal{S}_j$. Eigendecomposition is done on $[\mathbf{X}]$ in eq.(22) which results in the matrix $\mathbf{U}_{flag}$ which contains the eigenvectors and is assigned to $\mathbf{C}_j$. It is important to note that q still represents the subspace dimensionality, i.e., 2.

Table 4: E-step and M-step functions of the Grassmann Clustering algorithm

3.3.2 Weighted Grassmann Clustering

The Grassmannian, denoted as $\mathrm{Gr}(q, p)$, represents the space of all q-dimensional subspaces in a p-dimensional vector space. By incorporating the eigenvalues as weights, the Weighted Grassmann Clustering algorithm, or WGC, enhances GC, by providing a more nuanced representation of the data. These eigenvalues capture the varying importance of different subspace components, and it will help avoid the situation where too much weight is put on eigenvectors with little influence. Therefore, given a data matrix $\mathbf{X}$ of size $(n \times p \times q)$, where n is the number of observations, p is the number of features, q is the subspace dimensionality, the eigenvalue

weights are represented as a matrix $\mathbf{X}_{\text{weights}}$ of size $(n \times q)$. These weights correspond to the eigenvalues obtained from our data.

E-step - Weighted Chordal Distance

Each data point has a set of q eigenvalues associated with it. We can use these to weigh the distance between points. The weighted chordal distance function includes the eigenvalues $\Lambda = \text{diag}(\lambda_1, \ldots, \lambda_q)$:

$$d_{Gr(p,q)}(\mathbf{U}_1, \mathbf{U}_2, \mathbf{\Lambda}_1, \mathbf{\Lambda}_2) = \frac{1}{\sqrt{2}} || \left(\mathbf{U}_1 \mathbf{\Lambda}_1 \mathbf{U}_1^\top\right) - \left(\mathbf{U}_2 \mathbf{\Lambda}_2 \mathbf{U}_2^\top\right) ||_F^2 \tag{23}$$

$$= \frac{1}{\sqrt{2}} \text{tr}\left[\mathbf{\Lambda}_1 \mathbf{\Lambda}_1 + \mathbf{\Lambda}_2 \mathbf{\Lambda}_2 - 2\mathbf{U}_1 \mathbf{\Lambda}_1 \mathbf{U}_1^\top \mathbf{U}_2 \mathbf{\Lambda}_2 \mathbf{U}_2^\top\right] \tag{24}$$

$$= \frac{1}{\sqrt{2}} \left[\sum_{l=1}^{q} \lambda_{1l}^2 + \sum_{l=1}^{q} \lambda_{2l}^2 - 2\text{tr}\left[\mathbf{U}_2^\top \mathbf{U}_1 \mathbf{\Lambda}_1 \mathbf{U}_1^\top \mathbf{U}_2 \mathbf{\Lambda}_2\right]\right] \tag{25}$$

$$= \frac{1}{\sqrt{2}} \left[\sum_{l=1}^{q} \lambda_{1l}^2 + \sum_{l=1}^{q} \lambda_{2l}^2 - 2\text{tr}\left[\Lambda_2^{1/2} \mathbf{U}_2^\top \mathbf{U}_1 \mathbf{\Lambda}_1^{1/2} \mathbf{\Lambda}_1^{1/2} \mathbf{U}_1^\top \mathbf{U}_2 \mathbf{\Lambda}_2^{1/2}\right]\right] \tag{26}$$

$$= \frac{1}{\sqrt{2}} \left[\sum_{l=1}^{q} \lambda_{1l}^2 + \sum_{l=1}^{q} \lambda_{2l}^2 - 2||\mathbf{M}||_F^2\right] \tag{27}$$

Where $\mathbf{M}$ is simply a substitution that represents how the algorithm is formulated in the code.

M-step - Weighted Flag Mean

The flag mean can in addition be weighted by using the eigenvalues. First, we stack the bases horizontally:

$$[\mathbf{U}] = \left[\mathbf{U}_1 \mathbf{\Lambda}_1^{1/2}, \mathbf{U}_2 \mathbf{\Lambda}_2^{1/2}, \ldots, \mathbf{U}_N \mathbf{\Lambda}_N^{1/2}\right] \tag{28}$$

The solution to the flag mean is then the SVD of this matrix:

$$[\mathbf{U}] = \mathbf{U}_{flag} \mathbf{\Sigma}_{flag} \mathbf{V}_{flag}^\top \tag{29}$$

The centroid is then composed of both the first q left singular vectors and the squared first q singular values. And the centroid weights, are defined using $\mathbf{\Sigma}$, and normalized according to dimension p (Wittrup et al. (GitHub, Weighted Grassmann Clustering Walkthrough):

$$\mathbf{C} = \mathbf{U}_{flag}, \quad \mathbf{C}_{\text{weights}} = \frac{\mathbf{\Sigma}_{flag} \cdot p}{\|\mathbf{\Sigma}_{flag}\|} \tag{30}$$

Step	Function
E-step (Expectation step)	$$D_{i,j} = \frac{1}{\sqrt{2}} \left[\sum_{l=1}^{q} \lambda_{1l}^2 + \sum_{l=1}^{q} \lambda_{2l}^2 - 2\|\|\mathbf{M}\|\|_F^2 \right] \quad (31)$$ $$\mathbf{M} = \left[\mathbf{C}_{\text{weights}}^{1/2} \mathbf{C}^\top \mathbf{X} \mathbf{X}_{\text{weights}}^{1/2} \mathbf{X}_{\text{weights}}^{1/2} \mathbf{X}^\top \mathbf{C} \mathbf{C}_{\text{weights}}^{1/2} \right] \quad (32)$$ Where, λ_1 are eigenvalues from $\mathbf{X}_{\text{weights}}$, and λ_2 are eigenvalues from $\mathbf{C}_{\text{weights}}$.
M-step (Maximization step)	$$[\mathbf{X}]_j = \left[\mathbf{X}_1 \Lambda_{\mathbf{X}_1}^{1/2}, \mathbf{X}_2 \Lambda_{\mathbf{X}_2}^{1/2}, \ldots, \mathbf{X}_j \Lambda_{\mathbf{X}_j}^{1/2} \right] \quad \text{where} \quad \mathbf{X} \in \mathcal{S}_j \quad (33)$$ $$[\mathbf{X}] = \mathbf{U}_{flag} \mathbf{\Sigma}_{flag} \mathbf{V}_{flag}^\top \quad (34)$$ $$\mathbf{C} = \mathbf{U}_{flag}, \quad \mathbf{C}_{\text{weights}} = \frac{\mathbf{\Sigma}_{flag} \cdot p}{\|\mathbf{\Sigma}_{flag}\|} \quad (35)$$ Where $\Lambda_{\mathbf{X}_j}$ represents the eigenvalues for $\mathbf{X}_j$.

Table 5: E-step and M-step functions of the Weighted Grassmann Clustering algorithm

3.4 Comparing labels using NMI

With all the clustering algorithms introduced, only a method of comparing the clustering methods is now needed. First, the centroids from each method are converted to activation sequences, i.e. labels of cluster assignments, in our case, representing brain states and when they are activated, as extrapolated from the K number of clusters. Then these brain network activation sequences are compared to the labeled data, where physicians had judged the sleeping state of the patients. The problem that normally occurs when trying to compare labels is that the clustering models don't know what the different clusters "are" or what they represent. This means that the 3. cluster in one clustering method and the 5. cluster in another method might be referring to the same cluster in the data. For this problem, NMI is a powerful tool because it has the benefit of not caring about which specific labels were given to the data, only how the two sets of labels match overall (Wikipedia contributors [2024c]).

Mutual information is a straightforward measure of dependence between two variables. Defined as:

$$\text{MI}(\mathbf{U}; \mathbf{V}) = \sum_{x \in X} \sum_{y \in Y} p(x, y) \log \left[\frac{p(x, y)}{p(x)p(y)} \right] \quad (36)$$

Since Mutual Information has a slightly unintuitive unbounded range: $\text{MI}(\mathbf{U}, \mathbf{V}) \in [0 \ldots \inf[$. Using Normalized Mutual Information (NMI) instead provides a more reasonable range: $\text{NMI}(\mathbf{U}, \mathbf{V}) \in [0 \ldots 1]$. NMI is defined as:

$$\text{NMI}(\mathbf{U}; \mathbf{V}) = \frac{\text{MI}(\mathbf{U}; \mathbf{V})}{\text{mean}(\text{MI}(\mathbf{U}; \mathbf{U}),\ \text{MI}(\mathbf{V}; \mathbf{V}))} \quad (37)$$

4 Data

4.1 Data Introduction

The dataset originates from a research project conducted at Pennsylvania State University. It features a simultaneous collection of EEG (electroencephalogram) and BOLD (blood oxygenation level-dependent) signals across various sessions - however, in this project only the BOLD fMRI data will be considered for the Grassmann clustering models. BOLD fMRI signals were obtained from 33 healthy participants. Each participant underwent an anatomical session, two resting-state sessions, and several sleep sessions. During the sleep sessions, participants' sleep stages were recorded and classified into various stages, including wakefulness (denoted as "W") and sleep stages: N1, N2 and N3. These sleep stages are represented in the dataset as .TSV (tab-separated values) files, with each file containing sleep stage classifications for 30-second epochs. Additionally, epochs with uncertainty in classification or significant artifacts are labeled as "uncertain" or "unscorable," respectively. Blood oxygenation level-dependent, functional MRI, BOLD fMRI, scans use contrast imaging to detect changes in blood oxygenation levels, serving as a proxy for neural activity. This can then further be used to capture relative activity.
The collection and usage of the dataset adhere to ethical standards, with participants providing informed consent for data acquisition and analysis. The dataset is organized and stored systematically, with BOLD fMRI data, EEG data, and sleep stage classifications organized into separate folders. This structured organization facilitates efficient data retrieval.

4.2 Data Preparation

The data contains measurements from both rest and sleep runs. We will discard all the rest runs, as we are more interested in the sleep data. Most literature concludes, that the brain can be in four different sleep stages (N1, N2, N3, REM) during sleep. REM sleep usually begins about 90 minutes after falling asleep. Given how the measurements was collected, it makes sense that we don't see any REM sleep in the data, and indeed, we don't. Additionally, as mentioned earlier, it wasn't even possible for the experts to classify any observations as REM sleep. The N3 state is considered deep sleep, and given the relatively short interval of data measurements and the fact that sleeping in an unfamiliar setting can be challenging for some people, it also makes sense that we only see a few observations of the N3 state across all subjects. We have decided to remove the few sleep runs, that contained N3, as we would optimally like to see each cluster capturing only one kind of sleep stage. This does not seem likely to happen with only a few observations of N3 unless we build a model with a substantial amount of clusters. Therefore, the data we work with only contain three stages (including the awake state): W, N1, and N2.

With this in mind, we can move forward. There are still some issues with the data that will now be addressed. Firstly, not all subjects have a complete data portfolio; specifically, we are missing some of the sleep runs. At first glance, this might not seem like a problem, as the models will be built with all the data combined. However, we also want to analyze individual subjects. Therefore, we need the same amount of data for each subject to justify the analysis. Since this issue only affects a few subjects, we have decided to remove them completely.

The second issue to address is that some of the sleep runs from certain subjects contain only the awake state (W). In other words, the subject is fully awake throughout the measurements. Initially, this should not be a problem other than the fact that it would be optimal to have a balanced dataset. However, when we turn to individual analysis of the subjects, we encounter problems with the NMI score when we only have one state in a subset of the data. The NMI score will be zero in such cases. Therefore, we also removed runs where the state is constant throughout the entire measurement period.

To sum up, the data we are working with includes only those subjects for whom we have data for all four sleep runs, and where the runs contain more than one sleep stage. Additionally, there were a few subjects with a substantial amount of data classified as uncertain. Instead of removing the individual data points, we chose to remove these subjects entirely. This decision was made because the uncertain labels were confined to data within a few subjects, and the general data from these subjects appeared quite noisy. After these adjustments, we are left with data from 13 subjects that meet all the above rules.

4.3 Synthetic data

The synthetic data generation serves as a crucial component in this research project, enabling the creation of artificial datasets that can be used to evaluate methods relevant to the study's objectives. The code for synthetic data generation works by iterating over different parameters such as the number of clusters (K) and the dimensions of the data (p). For each combination of parameters, that follows $K \geq p$, the code proceeds to generate synthetic data using mixture models, specifically the Angular Central Gaussian (ACG) mixture model and the Matrix of Angular Central Gaussians (MACG) model.
In the ACG mixture generation process, covariance matrices are constructed for each cluster, encompassing both isotropic and anisotropic covariance structures among the clusters. Isotropic covariance matrices concentrate around a specific axe (stretched distribution), while anisotropic covariance matrices covary along two axes (round distribution), providing diversity in the synthetic data. The ACG model then samples points from the clusters (distributions). The MACG mixture model generates synthetic data in a slightly different way, sampling points in pairs enabling the data to be represented as subspaces.
For all the models the synthetic data is generated in two steps corresponding to training and test data. Data is also normalized to ensure it lies on the unit sphere.

The second method of sampling, referred to as the custom eigenvector sampling method, was one focused on generating valid coherence maps on which eigendecomposition can provide eigenvalues and eigenvectors. This is done by first generating a base consisting of $\frac{\pi}{2}$ values with dimensions $(p \times N)$ where an equidistant, over N, number of indices is chosen from this base and set to 0. Uniform noise varying from $[0, \ldots, 0.7]$ is then added to all points in the base, the coherence map is calculated and then eigendecomposition is applied on that, to get the eigenvectors and eigenvalues from these base vectors. Similarly to LEiDA, only the first two leading eigenvectors and eigenvalues are saved.

5 Results

The following results were gathered from code which can be found in our GitHub: Wittrup et al. (GitHub, 2024a)

5.1 Comparing Grassmann Clustering to Traditional Methods

Various clustering methods were defined to address the first research question, including Euclidean K-Means, Cosine K-Means, Diametrical Clustering, and the cutting-edge method Grassmann Clustering. As concluded in the method section, the Euclidean and Cosine K-Means variations differ only in the accuracy of the centroids. Still, since that won't be explored on the synthetic dataset, this is neglected and therefore the Cosine K-Means variation wasn't included in tests. And since the synthetic data generated using MACG doesn't create eigenvectors, it also doesn't have any eigenvalues that can be used for Weighted Grassmann clustering, which meant that this method couldn't be tested using this type of synthetic data.

Since the objective functions for each clustering method vary a lot in how they are measured and therefore also in their mean values etc., this was not considered as a proper variable to compare them with. This is why the clustering labels became the primary variable of interest, an example of how the labels could look for K=5 is shown in Figure 6, where it is obvious that Euclidean K-Means is hopeless at classifying correctly.

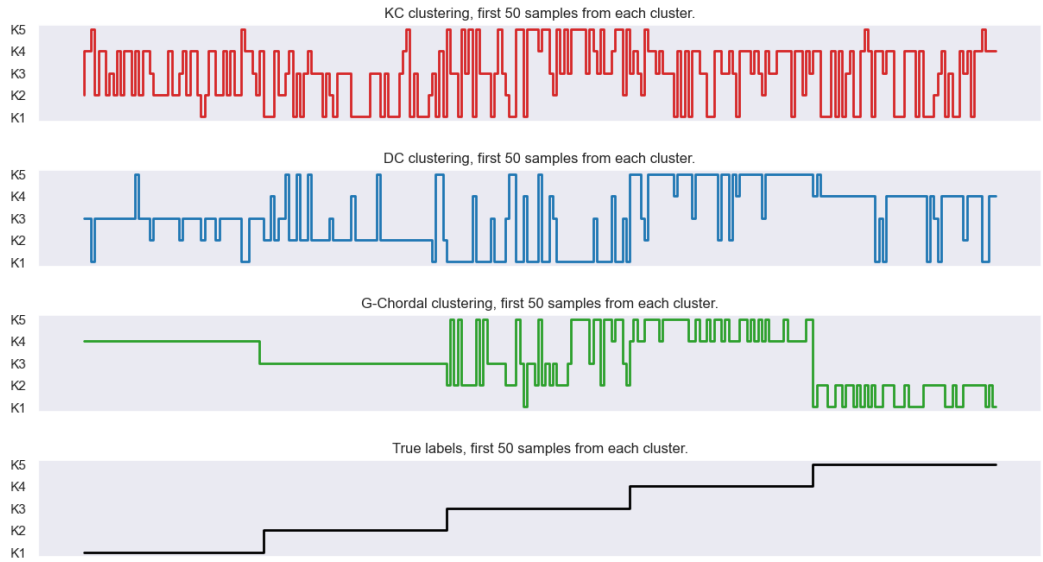


Figure 6: Labels after clustering with Euclidean K-Means (red), Diametrical Clustering (blue), and Grassmann Clustering (green), for K=5, including only the first 50 samples from each (true) cluster.

But to properly analyze the differences in performance when using Grassmann clustering or the weighted version of Grassmann clustering it was necessary to compare them using NMI. First, the MACG method of sampling data was attempted which yielded acceptable results, except that most of the clustering methods completely failed at labeling any MACG-generated data with 2 clusters. For this reason, it was necessary to create a secondary method of producing synthetic data, which could also return eigenvalues so the weighted version of Grassmann clustering could be tested. An example of how these labels could be distributed for $K = 5$ clusters is shown in

Figure 6.

In Figure 7 it is obvious that the Grassmann clustering algorithm performs better overall, although it has a slight dip at $p = 50, K = 4$, where Diametrical Clustering is slightly better. But it is also worth noting that all clustering results from MACG sampling are quite bad, with a max median of about 0.7. Furthermore, there is an interesting pattern for Euclidean K-Means and Grassmann Clustering where they are better at classifying a high number of clusters and somewhat better when the dimensions are higher as well, where both methods have a significant decline in performance when the number of clusters is smallest. It is worth noting that this isn't the case for diametrical clustering, which generally performs better for a higher number of clusters and dimensions.

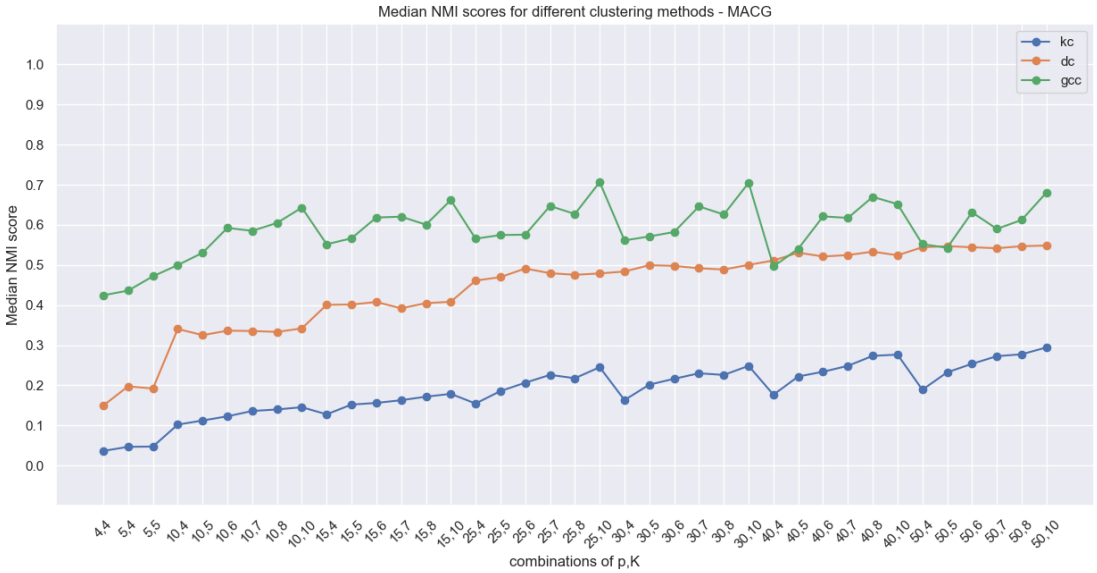


Figure 7: The median NMI calculated from 10 runs of each clustering method where the x-axis represents the (p, K) values, given on the x-axis, were used to generate the data via. MACG sampling. The three lines represent Euclidean K-means (blue), Diametrical Clustering (yellow), and Grassmann Clustering (green).

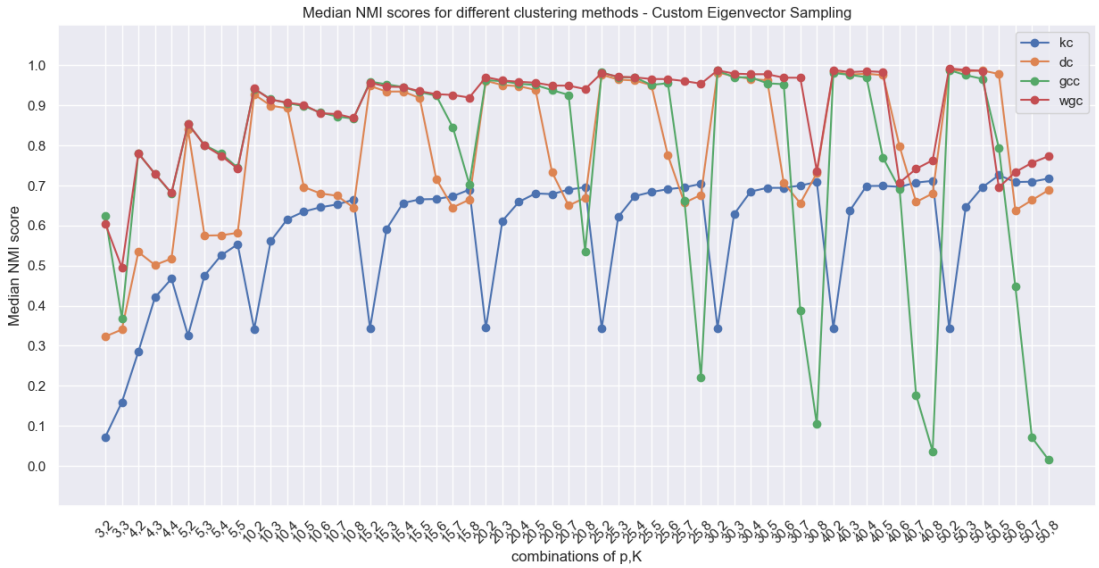


Figure 8: The median NMI calculated from 10 different seeds in the custom eigenvector sampling method, for each clustering method, where the x-axis represents the (p, K) values used to generate the data. The four lines represent Euclidean K-means (blue), Diametrical Clustering (yellow), Grassmann Clustering (green), and Weighted Grassmann Clustering (red).

But for the second method of sampling synthetic data, seen in Figure 8, there are both higher highs and lower lows of NMI values. For this method of sampling, there were a few more combinations of dimensions and clusters that were possible to test, and more importantly, it allowed us to test on weighted Grassmann clustering. For this method of sampling it is slightly different, and only Euclidean K-means performs badly on a low amount of clusters, and both GC and WGC perform their best on a low amount of clusters. Where Grassmann clustering was previously thought to best, it doesn't perform very well on this type of data, but the weighted variant WGC does perform better than all the other methods overall, only slightly outperformed by diametrical clustering in very few situations (p=40,K=6), (p=50,K=5).

5.2 Finding the optimal number of clusters

5.2.1 Objective Function Value and NMI Evaluation

To answer the second research question, namely, *Is there an optimal number of clusters, and how does this relate to dataset characteristics?*, we will work with the real BOLD fMRI data from Pennsylvania State University. Here is a brief recap to clarify the specifics of this part of the research. We will attempt to determine the optimal number of clusters by experimenting with different cluster configurations. This will help us understand the relationship between the number of clusters and the dataset characteristics.

For now, we will stick to the GC algorithm as introduced by Gruber and Theis [2006]. All the data from the 13 subjects is combined, resulting in a shape of (22,308, 116, 2) — that is, the number of data points, the dimensions of eigenvectors, and the number of eigenvectors, respectively. We used the following number of clusters for the analysis:

$$clusters = [2, 3, 4, 5, 6, 7, 8, 9, 10, 12, 15]$$

The Grassmann algorithm was executed with a constraint of 500 iterations maximum due to computational complexity; however, most of the time, the algorithm terminated well before

reaching this limit. Figure 9 illustrates a plot of the best objective function value, for each subject, as a function of the number of clusters. Only the values at the exact positions of the vertical red lines can be interpreted as true, as the rest are interpolated to fit the graphs.

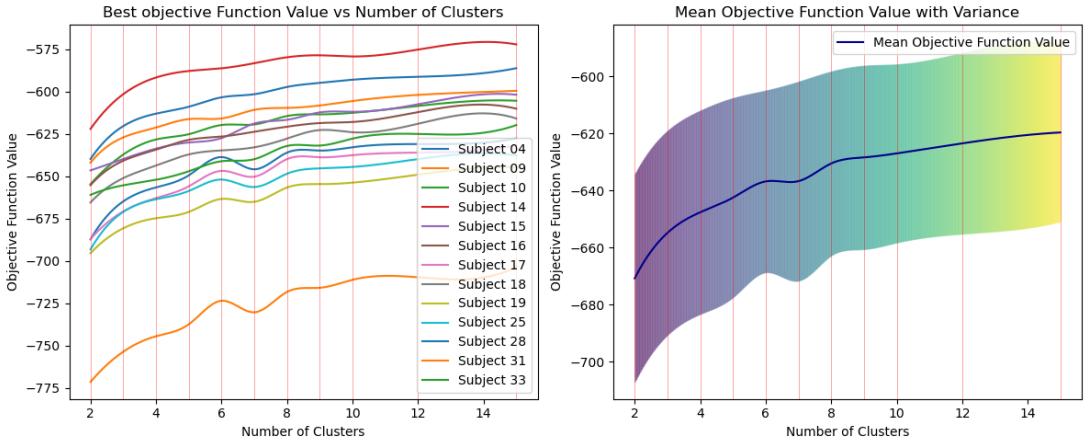


Figure 9: Objective function value

From the mean objective function value graph in figure 9, it is visible that the objective function value improves slightly more when approaching 3, 6 and 8 clusters, compared to the other numbers of clusters. This indicates that the data we are working with fits especially well for those number of clusters. The objective function value is simply the sum of the negative distances for all data points to their assigned cluster, so it is no surprise that we see a general trend of better objective function values as we increase the number of clusters. The average distance for the data points to their assigned clusters should intuitively decrease as we add more clusters, even for randomly placed centroids.

In the following analysis, we will shift our focus from evaluating the objective function value to calculating NMI score. As mentioned in the methods section, NMI can be used to compare the activation sequence derived from Grassmann clustering, or any other clustering method, with the true labels. This will tell us how well the predictions fit the true labels, even for different numbers of classes. For this to work, some further work on the data is necessary. The true labels are provided for each 30-second interval by the expert source. We have 30 true labels for each run, giving us the 15-minute runtime. However, we have BOLD fMRI measurements every 2.1 seconds. For NMI to work, we need to have a true label, for every measurement. We devised a sequence of length 30 that contains the numbers 14 and 15 spread out somewhat evenly. This sequence sums to 429 ($429 \cdot 2.1sek = 15min$) and repeats each true label either 14 or 15 times.

We then one-hot encode the true labels. This matrix will take the shape (3, 22,308) given that we have 3 stages. We further one-hot encode the predictions. This matrix will take the shape (K, 22,308), where K is the number of clusters. From here, we can calculate the NMI scores. Figure 10 illustrates the NMI score as a function of the number of clusters, K

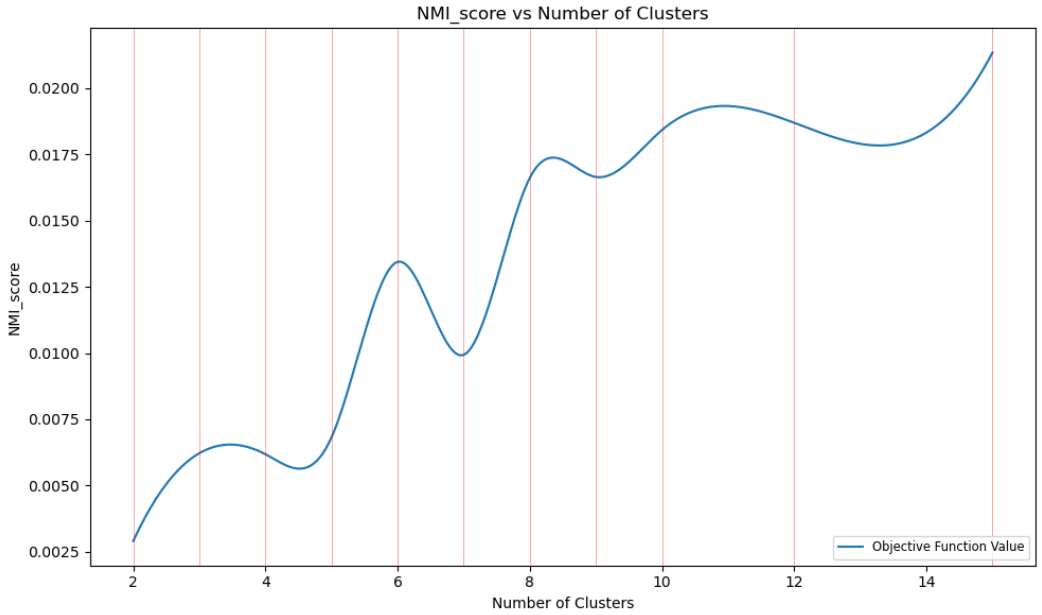


Figure 10: NMI score

Its clear that there exists a local peak in the NMI score at $K = 3$ and $K = 6$. Furthermore the NMI score for $K = 6$ is drastically better than for $K = 4$ and $K = 5$. From $K = 8$ and onwards the increase in NMI seems to be slow and steady. This indicates that as the number of clusters increases, we are better able to capture the true underlying sleep stages, even though each of them, presumably, are represented by multiple clusters. In general its also evident, that the NMI is relatively bad for all number of clusters. This will be further discussed later. One thing to note is that the random initialization of the Grassmann clustering algorithm seems to have a slight impact on the results. The overall trends remain, but we see slightly different results for each complete run of the code. We have attached a plot in the appendix that shows the mean and variance for each number of clusters, based on 10 random initializations.

5.2.2 Centroid Analysis and Functional Network Mapping

Looking at the actual centroids obtained from the Grassmann clustering algorithm, we can further explore their potential meanings. First of all, it is beyond the scope of this project to delve into the very complex biology and structure of the brain. In theory, this project aims to construct and explain new brain networks by clustering data and analyzing centroids. However, we will not pursue that direction. Instead, we will focus our analysis on existing networks and models of the brain. Specifically, we will compare the results to the seven large-scale functional networks Yeo et al. [2024] and briefly consider the lobes of the brain Wikipedia contributors [2024b] and their respective functions.

Figure 11 is an illustration of the seven large-scale functional networks as well as the Subcortex area, which is the inner part of the brain. As introduced in the methods section, we use the Schaefer 2018 atlas in this project. Each of the 116 areas from the atlas is then mapped to the seven large-scale functional networks. In other words, we know which network each of the 116 areas from the atlas belongs to.

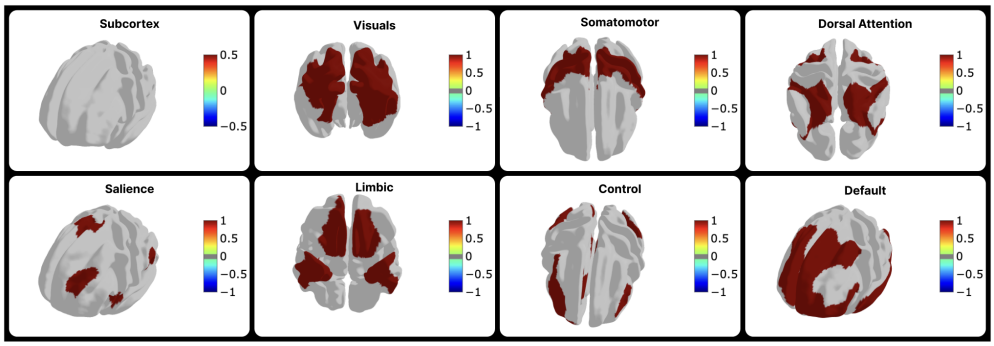


Figure 11: Seven large-scale functional networks + Subcortex

Each time we run the Grassmann clustering algorithm, it returns the centroids C with the shape (K, 116, 2), where K is the number of clusters. We can only plot one component of each centroid at a time, so this means we have to make $K \cdot 2$ plots for each run of the algorithm. This would be an overwhelming amount of information, so we have decided to narrow the analysis to the models $K = 5$ and $K = 6$, as we see a significant increase in the NMI score from 5 to 6 clusters.

Another point to mention is that the second component of all centroids, for all numbers of clusters, is "constant"; that is, all values are either positive or negative. This is most likely due to the fact that the instantaneous coherence matrices we work with (before converting to eigenvectors) generally have positive phase coherence. This means that the signals from different brain regions are more in sync than not. Consequently, this translates to a constant component, and one could argue that this constant component doesn't hold much information about the brain's biology. For this reason, we will only work with the first component from each centroid.

In figure 12 the distribution of true labels within each cluster for the models $K = 5$ and $K = 6$ is plotted. It is clear that both models struggle to isolate the sleep stages within their own clusters. However, it looks like the model with $K = 6$ has a higher NMI score compared to the model with $K = 5$. One explanation for this could be that the $K = 6$ model is better at capturing the sleep stage 'W', as the majority of 'W' observations lies in clusters 0, 4, and 5. We can further analyze the centroids using brain maps and bar plots. This will enable us to understand what each centroid might be capturing by comparing them to existing and well-known networks.

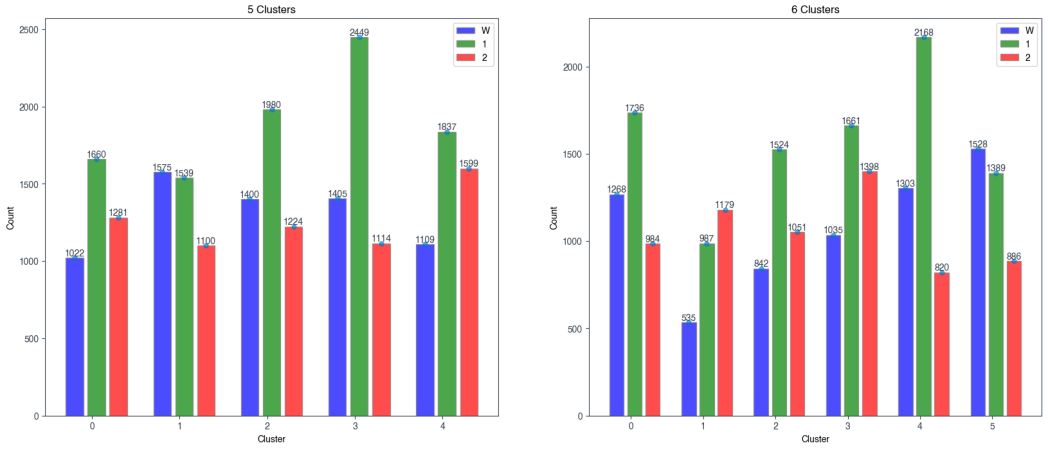


Figure 12: Distribution of true labels within each cluster

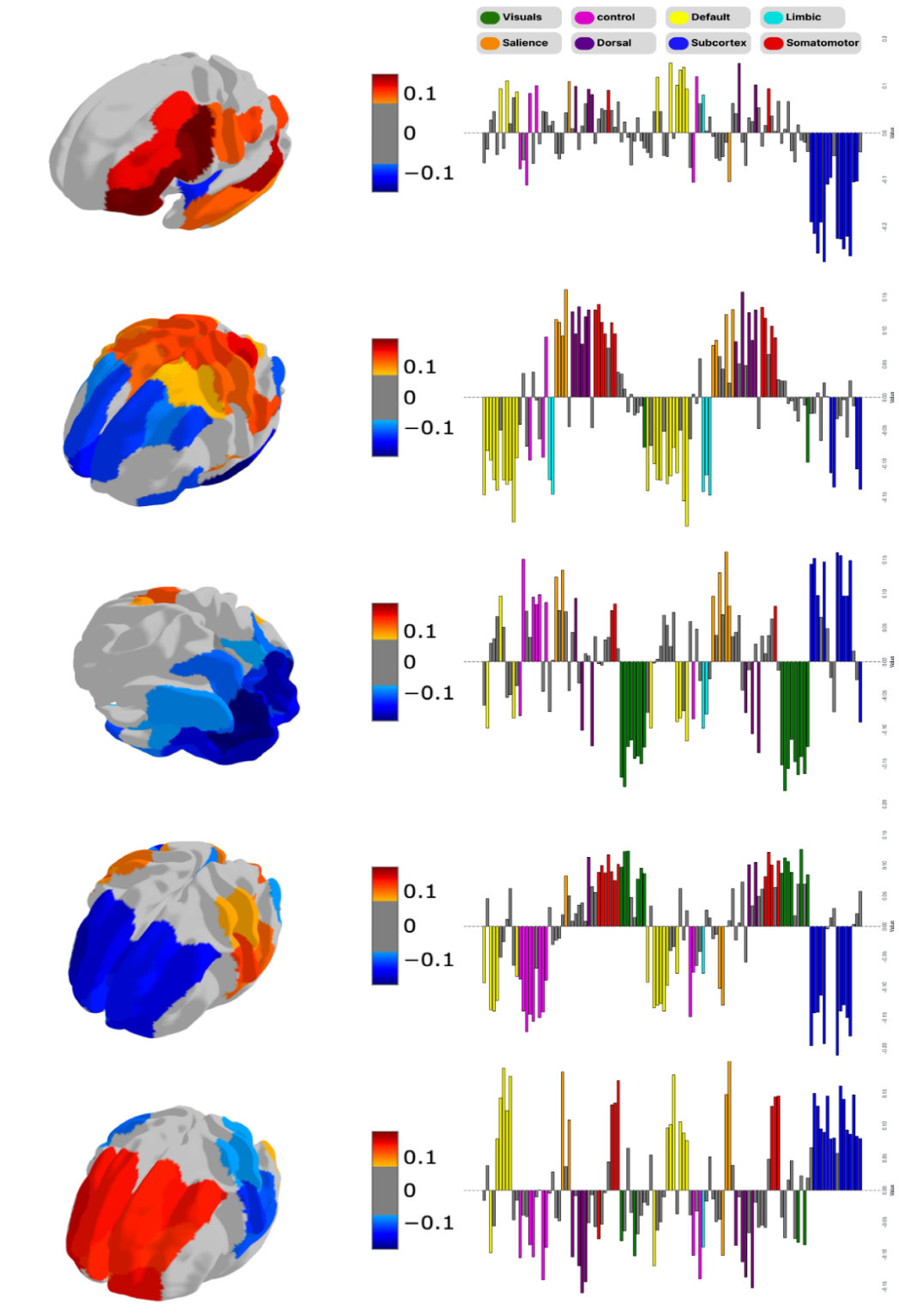


Figure 13: Centroids as brainmaps and barplots, 5 clusters

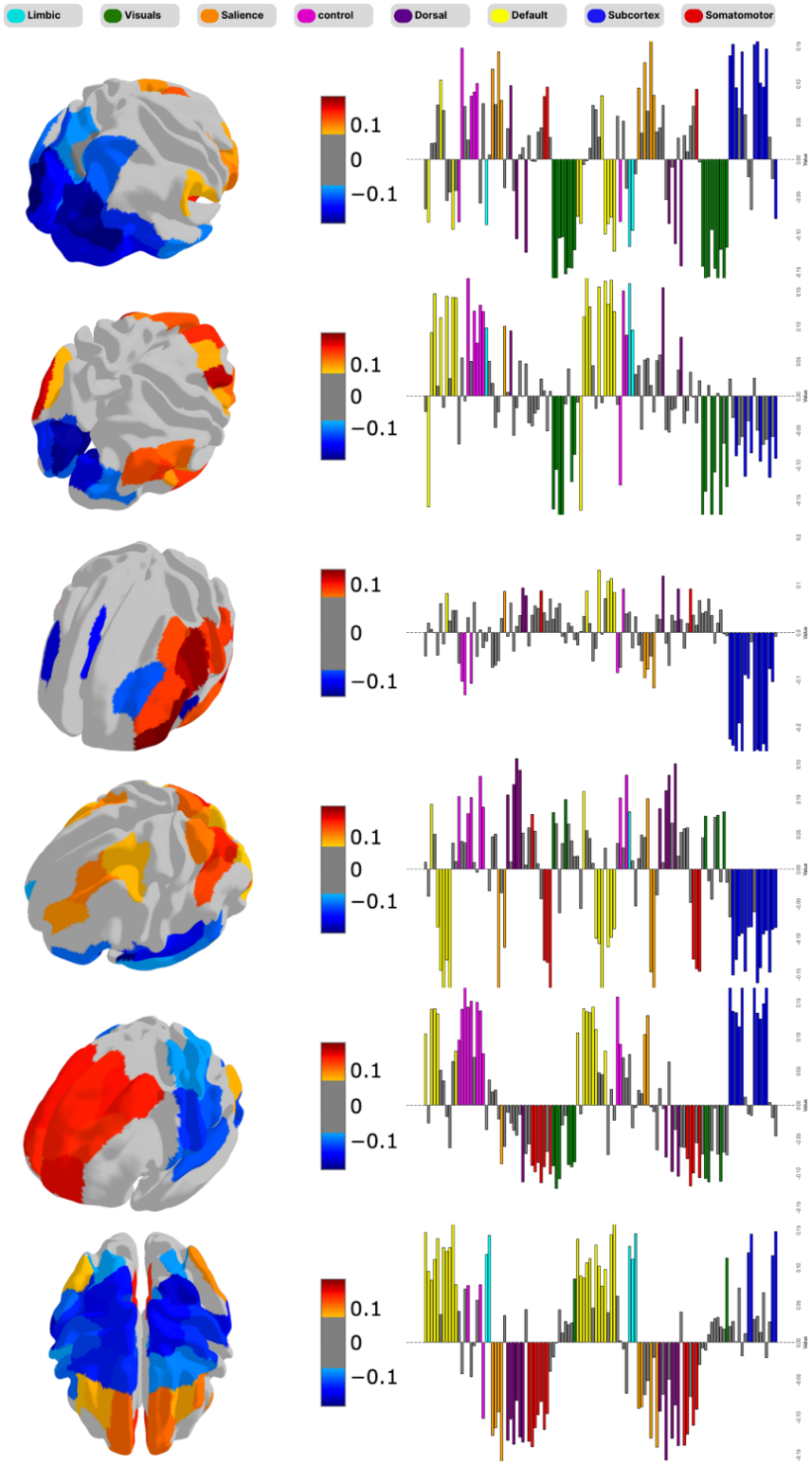


Figure 14: Centroids as brainmaps and barplots, 6 clusters

Looking at the two models and the resulting brain maps and bar plots, there are many aspects to explore. First of all, it is evident that all centroids in the $K = 5$ model (figure 13) are also represented to some degree in the $K = 6$ model (figure 14). Using zero-indexing, we can map the two models accordingly.

$K = 5$ Model, Cluster Number	$K = 6$ Model, Cluster Number
0	2
4	3
1	5
2	0
3	4

Table 6: Cluster Mapping Between Two Models

This leaves us with a completely new cluster in the $K = 6$ model, specifically cluster number 1. From Figure 12, we know that this centroid captures the fewest observations with the true label 'W', while still capturing a substantial amount of the data overall. Given that this is the only real difference between the two models, apart from small adjustments in corresponding centroids, it is fair to assume that cluster 1 in the $K = 6$ model causes the increase in the NMI score. Looking at Figure 11, we see the most dominant activation in the default mode, visual network, and control network for cluster 1. From the brain map, we see activation in the frontal lobe, occipital lobe, and temporal lobe. The Default mode network is both activated when an induvidual is awake and at rest. Its internally-oriented and handles tasks such as daydreaming, envisioning the future, retrieving memories and theory of mind (Wikipedia contributors [2024a]). The visual network processes visual information, while the control network, also known as the central executive network, handles cognitive control. The control network should not be confused with the somatomotor network, which manages actual body movement. We know that cluster 1 from model $K = 6$ primarily consists of observations with true labels N1 and N2 (sleep stages). This aligns somewhat with the known networks that are active, when looking at the centroid of cluster 1. Specifically, the default mode network and the control network are assumed to be active during sleep. However, we will only comment on this and conclude that the increase in the NMI score, between model $K = 5$ and $K = 6$ seems to be caused by cluster 1 in model $K = 6$.

To sum it up, we observe local spikes in the NMI score at $K = 3$ and $K = 6$, along with a general trend of increasing NMI scores with the number of clusters (up to $K = 15$). The most drastic change in the NMI score occurred from $K = 5$ to $K = 6$. Upon further investigation of the cluster centroids in both models, we found that the increase in the NMI score is likely due to the introduction of a completely new centroid in the $K = 6$ model. This stands in contrast to our initial theory that each sleep stage could be divided into subclasses. We expected that as the number of clusters increased, some clusters would disappear, and two new clusters would emerge, providing an even better fit for the data captured by the original cluster centroid.

5.3 Weighted Grassmann Clustering with Eigenvalues as Weights

To understand the impact of eigenvalues on clustering results, we will use the weighted Grassmann clustering algorithm (WGC) for this part of the project, as introduced in the methods section. We have demonstrated how the Grassmann algorithm, as introduced by Gruber & Theis (2006), performs across different numbers of clusters. To obtain comparable results, we will now repeat the same analysis with the same preprocessing of the data. The only thing changed is the clustering algorithm, that is now WGC.

Since WGC incorporates weights that influence the scale of distances computed for each observation, the objective function value cannot be directly compared to that of the normal GC. However, the relative increase or decrease in the objective function value across different numbers of clusters remains comparable. As shown in Figure 15, the general trend of better objective function values as we increase the number of clusters remains the same compared to GC. There are, however, some notable changes. Both models, GC and WGC, seem to have specific numbers of clusters that best, locally, fit the data. For normal GC, this was $K = 3$ and $K = 6$, as shown previously in Figure 7. For WGC, this appears to be $K = 3$, $K = 8$ and $K = 10$, as shown in both Figures 12 and 13.

We also observe that WGC seems to perform worse compared to GC, as evidenced by generally lower NMI scores across different number of clusters. This is, of course, somewhat disappointing since WGC is a brand-new clustering method with the promise of utilizing the importance of each component in each data point. It would have been nice to see this translate into better clustering and thereby better NMI scores. However, it is important to remember that we are working with presumably noisy data, given its labeled subjectively by an expert. Because the NMI score is calculated using the true labels, this might introduce noise into the results. It could be the case that the WGC models has mapped out more useful brain networks than the GC models when it comes to classifying sleep stages. Both GC and WGC, like many other clustering methods, do not know the true labels. They work independently to form the clusters. Therefore if we applied the learned GC and WGC models to another dataset, we might see different results. We will discuss this further in the discussion section.

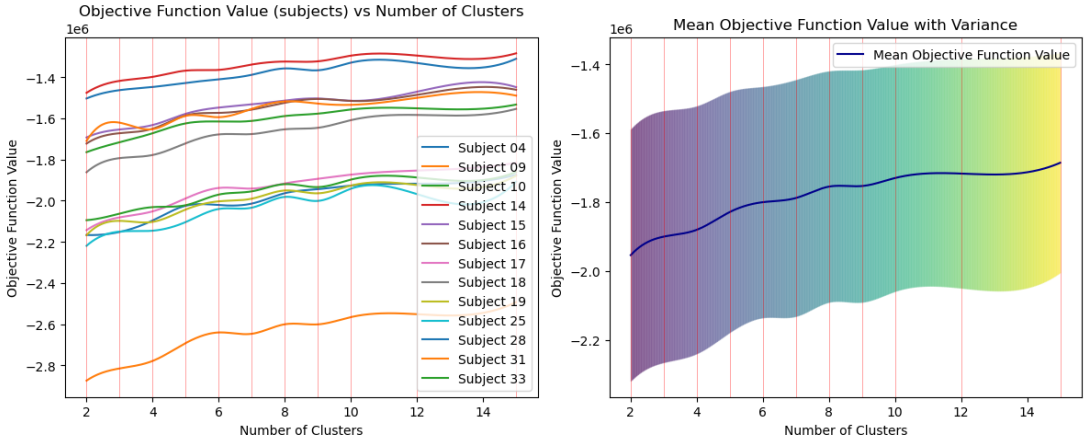


Figure 15: Objective function value, Weighted Grassmann

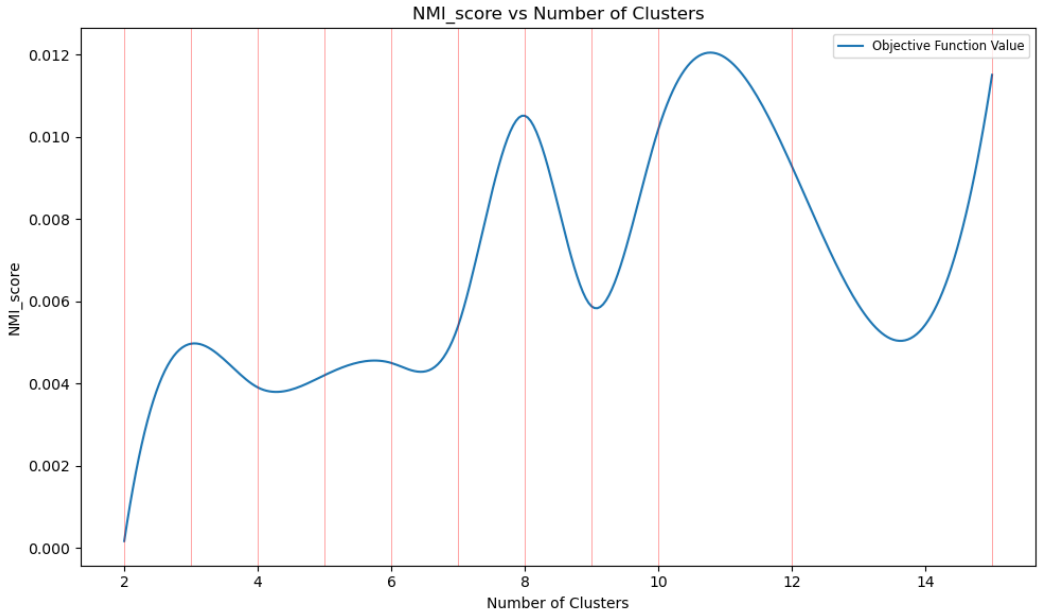


Figure 16: NMI score, Weighted Grassmann

Similarly to the segment about the GC, we ran the WGC algorithm 10 times with 10 random initializations. A plot of the NMI score with mean and variance is provided in the appendix. An argument regarding the poor NMI score for the WGC models compared to the GC models has been given earlier. We can further analyze how the two clustering methods differ by examining their centroids.

To do this, we focus on $K = 6$ for both clustering methods because we observed a significant difference in NMI scores for this number of clusters and therefore assume to see some differences in the centroids. Additionally, we only consider the first component of each centroid. The relevant plots are shown in Figure 14. Figure 15 includes a plot of the distribution of true labels within each cluster for both clustering methods.

Examining the bar plot, we see somewhat comparable centroids for the first four clusters between the two methods. The same combination of known networks is active within each pair of centroids. For cluster 0, this argument might be a bit of a stretch. However, for clusters 5 and 6, we observe completely different centroids between the GC and WGC model. Furthermore, the figure shows a very different distribution of true labels within clusters 5 and 6 when comparing the two models. In cluster 5 for the WGC model, when compaing to cluster 5 in the GC model, there are very few observations, while cluster 1 sees a significant increase in observations.

In our earlier analysis of the 6-cluster GC model, we argued that cluster 1 might play a crucial role in the increased NMI score compared to the 5-cluster GC model. This cluster seems to effectively capture observations where the subject is asleep (N1, N2). In the WGC model, cluster 1 now appears to have lost this ability, as clusters 4 and 5 have changed. This does not tell us anything about why WGC, as a methods, preforms worse in general than GC, but rather it gives us a direct explanation for why the NMI score differs for the two $K = 6$ models,

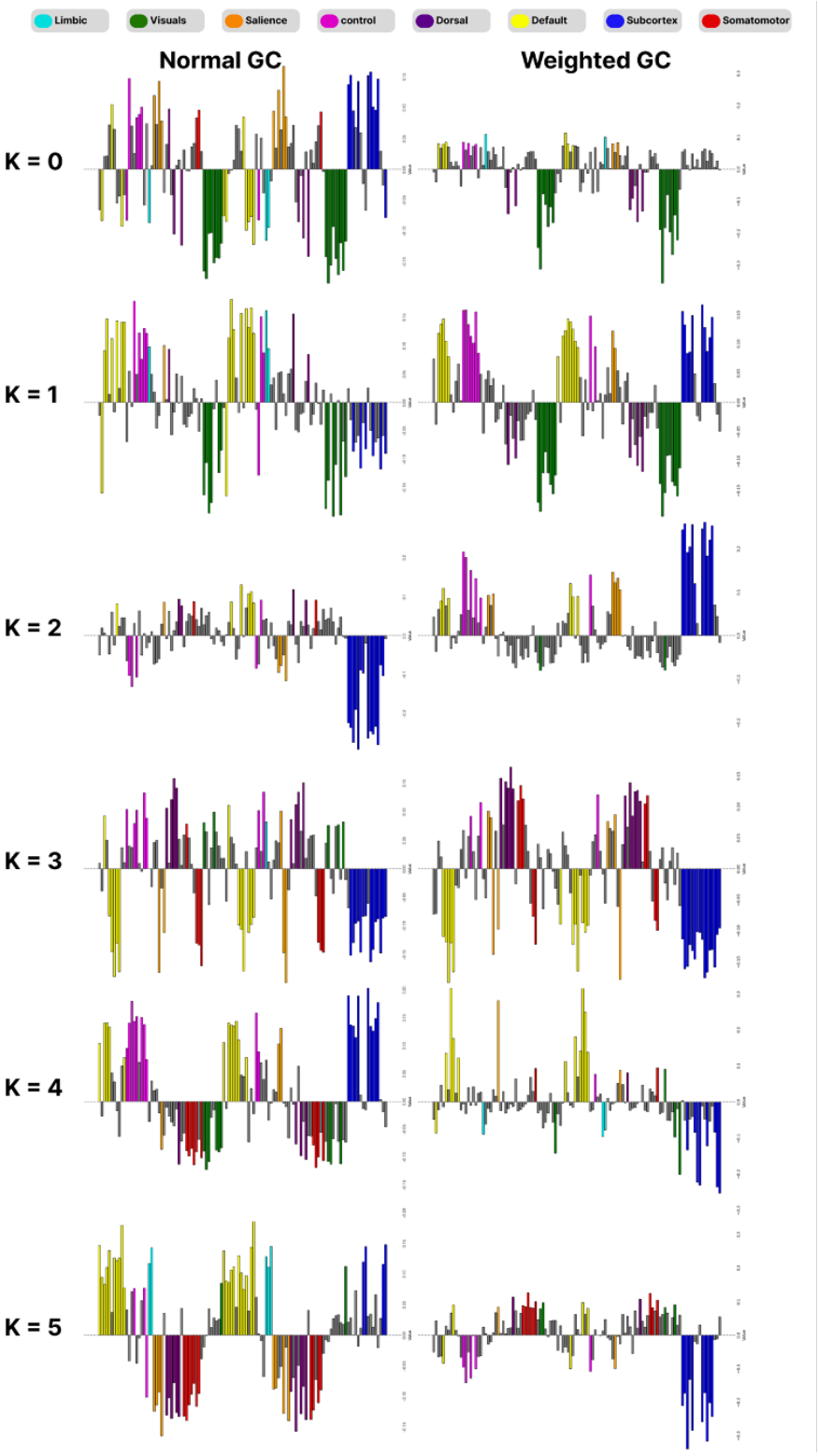


Figure 17: Centroids as barplots, 6 clusters, WGC vs GC

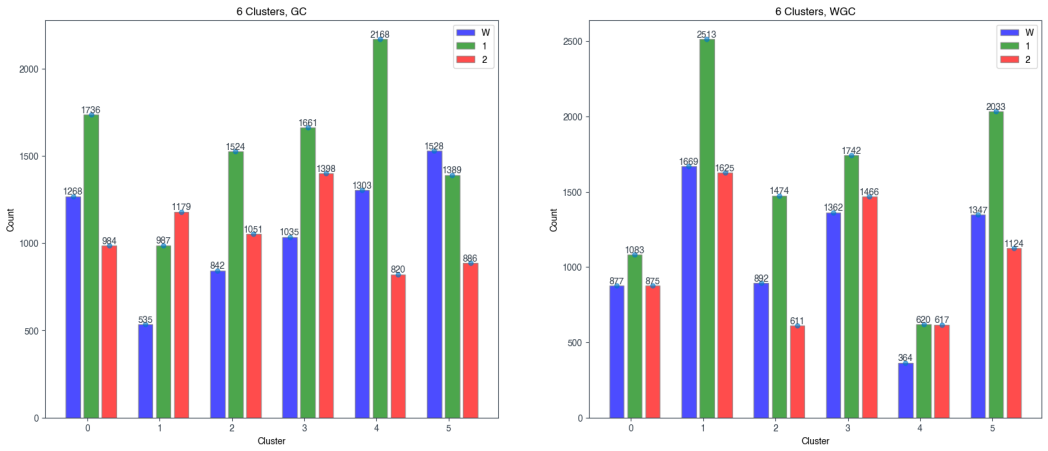


Figure 18: Distribution of true labels within each cluster

6 Discussion

For MACG sampling, no combination of dimensions and clusters seemed to give any results higher than an NMI of 0.8. This could just be because a good portion of the variance in MACG sampling can't be clustered without some uncertainty included in the models.

For the custom eigenvector sampling method, the NMI scoring patterns were a bit more erratic, with KC performing its worst with only a few clusters and the other algorithms performing their best for fewer clusters. We're unsure why this might be, but it could be a good idea to look into the stability of the sampling technique.

Ultimately, we only tested four different algorithms, KC, DC, GC, and WGC. And it might have been worth it to attempt Grassmann clustering with a geodesic distance function, but since we did not manage to find a good method of creating a weighted variant using geodesic distance, it wasn't attempted. There are many different distance functions for Grassmann manifolds which could be interesting to test (Lim et al. [2018]).

One of the biggest issues we faced when clustering using real data was that the labeled data revealed that a lot of the data was largely useless for our purposes, i.e. the true labels when each of the 4 states were counted, before any data preparation:

$$\{\text{W} : 1501,\ \text{NREM1} : 1153,\ \text{NREM2} : 789,\ \text{NREM3} : 39,\ \text{REM} : 0\}$$

This shows a significant lack of two entire stages of sleep NREM3 and REM, and it shows that a big portion of the data is labeled as the Wakeful (W) state, meaning that no information of value can be collected in that case. Overall this just means that this sleep study is very lacking in the various sleep stages, and also very lacking in overall data. Especially because a lot of the patients had to be completely excluded due to how much data was missing/useless. And in general, this type of hard labeling, referred to as sleep-staging, was performed by a Registered Polysomnographic Technologist (Gu et al. [2023]), is both limiting because of the hard boundaries, and we also suspect that the techniques used are too subjective to trust as "true labels".

In the same vein, the WGC algorithm provided results which on their own can be interpreted as mapping out some possibly useful brain networks, but with a worse NMI score in general, when compared to GC. This might be more due to the inaccuracy of the sleep-stage labels rather than the GC algorithm being better than the WGC.

6.1 FN's Sustanablility Development Goals

Our project focuses on analyzing sleep patterns by using different clustering methods, including Weighted Grassmann Clustering and Grassmann Clustering, to evaluate brain dynamics from fMRI measurements. This falls under the 3rd FN Sustainability Development Goal (SDG) "Good Health and Well-Being", as good sleep is cruical for a persons health and well-being. By researching sleep patterns and brain dynamics and improving our understanding we can consequentially aid in diagnosing and treating sleep disorders associated with health issues like cardiovascular diseases, diabetes, depression, and cognitive impairment. Utilizing these innovative methods can result in nuanced insigts into brain connectivity promoting earlier diagnosis of neurological conditions and supporting healthy lives for all ages (American Heart Association).

7 Conclusion

In this project, we employed Grassmann clustering and its weighed variant to analyze sleep stage data derived from fMRI measurements. The primary objective was to explore whether Grassmann clustering could provide new insights and if incorporating eigenvalues enhances the clustering of time-varying brain connectivity patterns compared to traditional methods

Grassmann Clustering demonstrated overall better performance compared to traditional methods such as Euclidean K-Means and Diametrical Clustering. This improvement is evident from the median NMI scores, where Grassmann Clustering consistently outperformed these traditional methods, using MACG sampling. Weighted Grassmann Clustering which also outperformed all methods, with few exceptions, using a custom eigenvector sampling method. In common for both sampling methods, it seemed a general pattern that a low number of clusters could lead to significantly worse performance (except GC), whereas a high number of dimensions generally lead to better performance.

From the Grassmann clustering on real fMRI data, it was found, that although the NMI scores increases with the number of cluster, there were some local spikes in NMI present for specific number of clusters, namely for models with $K = 3$ and $K = 6$. It should be noted however, that when looking at the average NMI score across all number of cluster from 10 random initializations, these results seems to be weakened, indicating that good local NMI score for especially $K = 6$ might be an extreme case.

We performed a detailed analysis of the centroids for the models with $K = 5$ and $K = 6$ clusters, noting a significant increase in the NMI score for the $K = 6$ model. Our findings revealed that the $K = 6$ model contained all the same centroids as the $K = 5$ model, with only minor variations. Additionally, the $K = 6$ model included a unique centroid that was particularly effective at capturing observations with the true labels 'N1' and 'N2'. Consequently, the increased NMI score between the two models can be attributed to Cluster 1 in the $K = 6$ model, which demonstrated clear activation in the default, visual, and subcortex networks.

The generally uniform distribution of true labels within each cluster across all models, after correcting for the unbalanced dataset, indicates that the NMI scores are low. This was confirmed, as we observed NMI scores in the range of 0.002-0.02 across all models. Given the assumed noise in the dataset and the inherent complexity of working with brain data, this result is not surprising. It should not lead to the conclusion that Grassmann clustering (GC) is an ineffective clustering method.

In the last part of this report, we analyzed the impact of the eigenvalues on clustering results using the WGC algorithm. We ran WGC across different numbers of clusters, similar to our approach with GC, allowing for a direct comparison between the two methods. The general trend observed indicates better objective function values and NMI scores with an increasing number

of clusters for both methods. However, the WGC algorithm generally resulted in lower NMI scores compared to the GC algorithm, which was disappointing given the expected advantages of WGC. This discrepancy could potentially be attributed to the highly subjectively labeled data.

To investigate the poorer performance of WGC, we compared the centroids of $K = 6$ models from both methods. Interestingly, we found that Cluster 1 from the $K = 6$ GC model, with some variation, is also present in the WGC model. However, this specific cluster seems to have become less useful now, as other centroids have changed substantially, potentially causing the decrease in NMI for the $K = 6$ WGC model.

References

American Heart Association. Sleep and heart health. https://www.heart.org/en/health-topics/sleep-disorders/sleep-and-heart-health#:~:text=In%20fact%2C%20sleep%20disorders%20have,high%20blood%20pressure%20and%20diabetes. Accessed: 2024-06-23.

Luigi De Gennaro. How america broke its supply chains, 2023. URL https://www.youtube.com/watch?v=5WJgDuPVdhk. Accessed: 2024-06-23, Figure at 3:00.

Peter Gruber and Fabian J. Theis. Grassmann clustering. In *Proceedings of the Conference on Pattern Recognition*, 2006. URL https://www.researchgate.net/publication/225037524_Grassmann_clustering. Accessed: 2024-06-23.

Yameng Gu, Lucas E. Sainburg, Feng Han, and Xiao Liu. Simultaneous eeg and functional mri data during rest and sleep from humans. *Data in Brief*, 48:109059, 2023. doi: 10.1016/j.dib.2023.109059. URL https://doi.org/10.1016/j.dib.2023.109059. Accessed: 2024-06-23.

Hermann Karcher. Riemannian center of mass and so called karcher mean. *arXiv preprint arXiv:1407.2087*, 2014. URL https://arxiv.org/abs/1407.2087v1. Slide 10.

Lek-Heng Lim, Ke Ye, and Rodolphe Sepulchre. Cross-dimensional distances. In *Presentation at University of Chicago*, 2016. URL http://arxiv.org/abs/1807.10883. Supported by DARPA D15AP00109, NSF DMS-1209136, NSF IIS-1546413, Frank Sottile (Texas A&M).

Lek-Heng Lim, Ke Ye, and Rodolphe Sepulchre. Distances between subspaces of different dimensions. http://helper.ipam.ucla.edu/publications/glws1/glws1_15465.pdf, 2018. Accessed: 2024-06-24.

Tim Marrinan, J. Ross Beveridge, Bruce Draper, Michael Kirby, and Chris Peterson. Finding the subspace mean or median to fit your need. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1–9. IEEE, 2014. URL https://www.cv-foundation.org/openaccess/content_cvpr_2014/papers/Marrinan_Finding_the_Subspace_2014_CVPR_paper.pdf. Slide 10.

Anders S. Olsen, Anders Lykkebo-Valløe, Brice Ozenne, Martin K. Madsen, Dea S. Stenbæk, Sophia Armand, Morten Mørup, Melanie Ganz, Gitte M. Knudsen, and Patrick M. Fisher. Psilocybin modulation of time-varying functional connectivity is associated with plasma psilocin and subjective effects. *NeuroImage*, 264:119716, 2022. doi: 10.1016/j.neuroimage.2022.119716. URL https://doi.org/10.1016/j.neuroimage.2022.119716.

Wikipedia contributors. Large-scale brain network. https://en.wikipedia.org/wiki/Large-scale_brain_network, 2024a. Accessed: 2024-06-24.

Wikipedia contributors. Lobes of the brain. https://en.wikipedia.org/wiki/Lobes_of_the_brain, 2024b. Accessed: 2024-06-24.

Wikipedia contributors. Mutual information. https://en.wikipedia.org/wiki/Mutual_information, 2024c. Accessed: 2024-06-24.

Alexander Wittrup, Bertram Nyvold Larsen, and Simon Kjølbye Kristensen. Fagprojekt dtu. GitHub, 2024a. URL https://github.com/AlecHero/Fagprojekt. Accessed: 2024-06-24.

Alexander Wittrup, Bertram Nyvold Larsen, and Simon Kjølbye Kristensen. Weighted grassmann clustering walkthrough. https://github.com/AlecHero/Fagprojekt/blob/main/weighted_grassmann_clustering_walkthrough.ipynb, 2024b. Accessed: 2024-06-24.

Ryoma Yataka and Kazuhiro Fukui. Three-dimensional object recognition via subspace representation on a grassmann manifold. In *Proceedings of the 6th International Conference on Pattern Recognition Applications and Methods (ICPRAM)*, pages 208–216, 2017. doi: 10.5220/0006204702080216. URL https://www.researchgate.net/publication/315865722_Three-dimensional_Object_Recognition_via_Subspace_Representation_on_a_Grassmann_Manifold. Accessed: 2024-06-23.

B. T. Thomas Yeo et al. Seven large-scale functional networks defined by yeo's atlas, 2024. URL https://www.researchgate.net/figure/Seven-large-scale-functional-networks-defined-by-Yeos-atlas-The-whole-cortical-brain_fig1_369268018. Accessed: 2024-06-24.

8 Appendix

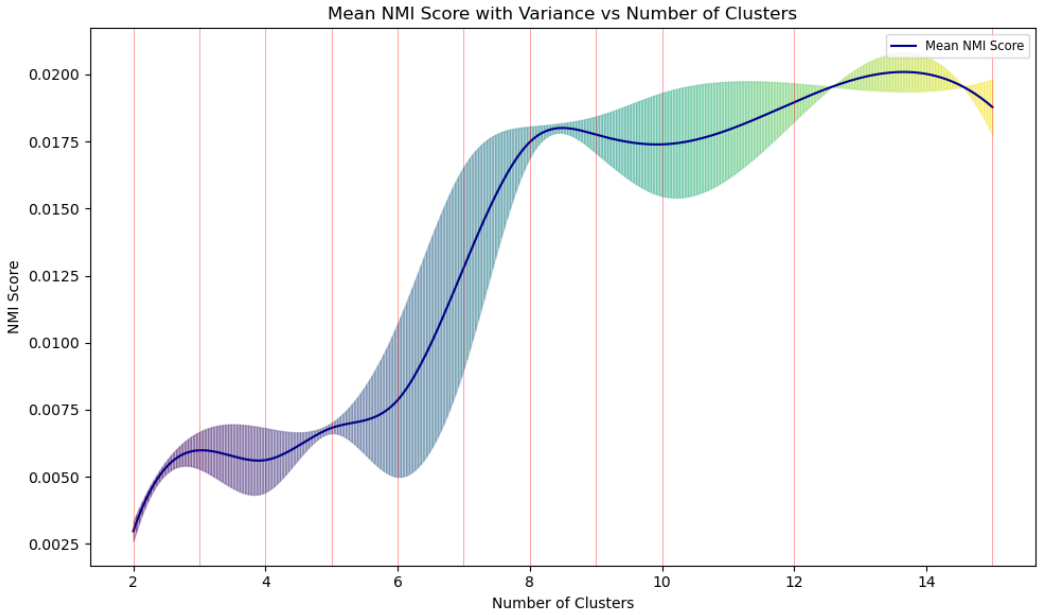


Figure 19: GC, NMI score with mean and variance

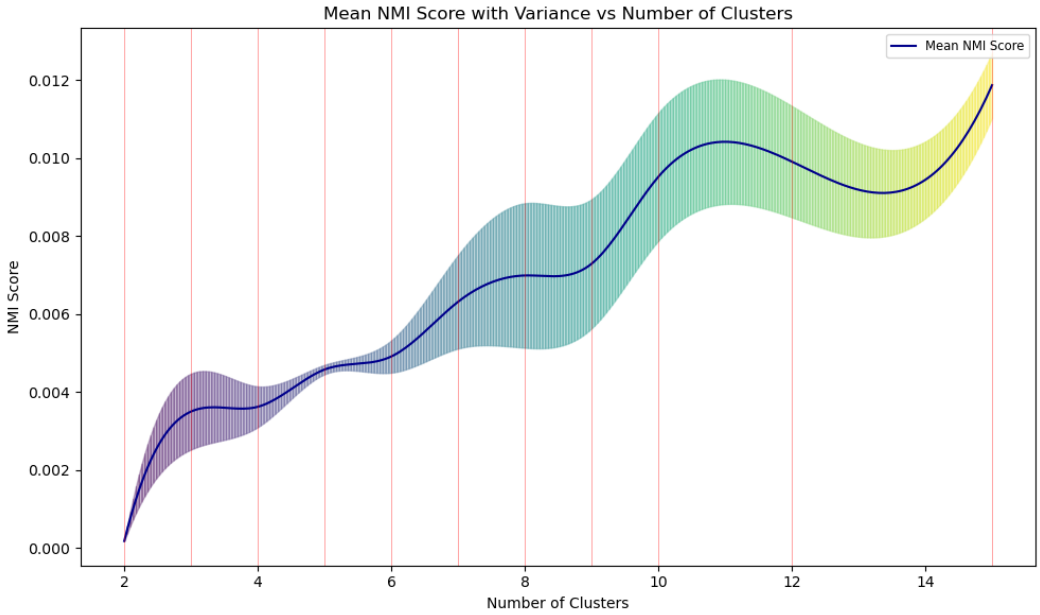


Figure 20: WGC, NMI score with mean and variance