Mixtures of large-scale dynamic functional brain network modes

Chetan Gohil*[1], Evan Roberts*[1], Ryan Timms*[1], Alex Skates[1], Cameron Higgins[1], Andrew Quinn[1], Usama Pervaiz[2], Joost van Amersfoort[3], Pascal Notin[3], Yarin Gal[3], Stanislaw Adaszewski[4], and Mark Woolrich[1]

[1]Oxford Centre for Human Brain Activity (OHBA), Wellcome Centre for Integrative Neuroimaging, Department of Psychiatry, University of Oxford, Oxford, OX3 7JX, United Kingdom

[2]Oxford Centre for Functional MRI of the Brain (FMRIB), Wellcome Centre for Integrative Neuroimaging, Nuffield Department of Clinical Neurosciences, University of Oxford, Oxford OX3 9DU, United Kingdom

[3]Oxford Applied and Theoretical Machine Learning (OATML), Department of Computer Science, University of Oxford, Oxford, OX1 3QD, United Kingdom

[4]Pharma Research and Early Development Operations, Roche Innovation Center Basel, F. Hoffman – La Roche AG, Basel CH-4070, Switzerland

(May 3, 2022)

Abstract

Accurate temporal modelling of functional brain networks is essential in the quest for understanding how such networks facilitate cognition. Researchers are beginning to adopt time-varying analyses for electrophysiological data that capture highly dynamic processes on the order of milliseconds. Typically, these approaches, such as clustering of functional connectivity profiles and Hidden Markov Modelling (HMM), assume mutual exclusivity of networks over time. Whilst a powerful constraint, this assumption may be compromising the ability of these approaches to describe the data effectively. Here, we propose a new generative model for functional connectivity as a time-varying linear mixture of spatially distributed statistical "modes". The temporal evolution of this mixture is governed by a recurrent neural network, which enables the model to generate data with a rich temporal structure. We use a Bayesian framework known as amortised variational inference to learn model parameters from observed data. We call the approach DyNeMo (for Dynamic Network Modes), and show using simulations it outperforms the HMM when the assumption of mutual exclusivity is violated. In resting-state MEG, DyNeMo reveals a mixture of modes that activate on fast time scales of 100-150 ms, which is similar to state lifetimes found using an HMM. In task MEG data, DyNeMo finds modes with plausible, task-dependent evoked responses without any knowledge of the task timings. Overall, DyNeMo provides decompositions that

*Equal contribution.

are an approximate remapping of the HMM's while showing improvements in overall explanatory power. However, the magnitude of the improvements suggests that the HMM's assumption of mutual exclusivity can be reasonable in practice. Nonetheless, DyNeMo provides a flexible framework for implementing and assessing future modelling developments.

1 Introduction

Functional connectivity (FC, [1]) has traditionally been studied across the duration of an experiment, be it metabolic (e.g. [2–5]) or electrophysiological in nature (e.g. [6–9]). Such studies have shown that the brain forms well-defined spatio-temporal networks which are seen both in task [10] and at rest [11]. However, there is a growing body of evidence supporting the idea that these networks are transient [12–14], and that they emerge and dissolve on sub-second time scales. It is now well established that the dynamics of these networks underpin healthy brain activity and cognition [15] and that the disruption of FC is implicated in disease [16, 17].

A systematic understanding of the neuroscientific significance of these networks of whole-brain activity is only facilitated by accurate modelling across the spatial, temporal and spectral domains. Sliding window analyses have been used successfully to study time-varying FC in both M/EEG [13, 18–25] and fMRI [26–35]. Recent studies have calculated very short, or even instantaneous, time-point-by-time-point estimates of FC, which are then combined with a second stage of clustering such as k-means (e.g. [13]) to pool over recurrent patterns of otherwise poorly estimated FC. These two-stage approaches allow access to FC on fast time scales [36, 37].

Although they remain popular, sliding window analyses are a heuristic approach to data analysis and lack a generative model. An alternative approach to studying dynamics of functional brain networks is via the adoption of a formal model. An Hidden Markov Model (HMM) [38] is one such option. As with the two-stage approaches mentioned above, HMMs can pool non-contiguous periods of data together to make robust estimations of the activity of brain networks, including FC. However, they do so by incorporating these two stages into one model. HMMs have been used to show that brain networks evolve at faster time scales than previously suggested by competing techniques (such as independent component analysis) [14]. In the context of M/EEG, HMMs have been used to elucidate transient brain states [12], model sensor level fluctuations in covariance [39] and reveal latent task dynamics attributed to distributed brain regions [10]. More recently, Seedat et al. applied an HMM to detect transient bursting activity which could explain the electrophysiological connectome [40], whilst Higgins et al. were able to show that replay in humans coincides with activation of the default mode network [41].

Although very powerful, convenient, and informative, traditional HMMs are themselves limited in two key ways. Firstly, there is the modelling choice that the state at any time point is only conditionally dependent on the state at the previous time point (i.e. the model is Markovian). This limits the modelling capability of the technique as there is no way for any long-range temporal dependencies between historic state occurrences and the current state to be established. While approaches that use Hidden Semi-Markov Models have been proposed, they are limited in the complexity of long-range temporal dependencies they can capture [42]. Secondly, HMMs adopt a mutually exclusive state model, meaning that data can only be generated by one set of observation model parameters at any given instance. True brain dynamics might be better modelled by patterns that can flexibly combine and mix over time. The mutual exclusivity constraint was found to lead to errors in inferred functional brain network metrics in [43].

We set to address these two limitations in this paper and do so by introducing a new generative model for neuroimaging data. Specifically, we model the time-varying mean and covariance of the data as a linear weighted sum of spatially distributed patterns of activity or "modes". Notably, we do not impose mutual exclusivity on mode activation. Similarly, we drop the assumption that the dynamics of the modes are a function of a Markovian process. This is achieved by using a unidirectional recurrent neural network (RNN) to model the temporal evolution of the weighted sum. The memory provided by the RNN facilitates a richer context to the changes in the instantaneous mean and covariance than what would be afforded by a traditional HMM.

We use Bayesian methods [44] to infer the parameters of our generative model. With this method, we learn a distribution for each parameter, which allows us to incorporate uncertainty into our parameter estimates. Having observed data, we update the distributions to find likely parameters for the model to have generated the data. In this work, we adapt a method used in variational autoencoders [45], which has been called black box stochastic variational inference [46]. One component of this is amortised inference, which works through the deployment of an inference network. In our case the inference network is another RNN, which is bidirectional and learns a mapping from the observed data to the model parameter distributions. The use of an inference network facilitates the scaling and application of this technique to very large datasets, without ever needing (necessarily) to increase the number of inference network parameters to be learnt.

To update our model parameter distributions, we minimise the variational free energy (see Section 2.2) using stochastic gradient descent [46]. We do this by sampling from the model parameter distributions using the reparameterisation trick [45]. The ability to estimate the variational free energy by sampling enables us to use sophisticated generative models that include highly non-linear transformations that would not be feasible when using classical Bayesian methods. Taken together, we call the generative model and inference framework DyNeMo (Dynamic Network Modes).

2 Methods

In this section we outline the generative model and describe the inference of model parameters. We also describe the datasets and preprocessing steps carried out in this work.

An implementation of DyNeMo written in Python using TensorFlow [47] can be accessed here: `https://github.com/OHBA-analysis/osl-dynamics`.

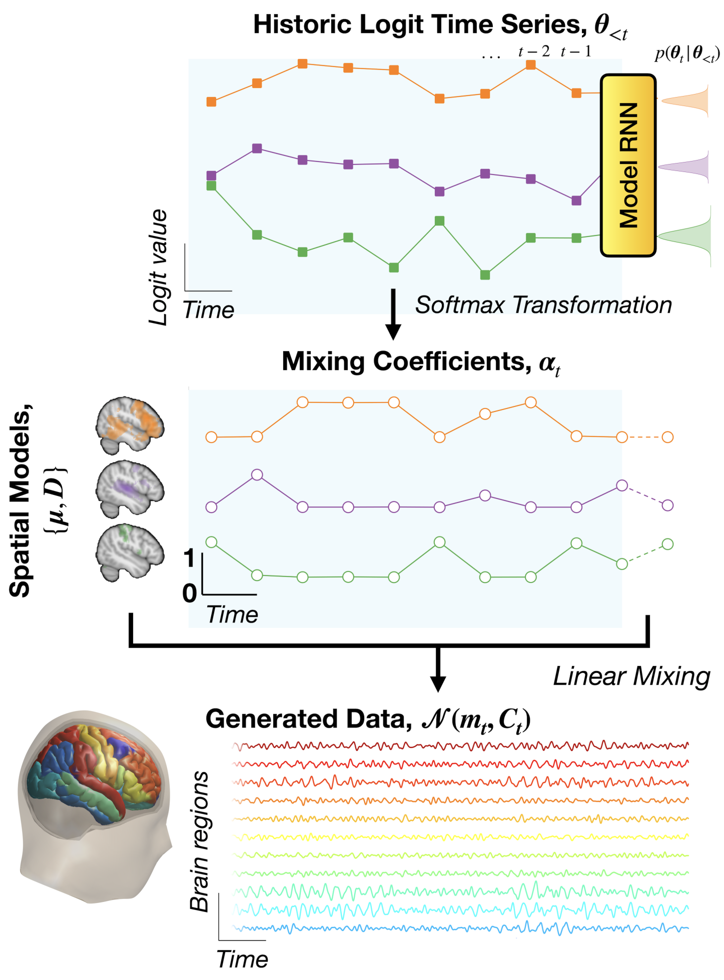

Historic Logit Time Series, $\theta_{<t}$
... $t-2$ $t-1$
$p(\theta_t|\theta_{<t})$
Model RNN
Logit value
Time
Softmax Transformation
Mixing Coefficients, α_t
Spatial Models, $\{\mu, D\}$
1
0
Time
Linear Mixing
Generated Data, $\mathcal{N}(m_t, C_t)$
Brain regions
Time

Figure 1: Generative model employed in DyNeMo. Historic values of a latent logit time series (solid squares, blue background), $\boldsymbol{\theta}_{<t}$, are fed into a unidirectional model RNN. The output of the model RNN parameterises a normal distribution, $p(\boldsymbol{\theta}_t|\boldsymbol{\theta}_{<t})$, which predicts the next logit, $\boldsymbol{\theta}_t$. These logits are transformed via the softmax operation to give the mixing coefficients, $\boldsymbol{\alpha}_t$, (unfilled circles) which are positive and must sum to one at any instance in time. Separate from the dynamics are the corresponding spatial models that describe the brain network activity of each mode (depicted in different colours here); via a mean vector, $\boldsymbol{\mu}$, and covariance matrix, $\boldsymbol{D}$, which captures FC through the metric of correlation. Note that we do not enforce the modes to be spatially separable and that they can overlap in time and space. The mode spatial models combine with the dynamic mixing coefficients to parameterise a multivariate normal distribution with a time-varying mean vector, $\boldsymbol{m}_t$, and covariance matrix, $\boldsymbol{C}_t$.

2.1 Generative Model

Here we propose a model for generating neuroimaging data that explicitly models functional brain networks, including a metric of their functional connectivity, as a dynamic quantity. The model describes the brain activity across multiple brain regions using a set of *modes*, which are constituent elements that can be combined to define time-varying statistics of the data. We refer to them as "modes" to emphasise that the model is not categorical, i.e. that modes should not be mistaken for mutually exclusive *states* (as would be the case in an HMM). Similar to an HMM, our generative model has two components: a latent representation and a data generating process given the latent representation, which is referred to as an *observation model*. In our case, the latent representation is a set of mixing coefficients $\boldsymbol{\alpha}_t$ and the observation model is a multivariate normal distribution. The mean and covariance of the multivariate normal distribution is determined by linearly mixing the modes' spatial models - i.e. means $\boldsymbol{\mu}$ and covariances $\boldsymbol{D}$ - with the coefficients $\boldsymbol{\alpha}_t$. The mixing coefficients are dynamic in nature whereas the modes are static. Therefore, dynamics in the observed data are captured in the dynamics of the mixing coefficients. The mixing coefficients provide a low-dimensional and interpretable description of the data and modes correspond to spatial distributions of activity/FC, where mode-specific FC is captured by the between-brain-region correlations in $\boldsymbol{D}$. Both of these quantities are useful for understanding the data. An overview of the generative model is shown in Figure 1 and a mathematical formulation is given below.

At each time point t there is a probabilistic vector of free parameters, referred to as a *logit* and denoted by $\boldsymbol{\theta}_t$. The logits are distributed in accordance with a multivariate normal distribution,

$$p(\boldsymbol{\theta}_t|\boldsymbol{\theta}_{1:t-1}) = \mathcal{N}(\boldsymbol{\mu}_{\theta_t}(\boldsymbol{\theta}_{1:t-1}), \boldsymbol{\sigma}^2_{\theta_t}(\boldsymbol{\theta}_{1:t-1})), \tag{1}$$

where $\boldsymbol{\theta}_{1:t-1}$ denotes a sequence of historic logits $\{\boldsymbol{\theta}_1, ..., \boldsymbol{\theta}_{t-1}\}$, $\boldsymbol{\mu}_{\theta_t}$ is a mean vector and $\boldsymbol{\sigma}^2_{\theta_t}$ is a diagonal covariance matrix. We use a unidirectional RNN to predict future values of $\boldsymbol{\mu}_{\theta_t}$ and $\boldsymbol{\sigma}_{\theta_t}$ based on previous logits $\boldsymbol{\theta}_{1:t-1}$. The historic values of the logits $\boldsymbol{\theta}_{1:t-1}$ are fed into the RNN:

$$\begin{aligned}\boldsymbol{\mu}_{\theta_t}(\boldsymbol{\theta}_{1:t-1}) &= g_\mu(\text{LSTM}(\boldsymbol{\theta}_{1:t-1})),\\ \boldsymbol{\sigma}_{\theta_t}(\boldsymbol{\theta}_{1:t-1}) &= \xi(g_\sigma(\text{LSTM}(\boldsymbol{\theta}_{1:t-1}))),\end{aligned} \tag{2}$$

where g_μ and g_σ are learnt affine transformations, ξ is a softplus function included to ensure the standard deviations $\boldsymbol{\sigma}_{\theta_t}$ are positive, and LSTM is a type of RNN known as a Long Short Term Memory network [48]. We refer to this network as the *model RNN*. The logits $\boldsymbol{\theta}_t$ are used to determine a set of mixing coefficients,

$$\boldsymbol{\alpha}_t = \zeta(\boldsymbol{\theta}_t), \tag{3}$$

where ζ is a softmax function which assures that the $\boldsymbol{\alpha}_t$ values are positive and sum to one. The mixing coefficients are then used together with a set of spatial modes to calculate a time-varying mean vector and covariance matrix:

$$\begin{aligned}\boldsymbol{m}_t &= \sum_{j=1}^{J} \alpha_{jt}\boldsymbol{\mu}_j,\\ \boldsymbol{C}_t &= \sum_{j=1}^{J} \alpha_{jt}\boldsymbol{D}_j,\end{aligned} \tag{4}$$

where J is the number of modes, $\boldsymbol{\mu}_j$ is the mean vector for each mode, $\boldsymbol{D}_j$ is the covariance matrix for each mode and α_{jt} are the elements of $\boldsymbol{\alpha}_t$.

2.2 Inference

In this section we describe the framework employed to infer the parameters of our generative model. Namely, the logits $\boldsymbol{\theta}_t$, mode means $\boldsymbol{\mu}_j$ and covariances $\boldsymbol{D}_j$. In this work, we use variational Bayesian inference to learn the full posterior distribution for $\boldsymbol{\theta}_t$ and point estimates for $\boldsymbol{\mu}_j$ and $\boldsymbol{D}_j$.

Variational Bayes. In Bayesian inference, we would like to learn a distribution, referred to as the *posterior distribution*, for the variable we are trying to estimate given some data we have observed. In *variational* Bayesian inference, we approximate the posterior distribution with a simple distribution, referred to as the *variational posterior distribution* $q(\boldsymbol{\theta}_t)$, and aim to minimise the Kullback-Leibler (KL) divergence between the variational and true posterior, which amounts to minimising the variational free energy (or equivalently, maximising the evidence lower bound). In classical variational Bayes [49–51], this involves formulating update rules for the parameters of the variational posterior distribution given some observed data. Deriving these update rules is only made possible by limiting the complexity of the generative model for the observed data and restricting the variational posterior to conjugate distributions. In addition to this, we have a separate variational distribution for each variable we are trying to estimate. Also in classical variational Bayes, we learn the parameters of each variational distribution separately, which becomes problematic in terms of computer memory requirements when we wish to estimate a large number of variables.

In brief, we overcome these difficulties with a technique adapted from variational autoencoders [45], known as black box stochastic variational inference [46]. This deploys a neural network (which we call the *inference network*) to perform amortised inference, which helps the approach to scale to large numbers of observations over time; and a sampling technique (known as the *reparameterisation trick*) that allows us to learn a full posterior distribution for $\boldsymbol{\theta}_t$ [45]. We learn point estimates of $\boldsymbol{\mu}_j$ and $\boldsymbol{D}_j$ using trainable free parameters. We update estimates for $\boldsymbol{\mu}_j$, $\boldsymbol{D}_j$, and the posterior distribution parameters of $\boldsymbol{\theta}_t$, to minimise the variational free energy using stochastic gradient descent.

Logits $\boldsymbol{\theta}_t$. Focusing on the full posterior inference of the logits $\boldsymbol{\theta}_t$, here, we use amortised inference [50]. This involves using an inference network to learn a mapping from the observed data to the parameters of the variational posterior. The rationale for this approach is that the computation from past inferences can be reused in future inferences. The use of an inference network fixes the number of trainable parameters to the number of internal weights and biases in the inference network. This is usually significantly smaller than the number of time points, which allows us to efficiently scale to bigger datasets.

Inference network. We now describe the inference network in detail. Having observed the time series $\boldsymbol{x}_{1:N}$, we approximate the variational posterior distribution for $\boldsymbol{\theta}_t$ as

$$q(\boldsymbol{\theta}_t|\boldsymbol{x}_{1:N}) = \mathcal{N}(\boldsymbol{m}_{\theta_t}(\boldsymbol{x}_{1:N}), \boldsymbol{s}^2_{\theta_t}(\boldsymbol{x}_{1:N})), \tag{5}$$

where $\boldsymbol{m}_{\theta_t}$ and $\boldsymbol{s}^2_{\theta_t}$ are the variational posterior mean and covariance of a multivariate normal distribution respectively. The variational posterior covariance is a diagonal matrix. We use a bidirectional RNN for the inference network, which we refer to as the *inference RNN*. This network outputs the parameters of the variational posterior distribution given the observed data:

$$\begin{aligned} \boldsymbol{m}_{\theta_t}(\boldsymbol{x}_{1:N}) &= f_m(\text{BLSTM}(\boldsymbol{x}_{1:N})) \\ \boldsymbol{s}_{\theta_t}(\boldsymbol{x}_{1:N}) &= \xi(f_s(\text{BLSTM}(\boldsymbol{x}_{1:N}))), \end{aligned} \tag{6}$$

where f_m and f_s are affine transformations and BLSTM denotes a bidirectional LSTM. The complete DyNeMo framework and interplay between the generative model and inference network is shown in Figure 2.

Loss function. Having outlined the inference network for the logits, we turn our attention to the loss function used in DyNeMo. In variational Bayesian inference we infer a parameter, in this case $\boldsymbol{\theta}_t$, by minimising the variational free energy [52],

$$F = -\int q(\boldsymbol{\theta}_{1:N}|\boldsymbol{x}_{1:N}) \log \left(\frac{p(\boldsymbol{x}_{1:N}|\boldsymbol{\theta}_{1:N}) p(\boldsymbol{\theta}_{1:N})}{q(\boldsymbol{\theta}_{1:N}|\boldsymbol{x}_{1:N})} \right) \mathrm{d}\boldsymbol{\theta}_{1:N}, \tag{7}$$

where $p(\boldsymbol{\theta}_{1:N})$ is the prior and $p(\boldsymbol{x}_{1:N}|\boldsymbol{\theta}_{1:N})$ is the likelihood. With this approach the inference problem is cast as an optimisation problem, which can be efficiently solved with the use of stochastic gradient descent [46]. Here, we make stochastic estimates of a loss function, and use the gradient of the loss function to update the trainable parameters in our model. However, to estimate the loss function we must calculate the integral in Equation (7). In DyNeMo, this is done using a sampling technique (i.e. the reparameterisation trick) to give Monte Carlo estimates of the loss function.

Insight into the loss function is gained by re-writing Equation (7) as two terms (see SI 8.1):

$$F = -\text{LL} + \text{KL}. \tag{8}$$

The first term is referred to as the *log-likelihood term* and the second term is referred to as the *KL divergence term*. The log-likelihood term acts to give the most probable estimate for the logits that could generate the training data and the KL divergence term acts to regularise the estimate. Relating this to components of DyNeMo, it is the inference RNN that infers the logits, which together with the learnt mode means and covariances determine the log-likelihood term, whilst the model RNN regularises the inferred logits through its role as the prior in the KL divergence term.

We now detail the calculation used to estimate the loss function. The log-likelihood term is given by

$$\text{LL} = \sum_{t=1}^{N} \log(p(\boldsymbol{x}_t|\boldsymbol{\theta}_t^1)), \tag{9}$$

where $p(\boldsymbol{x}_t|\boldsymbol{\theta}_t^1)$ is the likelihood of generating data $\boldsymbol{x}_t$ at time point t given the latent variable is $\boldsymbol{\theta}_t^1$, which is a sample from the variational posterior $q(\boldsymbol{\theta}_t|\boldsymbol{x}_{1:N})$. The superscript in $\boldsymbol{\theta}_t^1$ indicates that it is the first sample from $q(\boldsymbol{\theta}_t|\boldsymbol{x}_{1:N})$. Only one sample from the variational posterior at each time point is used to estimate the log-likelihood term. Note that the likelihood is a multivariate normal whose mean and covariance is determined by Equation

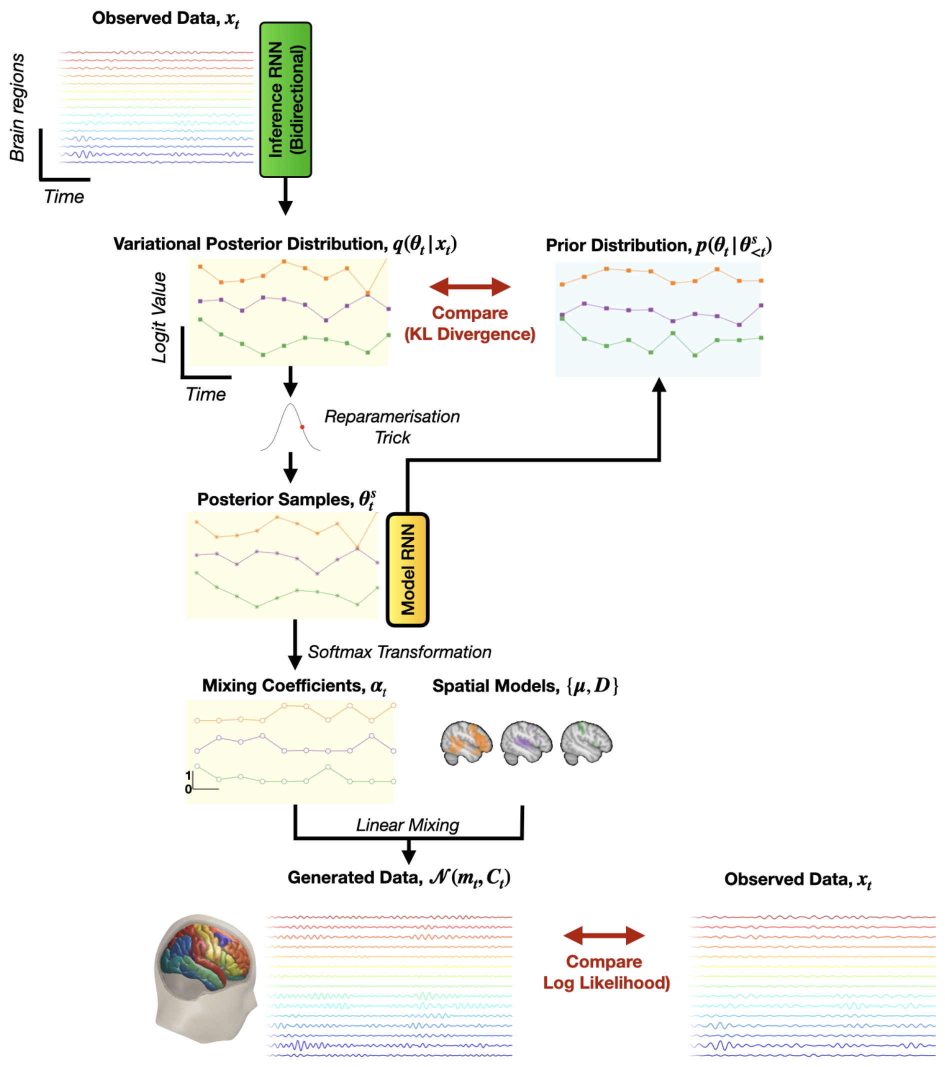

Observed Data, x_t
Brain regions
Time
Inference RNN (Bidirectional)
Variational Posterior Distribution, $q(\theta_t|x_t)$
Logit Value
Time
Compare (KL Divergence)
Prior Distribution, $p(\theta_t|\theta^s_{<t})$
Reparamerisation Trick
Posterior Samples, θ^s_t
Model RNN
Softmax Transformation
Mixing Coefficients, α_t
1
0
Spatial Models, $\{\mu, D\}$
Linear Mixing
Generated Data, $\mathcal{N}(m_t, C_t)$
Compare Log Likelihood)
Observed Data, x_t

Figure 2: The full DyNeMo framework. A sequence of observed data, $\boldsymbol{x}_t$, is fed into a bidirectional RNN which parameterises the approximate variational posterior distribution for the logit time series, $q(\boldsymbol{\theta}_t|\boldsymbol{x}_t)$. The variational free energy is estimated by sampling from the posterior distribution using the reparameterisation trick to give $\boldsymbol{\theta}^s_t$, which are fed into the model RNN and used to calculate the mixing coefficients $\boldsymbol{\alpha}_t$. The approximate variational posterior distribution is used with the prior distribution from the model RNN to estimate the KL divergence term of the variational free energy. The mixing coefficients are combined with point estimates for the spatial models of each mode to define the parameters of the observation model, $\boldsymbol{m}_t$ and $\boldsymbol{C}_t$, which are used to estimate the log-likelihood term of the variational free energy. This is done for a batch of sequences and the trainable parameters are updated using backpropagation until the variational free energy converges.

(4). Therefore, the likelihood depends on the logits $\boldsymbol{\theta}_t$, mode means $\boldsymbol{\mu}_j$ and covariances $\boldsymbol{D}_j$.

The KL divergence term is given by

$$\text{KL} = \sum_{t=2}^{N} D_{\text{KL}}(q(\boldsymbol{\theta}_t|\boldsymbol{x}_{1:N}) \;||\; p(\boldsymbol{\theta}_t|\boldsymbol{\theta}^1_{1:t-1})), \tag{10}$$

where $p(\boldsymbol{\theta}_t|\boldsymbol{\theta}^1_{1:t-1})$ is the prior distribution for $\boldsymbol{\theta}_t$ given a single sample for the previous logits $\boldsymbol{\theta}^1_1, ..., \boldsymbol{\theta}^1_{t-1}$ from their respective variational posteriors $q(\boldsymbol{\theta}_1|\boldsymbol{x}_{1:N}), ..., q(\boldsymbol{\theta}_{t-1}|\boldsymbol{x}_{1:N})$ and D_{KL} is the KL divergence [49] between the variational posterior and prior. A full derivation of the loss function is given in SI 8.1.

Reparameterisation trick. Next, we outline the method used to sample from the variational posterior distribution $q(\boldsymbol{\theta}_t|\boldsymbol{x}_{1:N})$. This is a multivariate normal distribution with mean vector $\boldsymbol{m}_{\theta_t}(\boldsymbol{x}_{1:N})$ and diagonal covariance matrix $\boldsymbol{s}^2_{\theta_t}(\boldsymbol{x}_{1:N})$. To obtain a sample $\boldsymbol{\theta}^s_t$ from $q(\boldsymbol{\theta}_t|\boldsymbol{x}_{1:N})$, we use the reparameterisation trick [45], where we sample from a normal distribution,

$$\boldsymbol{\epsilon} \sim \mathcal{N}(0, \boldsymbol{I}), \tag{11}$$

where $\boldsymbol{I}$ is the identity matrix. $\boldsymbol{\epsilon}^s$ denotes the s^{th} sample from $\mathcal{N}(0, \boldsymbol{I})$. We calculate the samples for the logits as

$$\boldsymbol{\theta}^s_t = \boldsymbol{m}_{\theta_t}(\boldsymbol{x}_{1:N}) + \boldsymbol{s}_{\theta_t}(\boldsymbol{x}_{1:N})\boldsymbol{\epsilon}^s, \tag{12}$$

where $\boldsymbol{s}_{\theta_t}(\boldsymbol{x}_{1:N})$ is a vector containing the square root of the diagonal from $\boldsymbol{s}^2_{\theta_t}(\boldsymbol{x}_{1:N})$. The use of the reparameterisation trick allows us to directly minimise the loss function using stochastic gradient descent. In addition, this method produces low variance estimates the gradient with respect to the inference RNN weights and baises. This allows us efficiently train the inference RNN with backpropagation [53].

Mode means $\boldsymbol{\mu_j}$ and covariances $\boldsymbol{D_j}$. Having detailed the inference of the logits $\boldsymbol{\theta}_t$ and the calculation of the loss function, we now turn our attention to the spatial models described by the means $\boldsymbol{\mu}_j$ and covariances $\boldsymbol{D}_j$. We performed fully Bayesian inference on the logits, as they are temporally local parameters, and hence will have reasonably large amounts of uncertainty in their estimation which needs to be propagated to the inference of $\boldsymbol{\theta}_t$ over time. By contrast, the mode means $\boldsymbol{\mu}_j$ and covariances $\boldsymbol{D}_j$ are global parameters whose inference can draw on information over all time points. As a result we choose to use point estimates for $\boldsymbol{\mu}_j$ and $\boldsymbol{D}_j$, which are learnt using trainable free parameters. Additionally, learning point estimates when they are sufficient has the advantage of simplifying inference.

The time-varying mean vector $\boldsymbol{m}_t$ constructed from the mode means $\boldsymbol{\mu}_j$ can take on any value, and can therefore be treated as free parameters. However, the time-varying covariance $\boldsymbol{C}_t$ constructed from the $\boldsymbol{D}_j$ matrices is required to be positive definite. We enforce this by parameterising the $\boldsymbol{D}_j$'s using the Cholesky decomposition,

$$\boldsymbol{D}_j = \boldsymbol{L}_j \boldsymbol{L}'_j, \tag{13}$$

where $\boldsymbol{L}_j$ is a lower triangular matrix known as a Cholesky factor and $'$ denotes the matrix transpose. We learn $\boldsymbol{L}_j$ as a vector of free parameters that is used to fill a lower triangular matrix. We also apply a softplus operation and add a small positive value to the diagonal of the Cholesky factor to improve training stability. Using this approach, we learn point estimates for the mode means and covariances.

Hyperparameters, initialisation and training. The full DyNeMo model contains several hyperparameters, for example the number of layers and hidden units in the RNNs, the batch size, the learning rate, and many more. These all must be specified before training the model. DyNeMo also contains a large number of trainable parameters, which must be initialised. A description of the hyperparameters and the initialisation of trainable parameters is given in SI 8.2. Hyperparameters for each dataset used in this work are summarised in Table 1. There are also several techniques that can be used to improve model training, such as KL annealing [54] and using multiple starts. These are also discussed in detail in SI 8.2.

Table 1: Hyperparameters (see SI 8.2) used in simulation and real data studies.

Hyperparameter	**Simulation 1**	**Simulation 2**	**MEG Data**
Number of modes, J	3	6	10
Sequence length, N	200	200	200
Inference RNN hidden units	64	64	64
Model RNN hidden units	64	64	64
KL annealing sharpness, A_S	10	10	10
KL annealing epochs, n_{AE}	100	100	300
Training epochs n_{E}	200	200	600
Batch size	16	16	32
Learning rate, η	0.01	0.01	0.0025
Gradient clip (norm.)	-	-	0.5
Number of multi-starts	-	-	10
Multi-start epochs	-	-	20

2.3 Datasets

In this section, we describe the data used to train DyNeMo. This includes simulated data, described in Sections 2.3.1 and 2.3.2, which was used to evaluate DyNeMo's modelling and inference capabilities, and real MEG data, described in Section 2.3.3, which was used for neuroscientific studies.

2.3.1 Simulation 1: Long-Range Dependencies

The first simulation dataset was used to examine DyNeMo's ability to learn long-range temporal dependencies in the underlying logits. In simulation 1, data were generated using a Hidden Semi-Markov Model[1] (HSMM) [55]. In this simulation, we used a gamma distribution (with shape and scale parameters of 5 and 10 respectively) to sample state lifetimes. We use a transition probability matrix with self-transitions excluded to determine the sequence of states to sample a lifetime for. The transition probability matrix and ground truth mode covariances are shown in Figures 4a and 4b respectively. A multivariate time series with

[1]Unlike in a vanilla HMM, the state lifetimes are explicitly modelled in an HSMM. This enables the possibility of long-lived states to exist, which would otherwise be improbable in an HMM.

80 channels, 25,600 samples and 3 hidden states was generated using an HSMM simulation with a multivariate normal observation model. A zero mean vector was used for each mode and covariances were generated randomly. The ground truth state time course and lifetime distribution of this simulation is shown in Figures 4c and 4d respectively.

2.3.2 Simulation 2: Linear Mode Mixing

The second simulation dataset was used to examine DyNeMo's ability to infer a linear mixture of co-activating modes. Here, we simulated a set of J sine waves with different amplitudes, frequencies and initial phases to represent the logits $\boldsymbol{\theta}_t$. We applied a softmax operation at each time point to calculate the ground truth mixing coefficients $\boldsymbol{\alpha}_t$. A multivariate normal distribution with zero mean and randomly generated covariances was used for the observation model. A multivariate time series with 80 channels, 25,600 samples and 6 hidden modes was simulated. The first 2,000 time points of the simulated logits and mixing coefficients are shown in Figures 5a and 5b respectively.

2.3.3 MEG Data

In addition to the simulation datasets, we trained DyNeMo on two real MEG datasets: a resting-state and a (visuomotor) task dataset. The MEG datasets were source reconstructed to 42 regions of interest. The raw data, preprocessing and source reconstruction are described below.

Raw data and preprocessing. Data from the UK MEG Partnership were used in this study. The data were acquired using a 275-channel CTF MEG system operating in third-order synthetic gradiometry at a sampling frequency of 1.2 kHz. Structural MRI scans were acquired with a Phillips Achieva 7 T. MEG data were preprocessed using the OHBA software library (OSL, [56]). The time series was downsampled to 250 Hz before a notch filter at 50 Hz (and harmonics) was used to remove power line noise. The data were then bandpass filtered between 1 and 98 Hz. Finally, an automated bad segment detection algorithm in OSL was used to remove particularly noisy segments of the recording. No independent component analysis was applied to identify artefacts.

Source reconstruction. Structural data were coregistered with the MEG data using an iterative close-point algorithm; digitised head points acquired with a Polhemous pen were matched to individual subject's scalp surfaces extracted with FSL's BET tool [57, 58]. We used the local spheres head model in this work [59]. Preprocessed sensor data were source reconstructed onto an 8 mm isotropic grid using a linearly constrained minimum variance beamformer [60]. Voxels were then parcellated into 42 anatomically defined regions of interest, before a time series for each parcel was extracted by applying Principal Component Analysis (PCA) to each region of interest. Spatial leakage between parcel time courses was reduced using the symmetric multivariate leakage reduction technique described in [61], which unlike pairwise methods has the benefit of reducing leakage caused by so-called ghost interactions [62]. We will refer to each parcel as a *channel*.

Resting-state dataset. The resting-state dataset is formed from the MEG recordings of 55 healthy participants (mean age 38.3 years, maximum age 62 years, minimum age 19 years, 27 males, 50 right handed). The participants were asked to sit in the scanner with their eyes open while 10 minutes of data were recorded.

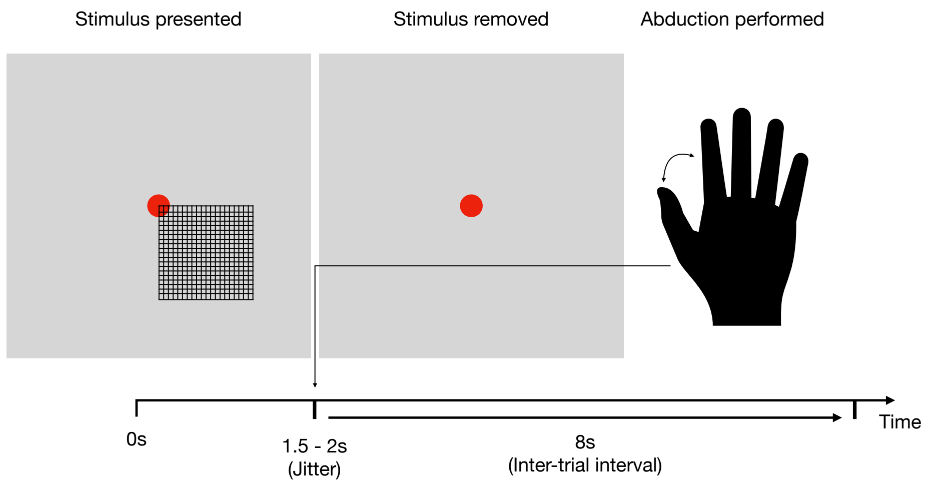


Figure 3: The structure of the visuomotor task. Participants are presented with an onscreen grid. After a period of between 1.5 and 2 seconds, the grid is removed. Upon grid removal, the participant performs a right-hand index finger abduction. Between the removal of the grid and its reappearance for the next trial, there is an 8 second inter-trial interval.

Task dataset. The task dataset is formed from MEG recordings of 51 healthy participants (mean age 38.4 years, maximum age 62 years, 24 males, 46 right handed). The recordings were taken while the participants performed a visuomotor task [63]. Participants were presented with a high-contrast grating. The grating remained on screen for a jittered duration between 1.5 and 2 seconds. When the grating was removed, the participants performed an abduction using the index finger and thumb of the right hand. This abduction response was measured using an electromyograph on the back of the hand. From the grating removal, an 8 second inter trial interval is incorporated until the grating re-appeared on the screen. The structure of the task is shown in Figure 3. A total of 1,837 trials are contained in this dataset. The majority of participants in the UK MEG Partnership study have both resting-state and task recordings. 48 of the participants in the resting-state and task dataset are the same.

Data preparation. Before training DyNeMo, we further prepare the preprocessed data by performing the following steps. The first step is used to encode spectral information into the observation model, whereas the other two are to help train the model. These steps were only performed on the MEG datasets. The steps are:

1. Time-delay embedding. This involves adding extra channels with time-lagged versions of the original data. We use 15 embeddings, which results in a total of 630 channels. By doing this, we introduce additional off-diagonal elements to the covariance matrix,

which contains the covariance of a channel with a time-lagged version of itself. This element of the covariance matrix is the autocorrelation function of the channel for a given lag [64]. As the autocorrelation function captures the spectral properties of a signal, this allows the model to learn spectral features of the data as part of the covariance matrix.

2. PCA. After time-delay embedding we are left with 630 channels. This is too much for modern GPUs to hold in memory. Therefore, we use PCA for dimensionality reduction down to 80 channels.

3. Standardisation (z-transform) across the time dimension. This is a common transformation that has been found to be essential in many optimisation problems [46]. Standardisation is the final step in preparing the training data.[2]

Time-delay embedding and PCA are summarised in Figure 15.

2.4 Post-hoc Analysis of Learnt Latent Variables

Once trained, DyNeMo provides a set of mode mixing coefficients, $\boldsymbol{\alpha}_t$, which contain a description of latent dynamics in the training data. We can use the mode mixing coefficients to estimate quantities that characterise the training data. We describe such analyses below.

Summary statistics. We can summarise the mixing coefficients with statistics, which can give a high-level description of the data. We can take inspiration from the Viterbi path of an HMM[3] [49], which is typically summarised with state lifetimes[4], interval times and the fractional occupancies [14]. One benefit of the mutual exclusivity assumption made by the HMM is that there are well defined time points when a state is active, making determining a lifetime and interval time straightforward. Contrastingly, DyNeMo provides a description where multiple modes are simultaneously present at each time point. To define when a mode is active we fit a two-component Gaussian Mixture Model (GMM) to the logit time course of each mode. One of the Gaussian components corresponds to time points when the mode is active whereas the other component corresponds to time points when the mode is inactive. This GMM therefore gives us a mode activation time course, which we can use to compute the usual summary statistics we would calculate with a Viterbi path. Note, we fit a GMM separately to each mode, which enables the possibility of there being time points where multiple modes or no modes activate. The mixing coefficients have a sum to one constraint, which means their distribution is non-Gaussian. Therefore, we transform the data using the logit function[5] and standardise before fitting the GMM to make the distribution more Gaussian.

[2]Note, standardisation was also performed before PCA.

[3]The Viterbi path is the argmax of the inferred state time course.

[4]Also known as the dwell time.

[5]$\text{logit}(x) = \ln\left(\frac{x}{1-x}\right)$.

Mode spectral properties. Neuronal activity in the brain has oscillatory dynamics. A useful quantity for examining these oscillations is the power spectral density (PSD), which displays the power at each frequency for a given channel (i.e. region of interest). Additionally, the cross spectral density, which displays the power coupling across two channels, is of interest. We can calculate power and cross spectra for each mode directly from the training data once we have inferred the mixing coefficients. Equation (4) defines a linear mixture of mode covariances[6]. A property of this model is that the PSD of each mode mixes with the same coefficients. We can exploit this property to estimate the spectral properties of each mode. We do this by first calculating a cross spectrogram using the dataset and fitting a linear regression model using the mixing coefficients:

$$\boldsymbol{P}_t(f) = \sum_{j=1}^{J} \alpha_{jt} \boldsymbol{P}_j(f) + \boldsymbol{P}_0(f), \tag{14}$$

where $\boldsymbol{P}_t(f)$ is the cross spectrogram of each pair of channels, $\boldsymbol{P}_j(f)$ are regression coefficients, which are our mode cross spectra, $\boldsymbol{P}_0(f)$ is an residual term and f is the frequency. We standardise (z-transform) the mixing coefficients across the time dimension before calculating the linear regression. This results in the residual term $\boldsymbol{P}_0(f)$ corresponding to the mean PSD (averaged over time) and the regression coefficients $\boldsymbol{P}_j(f)$ corresponding to PSDs relative to the mean. We calculate the cross spectrogram with the source reconstructed data using Welch's method [65], which involves segmenting the data into overlapping windows and performing a Fourier transform:

$$P_{mn,t}(f) = \frac{1}{f_s W^2} X_{m,t}(f) X^*_{n,t}(f), \tag{15}$$

where m and n are channels, f_s is the sampling frequency, W is the number of samples in the window, $X_{m,t}(f)$ is the Fourier transform of the data in a window centred at time t and $*$ denotes the complex conjugate.

Mode power maps. Additionally, the spatial distribution of power across the brain is of interest. We can examine the spatial power distribution of each mode separately, which can highlight the areas of the brain that are active for each mode. Using the cross spectrogram regression method (Equation (14)), we obtain a PSD for each channel for a given mode. As each channel corresponds to a region of interest, we obtain a PSD for the activity at each region of interest. The integral of a PSD is the power. Plotting the power at each region of interest as a two-dimensional heat map projected onto the surface of the brain shows the spatial distribution of power for a given mode. It is often more interesting to plot the power relative to a reference rather than the absolute value to highlight differences in the power maps of each mode. In this work, we calculate the power by integrating the mode PSDs $\boldsymbol{P}_j(f)$ without the mean term $\boldsymbol{P}_0(f)$. This gives us the power distribution relative to the mean power common to all modes. We also subtract the mean power across modes for each region of interest separately when displaying the power maps to help highlight relative differences between the modes.

[6]In this work, we do not model the mean vector for each mode. We set each mode's mean to zero.

Mode FC maps. In addition to power maps, FC maps reveal the brain networks that are present for each mode. We use the coherence as our measure of FC, which quantifies the stability of phase difference between two regions of interest, thereby providing a measure of synchronisation or phase-locking between brain regions. To calculate the coherence we use the mode cross spectra estimated with the linear regression model in Equation (14). Estimating the coherence for a window requires us to average multiple estimates of the cross spectra within that window. We do this by dividing each window into a set of sub-windows and taking the average cross spectra for each sub-window. Using the cross spectra we calculate a coherence for each pair of channels. However, most of these connections correspond to a background level of activity that is are common to all modes. To identify the most prominent connections, we fit a two-component GMM to the distribution of coherences over brain region edges. We standardise the coherence values before fitting the GMM. One component of the GMM corresponds to a population of background FC, whereas the other corresponds to prominently high coherence values. We retain the coherences that correspond to the latter component and plot the top 5% of connections in FC maps.

To calculate power and FC maps, we use a window length of 4 seconds, sub-window length of 0.5 seconds, window step size of 0.08 seconds, and apply a Hann windowing function to each sub-window before performing the Fourier transform to calculate the cross spectrogram. We calculate the cross spectrogram and perform the linear regression separately for each subject. We then average the subject-specific mode PSDs and coherences over subjects to obtain group-level spectra. We also apply a Hann function to sub-windows of the mixing coefficient time series to match the windowing applied to the data and average the values across a full window to calculate the α_{jt} value to use in the regression. To calculate the spectrograms used to the study the evoked response to task (Section 3.4) we use a window length of 0.5 seconds and do not separate the window into sub-windows.

3 Results

3.1 Simulation 1: Long-Range Dependencies

A simulation dataset was used to examine DyNeMo's ability to learn long-range temporal dependencies. DyNeMo was trained on the simulation dataset described in Section 2.3.1. An HMM was also trained on the simulated data for comparison. In this simulation, a mutually exclusive hidden state was used to generate the training data. The ground truth hidden state time course is shown in Figure 4c. DyNeMo was able to correctly infer mutually exclusive modes, which we can think of as states. The DyNeMo and HMM inferred state time courses are also shown in Figure 4c. Both DyNeMo and the HMM are able to infer the presence of long-range dependencies by matching the ground truth, non-exponential, state lifetime distributions (shown in Figure 4d). A dice coefficient (model inferred vs ground truth) of greater than 0.99 is achieved for both models. However, this does not mean that the HMM or DyNeMo generative models have necessarily learnt long-range dependencies, as the inferred state time courses could be a result of purely data-driven information. To test this, we can sample state time courses from the trained HMM and DyNeMo generative models and examine their lifetime distributions. Figure 4e shows the sampled state time courses. The state lifetime distribution of the sample from DyNeMo captures the non-exponential ground truth distribution, demonstrating its ability to learn long-range temporal dependencies over the scale of at least 50 samples. Contrastingly, the HMM was not able to generate any long-range temporal dependencies, indicating that, as expected, it is only able to capture short-range dependencies.

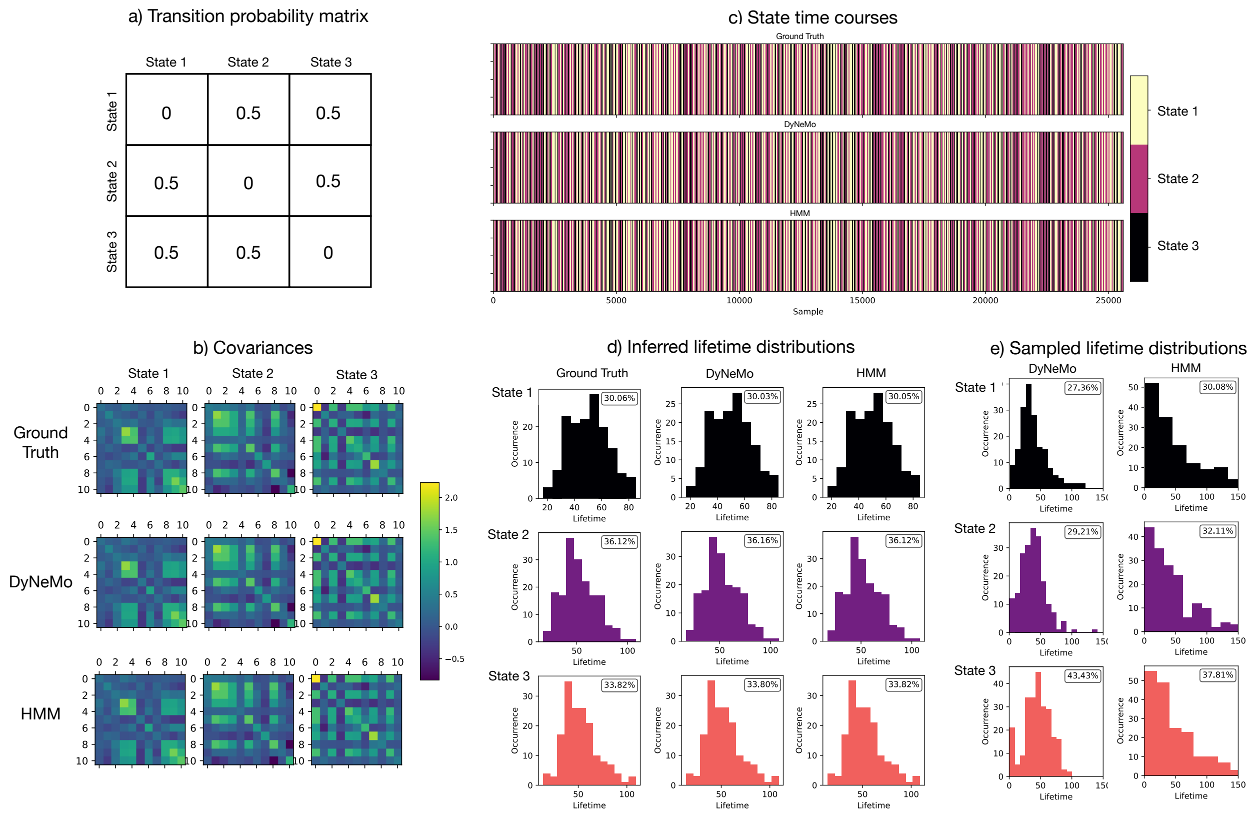


Figure 4: DyNeMo is able to learn long-range temporal dependencies in the latent dynamics of simulated data. Parameters of an HSMM simulation are shown along with the parameters inferred by DyNeMo and an HMM. While both DyNeMo and the HMM were able to accurately *infer* the hidden state time course and their lifetime distributions, actual *samples from each model* show that only DyNeMo was able to learn the lifetime distribution of the states within its generative model, demonstrating its ability to learn long-range temporal dependencies. a) Transition probability matrix used in the simulation. b) Covariances: simulated (top), inferred by DyNeMo (middle) and inferred by an HMM (bottom). c) State time courses: simulated (top), inferred by DyNeMo (middle) and inferred by an HMM (bottom). Each colour corresponds to a separate state. d) Lifetime distribution of inferred state time courses. e) Lifetime distribution of sampled state time courses. The fractional occupancy of each state is shown as a percentage in each histogram plot.

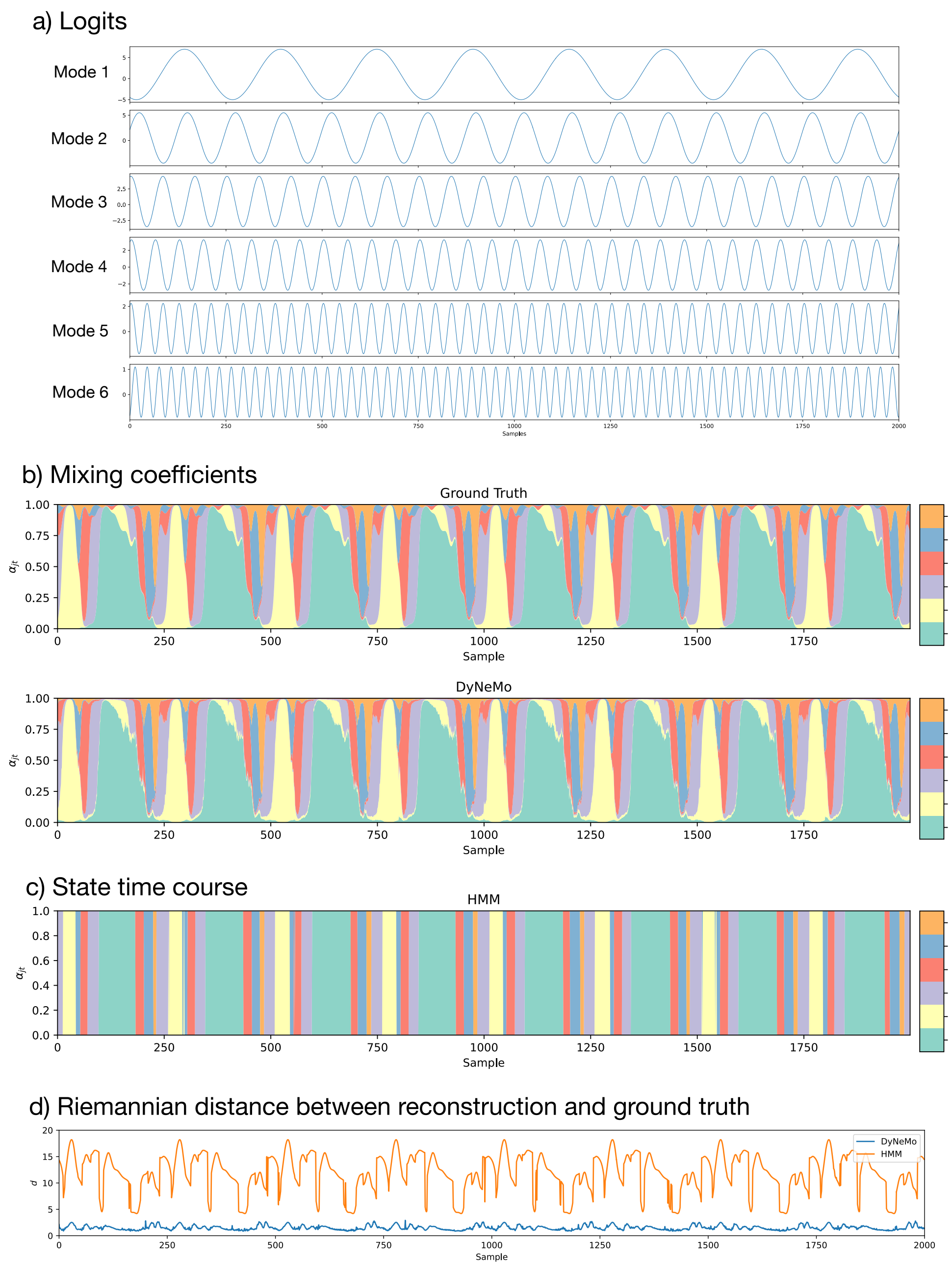


Figure 5: DyNeMo is able to accurately infer a linear mixture of modes. DyNeMo was trained on a simulation with co-activating modes. The mixing coefficients inferred by DyNeMo follow the same pattern as the ground truth. The failure of an HMM in modelling this type of simulation due to its inherent assumption of mutual exclusivity is also shown. a) Logits used to simulate training data. b) Mixing coefficients of the simulation (top) and inferred by DyNeMo (bottom). c) State time course inferred by an HMM. d) Riemannian distance between the reconstruction of the time-varying covariance, $\boldsymbol{C}_t$, (via Equation (4)) and the ground truth for DyNeMo and the HMM. Only the first 2000 time points are shown in each plot.

3.2 Simulation 2: Linear Mode Mixing

In contrast to the mutual exclusivity assumption of the HMM, DyNeMo has the ability to infer a linear a mixture of modes. To test this we trained DyNeMo and the HMM for comparison on the simulation dataset described in Section 2.3.2. Figure 5b shows the simulated mixing coefficients and those inferred by DyNeMo. For comparison, the state time course inferred by an HMM is also shown in Figure 5c. As the HMM is a mutually exclusive state model, it is unable to infer a linear mixture of modes, whereas DyNeMo's mixing coefficients estimate the ground truth very well, demonstrating its ability to learn a mixture of modes. Using the inferred mixing coefficients or state time course along with the inferred covariances, we can reconstruct the time-varying covariance, $\boldsymbol{C}_t$, of the training data. The Riemannian distance between the reconstruction and ground truth is shown in Figure 5d. The Riemannian distance for DyNeMo ($M = 1.5$, $SD = 0.42$) is significantly smaller than for the HMM ($M = 11.9$, $SD = 3.6$) indicating DyNeMo is a more accurate model for the time-varying covariance (p-value $< 10^{-5}$).

3.3 Resting-State MEG Data

DyNeMo identifies plausible resting-state networks. Figure 6 shows the power maps, FC maps and PSDs of 10 modes inferred by DyNeMo when trained on the resting-state MEG dataset described in Section 2.3.3. For the PSDs, we plot the regression coefficients $\boldsymbol{P}_j(f)$ to highlight differences relative to the mean PSD $\boldsymbol{P}_0(f)$ common to all modes. Mode 1 appears to be a low-power background network and does not show any significant power above the mean PSD for any frequency. Modes 2-10 show high power localised to specific regions associated with functional activity. Regions with high power also appear to have high FC. Modes 2 and 3 show power in regions associated with visual activity. Mode 4 shows power in parietal regions and can be associated with the posterior default mode network (see Figure 11). Mode 5 shows power in the sensorimotor region. Modes 6-8 show power in auditory/language regions. Modes 2-8 show power in the alpha band (8-12 Hz) and modes 4-6 and 8 include power at higher frequencies in the beta band (15-30 Hz). Mode 9 shows power in fronto-parietal regions and is recognised as an executive control network. Mode 10 shows power in frontal regions which can be associated with the anterior default mode network. Modes 9 and 10 exhibit low-frequency oscillations in the delta/theta band (1-7 Hz). The PSD of each mode is consistent with the expectation based on the functional role associated with each mode. A comparison with states inferred with this dataset using an HMM is presented in the section "Large-scale resting-state networks can be formed from a linear mixture of modes".

Power maps are reproducible across two split-halves of the dataset. To assess the reproducibility of modes across datasets, we split the full dataset into two halves of 27 subjects. We assess the reproducibility of the modes across halves using the RV coefficient [66]. We match the modes across halves in a pairwise fashion using the RV coefficient as a measure of similarity. Figure 7 shows the power maps of the matched modes. In general, the same regions are active in each pair of modes and the functional networks are reproducible across datasets. The main difference is small changes in how power is distributed across the visual

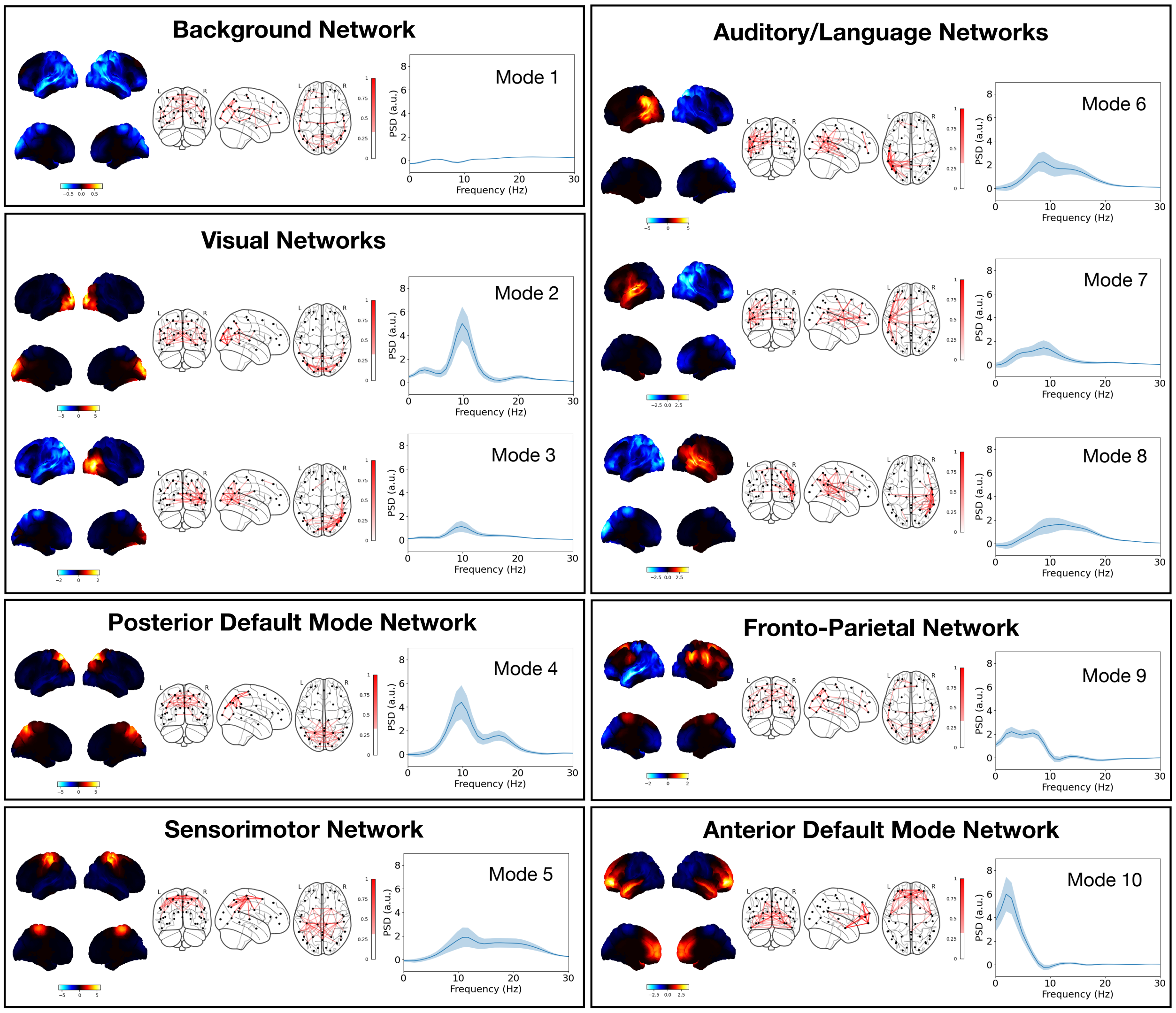


Figure 6: DyNeMo infers modes that form plausible resting-state MEG networks. Ten modes were inferred using resting-state MEG data from 55 subjects. Mode 1 appears to be a low-power background network, whereas modes 2-10 show high power in areas associated with functional networks. Modes are grouped in terms of their functional role. Each box shows the power map (left), FC map (middle) and PSD averaged over regions of interest (right) for each group. No thresholding was applied to the power maps. The top 5% of connections is shown in the FC maps. The shaded area in the PSD plots shows the standard error on the mean.

network modes (mode 4) and across the temporal/frontal regions (mode 9).

Mode activations are anti-correlated with a background mode and modes with activity in similar regions co-activate. A subset of the inferred mixing coefficients is shown in Figure 8. Figure 8a shows the raw mixing coefficients inferred directly from DyNeMo. However, these mixing coefficients do not account for a difference in the relative magnitude of each mode covariance. For example, a mode with a small mixing coefficient

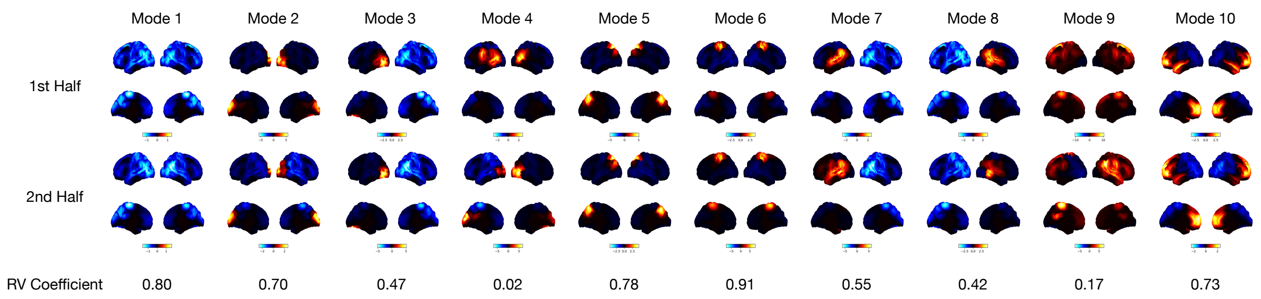


Figure 7: Power maps are reproducible across two split-halves of a dataset. Each half of the dataset contains the resting-state MEG data of 27 subjects. Power maps are shown for the the first half of the dataset (top) and second half of the dataset (middle). The RV coefficient of the inferred covariances from each half for a given mode (bottom) is also shown. The modes were matched in terms of their RV coefficient. Pairing the modes from each half we see the same functional networks are inferred. These networks also match the modes inferred on the full dataset of 55 subjects, suggesting these networks are reproducible across datasets. No thresholding was applied to the power maps.

may still be a large contributor to the time-varying covariance if the magnitude of its mode covariance is large. We can account for this by obtaining a weighted mixing coefficient mode time course by multiplying the raw mixing coefficients with the trace of its mode covariance. We also normalise the weighted mixing coefficient time course by dividing by the sum over all modes at each time point to maintain the sum-to-one constraint. Figure 8b and 8c show these normalised weighted mixing coefficients. Once we account for the magnitude of the mode covariances, we see each mode's contribution to the time-varying covariance is roughly equal. Figure 8d shows the correlation between the raw mixing coefficients α_{jt} for each mode. Modes 2-10 appear to be anti-correlated with mode 1. This arises due to the softmax operation (Equation (38)) that constrains the mixing coefficients to sum to one. For a mode to activate by contributing more to the time-varying covariance, another mode's contribution must decrease. The anti-correlation of mode 1 with every other mode suggests that it is primarily this mode's contribution that is decreased. This suggests that mode 1 can be thought of as a background mode that is deactivated by the other modes.

DyNeMo reveals short-lived (100-150 ms) mode activations. Using a GMM to identify when a mode is active we calculate summary statistics such as lifetimes, intervals and fractional occupancies. Mode activation time courses and summary statistics are shown in Figure 9. Mode 1 appears to have long activation lifetimes and a high fractional occupancy, which is consistent with the description of it being a background network that is largely present throughout. Modes 2-10 have mean lifetimes approximately over the range 100-150 ms, which is slightly longer than the state lifetimes obtained from an HMM, which are over the range 50-100 ms [67]. Both models reveal transient networks with lifetimes on the order of 100 ms, suggesting that this is a plausible time scale for these functional networks in resting-state MEG data, confirming that the short lifetimes previously found by the HMM are not likely to be caused by the mutual exclusivity assumption.

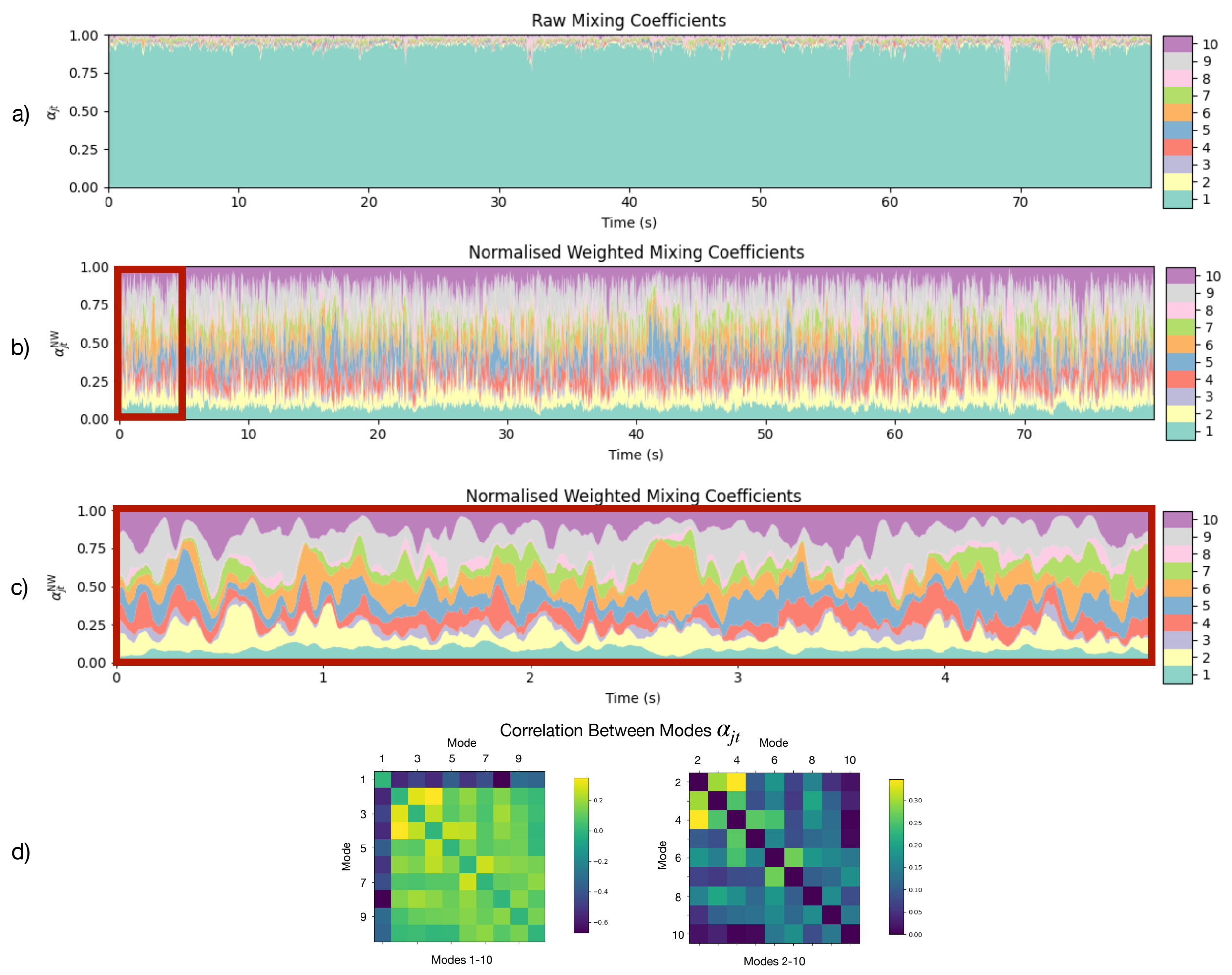


Figure 8: DyNeMo provides a mode description of resting-state MEG data. a) Raw mixing coefficients α_{jt} inferred by DyNeMo for one subject. b) Mixing coefficients α_{jt} weighted by the trace of each mode covariance and normalised to sum to one at each time point. c) Zoomed in normalised weighted mixing coefficients $\alpha_{jt}^{\mathrm{NW}}$ for the first 5 seconds. d) Correlation between the raw mixing coefficients α_{jt} for different modes j. Ordering is the same as Figure 6. We see DyNeMo's description of the data is a set of co-existing modes whose contribution to the time-varying covariance fluctuates. Once weighted by the covariance matrices we see each mode has a more equal contribution. We also see modes 2-10 are anti-correlated with the mode 1 and modes with activation in similar regions, e.g. modes 2, 3 and 4, are correlated.

DyNeMo learns long-range temporal correlations. Latent temporal correlations in MEG data can be seen by examining the inferred mixing coefficients, which are shown in Figure 8. Figure 10b (top left) shows the PSD of the inferred mixing coefficients, which has a $1/f$-like spectrum, indicating long-range temporal correlations are present. As in Section 3.1, this does not mean that DyNeMo's generative model has necessarily learnt long-range dependencies, as the presence of long-range temporal correlations could be a result of purely data-driven information. We can examine if the generative model in DyNeMo was able to learn these long-range temporal correlations by sampling a mixing coefficient time

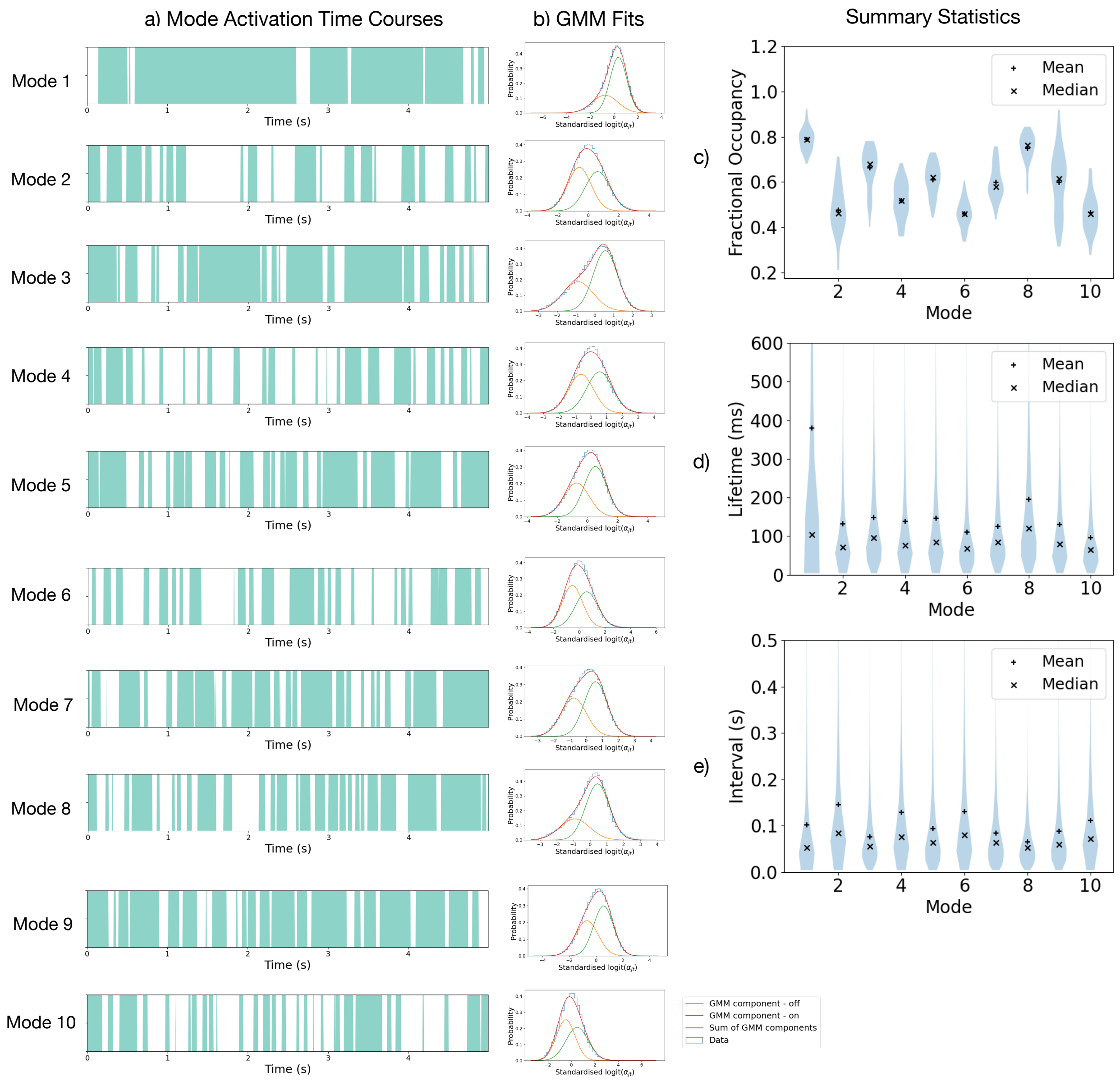


Figure 9: DyNeMo reveals short-lived mode activations with lifetimes of 100-150 ms. a) Mode activation time courses. Turquoise regions show when a mode is "active". Only the first 5 seconds of each mode activation time course is shown. b) GMM fits used to identify mode "activations". Distribution over activations and subjects of c) mode activation lifetimes and d) intervals. e) Distribution over subejcts of fractional occupancies. We see mode 1 has a significantly longer mean lifetime (approximately 400 ms) compared to the other modes (approximately 100-150 ms). There is also a wide distribution of fractional occupancies across subjects.

course from the model RNN. Figure 10a shows a sampled mixing coefficient time course. The PSD of the mixing coefficient time course sampled from the model RNN, Figure 10b (bottom left), shows the same $1/f$-like spectrum as the inferred mixing coefficient time course,

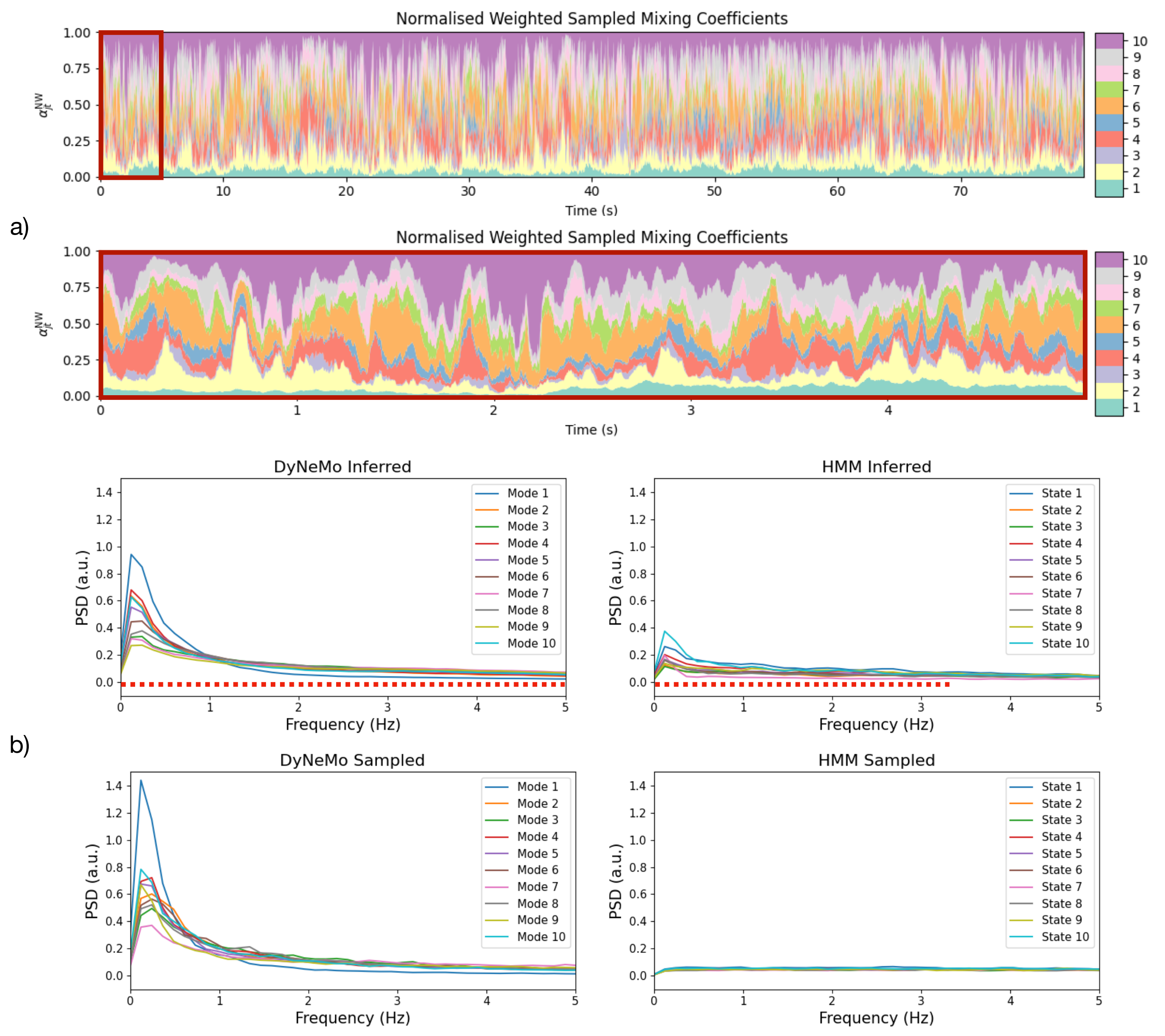
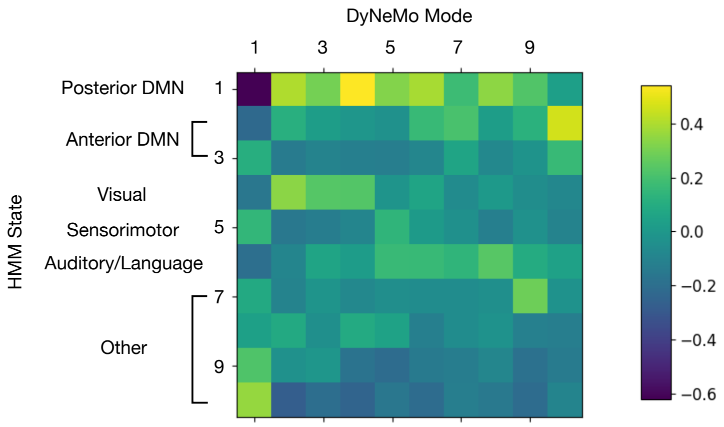
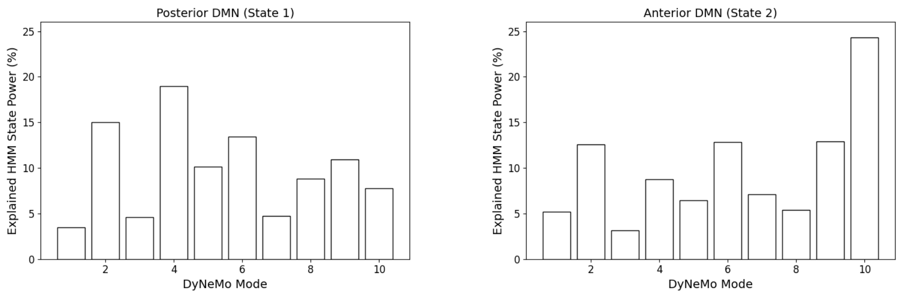


Figure 10: DyNeMo learns long-range temporal correlations in resting-state MEG data. a) Normalised weighted mixing coefficients sampled from the DyNeMo model RNN trained on resting-state MEG data. b) PSD of the sampled and inferred normalised weighted mixing coefficients from DyNeMo and sampled and inferred state time courses from an HMM. The red dashed line in b) shows statistically significant frequencies above a 95% confidence level when comparing the inferred time courses with a sample from the HMM using a paired t-test. The mixing coefficient time course sampled from the DyNeMo model RNN resembles the inferred mixing coefficient time course and shows a similar PSD. Contrastingly, the sampled state time course from an HMM does not have the same temporal correlations as the inferred state time course, which is demonstrated by the flat PSD for the sample. Each mixing coefficient time course was standardised (z-transformed) across the time dimension before calculating the PSD. The fractional occupancy in a 200 ms window was used to calculate the PSD of the HMM state time courses, see [14].

a)



b)



c)

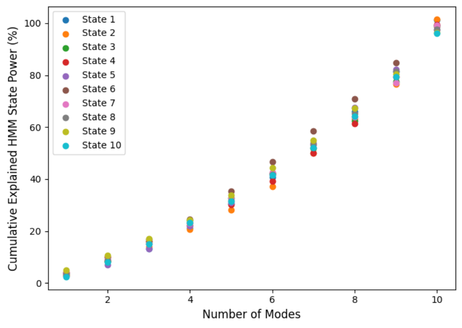


Figure 11: HMM states can be represented as a linear mixture of modes. a) Correlation of HMM state time courses with DyNeMo mode mixing coefficient time courses. The dynamics of multiple mode time courses correlate with each HMM state time course. In particular, many modes co-activate with the posterior default mode network (DMN) state. All elements are significant with a p-value < 0.05. b) Percentage of HMM state power explained by each DyNeMo mode for the posterior and anterior DMN. This was calculated as $\langle \alpha_{jt} \rangle \operatorname{Tr}(D_j)/\operatorname{Tr}(H_i)$, where D_j (H_i) is the DyNeMo (HMM) covariance for mode j (state i) and $\langle \alpha_{jt} \rangle$ is the time average mixing coefficient for mode j when state i is active. This shows all modes contribute to some extent to the power in these HMM states. c) The cumulative explained power for each HMM state. The modes were re-ordered in terms of increasing contribution before calculating the cumulative sum. Error bars are too small to be seen.

demonstrating it was able to learn long-range temporal correlations in the data. This is in contrast to an HMM, where the PSD of the inferred state time course, Figure 10b (top right), shows long-range temporal correlations, but the PSD of a sampled state time course, Figure 10b (bottom right), does not. It is also worth noting that the inferred long-range temporal correlations for the HMM are also less strong than for DyNeMo. This implies that the DyNeMo inferred long-range temporal correlations are not purely data driven, but also come from knowledge about long-range temporal correlations captured by DyNeMo through gathering information across the whole dataset.

Large-scale resting-state networks can be formed from a linear mixture of modes. The mixture model in DyNeMo allows it to construct large-scale patterns of covariance using a combination of modes with localised activity. This can be seen by comparing the modes inferred by DyNeMo with states that reveal large-scale networks inferred by an HMM. An HMM was trained on the same resting-state dataset. Power maps, FC maps and PSDs of the HMM states are shown in Figure 19. Two important networks identified by the HMM are the anterior and posterior default mode networks (states 1 and 2). The power map for DyNeMo mode 10 (see Figure 6) resembles the anterior state, however, there is no single mode that resembles the posterior state. Figure 11a shows the correlation of HMM state time courses with DyNeMo mode mixing coefficient time courses. We can see the modes that are correlated most with a state time course have activity in similar locations. Focusing on the default mode network states, DyNeMo mode 4 is the most correlated the posterior state and mode 10 is most correlated with the anterior state. In [67], it was shown that the default mode networks states have a high power in the alpha band for the posterior state and in the delta/theta band for the anterior state. The PSDs of the modes 4 and 10 also show this, providing further evidence that these modes are an alternative perspective on these states. The contribution of each mode to the default mode network HMM states is shown in Figure 11b. This shows the ratio of the total power in a mode relative to the total power in an HMM state. We can see that the power in the default mode network states is explained by many modes, i.e. DyNeMo has found a representation of these states that combines many modes. This is also true for the other HMM states. Figure 11 shows the fraction of power explained by a certain number of modes for each HMM state. The fraction of power explained increases monotonically with number of modes with no one particular mode explaining a large fraction of power. The mode description provided by DyNeMo appears to be fundamentally different to the HMM, no segments of time where one mode dominates are found. Instead, it is a representation where multiple modes co-exist and dynamics are captured by changes in the relative activation of each mode.

3.4 Task MEG Data

Resting-state networks are recruited in task. The power maps, FC maps and PSDs of 10 modes inferred by DyNeMo trained from scratch on the task MEG dataset described in Section 2.3.3 are shown in Figure 12. Very similar functional networks are found in task and resting-state MEG data (see Section 3.3). The main difference between the resting-state and task power maps is that the sensorimotor mode has split into two asymmetric modes. This could be due to the more frequent activation of this area in the task dataset, which incentivises the model to infer modes that best describe power at this location.

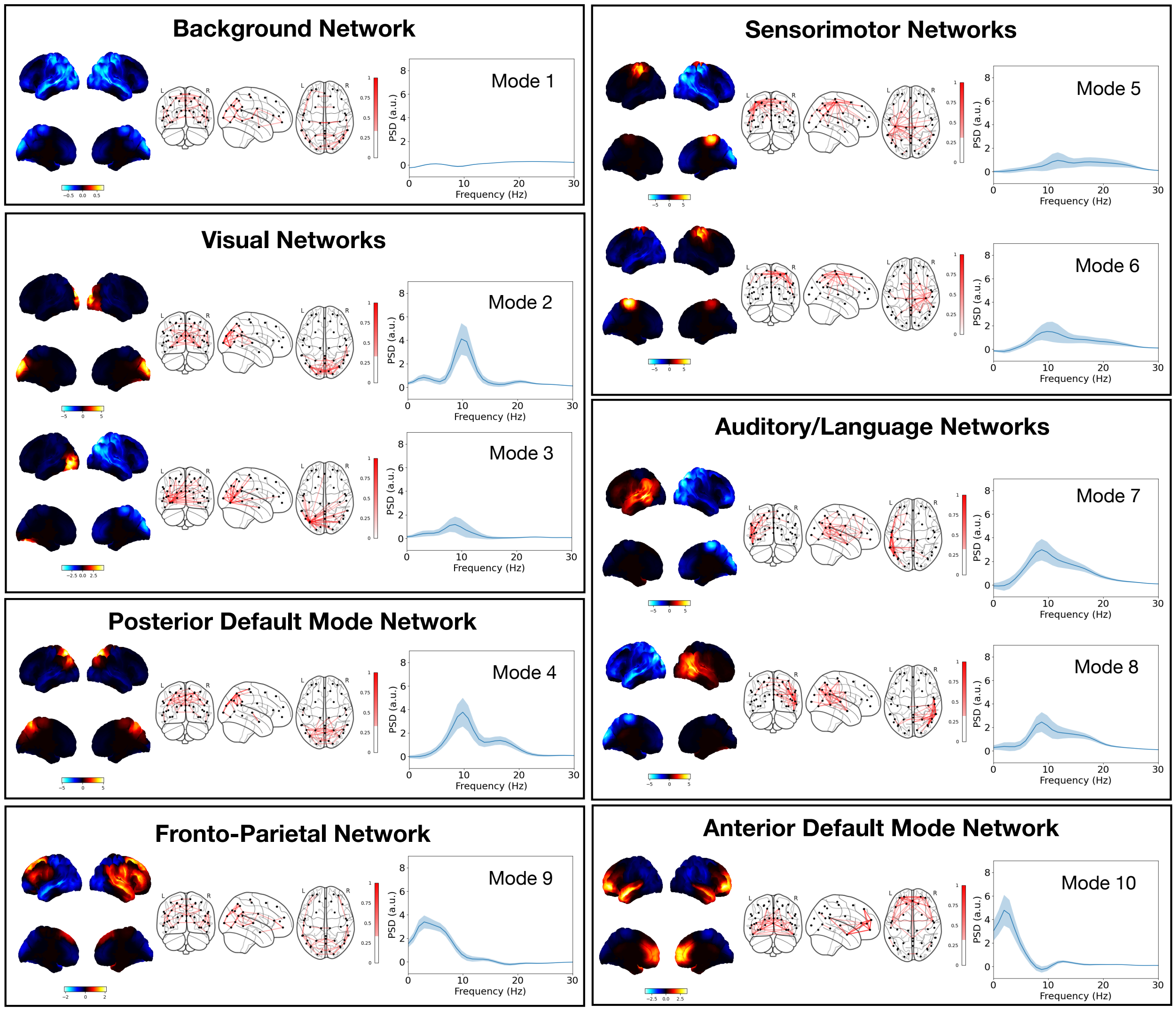


Figure 12: Resting-state networks are recruited in task. Ten modes were inferred using task MEG data from 51 subjects. Very similar functional networks are inferred as the resting-state data fit shown in Figure 6. Modes are grouped in terms of their functional role. Each box shows the power map (left), FC map (middle) and PSD averaged over regions of interest (right) for each group. No thresholding was applied to the power maps. The top 5% of connections are shown in the FC maps. The shaded area in the PSD plots shows the standard error on the mean.

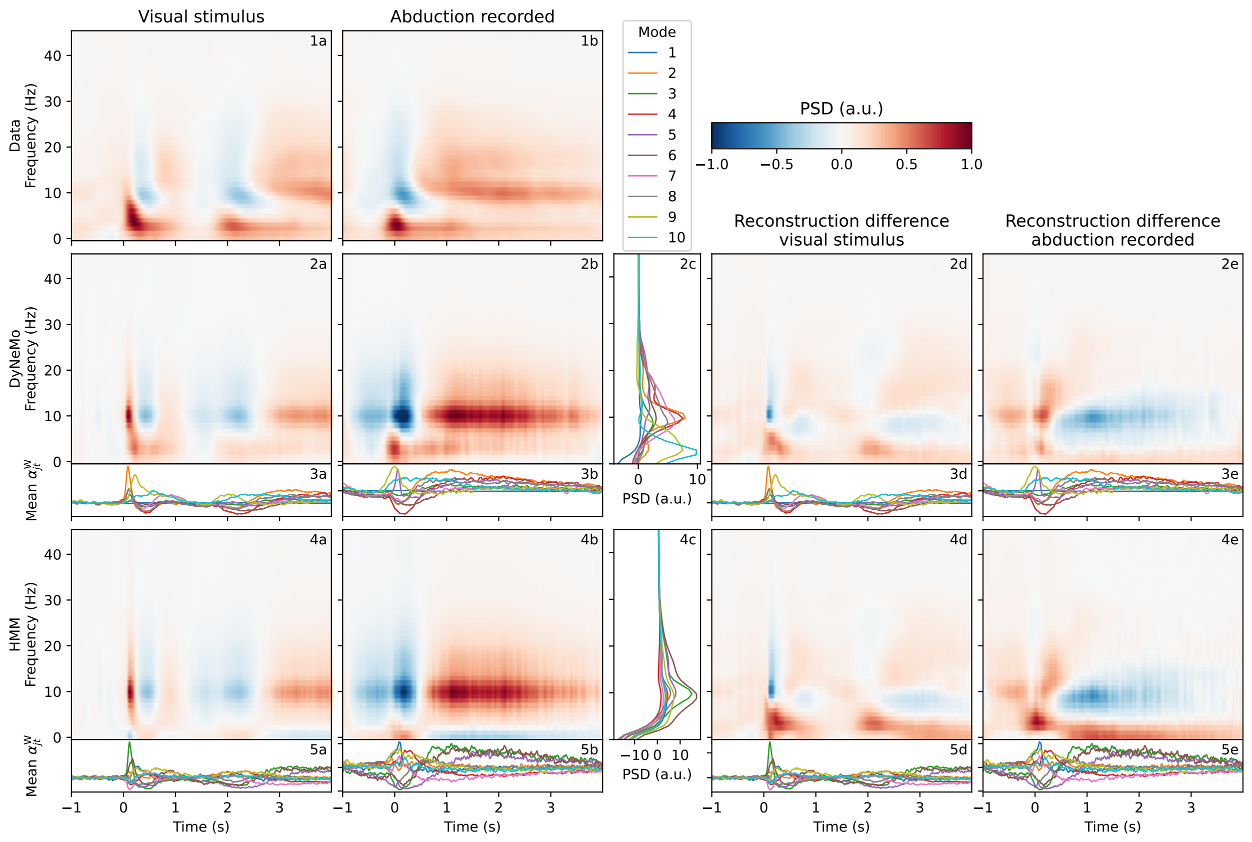


Figure 13: Time-frequency analysis shows the expected response to the visuomotor task. Power vs time and frequency is reconstructed from the model by multiplying the covariance-weighted mixing coefficient time courses with the mode PSDs. All plots share the same colour bar. This figure shows that DyNeMo is successfully capturing spectral information about its modes. Column *a* shows the reconstruction around the presentation of the visual stimulus. Column *b* shows the reconstruction around the recording of the abduction. Column *c* shows the mode PSDs. Columns *d* and *e* show the difference between the reconstruction and the frequency content of the data in the visual and abduction windows respectively. Row 1 shows the frequency content of the data. To make it equivalent to the expected result from the reconstruction, it is baseline corrected by subtracting the mean power in each frequency bin during the second prior to the presentation of the visual stimulus. Rows 2 and 4 show plots of DyNeMo and HMM reconstructions respectively. Rows 3 and 5 show the mixing coefficient time courses used in the reconstructions. Both the data and reconstructed PSDs are normalised such that their maximum absolute value is 1.

Modes show an evoked response to tasks. When the inferred mixing coefficient time courses are epoched around task events, a distinct response is seen. With the window around the presentation of the visual stimulus (Figure 13.3a), DyNeMo shows a strong activation in mode 2 which corresponds to activity in the visual cortex. It also shows smaller peaks in modes 4 (posterior default mode network) and 8 (auditory) followed by another larger peak in mode 9 (fronto-parietal network). These represent neural activity moving from the visual cortex to a broader posterior activation and finally to an anterior activation.

With the window around the abduction event (Figure 13.3b), DyNeMo shows a strong peak in mode 5 which corresponds to activity in the motor cortex. This is accompanied by a broader suppression of mode 4 which represents the posterior default mode network. The presence of task-related activations in the mixing coefficient time courses when DyNeMo is unaware of the task structure of the data shows that it is capable of learning modes and activations which are descriptive of underlying brain activity.

DyNeMo is a more accurate model of dynamic spectral properties compared to an HMM. From DyNeMo's mixing coefficients and mode PSDs, a time-frequency plot of the data can be reconstructed. The frequency profile of each mode is multiplied by its mixing coefficient at a given time to produce a reconstruction of a spectrogram (Figure 13.2a and b). This reconstruction bears a strong resemblance to the time-frequency content of the original data (Figure 13.1a and b). An HMM was also fitted to the task dataset. Power maps, FC maps and PSDs of states identified by the HMM are shown in Figure 20. Using the same procedure on the results of the HMM produces a reconstruction (Figure 13.4a and b) with a greater difference from the original data. The differences between the reconstructions and the original data are shown in Figure 13.2d and e, 4d and 4e. Calculating the mean absolute difference between the reconstructions and data, we find that in the visual window, DyNeMo has an error of 0.0579 while the HMM has 0.0691. In the motor window, these errors are 0.0690 and 0.0832 respectively. This implies that DyNeMo is a more accurate model of spectral content in the training data.

The same evoked response is seen across multiple trials. When considering all trials and not just the average evoked response over trials, we see that the visual mode is consistently activated when the visual stimulus is presented. We also see that the sensorimotor mode is consistently activated when the abduction is occurring. This result is shown in Figure 14. This shows that DyNeMo is finding activation on a trial-wise level, not just as an aggregation of thousands of trials. The HMM also shows trial-wise activation, although the binary nature of its state activations means that the contribution of a given state can be either wiped out by another state or falsely activated by reduced activity elsewhere. DyNeMo avoids this by allowing a mixture of states to be active at a given time.

4 Discussion

We have shown that MEG data can be described using multiple modes of spatio-temporal patterns which form large-scale brain networks (Figures 6 and 12). These modes have distinct spectral properties and correspond to plausible FC systems, such as visual, sensorimotor, auditory or other higher-order cognitive activity. These modes are more localised and can be more lateralised than the spatial patterns attributed with HMM states. Previous analysis of resting-state MEG data using an HMM [67] was able to identify large-scale transient networks which exist on time scales between 50 and 100 ms. We find DyNeMo infers transient networks at similar time scales of 100-150 ms (Figure 9). The implies the fast dynamics inferred by an HMM are not due to the assumption of mutually exclusive states.

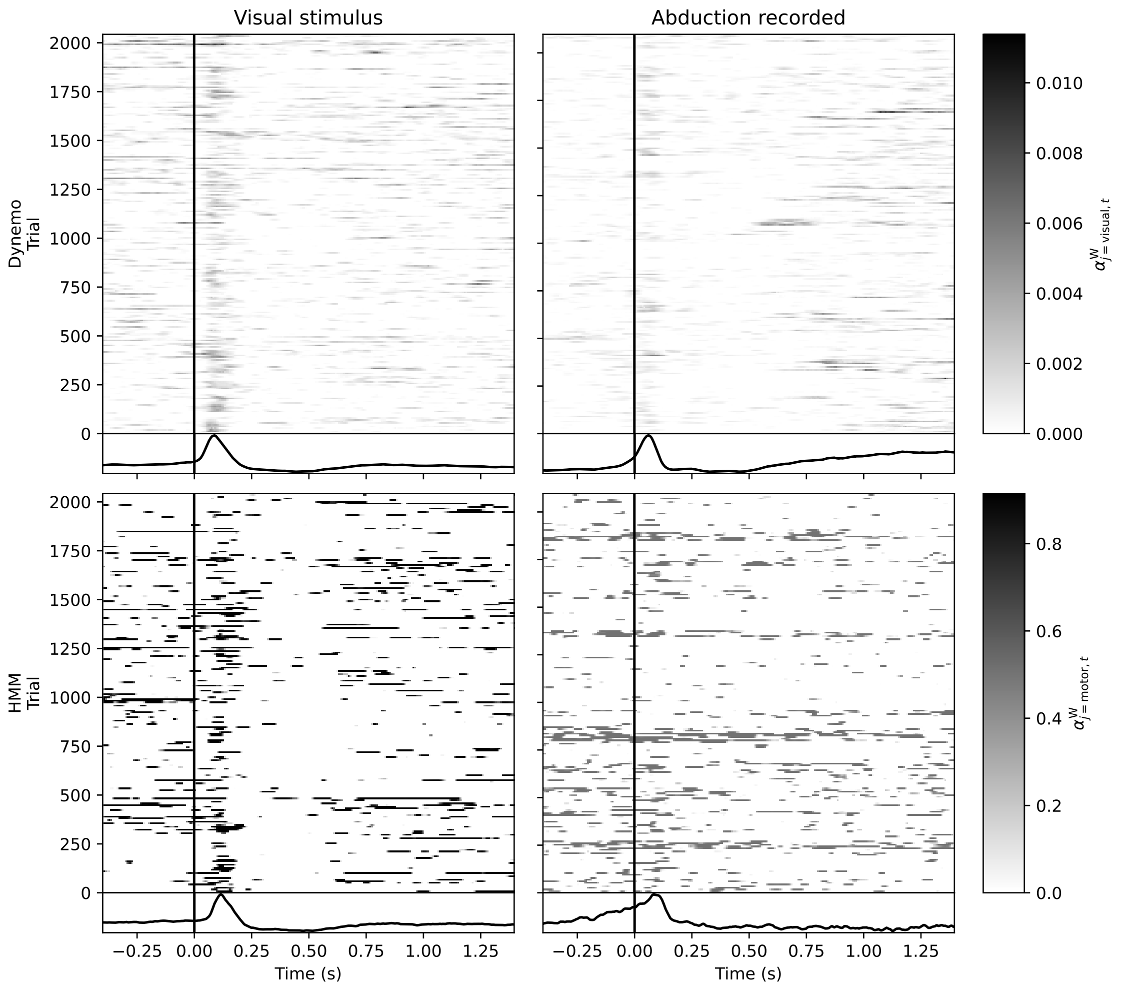


Figure 14: A consistent response to the visuomotor task is seen across trials. Trial-wise activations for sensorimotor and visual networks are shown. Rather than the trial-wise mean, we consider the individual trials to see if the activation of specific networks is well matched to the trial structure. In row 1, we consider the primary visual mode for each method. In row 2, we consider the primary sensorimotor mode for each method. Column 1 contains results from DyNeMo. Column 2 contains results from the HMM. Under each trial-wise plot, the mean activation across trials is presented. The band of grey which can be seen just after $t = 0$ in the DyNeMo plots shows that activation is seen across trials and not simply as an aggregate statistic.

An HMM trained on the resting-state MEG dataset used in this work suggested the default mode network was split into an anterior and posterior component [67]. In DyNeMo, the default mode network is further split up into many modes that combine to represent this network (Figure 11b). The modes that represent the default mode network show power in the same regions and frequency bands as the HMM states, supporting the fact that the modes represent an alternative perspective on the data.

Training DyNeMo on task MEG data, we find the same functional networks as inferred with resting-state data (Figure 12). This finding is supported in literature for other neuroimaging modalities, where the same networks are found in resting-state and task fMRI data [4]. The similarity in the functional networks could also be due to the fact that the majority of the subjects in the task dataset are also present in the resting-state dataset.

In an unsupervised fashion, DyNeMo was able to infer modes associated with the task. This is seen as an evoked response in the mixing coefficients of a mode to a task (Figure 13). This demonstrates that the modes inferred by DyNeMo meaningfully represent brain activity. The modes also reflect the expected time-frequency response to visual and motor tasks, which builds confidence in the description provided by DyNeMo. We find DyNeMo provides a more accurate model compared to an HMM of time-varying spectral features in the training data (Figure 13). However, both DyNeMo and the HMM show errors in modelling high-frequency spectral content in the task MEG dataset. We believe this arises from the PCA step in the data preparation, which retains components that explain large amounts of variance. In this data, lower frequencies have larger amplitudes and are able to explain more variance than high frequencies with smaller amplitudes, leading to high-frequency spectral content being filtered out. Avoiding the loss of this information could be investigated in future work with spectral pre-whitening techniques.

The smaller reconstruction error for the spectrogram of task MEG data from DyNeMo is due to the linear mixture affording the model a greater flexibility to precisely model dynamics. The fact that the reconstruction error is only slightly reduced compared to the HMM suggests that despite the constraint of mutual exclusivity the HMM was still able to provide a good description of dynamics.

4.1 Methodological Advancements

We believe that DyNeMo improves upon alternative unsupervised techniques in four key ways: the use of amortised inference; the use of the reparameterisation trick; the ability to model data as a linear mixture of modes (opposed to mutually exclusive states) and the ability to model long-range temporal dependencies in the data.

The amortised inference framework used in DyNeMo (described in Section 2) contains a fixed number of trainable parameters (inference RNN weights and biases). This means DyNeMo is readily trainable on datasets of varying size. Usually, the number of trainable parameters in the inference network is significantly smaller than the size of a dataset, making this approach very efficient when scaling to bigger datasets. As the availability of larger datasets grows, so does the need for models that can utilise them. Here, we believe deep learning techniques will play an important role, where with more data, models with a deep architecture begin to outperform shallower ones. Although, in this work we have studied a relatively small dataset (51-55 subjects) using a shallow model (one RNN layer), DyNeMo is readily scalable in terms of model complexity to include multiple RNN layers and more hidden units. In combination with bigger datasets this can reveal new insights into brain data. For example, previous modelling of a large resting-state fMRI dataset (Human Connectome Project, [68]) using an HMM revealed a link between FC dynamics and heritable and psychological traits [69].

Provided we are able to apply the reparameterisation trick to sample from the variational posterior distribution, we are able to infer the parameters for any generative model. This facilitates the use of more sophisticated and non-linear observation models and opens up a range of future modelling opportunities. This includes the use of an autoregressive model capable of learning temporal correlations in the observed data; the hierarchical modelling of inter-subject variability and the inclusion of dynamics at multiple time scales, similar to the approach used in [43].

A key modelling advancement afforded by DyNeMo is the ability to model data as a time-varying linear sum of modes. The extent to which modes mix is controlled by a free parameter referred to as the *temperature*, τ, which appears in the softmax transformation of the logits (see Equation (38) in SI 8.2). Low temperatures lead to mutually exclusive modes whereas high temperatures lead to a soft mixture of modes. In this work, we allow the temperature to be a trainable parameter. By doing this, the output of the softmax transformation is able to be tuned during training to find the appropriate level of mixing to best describe the data. Such a scheme can be interpreted as form of entropy regularisation [70, 71].

The inclusion of a model RNN in DyNeMo allows it to generate data with long-range temporal dependencies (Figures 4 and 10). This is because the future value of a hidden logit is determined by a long sequence of previous values, not just the most recent value. There is significant evidence for long-range temporal correlations in M/EEG data [72–74] and an association between altered long-range temporal correlations and disease [75, 76]. Models that are capable of learning long-range temporal correlations are advantageous in multiple ways: they can be more predictive of task or disease than models with a shorter memory; they can prevent overfitting to noise in the training data through regularisation and finally they can be used to synthesise data with realistic long-range neural dynamics.

In addition to the modelling and inference advancements discussed above, we also proposed a new method for calculating spectral properties for data described using a set of modes (see Section 2.4). With an HMM, methods such as a multitaper [12] can be used to provide high-resolution estimates of PSDs and coherences for each state. This approach relies on the state time course identifying segments of the training data where only one state is activate. This approach is no longer feasible with a description of the data as a set of co-existing modes. In this paper, we propose fitting a linear regression model to a cross spectrogram calculated using the data. This method relies on different time points having different ratios of mixing between the modes. Provided this is the case, this method produces high-resolution estimates of the PSD and coherence of each mode (Figures 6, 18, 12 and 13).

4.2 Drawbacks

As with most modern machine learning models, DyNeMo contains a large number of hyperparameters that need to be specified before the model can be trained. These are discussed in SI 8.2. An important hyperparameter that affects the interpretation of inferences from the model is the number of modes, J. We discuss the impact of varying the number of modes in SI 8.3. In short, as the number of modes is increased, the spatial activity of each mode becomes more localised and the variability of the inferred spatial patterns increases. We overcome this by ensuring any conclusions based on studies using DyNeMo are not sensitive to the number of modes chosen. We tune other hyperparameters by seeking the set of parameters that minimise the value of the loss function.

In addition to a large number of hyperparameters, we find the model is sensitive to the initialisation of trainable parameters. This includes the internal weights and biases of

RNN layers and the learnable free parameters for the mode means and covariances. The initialisations used in this work are listed in SI 8.2. We found the initialisation of the mode covariances to be particularly important. We overcome the issue of sensitivity to the initialisation of trainable parameters by training the model from scratch with different initialisations and only retaining the model with the lowest loss.

4.3 Outlook and Future Applications

The model presented here has many possible future applications. For example, it could be used to provide a dynamic and interpretable latent description, as done in this work, for other datasets. Alternatively, it could be used to facilitate future studies, examples of which are described below.

A common method to study the brain is the use of temporally unconstrained multivariate pattern analysis (*decoding*) to predict task, disease or behavioural traits [77]. The latent representation inferred by DyNeMo (unsupervised) provides a low-dimensional form of the training data, which is ideal for such analyses. This can overcome overfitting issues that are commonly encountered in decoding studies that use the raw data directly. Alternatively, the model architecture could be easily modified to form a semi-supervised learning problem where the loss function used has a joint objective to learn a low-dimensional representation that is useful for decoding as well as reconstructing the training data.

A useful feature of DyNeMo is the possibility of *transfer learning*, i.e. the ability to transfer information learnt from one dataset to another. This could be exercised by simply training DyNeMo on one dataset from scratch, before fine tuning the model on another dataset. Large resting-state datasets are commonplace in neuroimaging. A problem encountered in studies of small datasets (e.g. comprising of diseased cohorts) is the lack of statistical power for drawing meaningful conclusions [78]. Leveraging information gained from larger resting-state datasets could improve the predictions made on smaller datasets. For example, it has been shown resting-state data is predictive of task response [79, 80]. We believe DyNeMo offers the possibility of transferring information acquired from resting-state datasets with thousands of individuals to the individual subject level.

The generative model proposed here explictly models the covariance of the training data as a dynamic quantity. Such a model can be utilised in the field of M/EEG source reconstruction. Algorithms for source reconstruction often assume the covariance is static, which is rarely the case [81]. Using a dynamic estimate of the covariance, we can construct time-vaying reconstruction weights [39], which may improve source localisation.

Finally, whilst we focused on parcellated source reconstructed MEG data in this paper, DyNeMo could of course be applied to data from other neuroimaging modalities such as fMRI, sensor level MEG data and other electrophysiological techniques (EEG, ECOG, etc.).

5 Conclusions

We have proposed a new generative model and accompanying inference framework for neuroimaging data that is readily scalable to large datasets. Our application of DyNeMo to MEG data reveals fast transient networks that are spectrally distinct, in broad agreement with existing studies. We believe DyNeMo can be used to help us better understand the brain by providing an accurate model for brain data that explicitly models its dynamic nature using a linear mixture of modes. The modest improvement in modelling dynamic spectral properties compared to an HMM shows the assumption of mutual exclusivity does not necessarily impact the HMM's ability to model the data effectively. Nevertheless, DyNeMo is a novel and complementary tool that is useful for studying neuroimaging data.

6 Acknowledgements

We would like to thank Matt Brookes and his team at the University of Nottingham for providing us with the data analysed here. Data were collected in the context of the Medical Research Council (MRC)-funded MEG UK partnership. All participants gave written informed consent and ethical approval was granted by the University of Nottingham Medical School Research Ethics Committee.

This research was supported by the National Institute for Health Research (NIHR) Oxford Health Biomedical Research Centre. The Wellcome Centre for Integrative Neuroimaging is supported by core funding from the Wellcome Trust (203139/Z/16/Z).

7 Declaration of Interests

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this article.

References

[1] Karl J Friston. "Functional and effective connectivity in neuroimaging: a synthesis". In: *Human brain mapping* 2.1-2 (1994), pp. 56–78.

[2] Christian F Beckmann et al. "Investigations into resting-state connectivity using independent component analysis". In: *Philosophical Transactions of the Royal Society B: Biological Sciences* 360.1457 (2005), pp. 1001–1013.

[3] Michael W Cole et al. "Intrinsic and task-evoked network architectures of the human brain". In: *Neuron* 83.1 (2014), pp. 238–251.

[4] Stephen M Smith et al. "Correspondence of the brain's functional architecture during activation and rest". In: *Proceedings of the national academy of sciences* 106.31 (2009), pp. 13040–13045.

[5] Michael C Stevens. "The contributions of resting state and task-based functional connectivity studies to our understanding of adolescent brain network maturation". In: *Neuroscience & Biobehavioral Reviews* 70 (2016), pp. 13–32.

[6] Matthew J Brookes et al. "Investigating the electrophysiological basis of resting state networks using magnetoencephalography". In: *Proceedings of the National Academy of Sciences* 108.40 (2011), pp. 16783–16788.

[7] Henry Luckhoo et al. “Inferring task-related networks using independent component analysis in magnetoencephalography”. In: *Neuroimage* 62.1 (2012), pp. 530–541.

[8] Francesco De Pasquale et al. “A cortical core for dynamic integration of functional networks in the resting human brain”. In: *Neuron* 74.4 (2012), pp. 753–764.

[9] Joerg F Hipp et al. “Large-scale cortical correlation structure of spontaneous oscillatory activity”. In: *Nature neuroscience* 15.6 (2012), pp. 884–890.

[10] Andrew J Quinn et al. “Task-evoked dynamic network analysis through hidden markov modeling”. In: *Frontiers in neuroscience* 12 (2018), p. 603.

[11] Andreas K Engel et al. “Intrinsic coupling modes: multiscale interactions in ongoing brain activity”. In: *Neuron* 80.4 (2013), pp. 867–886.

[12] Diego Vidaurre et al. “Spectrally resolved fast transient brain states in electrophysiological data”. In: *Neuroimage* 126 (2016), pp. 81–95.

[13] George C O’Neill et al. “Dynamic recruitment of resting state sub-networks”. In: *Neuroimage* 115 (2015), pp. 85–95.

[14] Adam P Baker et al. “Fast transient networks in spontaneous human brain activity”. In: *Elife* 3 (2014), e01867.

[15] Pascal Fries. “Rhythms for cognition: communication through coherence”. In: *Neuron* 88.1 (2015), pp. 220–235.

[16] Diederick Stoffers et al. “Dopaminergic modulation of cortico-cortical functional connectivity in Parkinson’s disease: an MEG study”. In: *Experimental neurology* 213.1 (2008), pp. 191–195.

[17] CJ Stam et al. “Graph theoretical analysis of magnetoencephalographic functional connectivity in Alzheimer’s disease”. In: *Brain* 132.1 (2009), pp. 213–224.

[18] Viviana Betti et al. “Natural scenes viewing alters the dynamics of functional connectivity in the human brain”. In: *Neuron* 79.4 (2013), pp. 782–797.

[19] Matthew J Brookes et al. “Measuring functional connectivity using MEG: methodology and comparison with fcMRI”. In: *Neuroimage* 56.3 (2011), pp. 1082–1104.

[20] Matthew J Brookes et al. “Measuring temporal, spectral and spatial changes in electrophysiological brain network connectivity”. In: *Neuroimage* 91 (2014), pp. 282–299.

[21] Andrea Brovelli et al. “Dynamic reconfiguration of visuomotor-related functional connectivity networks”. In: *Journal of Neuroscience* 37.4 (2017), pp. 839–853.

[22] Ellen WS Carbo et al. “Dynamic hub load predicts cognitive decline after resective neurosurgery”. In: *Scientific Reports* 7.1 (2017), pp. 1–10.

[23] Francesco De Pasquale et al. “Temporal dynamics of spontaneous MEG activity in brain networks”. In: *Proceedings of the National Academy of Sciences* 107.13 (2010), pp. 6040–6045.

[24] Francesco De Pasquale et al. “A dynamic core network and global efficiency in the resting human brain”. In: *Cerebral Cortex* 26.10 (2016), pp. 4015–4033.

[25] George C O'Neill et al. "Measurement of dynamic task related functional networks using MEG". In: *NeuroImage* 146 (2017), pp. 667–678.

[26] Elena A Allen et al. "Tracking whole-brain connectivity dynamics in the resting state". In: *Cerebral cortex* 24.3 (2014), pp. 663–676.

[27] Catie Chang et al. "EEG correlates of time-varying BOLD functional connectivity". In: *Neuroimage* 72 (2013), pp. 227–236.

[28] Amanda Elton and Wei Gao. "Task-related modulation of functional connectivity variability and its behavioral correlations". In: *Human brain mapping* 36.8 (2015), pp. 3260–3272.

[29] R Matthew Hutchison et al. "Dynamic functional connectivity: promise, issues, and interpretations". In: *Neuroimage* 80 (2013), pp. 360–378.

[30] Aaron Kucyi and Karen D Davis. "Dynamic functional connectivity of the default mode network tracks daydreaming". In: *Neuroimage* 100 (2014), pp. 471–480.

[31] Maria Giulia Preti, Thomas AW Bolton, and Dimitri Van De Ville. "The dynamic functional connectome: State-of-the-art and perspectives". In: *Neuroimage* 160 (2017), pp. 41–54.

[32] Enzo Tagliazucchi et al. "Dynamic BOLD functional connectivity in humans and its electrophysiological correlates". In: *Frontiers in human neuroscience* 6 (2012), p. 339.

[33] Ünal Sakoğlu et al. "A method for evaluating dynamic functional network connectivity and task-modulation: application to schizophrenia". In: *Magnetic Resonance Materials in Physics, Biology and Medicine* 23.5 (2010), pp. 351–366.

[34] Martin A Lindquist et al. "Evaluating dynamic bivariate correlations in resting-state fMRI: a comparison study and a new approach". In: *NeuroImage* 101 (2014), pp. 531–546.

[35] Raphaël Liégeois et al. "Cerebral functional connectivity periodically (de) synchronizes with anatomical constraints". In: *Brain structure and function* 221.6 (2016), pp. 2985–2997.

[36] Prejaas Tewarie et al. "Tracking dynamic brain networks using high temporal resolution MEG measures of functional connectivity". In: *Neuroimage* 200 (2019), pp. 38–50.

[37] Olaf Sporns et al. "Dynamic expression of brain functional systems disclosed by fine-scale analysis of edge time series". In: *Network Neuroscience* 5.2 (2021), pp. 405–433.

[38] Lawrence Rabiner and Biinghwang Juang. "An introduction to hidden Markov models". In: *ieee assp magazine* 3.1 (1986), pp. 4–16.

[39] Mark W Woolrich et al. "Dynamic state allocation for MEG source reconstruction". In: *Neuroimage* 77 (2013), pp. 77–92.

[40] Zelekha A Seedat et al. "The role of transient spectral 'bursts' in functional connectivity: A magnetoencephalography study". In: *NeuroImage* 209 (2020), p. 116537.

[41] Cameron Higgins et al. "Replay bursts in humans coincide with activation of the default mode and parietal alpha networks". In: *Neuron* 109.5 (2021), pp. 882–893.

[42] Nelson J Trujillo-Barreto, David Araya, and Wael El-Deredy. "The discrete logic of the Brain-Explicit modelling of Brain State durations in EEG and MEG". In: *bioRxiv* (2019), p. 635300.

[43] Usama Pervaiz et al. "Multi-dynamic modelling reveals strongly time-varying resting fMRI correlations". In: *Medical Image Analysis* 77 (2022), p. 102366. ISSN: 1361-8415.

[44] Karl Friston et al. "Variational free energy and the Laplace approximation". In: *Neuroimage* 34.1 (2007), pp. 220–234.

[45] Diederik P. Kingma and Max Welling. "Auto-Encoding Variational Bayes". In: *International Conference on Learning Representations (ICLR)*. 2014.

[46] Aurélien Géron. *Hands-on machine learning with Scikit-Learn, Keras, and TensorFlow: Concepts, tools, and techniques to build intelligent systems*. O'Reilly Media, Inc., 2019.

[47] *TensorFlow: Large-Scale Machine Learning on Heterogeneous Systems*. Software available from tensorflow.org. 2015. URL: `https://www.tensorflow.org/`.

[48] Sepp Hochreiter and Jürgen Schmidhuber. "Long short-term memory". In: *Neural computation* 9.8 (1997), pp. 1735–1780.

[49] C.M. Bishop. *Pattern Recognition and Machine Learning*. Information science and statistics. Springer, 2007.

[50] Cheng Zhang et al. "Advances in variational inference". In: *IEEE transactions on pattern analysis and machine intelligence* 41.8 (2018), pp. 2008–2026.

[51] Mark W Woolrich et al. "Bayesian analysis of neuroimaging data in FSL". In: *Neuroimage* 45.1 (2009), S173–S186.

[52] Karl Friston, James Kilner, and Lee Harrison. "A free energy principle for the brain". In: *Journal of physiology-Paris* 100.1-3 (2006), pp. 70–87.

[53] David E Rumelhart, Geoffrey E Hinton, and Ronald J Williams. "Learning representations by back-propagating errors". In: *nature* 323.6088 (1986), pp. 533–536.

[54] Samuel R Bowman et al. "Generating sentences from a continuous space". In: *arXiv preprint arXiv:1511.06349* (2015).

[55] Shun-Zheng Yu. "Hidden semi-Markov models". In: *Artificial intelligence* 174.2 (2010), pp. 215–243.

[56] URL: `https://github.com/OHBA-analysis/osl-core`.

[57] Mark Jenkinson, Mickael Pechaud, Stephen Smith, et al. "BET2: MR-based estimation of brain, skull and scalp surfaces". In: *Eleventh annual meeting of the organization for human brain mapping*. Vol. 17. Toronto. 2005, p. 167.

[58] Stephen M Smith. "Fast robust automated brain extraction". In: *Human brain mapping* 17.3 (2002), pp. 143–155.

[59] MX Huang, John C Mosher, and RM Leahy. "A sensor-weighted overlapping-sphere head model and exhaustive head model comparison for MEG". In: *Physics in Medicine & Biology* 44.2 (1999), p. 423.

[60] Barry D Van Veen and Kevin M Buckley. “Beamforming: A versatile approach to spatial filtering”. In: *IEEE assp magazine* 5.2 (1988), pp. 4–24.

[61] Giles L Colclough et al. “A symmetric multivariate leakage correction for MEG connectomes”. In: *Neuroimage* 117 (2015), pp. 439–448.

[62] J Matias Palva et al. “Ghost interactions in MEG/EEG source space: A note of caution on inter-areal coupling measures”. In: *Neuroimage* 173 (2018), pp. 632–643.

[63] Benjamin AE Hunt et al. “Attenuated post-movement beta rebound associated with schizotypal features in healthy people”. In: *Schizophrenia bulletin* 45.4 (2019), pp. 883–891.

[64] Athanasios Papoulis and H Saunders. “Probability, random variables and stochastic processes”. In: (1989).

[65] Peter Welch. “The use of fast Fourier transform for the estimation of power spectra: a method based on time averaging over short, modified periodograms”. In: *IEEE Transactions on audio and electroacoustics* 15.2 (1967), pp. 70–73.

[66] Zhi Yang et al. “Ranking and averaging independent component analysis by reproducibility (RAICAR)”. In: *Human brain mapping* 29.6 (2008), pp. 711–725.

[67] Diego Vidaurre et al. “Spontaneous cortical activity transiently organises into frequency specific phase-coupling networks”. In: *Nature communications* 9.1 (2018), pp. 1–13.

[68] Stephen M Smith et al. “Resting-state fMRI in the human connectome project”. In: *Neuroimage* 80 (2013), pp. 144–168.

[69] Diego Vidaurre, Stephen M Smith, and Mark W Woolrich. “Brain network dynamics are hierarchically organized in time”. In: *Proceedings of the National Academy of Sciences* 114.48 (2017), pp. 12827–12832.

[70] Gabriel Pereyra et al. “Regularizing neural networks by penalizing confident output distributions”. In: *arXiv preprint arXiv:1701.06548* (2017).

[71] Eric Jang, Shixiang Gu, and Ben Poole. “Categorical reparameterization with gumbel-softmax”. In: *arXiv preprint arXiv:1611.01144* (2016).

[72] Biyu J He. “Scale-free brain activity: past, present, and future”. In: *Trends in cognitive sciences* 18.9 (2014), pp. 480–487.

[73] Klaus Linkenkaer-Hansen et al. “Long-range temporal correlations and scaling behavior in human brain oscillations”. In: *Journal of Neuroscience* 21.4 (2001), pp. 1370–1377.

[74] Maria Botcharova et al. “Resting state MEG oscillations show long-range temporal correlations of phase synchrony that break down during finger movement”. In: *Frontiers in physiology* 6 (2015), p. 183.

[75] Gabriela Cruz et al. “Long Range Temporal Correlations (LRTCs) in MEG-Data during Emerging Psychosis: Relationship to Symptoms, Medication-Status and Clinical Trajectory”. In: *NeuroImage: Clinical* (2021), p. 102722.

[76] James K Moran et al. “Long-range temporal correlations in resting state beta oscillations are reduced in schizophrenia”. In: *Frontiers in psychiatry* 10 (2019), p. 517.

[77] Diego Vidaurre et al. “Temporally unconstrained decoding reveals consistent but time-varying stages of stimulus processing”. In: *Cerebral Cortex* 29.2 (2019), pp. 863–874.

[78] Russell A Poldrack et al. “Scanning the horizon: towards transparent and reproducible neuroimaging research”. In: *Nature reviews neuroscience* 18.2 (2017), pp. 115–126.

[79] Ido Tavor et al. “Task-free MRI predicts individual differences in brain activity during task performance”. In: *Science* 352.6282 (2016), pp. 216–220.

[80] Robert Becker et al. “Transient spectral events in resting state MEG predict individual task responses”. In: *NeuroImage* 215 (2020), p. 116818.

[81] Gustavo Lucena Gómez et al. “Localization accuracy of a common beamformer for the comparison of two conditions”. In: *NeuroImage* 230 (2021), p. 117793.

[82] Diederik P Kingma and Jimmy Ba. “Adam: A method for stochastic optimization”. In: *arXiv preprint arXiv:1412.6980* (2014).

[83] Nitish Srivastava et al. “Dropout: A Simple Way to Prevent Neural Networks from Overfitting”. In: *Journal of Machine Learning Research* 15.56 (2014), pp. 1929–1958. URL: `http://jmlr.org/papers/v15/srivastava14a.html`.

[84] Sergey Ioffe and Christian Szegedy. “Batch normalization: Accelerating deep network training by reducing internal covariate shift”. In: *International conference on machine learning*. PMLR. 2015, pp. 448–456.

[85] Jimmy Lei Ba, Jamie Ryan Kiros, and Geoffrey E Hinton. “Layer normalization”. In: *arXiv preprint arXiv:1607.06450* (2016).

[86] Yoshua Bengio, Patrice Simard, and Paolo Frasconi. “Learning long-term dependencies with gradient descent is difficult”. In: *IEEE transactions on neural networks* 5.2 (1994), pp. 157–166.

[87] Razvan Pascanu, Tomas Mikolov, and Yoshua Bengio. “On the difficulty of training recurrent neural networks”. In: *International conference on machine learning*. PMLR. 2013, pp. 1310–1318.

[88] Xavier Glorot and Yoshua Bengio. “Understanding the difficulty of training deep feed-forward neural networks”. In: *Proceedings of the thirteenth international conference on artificial intelligence and statistics*. JMLR Workshop and Conference Proceedings. 2010, pp. 249–256.

8 Supplementary Information (SI)

8.1 Derivation of the Loss Function

In variational Bayesian inference we infer a parameter by minimising the variational free energy,

$$F = -\int q(\boldsymbol{\theta}_{1:N}|\boldsymbol{x}_{1:N}) \log\left(\frac{p(\boldsymbol{x}_{1:N}|\boldsymbol{\theta}_{1:N})p(\boldsymbol{\theta}_{1:N})}{q(\boldsymbol{\theta}_{1:N}|\boldsymbol{x}_{1:N})}\right) \mathrm{d}\boldsymbol{\theta}_{1:N}. \tag{16}$$

where $q(\boldsymbol{\theta}_{1:N}|\boldsymbol{x}_{1:N})$ is the posterior, $p(\boldsymbol{\theta}_{1:N})$ is the prior and $p(\boldsymbol{x}_{1:N}|\boldsymbol{\theta}_{1:N})$ is the likelihood, $\boldsymbol{\theta}_t$ is the logit at each time point, $\boldsymbol{x}_t$ is the observed data at each time point and $t = 1, ..., N$ denotes the time index. We can separate the logarithm into two terms,

$$\begin{aligned} F &= -\int q(\boldsymbol{\theta}_{1:N}|\boldsymbol{x}_{1:N}) \log\left(p(\boldsymbol{x}_{1:N}|\boldsymbol{\theta}_{1:N})\right) \mathrm{d}\boldsymbol{\theta}_{1:N} \\ &\qquad + \int q(\boldsymbol{\theta}_{1:N}|\boldsymbol{x}_{1:N}) \log\left(\frac{q(\boldsymbol{\theta}_{1:N}|\boldsymbol{x}_{1:N})}{p(\boldsymbol{\theta}_{1:N})}\right) \mathrm{d}\boldsymbol{\theta}_{1:N}. \\ &= -\mathrm{LL} + \mathrm{KL}. \end{aligned} \tag{17}$$

LL is referred to as the *log-likelihood term* and KL is referred to as the *KL divergence term*.

Considering the log-likelihood term,

$$\mathrm{LL} = \int q(\boldsymbol{\theta}_{1:N}|\boldsymbol{x}_{1:N}) \log\left(p(\boldsymbol{x}_{1:N}|\boldsymbol{\theta}_{1:N})\right) \mathrm{d}\boldsymbol{\theta}_{1:N}. \tag{18}$$

We use the mean field approximation for the posterior,

$$q(\boldsymbol{\theta}_{1:N}|\boldsymbol{x}_{1:N}) = \prod_{t=1}^{N} q(\boldsymbol{\theta}_t|\boldsymbol{x}_{1:N}). \tag{19}$$

Each factor $q(\boldsymbol{\theta}_t|\boldsymbol{x}_{1:N})$ is a multivariate normal distribution parameterised by a mean vector $\boldsymbol{m}_{\theta_t}(\boldsymbol{x}_{1:N})$ and diagonal covariance matrix $\boldsymbol{s}^2_{\theta_t}(\boldsymbol{x}_{1:N})$, i.e.

$$q(\boldsymbol{\theta}_t|\boldsymbol{x}_{1:N}) = \mathcal{N}(\boldsymbol{m}_{\theta_t}(\boldsymbol{x}_{1:N}), \boldsymbol{s}^2_{\theta_t}(\boldsymbol{x}_{1:N})) \tag{20}$$

We also assume the data at each time point is independent and only depends on the logit at that time point, i.e. we factorise the likelihood as

$$p(\boldsymbol{x}_{1:N}|\boldsymbol{\theta}_{1:N}) = \prod_{t=1}^{N} p(\boldsymbol{x}_t|\boldsymbol{\theta}_t). \tag{21}$$

We assume a multivariate normal distribution for the data, i.e.

$$p(\boldsymbol{x}_t|\boldsymbol{\theta}_t) = \mathcal{N}(\boldsymbol{m}(\boldsymbol{\theta}_t), \boldsymbol{C}(\boldsymbol{\theta}_t)) \tag{22}$$

Substituting Equations (19) and (21) into Equation (18),

$$\begin{aligned} \mathrm{LL} &= \int \left[\prod_{\tau=1}^{N} q(\boldsymbol{\theta}_\tau|\boldsymbol{x}_{1:N})\right] \log\left(\prod_{t=1}^{N} p(\boldsymbol{x}_t|\boldsymbol{\theta}_t)\right) \mathrm{d}\boldsymbol{\theta}_{1:N} \\ &= \sum_{t=1}^{N} \int \left[\prod_{\tau=1}^{N} q(\boldsymbol{\theta}_\tau|\boldsymbol{x}_{1:N})\right] \log\left(p(\boldsymbol{x}_t|\boldsymbol{\theta}_t)\right) \mathrm{d}\boldsymbol{\theta}_{1:N} \end{aligned} \tag{23}$$

For each term in the summation we can factorise the integral as

$$\begin{aligned}\mathrm{LL} &= \sum_{t=1}^{N}\left[\int q(\boldsymbol{\theta}_t|\boldsymbol{x}_t)\log\left(p(\boldsymbol{x}_t|\boldsymbol{\theta}_t)\right)\mathrm{d}\boldsymbol{\theta}_t \prod_{\tau=1,\tau\neq t}^{N}\int q(\boldsymbol{\theta}_\tau|\boldsymbol{x}_{1:N})\mathrm{d}\boldsymbol{\theta}_\tau\right] \\ &= \sum_{t=1}^{N}\int q(\boldsymbol{\theta}_t|\boldsymbol{x}_t)\log\left(p(\boldsymbol{x}_t|\boldsymbol{\theta}_t)\right)\mathrm{d}\boldsymbol{\theta}_t.\end{aligned} \tag{24}$$

We can use a Monte Carlo estimate to calculate this as

$$\mathrm{LL} \approx \sum_{t=1}^{N}\frac{1}{M}\sum_{s=1}^{M}\log\left(p(\boldsymbol{x}_t|\boldsymbol{\theta}_t^s)\right), \tag{25}$$

where $\boldsymbol{\theta}_t^s$ denotes the s^{th} sample from the posterior distribution $q(\boldsymbol{\theta}_t|\boldsymbol{x}_{1:N})$ at time point t. In practice we use just one sample, i.e. $M = 1$. Therefore, the log-likelihood term is approximated by

$$\mathrm{LL} \approx \sum_{t=1}^{N}\log\left(p(\boldsymbol{x}_t|\boldsymbol{\theta}_t^1)\right). \tag{26}$$

Considering the KL divergence term,

$$\mathrm{KL} = \int q(\boldsymbol{\theta}_{1:N}|\boldsymbol{x}_{1:N})\log\left(\frac{q(\boldsymbol{\theta}_{1:N}|\boldsymbol{x}_{1:N})}{p(\boldsymbol{\theta}_{1:N})}\right)\mathrm{d}\boldsymbol{\theta}_{1:N}. \tag{27}$$

We use the mean field approximation for the posterior (Equation (19)) and factorise the prior as

$$p(\boldsymbol{\theta}_{1:N}) = p(\boldsymbol{\theta}_1)\prod_{t=2}^{N}p(\boldsymbol{\theta}_t|\boldsymbol{\theta}_{1:t-1}). \tag{28}$$

With these substitutions the KL divergence term becomes

$$\mathrm{KL} = \int\left[\prod_{\tau=1}^{N}q(\boldsymbol{\theta}_\tau|\boldsymbol{x}_{1:N})\right]\log\left(\frac{\prod_{t=1}^{N}q(\boldsymbol{\theta}_t|\boldsymbol{x}_{1:N})}{p(\boldsymbol{\theta}_1)\prod_{\kappa=2}^{N}p(\boldsymbol{\theta}_\kappa|\boldsymbol{\theta}_{1:\kappa-1})}\right)\mathrm{d}\boldsymbol{\theta}_{1:N}. \tag{29}$$

We split up the logarithm as

$$\mathrm{KL} = \int\left[\prod_{\tau=1}^{N}q(\boldsymbol{\theta}_\tau|\boldsymbol{x}_{1:N})\right]\left[\log\left(\frac{q(\boldsymbol{\theta}_1|\boldsymbol{x}_1)}{p(\boldsymbol{\theta}_1|\boldsymbol{x}_1)}\right)+\log\left(\prod_{t=2}^{N}\frac{q(\boldsymbol{\theta}_t|\boldsymbol{x}_{1:N})}{p(\boldsymbol{\theta}_t|\boldsymbol{\theta}_{1:t-1})}\right)\right]\mathrm{d}\boldsymbol{\theta}_{1:N} \tag{30}$$

and clip the first logarithm to give

$$\begin{aligned}\mathrm{KL} &\approx \int\left[\prod_{\tau=1}^{N}q(\boldsymbol{\theta}_\tau|\boldsymbol{x}_{1:N})\right]\log\left(\prod_{t=2}^{N}\frac{q(\boldsymbol{\theta}_t|\boldsymbol{x}_{1:N})}{p(\boldsymbol{\theta}_t|\boldsymbol{\theta}_{1:t-1})}\right)\mathrm{d}\boldsymbol{\theta}_{1:N} \\ &\approx \sum_{t=2}^{N}\int\left[\prod_{\tau=1}^{N}q(\boldsymbol{\theta}_\tau|\boldsymbol{x}_{1:N})\right]\log\left(\frac{q(\boldsymbol{\theta}_t|\boldsymbol{x}_{1:N})}{p(\boldsymbol{\theta}_{1:N}|\boldsymbol{\theta}_{1:t-1})}\right)\mathrm{d}\boldsymbol{\theta}_{1:N}\end{aligned} \tag{31}$$

For each term in the summation, we can factorise the integral as

$$\begin{aligned}\mathrm{KL} &\approx \sum_{t=2}^{N} \int \left[\prod_{\tau=1}^{t} q(\boldsymbol{\theta}_\tau|\boldsymbol{x}_{1:N})\right] \log\left(\frac{q(\boldsymbol{\theta}_t|\boldsymbol{x}_{1:N})}{p(\boldsymbol{\theta}_t|\boldsymbol{\theta}_{1:t-1})}\right) \mathrm{d}\boldsymbol{\theta}_{1:t} \left[\prod_{\kappa=t+1}^{N} \int q(\boldsymbol{\theta}_\kappa|\boldsymbol{x}_{1:N}) \mathrm{d}\boldsymbol{\theta}_k\right] \\ &\approx \sum_{t=2}^{N} \int \left[\prod_{\tau=1}^{t} q(\boldsymbol{\theta}_\tau|\boldsymbol{x}_{1:N})\right] \log\left(\frac{q(\boldsymbol{\theta}_t|\boldsymbol{x}_{1:N})}{p(\boldsymbol{\theta}_t|\boldsymbol{\theta}_{1:t-1})}\right) \mathrm{d}\boldsymbol{\theta}_{1:t}\end{aligned} \tag{32}$$

We denote the integral over $\mathrm{d}\boldsymbol{\theta}_t$ by

$$D_{\mathrm{KL}}\left(q(\boldsymbol{\theta}_t|\boldsymbol{x}_{1:N}) \mid\mid p(\boldsymbol{\theta}_t|\boldsymbol{\theta}_{1:t-1})\right) = \int q(\boldsymbol{\theta}_t|\boldsymbol{x}_{1:N}) \log\left(\frac{q(\boldsymbol{\theta}_t|\boldsymbol{x}_{1:N})}{p(\boldsymbol{\theta}_t|\boldsymbol{\theta}_{1:t-1})}\right) \mathrm{d}\boldsymbol{\theta}_t. \tag{33}$$

Substituting this into the KL divergence term, we get

$$\mathrm{KL} \approx \sum_{t=2}^{N} \int \prod_{\tau=1}^{t-1} q(\boldsymbol{\theta}_\tau|\boldsymbol{x}_{1:N}) D_{\mathrm{KL}}\left(q(\boldsymbol{\theta}_t|\boldsymbol{x}_{1:N}) \mid\mid p(\boldsymbol{\theta}_t|\boldsymbol{\theta}_{1:t-1})\right) \mathrm{d}\boldsymbol{\theta}_{1:t-1} \tag{34}$$

We use a Monte Carlo estimate using a single sample from each posterior $q(\boldsymbol{\theta}_1|\boldsymbol{x}_{1:N}), ..., q(\boldsymbol{\theta}_{t-1}|\boldsymbol{x}_{1:N})$. Therefore, our KL divergence term is

$$\mathrm{KL} \approx \sum_{t=2}^{N} D_{\mathrm{KL}}\left(q(\boldsymbol{\theta}_t|\boldsymbol{x}_{1:N})) \mid\mid p(\boldsymbol{\theta}_t|\boldsymbol{\theta}^1_{1:t-1})\right). \tag{35}$$

The prior $p(\boldsymbol{\theta}_t|\boldsymbol{\theta}^1_{1:t-1})$ is a multivariate normal distribution parameterised by a mean vector $\boldsymbol{\mu}_{\theta_t}(\boldsymbol{\theta}^1_{1:t-1})$ and diagonal covariance matrix $\boldsymbol{\sigma}^2_{\theta_t}(\boldsymbol{\theta}^1_{1:t-1})$, i.e.

$$p(\boldsymbol{\theta}_t|\boldsymbol{\theta}^1_{1:t-1}) = \mathcal{N}(\mu_{\theta_t}(\boldsymbol{\theta}^1_{1:t-1}), \boldsymbol{\sigma}^2_{\theta_t}(\boldsymbol{\theta}^1_{1:t-1})). \tag{36}$$

The parameters $\boldsymbol{\mu}_{\theta_t}(\boldsymbol{\theta}^1_{1:t-1})$ and $\boldsymbol{\sigma}^2_{\theta_t}(\boldsymbol{\theta}^1_{1:t-1})$ are calculated using the model RNN.

We use stochastic gradient descent to minimise a loss function. The loss function we use is

$$\begin{aligned}\mathcal{L} &= F = -\mathrm{LL} + \mathrm{KL} \\ \mathcal{L} &= -\sum_{t=1}^{N} \log\left(p(\boldsymbol{x}_t|\boldsymbol{\theta}^1_t)\right) + \sum_{t=2}^{N} D_{\mathrm{KL}}\left(q(\boldsymbol{\theta}_t|\boldsymbol{x}_{1:N}) \mid\mid p(\boldsymbol{\theta}_t|\boldsymbol{\theta}^1_{1:t-1})\right).\end{aligned} \tag{37}$$

Using this loss function will minimise the variational free energy, or equivalently, it will maximise the evidence lower bound [49].

8.2 Training and Hyperparameters

Before training the model we prepare the source reconstructed data. We applied time-delay embedding, PCA and standardisation. Time-delay embedding and PCA are summarised in Figure 15. The procedure used to train the model and choices for hyperparameters are discussed below.

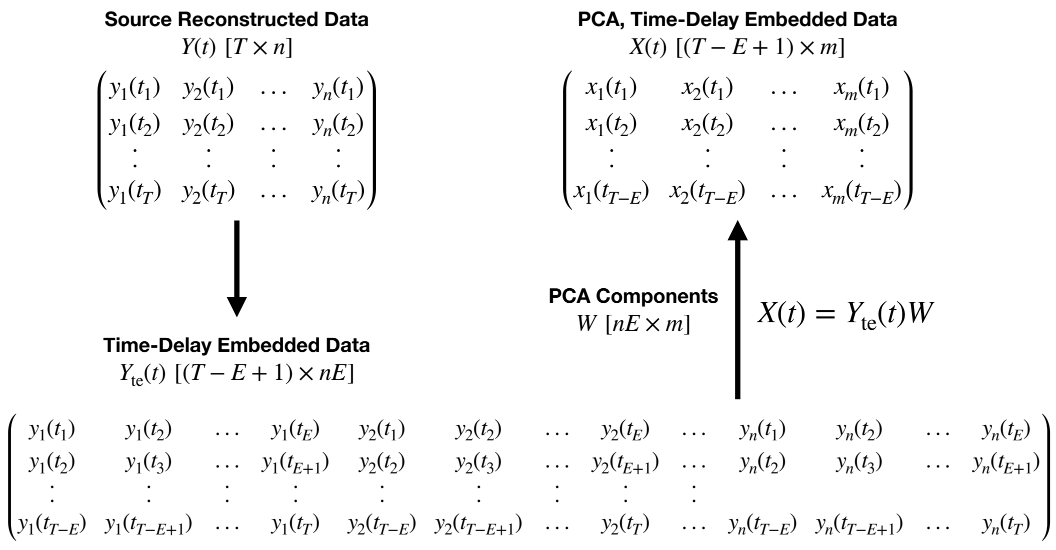

Source Reconstructed Data
$Y(t)$ $[T \times n]$
PCA, Time-Delay Embedded Data
$X(t)$ $[(T-E+1) \times m]$
PCA Components
W $[nE \times m]$
$X(t) = Y_{\text{te}}(t)W$
Time-Delay Embedded Data
$Y_{\text{te}}(t)$ $[(T-E+1) \times nE]$

Figure 15: Preparation applied to the source reconstructed data before training. The use of time-delay embedding encode spectral properties of the training data into the covariance. PCA is performed reduce the number of channels so that the data is not too large to fit within GPU memory. Standardisation is also applied after PCA. T is the number of time points, n is the number of parcels/regions of interest, E is the number of time-delay embeddings and m is the number of channels after PCA.

Logit activation function. To calculate the mixing coefficients from the logits we use a softmax function,

$$\alpha_{jt} = \{\zeta(\boldsymbol{\theta}_t)\}_j = \frac{\exp\left(\frac{\theta_{jt}}{\tau}\right)}{\sum_{j=1}^{J} \exp\left(\frac{\theta_{jt}}{\tau}\right)}, \tag{38}$$

where τ is a hyperparameter known as the temperature, discussed further below. The use of a softmax function imposes the constraint that the α_{jt}-values are positive and the sum of α_{jt} over j is equal to one.

Alpha temperature. We can specify a temperature τ for the softmax activation function. The temperature determines the amount of mixing between modes. A high temperature corresponds to a more even mixture, whereas a low temperature leads to more mutually exclusive modes. In this work, we allow the temperature to be a learnable parameter.

Learning rate. In this work, we use the Adam optimiser [82] to update trainable parameters. The learning rate η is a key hyperparameter of the Adam optimiser, which can affect training. Using a value that is too high can lead to divergence in the loss function, alternatively, one that is too small can lead to slow convergence.

Sequence length. The input to the model is a sequence of data points $\boldsymbol{x}_{1:N}$, where N is the length of the sequence. The sequence length determines the number of previous time points the model has access to. Therefore, we would like to use the longest sequence length possible. However, the sequence length is limited by the GPU memory that is available and the ability to train the model in a reasonable time frame.

Batch learning. Stochastic gradient descent is performed by estimating the loss for a small group of sequences, referred to as a *batch*. This loss is used to update the trainable parameters before the loss for a new batch is calculated. The number of sequences in a batch is referred to as the *batch size*. We calculate the loss for a batch by averaging the loss for each sequence in the batch. When performing batch learning it is important to shuffle the ordering of the sequences and batches. We did this by first separating the entire dataset into sequences, shuffling the order of the sequences, then grouping sequences in batches and performing one final random reordering to give the training dataset. One training loop through all of the batches is referred to as an *epoch*. The batches are not reshuffled between epochs.

Hidden units and number of LSTM layers. When using an LSTM we must specify a number of hidden units. We found small networks had more stable training than large networks, so only used 64 units in the model and inference LSTM. We also found stacking multiple LSTMs did not improve the model, so in this work we only use one LSTM layer.

Dropout [83]. This is well-known technique used when training a neural network to mitigate overfitting. The use of dropout was found not to benefit the model so no dropout layers have been used in this work.

Normalisation layers [84, 85]. Normalisation layers are often used to train deep neural networks because they help alleviate the the vanishing/exploding gradient problem [86]. In this work, we include a single Layer Normalisation [85] transformation to the output of the LSTMs in Equations (2) and (6) before the affine transformation.

Gradient clipping. RNNs can suffer from exploding gradients when backpropagation occurs through each time step. A strategy proposed in [87] to avoid this is gradient clipping, where we rescale the gradients so that their norm is a particular value when gradient norm would otherwise exceed this value. This strategy has been shown to improve training stability. In this work, we use gradient clipping when training on real MEG data.

Trainable parameters and initialisation. The trainable parameters in this model are:

- The weights and biases of the model and inference LSTM. At the start of training these are set randomly using Glorot initialisation [88].
- Layer Normalisation weights. The output of Layer Normalisation is centred around a learnable β-parameter and scaled with a γ-parameter [47, 85], these parameters were initialised using zeros and ones respectively.
- Dense layer weights and biases for the affine transformations. Glorot initialisation [88] was used for these parameters.
- The alpha temperature. This was initialised using a value of one.
- Elements of the mode means and covariances. In this work, we use zero vectors for the means and an identity matrix for the covariances. When training on MEG data we found the initialisation of the mode covariances to be important. We propose a strategy of initially training the model on the data for a randomly selected single subject to estimate its covariances. The covariances are initialised with the identity matrix when

training on a single subject. This removes the subject-to-subject variability in the data. Then, the model can be trained on the full dataset initialising with the single-subject mode covariances. We found this strategy helped to avoid local optima when learning a high latent dimensionality (e.g. $J > 10$).

Multi-start training. We find the model is sensitive to the initialisation of the trainable parameters, in particular the inference and model RNNs. When training on real MEG data, we observe that the model can converge to different local optima with the same dataset. To help find the global optimum, we propose a multi-start approach, where we train the model for a small number of epochs a few times and performing the full training on the model with the lowest loss at the end of the initial training period. This procedure was used when training on real MEG data and was found to reduce the run-to-run variability.

KL annealing [54] is a technique used at the start of training. An annealing factor λ is introduced into the loss function,

$$\mathcal{L} = -\mathrm{LL} + \lambda \cdot \mathrm{KL}. \tag{39}$$

The annealing factor is a smoothly varying function of the number of training epochs. It takes a value between zero and one. The function used to calculate the annealing factor in this work is

$$\lambda = \frac{1}{2} \tanh\left(\frac{A_S(n_\mathrm{E} - \frac{1}{2} n_\mathrm{AE})}{n_\mathrm{AE}}\right) + \frac{1}{2}, \tag{40}$$

where A_S is the annealing sharpness, which determines the shape of the annealing curve, n_E is the number of training epochs and n_AE is the number of annealing epochs.

At the start of training, the weights and biases of the model RNN are initialised with samples from a uniform distribution (Glorot initialisation [88]). As a consequence, the model RNN is prone to giving naive outputs/estimates for the logit time course in these early epochs. KL annealing with λ close to zero helps in this period because it prevents the model RNN from influencing the inference RNN before it has learnt a logit time course that is useful for describing the training data. As training progresses, λ tends to one and the model RNN learns the temporal dynamics in the latent representation by predicting probability distribution of the next logit $\boldsymbol{\theta}_t$ from the samples of previous logits $\boldsymbol{\theta}_{1:t-1}$. By doing this, it also regularises the inferred logits for the training data.

8.3 Run-to-Run Variability and Number of Modes

When training DyNeMo, our objective is to the minimise the loss function by updating the trainable parameters of the model. Due to the random initialisation of the RNN weights and the use of stochastic gradient descent to update the parameters, it is possible for DyNeMo to converge to different local optima in the loss function. Neuroimaging data itself is complex and it is reasonable to expect there could be multiple models with different parameters that can lead to similar loss values. In this section we look at the variability in the final loss value for different training runs and look at the impact of varying the number of modes.

To evaluate the run-to-run variability in the loss we use the resting-state MEG dataset described in Section 2.3.3. We split the dataset into a 45 subject training dataset and 10 subject validation dataset. We trained DyNeMo 10 times on the training dataset with a

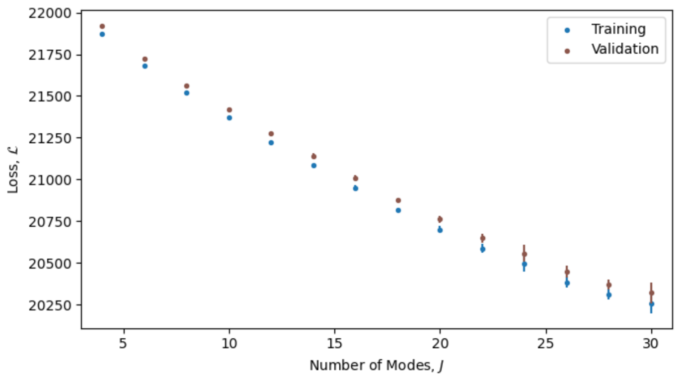


Figure 16: More modes provide a better description of the data. The mean training (blue) and validation loss (brown) for a batch vs the number of modes. Each data point is the average of ten runs. The error bar is one standard deviation. For 25 modes or less the error bar is too small to be seen.

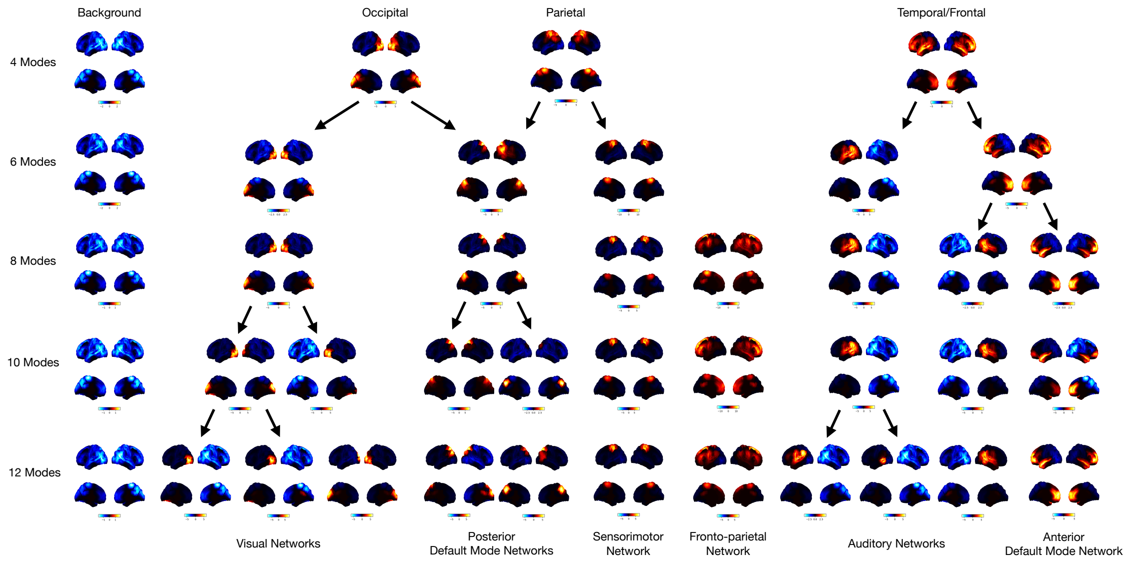


Figure 17: Power maps show more localised activation with an increasing number of modes. These power maps were obtained from a model trained on resting-state MEG data from 45 subjects. Arrows indicate when a power map has split as the number of modes was increased. No thresholding was applied to the power maps.

varying number of modes, J. All other hyperparameters for the 10 runs remained the same. Figure 16 shows the average training and validation loss for each number of modes. Although there's a systematic offset in the validation loss compared to the training loss, which suggests overfitting to the 45 subject dataset, the difference remains the same as the number of modes is increased, which means overfitting does not get worse with more modes. Over this range, the loss decreases monotonically meaning more modes provide a better description of the data. However, as the number of modes increases so does the variability in the loss. Our strategy for finding the global optimum involves training DyNeMo multiple times and selecting the model with the lowest loss.

Figure 17 shows the power maps for the run with the best loss when fitting 4-12 modes. As the number of modes increases the activation of each brain region becomes more localised. This is expected as increasing the number of modes allows greater precision in reconstructing the time-varying covariance $\boldsymbol{C}_t$ from the mode covariances $\boldsymbol{D}_j$ (see Equation (4)). When we fit 4 modes, we see power is divided amongst the four lobes: the occipital, parietal, temporal and frontal lobe. As we increase the number of modes, the number of possible ways to distribute power amongst the modes also increases. In this work, we choose to fit 10 modes to MEG data as a trade off between obtaining an interesting description of the data and minimising the variability in the power maps of each run.

8.4 Mode PSDs

Figure 18 shows the PSD of each mode calculated with the regression method described in Section 2.4 for the resting-state MEG dataset. In Figure 18 (left) we see all modes have a PSD with a $1/f$ profile and exhibit a prominent 10 Hz peak. The drop off towards 0 Hz is due to filtering applied during preprocessing. In Figure 18 (right) we see the mode PSD relative to the activity shared across modes. We see differences in the PSD of each mode consistent with expected frequency content of activity at locations shown in the corresponding power maps.

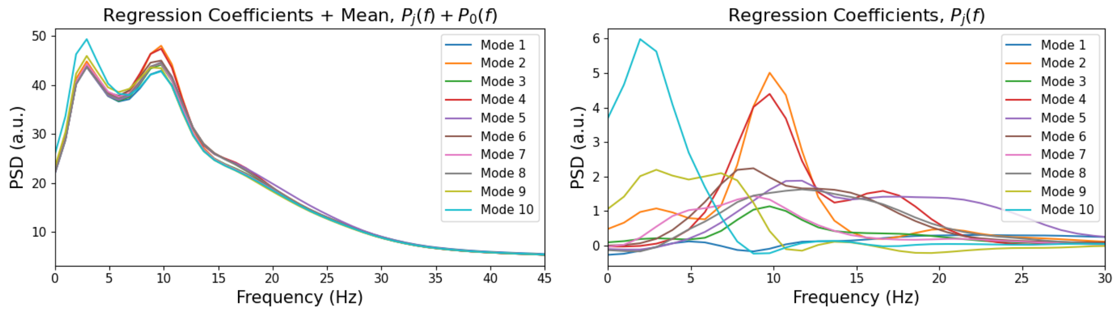


Figure 18: DyNeMo learns spectrally distinct modes. Left: mode PSDs including the activity common to all modes. Right: mode PSD relative to the mean activity common to all modes. The mean PSD across channels is shown.

8.5 HMMs Trained on the MEG Datasets

The HMM was also fitted to the MEG datasets described in Section 2.3.3 for comparison with DyNeMo. Figure 19 shows the power maps, FC maps and PSDs of the HMM states inferred on the resting-state MEG dataset and Figure 20 shows the power maps, FC maps and PSDs of the HMM states inferred on the task MEG dataset.

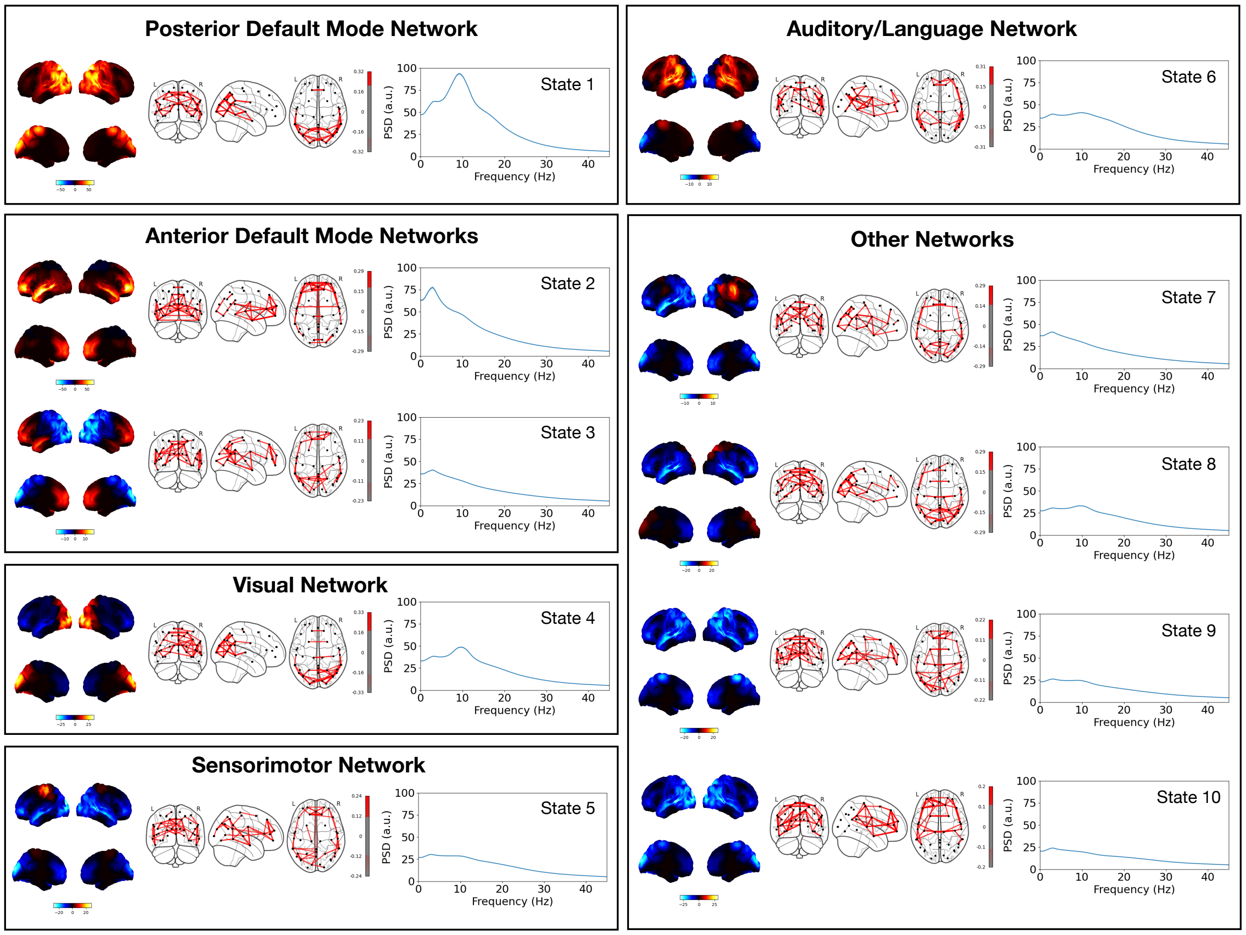


Figure 19: HMM states inferred on a resting-state MEG dataset consisting of 55 subjects. Each box shows the power map (left), FC map (middle) and PSD averaged over regions of interest (right) for each group. No thresholding was applied to the power maps. The top 5% of connections is shown in the FC maps. The shaded area in the PSD plots shows the standard error on the mean. These power and FC maps are the first wideband (1-30 Hz) component calculated using the multitaper and non-negative matrix factorisation approach described in [67].

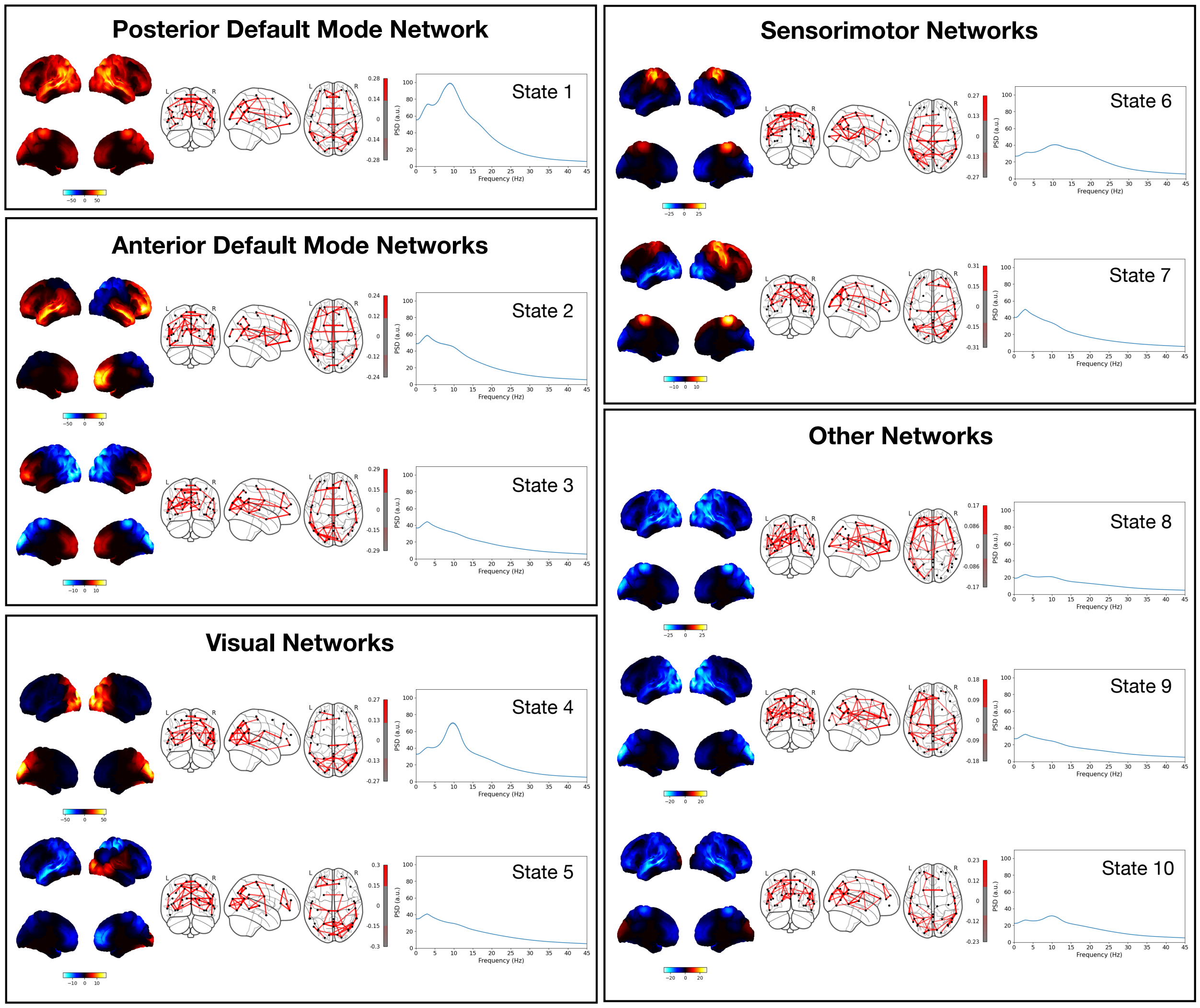


Figure 20: HMM states inferred on a visuomotor task MEG dataset consisting of 51 subjects. Each box shows the power map (left), FC map (middle) and PSD averaged over regions of interest (right) for each group. No thresholding was applied to the power maps. The top 5% of connections is shown in the FC maps. The shaded area in the PSD plots shows the standard error on the mean. These power and FC maps are the first wideband (1-30 Hz) component calculated using the multitaper and non-negative matrix factorisation approach described in [67].