Differentiable programming for functional connectomics

Editors: List of editors' names

Abstract

Mapping the functional connectome has the potential to uncover key insights into brain organisation. However, existing workflows for functional connectomics are limited in their adaptability to new data, and principled workflow design is a challenging combinatorial problem. We introduce an analytic paradigm that implements common operations used in functional connectomics as fully differentiable processing blocks. Under this paradigm, workflow configurations exist as reparameterisations of a differentiable functional that interpolates them. The differentiable program that we ultimately envision occupies a niche midway between traditional pipelines and end-to-end neural networks, combining the glass-box tractability and domain knowledge of the former with the amenability to optimisation of the latter. In this preliminary work, we provide a proof of concept for differentiable connectomics, demonstrating the capacity of our processing blocks across three separate problem domains critically important to brain mapping. We also provide a software library to facilitate adoption. Our differentiable framework is competitive with state-of-the-art methods in functional brain parcellation, time series denoising, and covariance modelling. Taken together, our results demonstrate the promise of differentiable programming for functional connectomics.

Keywords: fMRI, brain connectivity, differentiable programming, self-supervised, parcellation

1. Introduction

Many scientific disciplines depend on complex analytic workflows to extract salient information from data. In the health sciences, and in large-scale brain mapping in particular, the development of these workflows has grown to become a subdiscipline in its own right. Although the introduction of better tools is a necessary vector of progress, informatic practice also comes at the cost of increased analytic flexibility, or "researcher degrees of freedom" (Carp, 2012; Botvinik-Nezer et al., 2020). In other words, the development of new instruments presents researchers with the combinatorial challenge of selecting, from among the growing set of available informatic tools, an analytic workflow conditioned on their dataset and scientific question.

This challenge is significant for two reasons. First, many reported results fail to replicate across the space of workflow configurations (Carp, 2012; Botvinik-Nezer et al., 2020). Second, publication bias obscures the often iterative process of tool development, and a skewed scientific incentive structure motivates informaticians to refine algorithms until they outperform state-of-the-art (SOTA) methods on benchmark datasets. Anecdotally, this is often accompanied by either improper or incomplete implementation of the SOTA baselines, or insufficient consideration of benchmark dataset properties that might lead to inflated performance (e.g. Gorman and Bedrick (2019); Lipton and Steinhardt (2019)).

The problem of designing a data transformation workflow in a principled way is

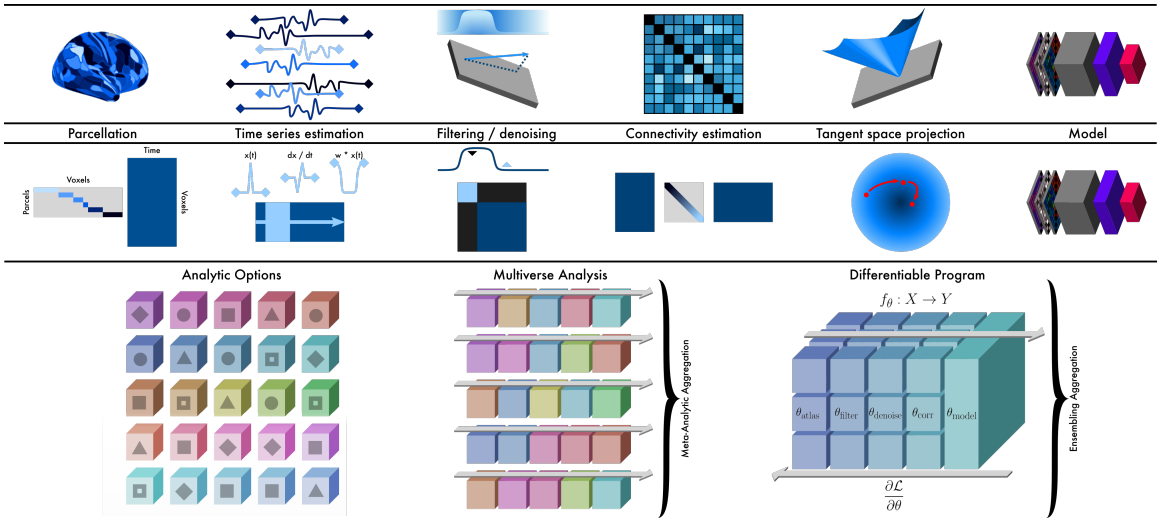


Figure 1: Problem setting and aspirational overview. Details in text.

thus of interest to many scientific disciplines, particularly areas such as medical imaging and brain mapping where ground truths are largely unknown or inaccessible. *Differentiable programming* promises one potential resolution to this problem. A differentiable program can instantiate each block of a workflow as a neural network module that subsumes existing analytic options under reparameterisation. In other words, this program is a differentiable functional that interpolates over existing workflow configurations, relaxing the combinatorial problem of principled workflow design into one that can be optimised locally using gradient methods.

Functional connectomics is the enterprise of creating, refining, and explaining whole-brain maps of synchrony and statistical dependence in order to develop an understanding of how neuronal populations communicate with one another (Smith et al., 2013). The standard functional connectivity workflow (Figure 1, *Top*) begins with a preprocessed blood oxygenation level-dependent (BOLD) time series—a proxy for underlying neural activity (Logothetis et al., 2001). A *parcellation* block first reduces the dimension of this input by mapping whole-brain activity to a set of functionally independent regions or parcels. The estimated parcel-wise time series are then *denoised* to mitigate the influence of artefacts. Next, some measure of *connectivity* among the denoised time series is estimated before it is passed to a final *model* that is fit to some research objective. Differentiable programming enables gradients to propagate back from the model to reconfigure the parameters of the workflow itself. While our aspirational objective is an end-to-end differentiable workflow, in the current work we focus on the near-term goals of solving specific subproblems within each block of the workflow.

Functional connectomics is an attractive test bed for prototyping differentiable programs because nearly all atoms of the workflow described above are either immediately differentiable or have differentiable relaxations. We developed an open source software library, `hypercoil`[1], to facilitate differentiable programming for functional connectomics. Using this library, we then conducted a series of experiments, detailed below, as a proof of concept for the differen-

1. `URL redacted for anonymity`

tiable programming paradigm in functional connectomics.

2. Related work

Parameterising the entire neuroimaging software stack as a neural network was first proposed by Vilamala et al. (2016). The current work was directly inspired by comprehensive evaluations of functional connectivity workflows in the prediction setting (Pervaiz et al., 2020; Dadi et al., 2019). Churchill et al. (2017) introduced an algorithm for adaptive neuroimaging workflow optimisation using a cross-validation framework. Dafflon et al. (2020) developed a learning method for efficiently estimating an approximate map of the space of workflow configurations and of their performance on a prediction objective.

Differentiable programming is not the only promising approach for resolving the problem of principled workflow design. A complementary paradigm, *multiverse analysis* (Botvinik-Nezer et al., 2020), is based on evidence that, although results gleaned from individual workflow configurations are often brittle and workflow-sensitive, meta-analytic aggregation across the "multiverse" of workflow configurations can separate workflow-sensitive results from those that are robust and reproducible (Botvinik-Nezer et al., 2020). Differentiable programming enables a complementary kind of aggregation: ensembling across parallel program blocks parameterised to exploit complementary sources of information (Figure 1, *Bottom*).

3. Experiments

Functional parcellation The delineation of functional subunits of the brain is among the longest standing problems in neuroscience. Practically, brain parcellation is also a critical dimension-reducing step that reduces the computational requirements of all downstream workflow blocks. For BOLD fMRI data, the objective of the parcellation problem is to learn a mapping $\mathbf{A} \in \mathbb{R}^{p \times v}$ from the vertex-wise time series $\mathbf{T_i} \in \mathbb{R}^{v \times t}$ to the parcel-wise time series $\mathbf{T_o} \in \mathbb{R}^{p \times t}$ subject to two desiderata: (i) minimal loss of information and (ii) parcels that are neuroscientifically relevant. In practice, this mapping is usually operationalised as either a linear parcellation matrix ($\mathbf{T_o} = \mathbf{A}\mathbf{T_i}$) or a linear projection ($\mathbf{T_o} = (\mathbf{A}\mathbf{A}^\intercal)^{-1}\,\mathbf{A}\mathbf{T_i}$) (Beckmann et al., 2009; Dadi et al., 2020). Here we take the former approach for its simplicity, although our method is easily extended to the latter. Each entry $\mathbf{A}_{ij}$ in the parcellation matrix encodes the model's estimate of the probability that vertex j is assigned to parcel i. We initialise each column of $\mathbf{A}$ by first sampling from a Dirichlet distribution and then log-transforming the Dirichlet samples. During each forward pass, the parcellation logits are projected to the probability simplex using a softmax mapping.

To learn the parcellation matrix $\mathbf{A}$, we combine four terms into a differentiable temporal-spatial clustering (dTSC) objective (Figure 2, *Top left*; Table 1). A *compactness* loss penalises the distance from each vertex in a parcel to the parcel's centre of mass using vertex coordinates $\mathbf{C} \in \mathbb{R}^{v \times 3}$, thereby promoting compact parcels. A complementary *dispersion* objective promotes separation of the parcels' centres of mass. All distances are computed as spherical geodesics after projecting data from each cortical hemisphere onto a spherical mesh (Fischl et al., 1999; Robinson et al., 2014).

To minimise information loss and learn functionally uniform parcels, we wish to assign vertices with similar signals to the same parcel and vertices with different signals to different parcels. Accordingly, in addition to the two spatial terms, temporal loss terms promote parcel homogeneity. A *second moment* objective quadratically pe-

Table 1: The four loss terms of the differentiable temporal-spatial clustering (dTSC) objective. Detailed definitions in Appendix A.

	Spatial	**Temporal**
Within	*Compactness* $\mathbf{A} \circ \left\|\mathbf{C} - \frac{\mathbf{AC}}{\mathbf{A1}}\right\|_{\text{cols}}$	*Second Moment* $\left[\mathbf{A} \circ \left(\mathbf{T_i} - \frac{\mathbf{T_o}}{\mathbf{A1}}\right)^2\right] \frac{\mathbf{1}}{\mathbf{A1}}$
Between	*Dispersion* $-\sum_{i,j} \mathrm{d}\left(\frac{\mathbf{A_i C}}{\mathbf{A_i 1}} - \frac{\mathbf{A_j C}}{\mathbf{A_j 1}}\right)$	*Determinant* $-\log\det \mathrm{corr}(\mathbf{T_o})$

nalises parcels for including vertices whose time series differ from the parcel's time series. Finally, we promote temporally independent parcels by placing a penalty on the negative *log-determinant* of the correlation matrix among parcel time series. Our approach is readily modified to produce null parcellations: ablating the temporal loss terms leaves a model that learns entirely from spatial relationships, without using any information from the brain time series.

We supplement the dTSC objective with three simple regularisations. First, to obtain parcels of similar size, we impose an L2 parcel equilibrium penalty. Second, we promote analogy across cortical hemispheres by tethering each parcel's centre to an analogue in the opposite hemisphere. Third, to obtain deterministic parcels, we penalise the entropy of each vertex's parcel assignment distribution. Because a strong entropy penalty can lead the model to fixate irreversibly on its maximum assignment, we begin with a small multiplier for the entropy term and progressively cascade it upward over the course of training. The cascading entropy loss induces parcel probabilities to converge toward a deterministic assignment, at the cost of worsening parcel homogeneity as reflected by a growing second-moment term (Figure 3a, *Top*), with boundary regions slowest to converge (Figure 3a, *Bottom*, entropy multipliers denoted in black boxes). Because the boundaries of brain subsystems vary substantially between individuals (Laumann et al., 2015; Gordon et al., 2017), we smooth the objective using a modification of the stochastic weight averaging (SWA; Izmailov et al. (2019)) algorithm for better compatibility with entropy penalties.

We used the dTSC objective to train parcellation models corresponding to 5 spatial scales (300, 400, 666, 800, 1000 cortical parcels) using 2457 images from 618 subjects from the Human Connectome Project (HCP) dataset (Van Essen et al., 2013) (training details in appendices). In comparison with a null model, the dTSC parcellations better reflected the contours of the brain's canonical functional subsystems (Figure 2, *Top*; Yeo et al. (2011)), with starkest differences at coarser resolutions (e.g., 300 parcels). On average, the deterministic parcels learned by our model also achieved better signal homogeneity than comparably sized parcels from the null model and performed competitively with state-of-the-art methods (gwMRF: gradient-weighted Markov Random Field (Schaefer et al., 2018); MMP: multimodal parcellation (Glasser et al., 2016); Grad. BM: gradient boundary mapping (Gordon et al., 2016)) in held-out data (Figure 2, *Middle*). (We show the homogeneity as a function of parcel size because, *a priori*, smaller parcels are likely to be more homogeneous.)

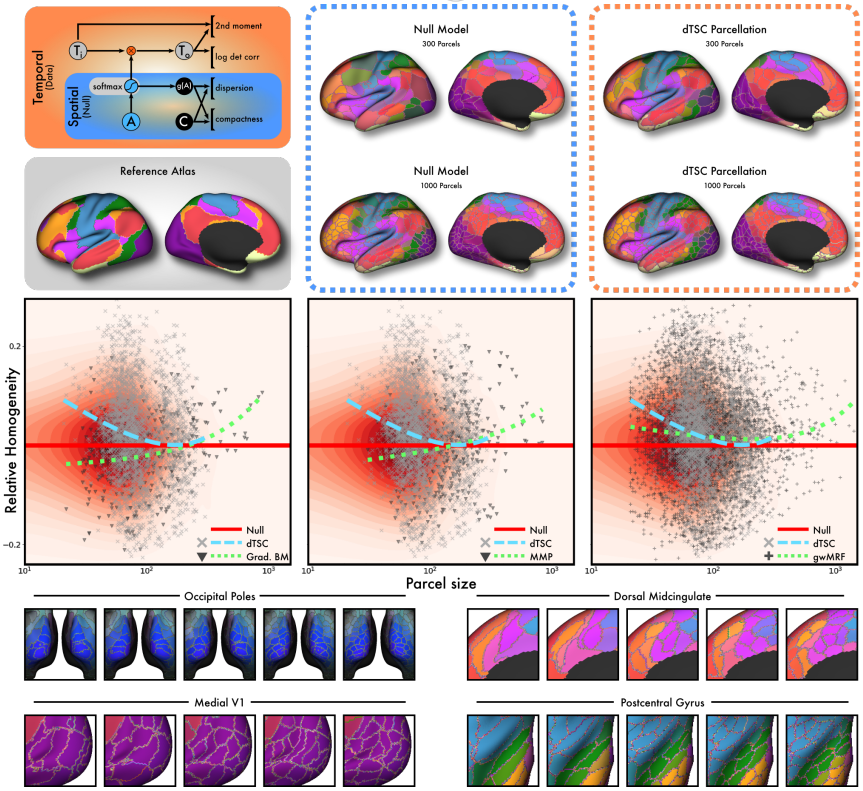
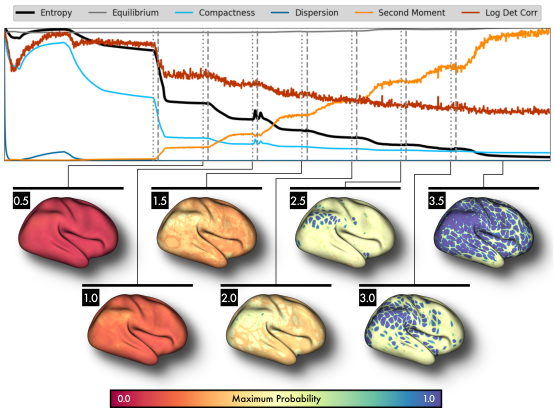


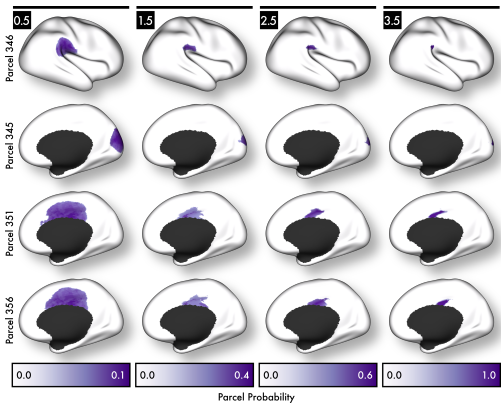
Figure 2: dTSC parcellation. *(Top left)* Schematic of the dTSC model. *(Top)* The model recovers reference functional boundaries (Yeo et al., 2011) with success relative to a null model. *(Middle)* Quantitative assessment of parcellation homogeneity (vertical axis) as a function of parcel size (horizontal axis) (higher is better). The dTSC method is compared against three different parcellations from the literature, as well as a spatial null model. The point clouds show the mean homogeneity (across the test sample) of all individual parcels from the designated algorithm, and the red density in the background similarly represents the mean homogeneity of parcels from the null model. The curves are least-squares fits of homogeneity as a logarithmic function of parcel size, adjusted via subtraction so that the fit to the null model is zero. *(Bottom)* Consistency and difference across five parcellation resolutions in four representative regions.

Across spatial resolutions, we observed consistency and difference in parcel structure (Figure 2, *Bottom*). Although some areal boundaries (notably of the occipital visual cortex and medial V1) were reproduced with high fidelity across resolutions, we qualitatively found greater consistency in the orientation of local axes of signal similarity and difference, as reflected in the eccentricities and orientations of parcel contours. This pattern suggests a limitation of areal parcellation schemes such as dTSC; parcellations informed by connectopic gradients (e.g., Tian et al. (2020)) offer a potential direction for future improvement.

Increasing the entropy penalty over the course of training (Figure 3b) lends our scheme another interesting characteristic: the spatial extent of parcels decreases, and their maximum assignment probability increases toward unity. Qualitatively, this is reflected in a shift from spatially overlapping, probabilistic modes of brain activity (e.g., Dadi et al. (2020)) toward determin-



(a) Parcel assignment uncertainty is modulated by a progressive penalty on the distribution entropy. The paired vertical lines denote training cycles during which the entropy multiplier is cascaded upward.



(b) The learned parcellation proceeds from distributed functional modes to deterministic areal parcels. Here, we show four exemplar parcels from the right cortical hemisphere of a 666-region parcellation at the end of each of four stages of the entropy cascade.

Figure 3: Overview of the parcellation learning process using the dTSC model.

istic areal assignments. In particular, the shown areal parcels 351 and 356 begin with near complete overlap and substantially differentiate only in the late stages of training. The progression of parcel differentiation during training can potentially be used to investigate hierarchically nested modes of brain function (Pines et al., 2022). As complementary information might be offered by overlapping probabilistic modes and circumscribed parcels, the stopping entropy also becomes a hyperparameter of the differentiable program.

Our approach differs from previous work in that it is both fully differentiable and designed to fit entirely in GPU memory. We accomplish this principally using functional transformations that serialise the most expensive loss computations over sub-blocks of the input dataset. However, we find that (even subject to the constraints of GPU memory) we can achieve parcel homogeneities that are competitive with SOTA in held-out data (Figure 2, middle row).

Artefact removal The movement of subjects during fMRI acquisition introduces artefactual fluctuations into the BOLD signal (Power et al., 2012), which typically inflate estimates of functional connectivity. To make matters worse, in-scanner movement is correlated with measures of scientific or clinical interest, such as age and diagnostic labels (Satterthwaite et al., 2012). In the functional connectivity workflow, denoising is typically implemented by residualising the BOLD time series with respect to a *confound model* that is thought to approximately explain structured sources of artefact. We adopt this framework here.

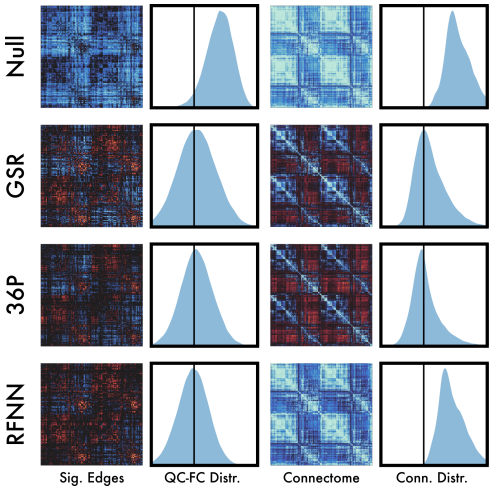
In particular, let Σ_{XX} be the covariance matrix of BOLD time series, Σ_{YY} be the covariance matrix of confound time series, and Σ_{XY} be the matrix of covariances between BOLD and confound time series. We operationalise the residual functional connectivity as the *conditional covariance*

$$\Sigma_{X|Y} = \Sigma_{XX} - \Sigma_{XY}\Sigma_{YY}^{-1}\Sigma_{XY}^{\intercal}$$

This is equivalent to the covariance of BOLD data that have been residualised with respect to a linear least-squares fit of the confounds. *Ceteris paribus*, a more parsimonious confound model—one with fewer time series—

Figure 4: A shallow denoising model (RFNN) trained on the QC-FC loss outperforms SOTA methods in held-out data.

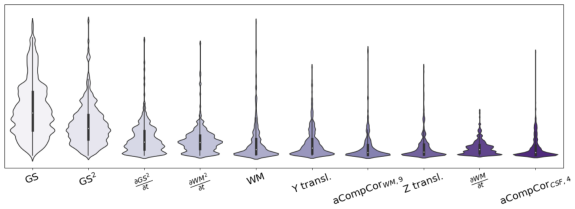
(a) The RFNN model learned to remove a single nuisance regressor from BOLD images so as to reduce motion-related variance.



(b) QC-FC benchmark results (lower is better).

Model	Abs. Med. Corr.	N. Sig. Edges
Null	0.16640	50765
GSR	0.09036	14558
36P	0.07487	11099
RFNN	**0.07185**	**6439**

(c) Top *a priori* confounds that loaded onto the RFNN model were global signal (GS)-related.



is thought to be better because it removes fewer degrees of freedom from the BOLD data (Pruim et al., 2015). Thus, in this experiment, our objective is to remove as much motion-related variance as possible from the BOLD data using only a single learned confound time series.

We pursue this objective by parameterising a simple, shallow neural network model (RFNN) to learn a single linear combination of 57 confound time series selected *a priori* for their demonstrated efficacy in denoising (Ciric et al., 2017) (further details in Appendix B). These include direct estimates of subject movement; mean signals from high-noise tissue compartments (white matter—WM, cerebrospinal fluid—CSF); the overall global signal across the entire brain (GS); and localised signals obtained using singular value decompositions of WM and CSF (CompCor; Behzadi et al. (2007)). We parameterise the RFNN to also learn 5 response functions; the RFNN is permitted to select any linear combination of the 57 *a priori* confound time series and their $285 = 5 \times 57$ response function convolutions.

To train the RFNN to optimally remove motion artefact, we introduce the *QC-FC* loss function, a differentiable implementation of the widely used QC-FC benchmark for residual motion artefact (Power et al., 2014). The QC-FC loss computes the correlation between a gross estimate of subject motion (QC, detailed in Appendix B) and each edge of the functional connectivity matrix (FC) across the batch dimension. Thus, QC-FC loss goes to zero when subject movement and functional connectivity are uncorrelated. We trained the RFNN model on a dataset of 1393 images from 356 subjects.

In a held-out sample of 351 BOLD images from the HCP dataset (Figure 4), the RFNN model outperformed both the top-performing *a priori* one-confound model (GSR, global signal regression) and a 36-confound model (36P, a superset of GSR) that has previously been demonstrated to give SOTA performance (Ciric et al., 2017). Both the number of connections significantly related to motion (Sig. Edges; $p < 0.01$, uncorrected) and the median of the size of motion effects (QC-FC Distr.) were reduced

relative to all evaluated *a priori* models. We repeated this evaluation with fairly convergent results across 12 different splits of the HCP dataset (Appendix B). While these results show promise, further research is required to understand potential limitations: the functional connectome denoised using the RFNN did not feature the zero-centred correlations that are a hallmark of successful GSR-based denoising (Murphy et al., 2009). It is possible that this reflects the specificity of the learned model for removing motion-related variance—by contrast, GSR-based models also effectively remove respiratory and cardiac artefacts, which typically inflate global estimates of connectivity (Power et al., 2017). With this and additional benefits of GSR in consideration (Li et al., 2019), we recommend use of GSR in functional connectivity workflows; future work will explore integration of RFNN-like models with GSR. Notably, when we computed the extent to which *a priori* confounds captured variance in the RFNN confound, the top three loaded *a priori* confounds were all GSR-related: the global signal (GS), its square (GS^2), and the temporal derivative of the square ($\frac{\partial GS^2}{\partial t}$) (Figure 4C, distribution across images).

Covariance modelling The functional connectome is typically estimated using a derivative of covariance, such as Pearson correlation, among all pairs of denoised parcel time series. The usual definition of empirical covariance among a set of time series $\mathbf{T}$ can be generalised via parameterisation by the (typically diagonal) weight matrix $\mathbf{\Theta}$: $\mathbf{C}_{\mathbf{\Theta}} = \frac{1}{n-1}\left(\mathbf{T} - \bar{\mathbf{T}}\right)\mathbf{\Theta}\left(\mathbf{T} - \bar{\mathbf{T}}\right)^{\intercal}$.

In this proof of concept, we use this parameterisation to track the dynamics of community structure in the brain. The objective of community detection is to learn a partition of graph vertices (here, brain parcels) into *communities* whose members preferentially connect to one another (e.g., Blondel et al. (2008)). In connectomics, this corresponds to identification of modular subsystems of brain function. When combined with a community detection model, the covariance parameterisation $\mathbf{\Theta}$ can be trained to represent dynamic time courses that track the modularisation and demodularisation of these functional subsystems.

Early algorithms for community detection were not differentiable, although more recently neural networks have been applied to this problem domain (Su et al., 2022). Here, we opt for a simpler approach, because parcellated connectomes typically have fewer vertices than the large-scale graphs for which deep methods were developed. Our approach also differs from dynamic community detection algorithms (e.g., Thompson et al. (2019); Martinet et al. (2020)) in that existing algorithms typically aim to characterise the temporal evolution of community boundaries, whereas the objective of our approach is to delineate epochs during which communities coalesce and dissipate.

We train our model using a differentiable relaxation of the Girvan-Newman modularity objective (Newman and Girvan, 2004), which indexes the ability of a proposed community structure to explain the arrangement of edges in a real graph relative to a null model. Formally, let $\mathbf{A} \in \mathbb{R}^{v \times v}$ be the graph adjacency matrix and $\mathbf{C} \in \mathbb{R}^{v \times c}$ be a proposed assignment of v vertices to c communities. The *modularity matrix* $\mathbf{B} = \mathbf{A} - \gamma\mathbf{P}$ expresses the extent to which edges in the observed graph deviate from expectation under the null model $\mathbf{P}$, subject to a resolution hyperparameter γ. Here, we use $\gamma = 5$ and the original Girvan-Newman null model, $\mathbf{P}_{GN} = \frac{\mathbf{A}\mathbf{1}\mathbf{1}^{\intercal}\mathbf{A}}{\mathbf{1}^{\intercal}\mathbf{A}\mathbf{1}}$, which can be interpreted as the expected weight of connections between each pair of vertices if all existing edges are cut and then randomly rewired. The modularity objective is then $\mathcal{L}_Q = \mathbf{1}^{\intercal}(\mathbf{H} \circ \mathbf{B})\mathbf{1}$, where $\circ$ denotes the Hadamard product and

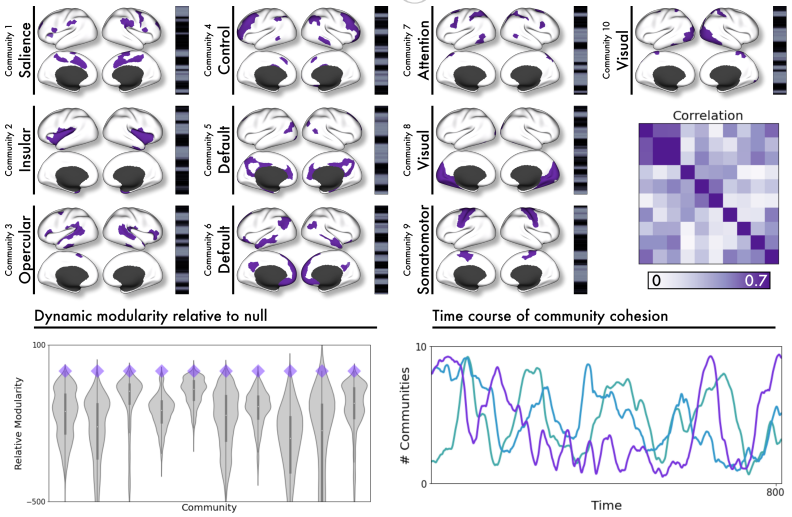


Figure 5: A community detection model learns to track the modularisation and demodularisation of brain subsystems.

$\mathbf{H} = \mathbf{C}\mathbf{C}^\intercal$ is the community coaffiliation matrix. Our goal is to learn a $\mathbf{C}$ that maximises the modularity objective. If we constrain all entries in $\mathbf{C}$ to $\{0, 1\}$, this relaxation converges to the original definition of modularity (Newman and Girvan, 2004). To maintain differentiability, we instead learn the logits of $\mathbf{C}$, which we pass through a softmax, permitting community assignments to vary continuously in the probability simplex.

The modularity objective is applied to the time-averaged connectome and combined with time-selective modularity objectives applied separately for each community. Specifically, for each community $\mathcal{C}_i$, we let $\mathbf{A}_{\mathcal{C}_i}$ be the covariance parameterised by a learnable time course $\mathbf{\Theta}_{\mathcal{C}_i}$ of the community's modularisation, and $\mathbf{H}_{\mathcal{C}_i} = \mathbf{C}_i\mathbf{C}_i^\intercal$.

Training this unsupervised model on 30 parcellated time series from the Midnight Scan Club (MSC) dataset (Gordon et al., 2017), we identify a set of dynamic communities that accord well with previously characterised brain subnetworks. At the top of Figure 5, we show the learned spatial distributions $\mathbf{C}_i$ of each community and an example time course $\mathbf{\Theta}_{\mathcal{C}_i}$ of the community's modularisation (grey: present, black: absent) in a single BOLD image. Relative to a null model obtained by randomly shifting modularisation time courses, the learned model consistently obtained better modularity (Figure 5, *Bottom left*: the purple shows the learned model, and the grey silhouette the modularity distribution of all null models relative to the learned model). We also find that communities tend to coalesce and dissipate in tandem (Figure 5 *Right*, correlation matrix and time course showing the number of modularised communities for 3 example images), consistent with previous work that characterised transient "excursions" of connectivity (Betzel et al., 2016).

4. Discussion

Our experimental results demonstrate the promises of fully differentiable methods across several domains of functional connectomics. However, there exists a central philosophical objection against the differentiable programming paradigm. The argument goes as follows: differentiable programming does not in fact resolve the problem of workflow design; instead, it is merely a re-delegation of the workflow design burden from selection of analytic options to specification of workflow hyperparameters. Different hyperparame-

ter configurations (e.g., the balances of loss and regularisation multipliers) are inherently going to produce different "optimal" workflows. Although it can be argued that the hyperparameter selection process sometimes offers researchers more transparent control over workflow design, because a numerical objective is more intuitively related to a research objective, this is often not the case.

Although responses to this argument (e.g., mapping and understanding the most relevant subsets of the hyperparameter space; strides in automating the process of hyperparameter optimisation, e.g. He et al. (2021)) are under active development in the machine learning world, this does not fully address doubts. Nevertheless, even if a differentiable program does not completely address the problem of principled workflow optimisation, it has the potential to provide both an engine for new discoveries and an informative complement to multiverse analysis.

References

Soroosh Afyouni and Thomas E. Nichols. Insight and inference for DVARS. *NeuroImage*, 172:291–312, May 2018. doi: 10.1016/j.neuroimage.2017.12.098.

Alex Aizman, Gavin Maltby, and Thomas Breuel. High Performance I/O For Large Scale Deep Learning. In *2019 IEEE International Conference on Big Data (Big Data)*, pages 5965–5967, Los Angeles, CA, USA, December 2019. IEEE. ISBN 978-1-72810-858-2. doi: 10.1109/BigData47090.2019.9005703.

Elena A. Allen, Eswar Damaraju, Sergey M. Plis, Erik B. Erhardt, Tom Eichele, and Vince D. Calhoun. Tracking Whole-Brain Connectivity Dynamics in the Resting State. *Cerebral Cortex*, 24(3):663–676, March 2014. doi: 10.1093/cercor/bhs352.

C.F. Beckmann, C.E. Mackay, N. Filippini, and S.M. Smith. Group comparison of resting-state FMRI data using multi-subject ICA and dual regression. *NeuroImage*, 47:S148, July 2009. doi: 10.1016/S1053-8119(09)71511-3.

Yashar Behzadi, Khaled Restom, Joy Liau, and Thomas T. Liu. A component based noise correction method (CompCor) for BOLD and perfusion based fMRI. *NeuroImage*, 37(1):90–101, August 2007. doi: 10.1016/j.neuroimage.2007.04.042.

Richard F. Betzel, Makoto Fukushima, Ye He, Xi-Nian Zuo, and Olaf Sporns. Dynamic fluctuations coincide with periods of high and low modularity in resting-state functional brain networks. *NeuroImage*, 127:287–297, February 2016. doi: 10.1016/j.neuroimage.2015.12.001.

Vincent D Blondel, Jean-Loup Guillaume, Renaud Lambiotte, and Etienne Lefebvre. Fast unfolding of communities in large networks. *Journal of Statistical Mechanics: Theory and Experiment*, 2008(10):P10008, October 2008. doi: 10.1088/1742-5468/2008/10/P10008.

Rotem Botvinik-Nezer, Felix Holzmeister, Colin F Camerer, Anna Dreber, Juergen Huber, et al. Variability in the analysis of a single neuroimaging dataset by many teams. *Nature*, 582(7810):84–88, June 2020.

Joshua Carp. On the Plurality of (Methodological) Worlds: Estimating the Analytic Flexibility of fMRI Experiments. *Frontiers in Neuroscience*, 6, 2012. doi: 10.3389/fnins.2012.00149.

Nathan W. Churchill, Pradeep Raamana, Robyn Spring, and Stephen C. Strother. Optimizing fMRI preprocessing pipelines for block-design tasks as a function of age.

NeuroImage, 154:240–254, July 2017. doi: 10.1016/j.neuroimage.2017.02.028.

Rastko Ciric, Daniel H. Wolf, Jonathan D. Power, et al. Benchmarking of participant-level confound regression strategies for the control of motion artifact in studies of functional connectivity. *NeuroImage*, 154:174–187, July 2017. doi: 10.1016/j.neuroimage.2017.03.020.

Kamalaker Dadi, Mehdi Rahim, Alexandre Abraham, Darya Chyzhyk, Michael Milham, Bertrand Thirion, and Gaël Varoquaux. Benchmarking functional connectome-based predictive models for resting-state fMRI. *NeuroImage*, 192:115–134, May 2019. doi: 10.1016/j.neuroimage.2019.02.062.

Kamalaker Dadi, Gaël Varoquaux, Antonia Machlouzarides-Shalit, Krzysztof J. Gorgolewski, Demian Wassermann, Bertrand Thirion, and Arthur Mensch. Fine-grain atlases of functional modes for fMRI analysis. *NeuroImage*, 221:117126, November 2020. doi: 10.1016/j.neuroimage.2020.117126.

Jessica Dafflon, Pedro F. Da Costa, František Váša, Ricardo Pio Monti, Danilo Bzdok, Peter J. Hellyer, Federico Turkheimer, Jonathan Smallwood, Emily Jones, and Robert Leech. Neuroimaging: into the Multiverse. preprint, Neuroscience, October 2020.

Damien A. Fair, Oscar Miranda-Dominguez, Abraham Z. Snyder, et al. Correction of respiratory artifacts in MRI head motion estimates. *NeuroImage*, 208:116400, March 2020. doi: 10.1016/j.neuroimage.2019.116400.

Bruce Fischl, Martin I. Sereno, and Anders M. Dale. Cortical Surface-Based Analysis II: Inflation, Flattening, and a Surface-Based Coordinate System. *NeuroImage*, 9(2):195–207, February 1999. doi: 10.1006/nimg.1998.0396.

Matthew F. Glasser, Stamatios N. Sotiropoulos, J. Anthony Wilson, et al. The minimal preprocessing pipelines for the Human Connectome Project. *NeuroImage*, 80:105–124, October 2013. doi: 10.1016/j.neuroimage.2013.04.127.

Matthew F. Glasser, Timothy S. Coalson, Emma C. Robinson, et al. A multi-modal parcellation of human cerebral cortex. *Nature*, 536(7615):171–178, August 2016. doi: 10.1038/nature18933.

Evan M. Gordon, Timothy O. Laumann, Babatunde Adeyemo, Jeremy F. Huckins, William M. Kelley, and Steven E. Petersen. Generation and Evaluation of a Cortical Area Parcellation from Resting-State Correlations. *Cerebral Cortex*, 26 (1):288–303, January 2016. doi: 10.1093/cercor/bhu239.

Evan M. Gordon, Timothy O. Laumann, Adrian W. Gilmore, et al. Precision Functional Mapping of Individual Human Brains. *Neuron*, 95(4):791–807.e7, August 2017. doi: 10.1016/j.neuron.2017.07.011.

Kyle Gorman and Steven Bedrick. We need to talk about standard splits. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 2786–2791, Florence, Italy, July 2019. Association for Computational Linguistics. doi: 10.18653/v1/P19-1267.

Michael N. Hallquist, Kai Hwang, and Beatriz Luna. The nuisance of nuisance regression: Spectral misspecification in a common approach to resting-state fMRI preprocessing reintroduces noise and obscures functional connectivity. *NeuroImage*, 82: 208–225, November 2013. doi: 10.1016/j.neuroimage.2013.05.116.

Xin He, Kaiyong Zhao, and Xiaowen Chu. AutoML: A survey of the state-of-the-art. *Knowledge-Based Systems*, 212: 106622, January 2021. doi: 10.1016/j.knosys.2020.106622.

Pavel Izmailov, Dmitrii Podoprikhin, Timur Garipov, Dmitry Vetrov, and Andrew Gordon Wilson. Averaging Weights Leads to Wider Optima and Better Generalization, February 2019. Number: arXiv:1803.05407 arXiv:1803.05407 [cs, stat].

Diederik P. Kingma and Jimmy Ba. Adam: A Method for Stochastic Optimization, January 2017. Number: arXiv:1412.6980 arXiv:1412.6980 [cs].

Timothy O. Laumann, Evan M. Gordon, Babatunde Adeyemo, et al. Functional System and Areal Organization of a Highly Sampled Individual Human Brain. *Neuron*, 87(3):657–670, August 2015. doi: 10.1016/j.neuron.2015.06.037.

Jingwei Li, Ru Kong, Raphaël Liégeois, Csaba Orban, Yanrui Tan, Nanbo Sun, Avram J. Holmes, Mert R. Sabuncu, Tian Ge, and B.T. Thomas Yeo. Global signal regression strengthens association between resting-state functional connectivity and behavior. *NeuroImage*, 196:126–141, August 2019. doi: 10.1016/j.neuroimage.2019.04.016.

Zachary C. Lipton and Jacob Steinhardt. Troubling trends in machine learning scholarship: Some ml papers suffer from flaws that could mislead the public and stymie future research. *Queue*, 17(1): 45–77, feb 2019. doi: 10.1145/3317287.3328534.

Nikos K Logothetis, Jon Pauls, Mark Augath, Torsten Trinath, and Axel Oeltermann. Neurophysiological investigation of the basis of the fMRI signal. *Nature*, 412: 8, 2001.

N. R. Lomb. Least-squares frequency analysis of unequally spaced data. *Astrophysics and Space Science*, 39(2):447–462, February 1976. doi: 10.1007/BF00648343.

L.-E. Martinet, M. A. Kramer, W. Viles, L. N. Perkins, E. Spencer, C. J. Chu, S. S. Cash, and E. D. Kolaczyk. Robust dynamic community detection with applications to human brain functional networks. *Nature Communications*, 11 (1):2785, December 2020. doi: 10.1038/s41467-020-16285-7.

John D. Medaglia, Theodore D. Satterthwaite, Apoorva Kelkar, et al. Brain state expression and transitions are related to complex executive cognition in normative neurodevelopment. *NeuroImage*, 166: 293–306, February 2018. doi: 10.1016/j.neuroimage.2017.10.048.

Kevin Murphy, Rasmus M. Birn, Daniel A. Handwerker, Tyler B. Jones, and Peter A. Bandettini. The impact of global signal regression on resting state correlations: Are anti-correlated networks introduced? *NeuroImage*, 44(3):893–905, February 2009. doi: 10.1016/j.neuroimage.2008.09.036.

M. E. J. Newman and M. Girvan. Finding and evaluating community structure in networks. *Physical Review E*, 69 (2):026113, February 2004. doi: 10.1103/PhysRevE.69.026113. arXiv:cond-mat/0308217.

Linden Parkes, Ben Fulcher, Murat Yücel, and Alex Fornito. An evaluation of the efficacy, reliability, and sensitivity of motion correction strategies for resting-state functional MRI. *NeuroImage*, 171:415–436, May 2018. doi: 10.1016/j.neuroimage.2017.12.073.

Usama Pervaiz, Diego Vidaurre, Mark W. Woolrich, and Stephen M. Smith. Optimising network modelling methods for fMRI. *NeuroImage*, 211:116604, May 2020. doi: 10.1016/j.neuroimage.2020.116604.

Adam R. Pines, Bart Larsen, Zaixu Cui, et al. Dissociable multi-scale patterns of development in personalized brain networks. *Nature Communications*, 13(1): 2647, December 2022. doi: 10.1038/s41467-022-30244-4.

Jonathan D. Power, Kelly A. Barnes, Abraham Z. Snyder, Bradley L. Schlaggar, and Steven E. Petersen. Spurious but systematic correlations in functional connectivity MRI networks arise from subject motion. *NeuroImage*, 59(3):2142–2154, February 2012. doi: 10.1016/j.neuroimage.2011.10.018.

Jonathan D. Power, Anish Mitra, Timothy O. Laumann, Abraham Z. Snyder, Bradley L. Schlaggar, and Steven E. Petersen. Methods to detect, characterize, and remove motion artifact in resting state fMRI. *NeuroImage*, 84:320–341, January 2014. doi: 10.1016/j.neuroimage.2013.08.048.

Jonathan D. Power, Mark Plitt, Timothy O. Laumann, and Alex Martin. Sources and implications of whole-brain fMRI signals in humans. *NeuroImage*, 146:609–625, February 2017. doi: 10.1016/j.neuroimage.2016.09.038.

Raimon H.R. Pruim, Maarten Mennes, Jan K. Buitelaar, and Christian F. Beckmann. Evaluation of ICA-AROMA and alternative strategies for motion artifact removal in resting state fMRI. *NeuroImage*, 112:278–287, May 2015. doi: 10.1016/j.neuroimage.2015.02.063.

Emma C. Robinson, Saad Jbabdi, Matthew F. Glasser, Jesper Andersson, Gregory C. Burgess, Michael P. Harms, Stephen M. Smith, David C. Van Essen, and Mark Jenkinson. MSM: A new flexible framework for Multimodal Surface Matching. *NeuroImage*, 100:414–426, October 2014. doi: 10.1016/j.neuroimage.2014.05.069.

Theodore D. Satterthwaite, Daniel H. Wolf, James Loughead, et al. Impact of in-scanner head motion on multiple measures of functional connectivity: Relevance for studies of neurodevelopment in youth. *NeuroImage*, 60(1):623–632, March 2012. doi: 10.1016/j.neuroimage.2011.12.063.

Theodore D. Satterthwaite, Mark A. Elliott, Raphael T. Gerraty, et al. An improved framework for confound regression and filtering for control of motion artifact in the preprocessing of resting-state functional connectivity data. *NeuroImage*, 64: 240–256, January 2013. doi: 10.1016/j.neuroimage.2012.08.052.

Alexander Schaefer, Ru Kong, Evan M Gordon, Timothy O Laumann, Xi-Nian Zuo, Avram J Holmes, Simon B Eickhoff, and B T Thomas Yeo. Local-Global Parcellation of the Human Cerebral Cortex from Intrinsic Functional Connectivity MRI. *Cerebral Cortex*, 28(9):3095–3114, September 2018. doi: 10.1093/cercor/bhx179.

Stephen M. Smith, Diego Vidaurre, Christian F. Beckmann, et al. Functional connectomics from resting-state fMRI. *Trends in Cognitive Sciences*, 17(12):666–682, December 2013. doi: 10.1016/j.tics.2013.09.016.

Xing Su, Shan Xue, Fanzhen Liu, Jia Wu, Jian Yang, Chuan Zhou, Wenbin Hu, Cecile Paris, Surya Nepal, Di Jin, Quan Z.

Sheng, and Philip S. Yu. A Comprehensive Survey on Community Detection with Deep Learning. *IEEE Transactions on Neural Networks and Learning Systems*, pages 1–21, 2022. doi: 10.1109/TNNLS.2021.3137396. arXiv:2105.12584 [cs].

William Hedley Thompson, Jessey Wright, James M. Shine, and Russell A. Poldrack. The identification of temporal communities through trajectory clustering correlates with single-trial behavioural fluctuations in neuroimaging data. preprint, Neuroscience, April 2019.

Ye Tian, Daniel S. Margulies, Michael Breakspear, and Andrew Zalesky. Topographic organization of the human subcortex unveiled with functional connectivity gradients. *Nature Neuroscience*, 23(11):1421–1432, November 2020. doi: 10.1038/s41593-020-00711-6.

David C. Van Essen, Stephen M. Smith, Deanna M. Barch, Timothy E.J. Behrens, Essa Yacoub, and Kamil Ugurbil. The WU-Minn Human Connectome Project: An overview. *NeuroImage*, 80:62–79, October 2013. doi: 10.1016/j.neuroimage.2013.05.041.

D.C. Van Essen, K. Ugurbil, E. Auerbach, et al. The Human Connectome Project: A data acquisition perspective. *NeuroImage*, 62(4):2222–2231, October 2012. doi: 10.1016/j.neuroimage.2012.02.018.

Albert Vilamala, Kristoffer Hougaard Madsen, and Lars Kai Hansen. Towards end-to-end optimisation of functional image analysis pipelines, October 2016. Number: arXiv:1610.04079 arXiv:1610.04079 [cs, q-bio, stat].

B. T. Thomas Yeo, Fenna M. Krienen, Jorge Sepulcre, et al. The organization of the human cerebral cortex estimated by intrinsic functional connectivity. *Journal of Neurophysiology*, 106(3):1125–1165, September 2011. doi: 10.1152/jn.00338.2011.

Appendix A. Parcellation

Dataset, preprocessing, and computational resources The openly available Human Connectome Project (HCP) dataset comprises scans from 1200 participants performing a number of in-scanner tasks. For each participant, the study acquisition protocols included 4 separate resting-state (task-free) scans, each acquired over the course of approximately 15 minutes of scanning using an accelerated multiband sequence with a sampling rate of 0.72 seconds (Van Essen et al., 2012, 2013). In practice, the protocols were incomplete for many participants, resulting in fewer than 4 scans for a subset of participants.

To obtain our training and evaluation datasets, we first performed a random 12-way split on 1098 HCP subjects with resting-state fMRI data. Note that our dataset split was performed across subjects rather than images, such that data from subjects in test and validation splits would not be seen during training. We then selected the first 7 of these splits, comprising 2457 total resting-state BOLD time series from 618 subjects, to train our model. To create an evaluation set, we selected another split, from which a single data shard of 69 time series was randomly selected for homogeneity evaluation.

Data were preprocessed according to the minimal preprocessing pipeline of the Human Connectome Project (Glasser et al., 2013), and were acquired following a standardised protocol in accordance with the host institution's IRB (Van Essen et al., 2012). To better respect the topology of the cortex and to reduce the effects of inter-subject variability, we used time series projected onto a spherical surface (Fischl et al.,

1999) and aligned using the MSMAll algorithm (Robinson et al., 2014). Before each time series was passed to the model, we performed several additional preprocessing steps. First, we projected each time series to the orthogonal complement of a subspace defined as the span of (i) a quadratic polynomial (to mitigate scanner drift artefact) and (ii) the average global signal across all voxels (following previous parcellation work, and in accordance with the overall demonstrated benefits of this approach; Schaefer et al. (2018); Li et al. (2019)). Next, we normalised each time series to a mean of 0 and a variance of 1. Finally, to reduce the memory footprint, we selected a random 500-sample window from each time series as input to the model.

The model was trained on a computer running Ubuntu Linux, using only a single commercial-grade (RTX 2080 Ti) GPU with 11 GB of RAM. Memory usage was reduced through block-wise serialisation of expensive loss computations (details in *Loss* section below). Each of the 5 parcellations required approximately 18-24 hours of GPU time to train. Data were stored as `tar` shards for compatibility with the `webdataset` format (Aizman et al., 2019). Due to the large size of each BOLD time series, the most significant bottleneck during training was reading from the disk. Unfortunately, we found that data loaders were not well equipped to expedite this process for the large samples used, although it is likely that this was due to insufficient optimisation in sharding and worker deployment. We used a fixed batch size of 3 time series for training. To reduce disk I/O operations, we trained the model for 5 steps with each batch before sampling the next batch. The model was trained for a total 6000 steps using the Adam optimiser (Kingma and Ba, 2017). On account of the substantial overhead incurred when reading a large BOLD time series from disk, we do not use an epoch-based training regime. This does mean that only a small subset of all images are seen at every stage of training, but our quantitative and qualitative assessments suggest that this approach is nevertheless sufficient to produce a good, representative group-level parcellation.

Loss function As detailed in the main text, the loss function we use comprises four clustering terms (spatial compactness of parcels, spatial dispersion among parcels, second moment of parcel time series, and negative log determinant of the correlation among parcel time series), as well as three regularisation terms (parcel equilibrium, vertex-wise distribution entropy, and inter-hemispheric spatial tether).

A detail of loss function definitions and considerations follows. Across all definitions, we use the following notation for dimensions: v denotes the number of vertices in the input BOLD time series, p denotes the total number of parcels, and t denotes the number of time points in the BOLD time series.

The spatial compactness loss for a single parcel is defined as

$$\mathcal{L} = \mathbf{a}^\intercal \mathrm{d}\left(\mathbf{C}, \quad \frac{\mathbf{a}^\intercal \mathbf{C}}{\mathbf{a}^\intercal \mathbf{1}}\right)$$

Let $\mathbf{a} \in \mathbb{R}^v$ be the assignment of all vertices to a single parcel, let $\mathbf{C} \in \mathbb{R}^{v\times 3}$ be a matrix containing the three spatial coordinates of each vertex, and let $\mathbf{1}$ denote a vector of all ones. The term $\mu_{\mathbf{C}} = \frac{\mathbf{a}^\intercal \mathbf{C}}{\mathbf{a}^\intercal \mathbf{1}} \in \mathbb{R}^3$ is then the weighted mean of vertex coordinates assigned to the parcel – i.e., it is a matrix containing the centre of mass of the parcel. We then compute the vector of distances between the parcel's centre of mass coordinates and the coordinates of all vertices. With some abuse of notation, we denote this vector above as $\mathbf{d} \in \mathbb{R}^v = \|\mathbf{C} - \mu_{\mathbf{C}}\|_{\text{rows}}$, where we implicitly broadcast the centre of mass and where we use the "rows" subscript to denote

that the distance is computed over each row separately. In our setting, we approximate the cortical surface as a sphere and operationalise the distance as a spherical geodesic. We finally define the loss by weighting each distance according to the corresponding vertex's assignment to the parcel to obtain the compactness loss $\mathbf{a}^\intercal\mathbf{d}$.

It is clear that the compactness loss of a parcel decreases as the weights in its assignment vector $\mathbf{a}$ are concentrated on vertices close to its centre of mass $\mu_{\mathbf{C}}$. The compactness score can also be loosely interpreted as a weighted error loss where the "predictions" are the vertex coordinates and the "target" is the centre of mass. We compute the compactness score independently for each parcel and scalarise using the mean.

We define the vector dispersion loss between a pair of parcels as

$$\mathcal{L} = -\mathrm{d}\left(\frac{\mathbf{a_i}^\intercal\mathbf{C}}{\mathbf{a_i}^\intercal\mathbf{1}}, \frac{\mathbf{a_j}^\intercal\mathbf{C}}{\mathbf{a_j}^\intercal\mathbf{1}}\right)$$

where we have used $\mathbf{a}_i, \mathbf{a}_j \in \mathbb{R}^v$ to denote the vertex assignment vectors of the two parcels. Analogously to the compactness loss, let $\mathbf{C} \in \mathbb{R}^{v\times 3}$ be a matrix containing the three spatial coordinates of each vertex, and let $\mathbf{1}$ denote a vector of all ones. The terms $\frac{\mathbf{a_i}^\intercal\mathbf{C}}{\mathbf{a_i}^\intercal\mathbf{1}}$ and $\frac{\mathbf{a_j}^\intercal\mathbf{C}}{\mathbf{a_j}^\intercal\mathbf{1}} \in \mathbb{R}^3$ then correspond to the centres of mass of parcels i and j, respectively, and the dispersion is simply a negation of some notion of distance between those centres of mass. (It is also possible to use a similarity measure rather than a negated distance.) In our work, we again use a spherical geodesic distance. The total dispersion is the sum of all unique pairwise distances between parcels, i.e. $-\sum_{j>i}\mathrm{d}\left(\frac{\mathbf{a_i}^\intercal\mathbf{C}}{\mathbf{a_i}^\intercal\mathbf{1}}, \frac{\mathbf{a_j}^\intercal\mathbf{C}}{\mathbf{a_j}^\intercal\mathbf{1}}\right)$. As the mutual separation among parcels increases, the dispersion loss decreases; this loss can therefore be used to promote spatially separated parcels. The dispersion is constrained from decreasing without bound because all coordinates and their centroids are bounded to a spherical manifold that approximates the cortical surface.

The temporal second moment loss for a single parcel is defined as

$$\mathcal{L} = \left(\frac{\mathbf{a}}{\mathbf{a}^\intercal\mathbf{1}}\right)^\intercal (\mathbf{T_v} - \mathbf{t_p})^2 \frac{\mathbf{1}}{t}$$

Let $\mathbf{a} \in \mathbb{R}^v$ be the assignment of all vertices to a single parcel, let $\mathbf{T_v} \in \mathbb{R}^{v\times t}$ be a matrix containing the vertex-wise BOLD time series, and let $\mathbf{1}$ denote a vector of all ones. We have also defined $\mathbf{t_p} \in \mathbb{R}^t$ as the time series of parcel $\mathbf{p}$. In our work, we operationalise each parcel's time series as the weighted mean of the time series assigned to that parcel, i.e. $\mathbf{t_p} = \frac{\mathbf{a}^\intercal\mathbf{T_v}}{\mathbf{a}^\intercal\mathbf{1}}$. (Note that, when we use this definition, the parallels between the *spatial* compactness loss and the *temporal* second moment loss become evident.) With implicit broadcasting, the term $\mathbf{D} \in \mathbb{R}^{v\times t} = (\mathbf{T_v} - \mathbf{t_p})^2$ represents the square deviation of all voxel time series from the parcel time series. The rationale for the "second moment" moniker is now evident: the weighting term $\frac{\mathbf{a}}{\mathbf{a}^\intercal\mathbf{1}}$ can be framed as the probability mass assigned to each vertex, and the deviation term $(\mathbf{T_v} - \mathbf{t_p})^2$ is the squared deviation of each observed time point from the mean. The last part of the second moment term is the scalarisation; here we use the mean $\frac{\mathbf{1}}{t}$ but the sum or another scalarisation would also be valid. (Note the analogy to the compactness if the square root of the sum is used.)

The second moment loss is minimised by concentrating a parcel's weight $\mathbf{a}$ on vertices whose deviation from the parcel mean is small. This loss term will thereby promote learning of parcels that have internally homogeneous activation patterns across all subjects. As with the compactness, we can loosely interpret the second moment term as

a weighted error loss where the "predictions" are the vertex-wise time series and the "target" is the parcel mean time series. Through experimentation, we also made one further change to the second moment loss term: we removed the normalisation $\frac{\mathbf{1}}{\mathbf{a}^\intercal\mathbf{1}}$, the use of which we found could yield a small number of very large parcels that did not correspond with any well-known functional systems of the brain. We compute the second moment loss independently for each parcel and scalarise using the mean.

Given a set of parcel time series $\mathbf{T_p} \in \mathbb{R}^{p\times t}$, the negative log determinant loss is defined as

$$\mathcal{L} = -\log\det\mathbf{K}(\mathbf{T_p})$$

where $\mathbf{K}$ is an appropriately selected kernel function and $\mathbf{K}(\mathbf{T_p}) \in \mathbb{R}^{p\times p}$ is the Gram matrix of time series vectors induced by that kernel function. Note that the parcel time series are again the weighted means of input BOLD time series: $\mathbf{T_p} = \frac{\mathbf{AT_v}}{\mathbf{A1}}$ for the parcellation matrix $\mathbf{A} \in \mathbb{R}^{p\times v}$ whose rows are the parcel vectors $\mathbf{a} \in \mathbb{R}^v$ that we have been dealing with until now. Here, for reasons that we will shortly elaborate, we let the Pearson correlation among time series operationalise the kernel. Apart from its positive definiteness (and the attendant convexity of its log determinant), the Pearson correlation matrix has the favourable property of a determinant that ranges from a minimum of 0 for a nonsingular matrix up to a maximum of 1 for fully orthogonal time series (corresponding to an identity matrix). The negative log determinant, therefore, goes to infinity for a singular matrix and to zero for an identity matrix. Placing a strong multiplier on this loss term can be used to obtain orthogonal parcel time series. As this is likely undesirable in the brain mapping context, we instead use a weaker multiplier to promote independence of parcel time series.

As stated, the negative log determinant term has the potential to go to infinity if parcel-wise time series are not independent. While the purpose of this term is to promote parcel independence, an exploding loss can irreversibly cause parameter values to go out of bounds when it is propagated back. To minimise the risk of this occurrence while preserving the efficacy of the determinant term, we added a small reconditioning term to the correlation matrix. For each entry along the diagonal of the correlation matrix, we randomly sampled i.i.d. noise from $\mathrm{Uniform}(0, 0.001)$ and added this noise to the diagonal. Although this reconditioning added further stochasticity to the determinant term, we observed an overall downward trend of the determinant loss during training as seen in Figure 3.

The entropy of our parcellation matrix $\mathbf{A} \in \mathbb{R}^{p\times v}$ is defined as

$$\mathcal{L} = \mathbf{1}^\intercal\mathbf{A} \circ \log\mathbf{A1}$$

where $\mathbf{1}$ is again the vector of all ones, log denotes the elementwise (not matrix) logarithm, and $\circ$ denotes the Hadamard (elementwise) product. In general, the entropy is a concave function that can decrease without bound. However, we use a softmax to project each column of $\mathbf{A}$ onto the probability simplex such that $\mathbf{1}^\intercal\mathbf{A} = \mathbf{1}$. Over this restricted domain, the entropy is a bounded function that attains its minimum when all probability mass is concentrated in a single outcome – i.e., when each vertex is deterministically assigned to a single parcel.

The equilibrium loss for a single parcel is here defined simply as the squared sum (or mean) of that parcel's assignment vector $\mathbf{a} \in \mathbb{R}^v$

$$\mathcal{L} = (\mathbf{a}^\intercal\mathbf{1})^2$$

The total equilibrium loss is then defined as the mean equilibrium across parcels, $\left(\frac{\mathbf{1}}{p}\right)^{\intercal}(\mathbf{A1} \circ \mathbf{A1})$ for $\mathbf{A} \in \mathbb{R}^{p \times v}$. This quantity attains its minimum when the total assignment weight of all parcels is equal, i.e., when all parcels are the same size. (Note that newer versions of our software library by default use a different formulation of the equilibrium that has a more intuitive relationship to the notion of equilibrium; please consult our online documentation hub for details.)

Finally, the inter-hemispheric tether loss has a similar definition to the dispersion loss, but is defined pairwise and acts in the opposite direction. For a pair of parcels, one in the left cortical hemisphere and another in the right cortical hemisphere, the inter-hemispheric tether is

$$\mathcal{L} = \mathrm{d}\left(\frac{\mathbf{a}_{i,\mathrm{left}}^{\intercal}\mathbf{C}_{\mathrm{left}}}{\mathbf{a}_{i,\mathrm{left}}^{\intercal}\mathbf{1}}, f\left(\frac{\mathbf{a}_{i,\mathrm{right}}^{\intercal}\mathbf{C}_{\mathrm{right}}}{\mathbf{a}_{i,\mathrm{right}}^{\intercal}\mathbf{1}}\right)\right)$$

We let $\mathbf{a}_{i,\mathrm{left}}, \mathbf{a}_{i,\mathrm{right}} \in \mathbb{R}^{v}$ be the vertex assignment vectors of the ith parcel in the left and right hemispheres respectively, between which we wish to promote analogy or approximate symmetry. Correspondingly, let $\mathbf{C}_{\mathrm{left}}, \mathbf{C}_{\mathrm{right}} \in \mathbb{R}^{v \times 3}$ be the spatial coordinates of all vertices in the left and right hemispheres and again let $\mathbf{1}$ denote the vector of all ones. Thus, expressions of the form $\frac{\mathbf{a}_i^{\intercal}\mathbf{C}}{\mathbf{a}_i^{\intercal}\mathbf{1}}$ denote the centre of mass of the ith parcel. Next, define a transformation f such that applying f to a coordinate in the right hemisphere returns an analogous coordinate in the left hemisphere. Here, we operationalise f as a simple negation of the x-coordinate, i.e., $f(\mathbf{x}) = \begin{pmatrix} -1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix}\mathbf{x}$. (Note that the formulation is still valid if the left and right hemispheres have different numbers of vertices, since only the centres of mass are required for the computation.)

Null model To construct a null model for brain parcellation, we used a loss function ablation. Specifically, we removed the two *temporal* loss terms: the second moment and the negative log determinant. In addition to the *spatial* clustering terms based on compactness and dispersion, entropy, equilibrium, and interhemispheric tethering terms were retained. Also retained was the softmax projection of parcel assignments onto the probability simplex. This null model scheme resulted in a model that learned from only spatial information without reference to any patterns of synchrony or homogeneity in the brain image dataset. The null model was trained using a combination of these loss functions for 1000 epochs without any input data. We created an ensemble of null models that matched the resolutions of all pre-existing and learned parcellations: 200, 300, 333, 400, 500, 600, 666, 700, 800, and 1000 parcels. The homogeneity of null model parcels was then used to calibrate our quantitative evaluation of parcel homogeneity; see the *Evaluation* section below for details.

SWAPR algorithm SWAPR is a simple modification of Stochastic Weight Averaging (SWA; Izmailov et al. (2019)) that adds revolution of parameters between the averaged model and the data-facing model. The rationale for this modification follows: when entropy is sufficiently penalised, as it is in the final stages of the entropy cascade that we use, a data-facing model will eventually converge to a deterministic solution that receives weak gradients and negligible parameter updates. Given sufficient training steps, any model that averages over this data-facing model will converge to the same deterministic solution, thereby obviating it altogether. To allow for some useful weight averaging in this high-negentropy setting (while also allowing for an averaged model that is eventually approximately deterministic), we revolve

parameters from the averaged model into the data-facing model at the end of each cascade stage. Practically, the average parameters from the previous stage of the cascade become the new data-facing parameters at the current stage, and a new averaged model is initialised (although it is also possible to decrease the weight of the running average instead of initialising a completely new averaged model). We use a large, constant learning rate of 0.05 during SWA steps.

Multiplier schedules and hyperparameters To train the parcellation model, we scheduled changes in the relative balance of loss multiplier hyperparameters over the course of training. We highlight the most critical aspect of this schedule, the entropy cascade, in the main text: we periodically increase the penalty on each vertex's parcel assignment distribution entropy to transition the parcellation solution from distributed and overlapping functional modes to compact and circumscribed deterministic parcels. Here, we discuss several additional scheduling decisions; the exact values of all multiplier hyperparameters and schedules are included in the associated code.

First, the loss terms that favour spatial and temporal separation of parcels—the dispersion and determinant terms—are initialised with large multipliers of 10 and .005 respectively in order to achieve rapid differentiation of parcels at the beginning of training. We anneal these multipliers to 0.5 and 0.0001 respectively before the 400th training step. Second, approximately following a practice established in previous parcellation efforts (e.g., Schaefer et al. (2018)), we progressively increase the compactness multiplier to collapse the spatial extent of parcels; we find that this increase works synergistically with the entropy cascade. Third, to reduce the chances of immediate fixation and convergence at the start of each entropy cascade step, we temporarily pump the multipliers for two terms that tend to compete with the entropy, the parcel equilibrium and second moment. These terms decay exponentially back to a baseline over the course of the cascade stage. Future work will more comprehensively evaluate the roles of these schedules in order to streamline the parcellation training regime.

Evaluation The quality of the learned parcellation was evaluated using a measure of parcel homogeneity, operationalised as follows. For fairer comparison with reference parcellations, which are defined deterministically, we first took the arg max over vertex probability assignments to create a deterministic parcellation at each of the 5 evaluated parcel scales. (In practice, the final stages of the entropy cascade already yielded maximum assignment probabilities close to unity for nearly all vertices.) For each parcel at each scale, we then computed the pairwise Pearson correlation among all of that parcel's assigned vertex-wise time series for each BOLD image in the held-out test set. For each BOLD image, the parcel homogeneity was operationalised as the mean of these correlations:

$$h = \frac{1}{\left(\sum_v \mathbf{1}_{[v\in\mathcal{P}]}(v)\right)^2} \sum_{v_i,v_j\in\mathcal{P}} \frac{\sigma_{ij}}{\sqrt{\sigma_{ii}\sigma_{jj}}}$$

where $\mathbf{1}_{[v\in\mathcal{P}]}(v)$ is the indicator function denoting membership of vertex v in parcel $\mathcal{P}$ and σ_{ij} denotes the covariance between the time series of vertices i and j.

We then defined the overall homogeneity of that parcel as its mean homogeneity across all images in the test set. We note that the homogeneity scores we obtained here were low in relation to many previous reports; this might be attributable in part to the lack of spatial smoothing in our data. Additionally, of the reference parcellations, only the

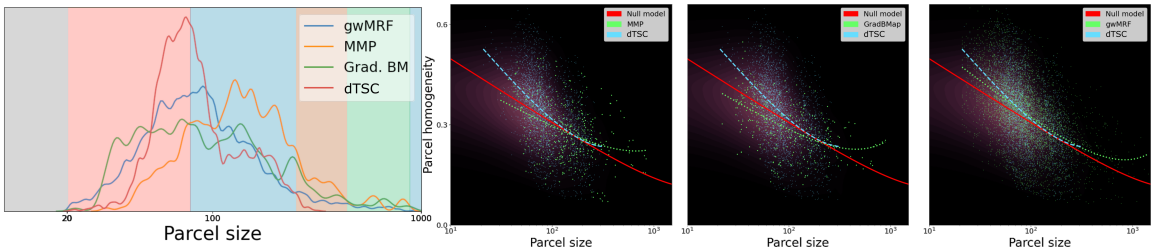


Figure 6: Parcellation evaluation. *Left*, distributions of parcel sizes for different evaluated parcellations. Background colour indicates maximum scoring parcellation for that size according to log fit. *Right*, Absolute homogeneity evaluation without adjustment for null.

MMP parcellation (Glasser et al., 2016) was defined using the HCP dataset; because of differences between datasets, acquisition protocols, and coordinate spaces in which the different parcellations were defined, comparison results should be interpreted with caution. As an additional note of caution, it is possible that the "unseen" data used for evaluation were not held out for the MMP parcellation. Furthermore, it should be noted that the reference gwMRF parcellation that we evaluated, like the dTSC parcellation, is in fact a set of parcellations defined across different spatial scales (all multiples of 100 parcels between 100 and 1000, inclusive).

Parcel homogeneity is related to parcel size—it is more likely for a smaller parcel to be more homogeneous. Accordingly, after computing homogeneity scores for each parcel, we also computed the best logarithmic least-squares fit of parcel size to parcel homogeneity. This fit was computed for all evaluated parcellations, including the null model. To generate the *Relative Homogeneity* plots in the main text, we subtracted the null model's best fit from all other fits and homogeneity scores. As a reference, we also plot the analogous figures without this adjustment here (Figure 6, *Right*). Parcels with a size of under 20 vertices were excluded from evaluation. We also include KDE plots to facilitate visualisation of size distributions for different parcellations (Figure 6, *Left*). The parcel size distribution for our method reveals a tighter distribution than for other methods, likely due to a fairly stringent parcel equilibrium regularisation. In future work, we will explore the impact of relaxing this regularisation, particularly in consideration of the desirability for more variable parcel sizes across the brain in certain applications. More flexible alternatives to a parcel equilibrium penalty, such as a unilateral L2 loss imposed on parcels whose total weight is less than some minimum, could also be explored.

Appendix B. Denoising

Dataset characteristics We again use the minimally preprocessed Human Connectome Project dataset detailed in the parcellation section above; however, we make a few adjustments. First, we use BOLD time series whose dimension has already been reduced via a mapping to parcels. We use the popular 400-region gwMRF parcellation introduced by Schaefer et al. (2018), which is based on a Markov random field approach and recapitulates previously characterised boundaries of functional subsystems. Second, we use a different data split, training on 4 of the 12 random splits, selecting another 4 for validation, and setting the last 4 aside for evaluation. Following this split, we train on a subset of 356 subjects and 1393 images. Due to

computational limitations (because all data must fit into memory simultaneously for evaluation), we use a 351-image subset of the test split for the main evaluation results. In the supplement, we report benchmark results separately using each of the 12 data splits (Figure 7).

QC measures Assessing denoising performance in fMRI data is challenging because researchers do not have access to a noise-free ground truth. Instead, the typical approach involves first quantifying the presence of artefact in some way and then measuring the extent to which that quantification of artefact relates to measurements of interest obtained from the data. Here, following the corpus of previous work (Power et al., 2014), we derive our quantification of motion-related artefact from head realignment estimates. To further elaborate, subjects move their heads while in the scanner, and this motion introduces spurious, largely non-neural variance into the functional MR image. When subjects move, the relative positions of their heads accordingly change in successive images acquired by the scanner. As part of the fMRI preprocessing workflow, their heads are realigned to a common reference frame using a rigid-body transformation matrix admitting a combination of translations and rotations. It is then possible to use these rigid-body transformation matrices to quantify how much subjects move their heads along the three axes of translation and three axes of rotation. This provides gross, incomplete estimates of motion, but these estimates have nevertheless proven highly effective in demonstrating the influence of motion artefact in previous work (Power et al., 2012; Satterthwaite et al., 2012; Ciric et al., 2017; Parkes et al., 2018). We accordingly use a derivative of these estimates here.

In particular, it is standard practice to obtain a single scalar summary measure of the amount a subject moves in a single time frame by summing the absolute values of these motion estimates. This summary measure is called framewise displacement (FD; Power et al. (2012)). Following reports that head motion estimates can be contaminated by respiratory artefact (Fair et al., 2020), we applied a notch filter to the 6 gross estimates of head motion included with the HCP dataset. We then used these filtered estimates to compute the FD that we used as our main QC metric: the sum of the absolute values of the 6 filtered motion estimates (Power et al., 2012).

Confound time series The standard approach for removal of motion artefact in fMRI is called "confound regression" (Satterthwaite et al., 2013). Confound regression is a two-step process: in the first stage, the researcher curates a *confound model*, compiling a collection of time series that they hypothesise index artefactual processes (but not signal of interest). The second step is the *residualisation* step, in which the observed BOLD time series are residualised with respect to the selected confound model, usually by means of a least-squares fit. Selecting a good confound model has proven to be a source of considerable ambiguity for researchers (Ciric et al., 2017; Parkes et al., 2018), and it is this problem that our approach aims to address. In particular, we will use a simple neural network to learn a maximally parsimonious confound model (ultimately comprising only a single nuisance regressor) that is also highly effective in artefact removal.

We now take a brief detour to highlight several of the candidate time series (or *nuisance regressors*) that researchers frequently use when building a confound model for denoising fMRI data. We provide each of these time series as an input to the neural network that we will shortly introduce. First

are the gross head motion estimates themselves, which include reconstructions of both head *position* and head *motion*, the latter obtained as the backward difference of head position estimates. Next are representative signals associated with white matter (WM) and cerebrospinal fluid (CSF) regions of the brain. We obtained these in two ways: first, we computed the mean time series across voxels in binary masks indicating membership in WM and cerebrospinal CSF compartments; these masks were included with the HCP dataset. We also computed singular value decompositions of WM and CSF voxels, yielding orthogonal sets of principal component time series (aCompCor; Behzadi et al. (2007)); we included the first 10 from each compartment in the input to our model. We also computed a commonly used confound measure called the DVARS, defined as the standard deviation across all voxels in temporal difference images (Afyouni and Nichols, 2018). Finally, we included the mean time series across all brain voxels – the global signal (GS). The global signal is the most effective and most controversial of candidate confounds. Although it is singular in its ability to remove artefact related to both motion and respiration (Power et al., 2017) and its use has additional benefits (Li et al., 2019), use of the global signal in confound models remains controversial (i) on account of its clear inclusion of signal from grey matter voxels and (ii) because a mathematically necessary consequence of its inclusion is that the distribution of correlations among residualised time series becomes zero-centred.

We also computed an expansion of the 6 gross head motion estimates (translation and rotation along the x-, y-, and z- axes) and 3 compartment mean signals (GS, WM, and CSF); this expansion consisted of temporal derivatives obtained via backward differences, quadratic terms, and squares of derivatives. This expansion resulted in 36 terms, which together comprised the 36P model we used that has previously been demonstrated to give excellent performance among *a priori* confound models (Satterthwaite et al., 2013; Ciric et al., 2017; Parkes et al., 2018). Together with the 20 CompCor time series and the single DVARS time series, we presented a total 57 candidate time series to our neural network model.

Preprocessing and computational resources Because of incomplete acquisitions, data might be missing for some images. To address this, we created a missing data mask for each batch of time series. We then synthesised data to impute the missing time frames while minimally impacting actual time frames in downstream analyses. We used a hybrid approach for time series imputation: any missing epochs comprising 3 or fewer frames were imputed using a weighted average of the closest frames, while longer missing epochs were imputed using a periodographic approach. To elaborate, we created a basis of sine and cosine functions, selected the frames of each basis function that corresponded to seen epochs in the data, and fit each basis function to the data. We then used the estimated coefficients from all basis function fits to reconstruct unseen data as a linear combination of basis functions. This approach is inspired by a previous method introduced by Power et al. (2014), which in turn draws inspiration from the Lomb-Scargle periodogram (Lomb, 1976). (We did not apply censoring to denoise the data for this experiment because, for this proof of concept, we were interested in optimising denoising model performance without censoring.) After this imputation step, time series were filtered to retain frequencies between 0.01 and 0.1 Hz using a brick-wall ideal filter applied multiplicatively in the frequency domain. The selected low-frequency band is relatively artefact-free

(Satterthwaite et al., 2013) and corresponds reasonably with the frequency of the haemodynamic BOLD response. Filter weights were frozen and not configured to be learnable. To prevent reintroduction of previously orthogonalised signals during the denoising step (Hallquist et al., 2013), identical imputation and filter transformations were applied to BOLD and confound time series. For the denoising experiment, we used approximately 4 hours of compute time, again on a single RTX 2080 Ti GPU on a machine running Ubuntu Linux to train the RFNN model.

In order to obtain relatively stable QC-FC correlations, the model was trained with a large batch size of 100 time series. After parcellation, data were further preprocessed as follows. First, BOLD data and *a priori* confounds were normalised such that each time series had a mean of 0 and a variance of 1. Next, at training and validation time (but not evaluation time) data were randomly windowed to select 500 contiguous time frames.

RFNN architecture, denoising, and loss The model we used for denoising is a minimal, shallow neural network with a single hidden convolutional layer. The hidden layer takes one channel of dimension $c \times t$—the $c = 57$ *a priori* confound time series—as its input. It has a total $f = 5$ learnable filters, each of dimension 1×9. With appropriate padding and a stride of 1, the output of the convolutional layer is thus a set of f $c \times t$ time series, which are the input time series convolved with each of the f "response functions". These are passed through a thresholding leaky ReLU nonlinearity (subtracting a bias term, applying the nonlinearity, and adding back the bias term) before they are concatenated together with the input confound time series. The result is a $(f+1)c \times t$ model matrix. The output layer of the RFNN linearly combines the $(f+1)c$ confound time series into a single model confound.

To obtain a denoised connectome, we compute the covariance of the parcellated BOLD time series conditioned on this confound as defined in the main text. Weights of any imputed observations were set to 0, so denoised connectomes were based only on actual, seen data. We then normalised the conditional covariance to a Pearson correlation before computing the loss function. This symmetric correlation matrix is our estimate of the functional connectome. As described in the main text, the loss function (QC-FC) is a "second-order" correlation computed across the batch dimension between the correlation that represents each connectome edge and a measure of artefact, in this case the mean filtered framewise displacement across all time frames. Because an edge with an inverse relationship to motion is just as undesirable as one with a direct relationship, the loss is operationalised as the mean of the absolute value of QC-FC correlations. Formally, let n be the batch size and let p be the number of regions. The QC-FC loss for a single edge of the connectome is then defined as:

$$\mathcal{L} = \frac{|\sigma_{\mathrm{FD,FC}}|}{\sqrt{\sigma_{\mathrm{FD,FD}}}\sqrt{\sigma_{\mathrm{FC,FC}}}}$$

where $\sigma_{\mathrm{i,j}}$ is the covariance between vectors i, j $\in \mathbb{R}^n$. FD $\in \mathbb{R}^n$ is the vector of QC measures – here, the framewise displacements (averaged across time) measured for each image in the batch. FC $\in \mathbb{R}^n$ is the vector of functional connectivity estimates for that edge, estimated for each image in the batch. The total QC-FC loss is obtained as the average or sum of edge-wise QC-FC losses. The QC-FC loss goes to zero when there is no linear relationship between framewise displacement and functional connectivity in the batch. We train the RFNN model

Model	Dist. Dep.
Null	-0.083395
GSR	-0.121429
36P	-0.138829
RFNN	-0.117902

Table 2: Distance dependence of QC-FC correlations (lower absolute value is better).

for 200 epochs using stochastic gradient descent (SGD), sampling 10 batches randomly per epoch. We also create a null model by randomly initialising a RFNN without training it. Hyperparameters are summarised in Appendix F.

Evaluation We evaluate model performance using standard benchmarks derived from edge-wise QC-FC correlations (Power et al., 2014). Benchmarks are computed on a held-out dataset of 351 BOLD time series. Benchmarks include the number of edges for which a significant relationship with motion is detected across subjects and the median of the absolute value of all QC-FC correlations.

Although it is not a focus of the current proof of concept, QC-FC correlations also tend to be distance-dependent: motion artefactually inflates estimates of short-range connections more than it does estimates of long-range connections. Furthermore, methods that remove the global signal are especially effective at removing widespread artefact that acts over longer ranges, so this distance-dependence is apparently exacerbated after GSR (e.g., Satterthwaite et al. (2013); Power et al. (2014)). We report the "third-order" correlation between QC-FC correlations and inter-node separation in Supplemental Table 2.

There is arguably some circularity in using QC-FC correlations as the outcome measure after directly training our model to minimise the mean of absolute QC-FC correlations. We leave to future work any investigation of the limitations of QC-FC measures and a more complete characterisation of the reasons that the RFNN was able to ostensibly outperform or compete with SOTA methods on these benchmarks without concomitant zero-centring of connectome edges. In the interim, we reiterate our tentative recommendation from the main text against use of this new method until a more thorough investigation is conducted.

Appendix C. Covariance clustering

The parameterised form of the covariance also finds an application in data augmentation: setting the weights θ_{ii} along the diagonal to random nonnegative integers whose sum is the total number of observations is equivalent to a resample. Relaxing this constraint to require only a nonnegative support and a mean of 1, satisfied *inter alia* by noise sampled from an appropriately chosen gamma or truncated normal distribution, leads to a simple method for augmenting covariance datasets that complements random windowing of the input time series (Figure 8, *Left*).

We performed additional experiments using the parameterised covariance in the clustering setting. Specifically, we asked, how can we cluster observations such that the covariance matrices estimated from different clusters are, for some measure of separation, maximally different? This nonconvex maximisation problem has two immediate applications in functional connectomics: *subnetwork detection* and *state detection*. To further elaborate, given the time series matrix $\mathbf{T} \in \mathbb{R}^{p \times t}$, we can obtain two covariance matrices, the $p \times p$ connectome matrix of interparcel correlations and the $t \times t$ matrix of correlations among brain activity profiles across time (e.g., Medaglia et al. (2018)). For the $p \times p$ connectome, observations correspond

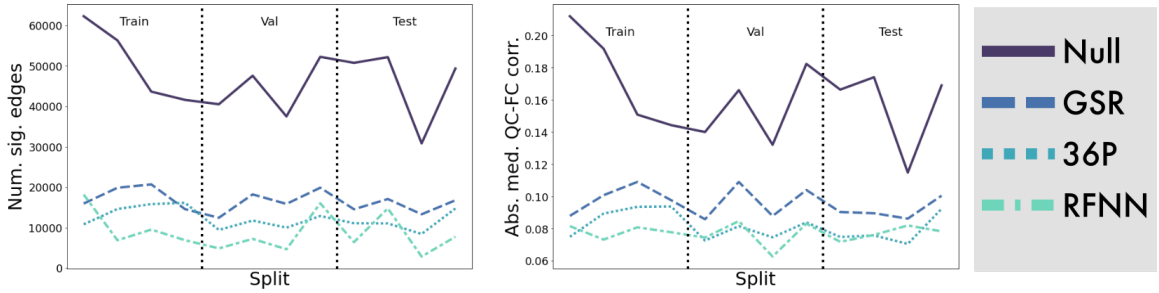


Figure 7: Denoising evaluation across all 12 data splits, each comprising around 350 images. The results shown in the main text are from the first test split. If the QC-FC benchmarks are taken at face value, the single-confound RFNN model outperforms other single-confound models with substantial consistency and performs competitively with the SOTA 36-confound model.

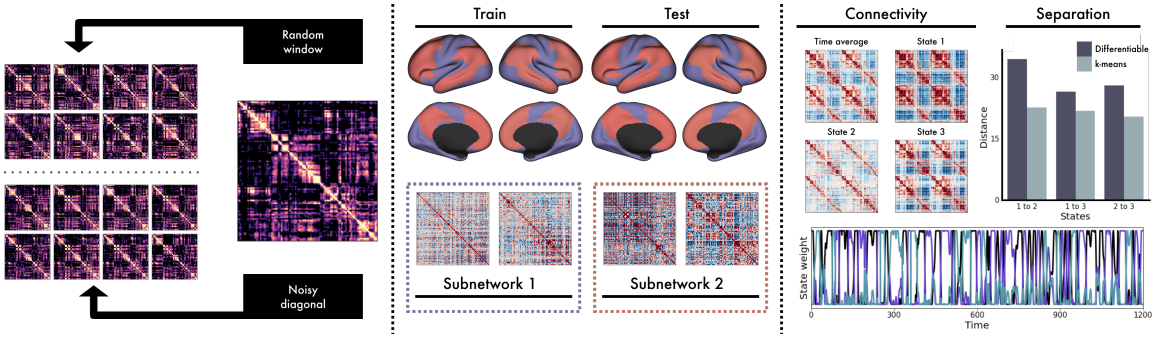


Figure 8: Results of covariance experiments. *Left*, Methods for augmenting covariance data. *Centre*, Clustering on the time-by-time covariance matrix partitions the brain into unimodal and higher-order subnetworks with distinct temporal reconfiguration profiles. *Right*, Clustering on the parcel-by parcel covariance matrix partitions each time series into connectivity states.

to time points, and clustering them to produce maximally distinct connectomes can be interpreted as brain state detection. For the $t \times t$ matrix, observations correspond to different brain parcels; clustering them is a form of subnetwork detection.

For each problem setting, we train a clustering model to learn $\Theta \in \mathbb{R}^{c \times p}$ or $\mathbb{R}^{c \times t}$, where c is the number of clusters, here selected *a priori*. We instantiate a clustering loss to maximise the total L2 dispersion among the covariance matrices corresponding to different detected clusters. As in the parcellation problem, we penalise the entropy to promote deterministic assignment of each parcel or time point to a single subnetwork or state. For the state detection problem, we also encourage the model to learn persistent states by imposing an L2 smoothness penalty on the backward difference of $\mathbf{\Theta}$. Because state detection (and numerous other neuroimaging applications, such as subject-specific parcellation) requires learning a unique time course for each data instance, the `hypercoil` package also includes extensions of optimisers that accept ephemeral, instance-specific parameter groups for optimisation and optionally clear them from memory after they have been updated.

The results of the clustering experiment in subsamples of the HCP dataset are shown in Figure 8. Clustering $t \times t$ covariances into $c = 2$ subnetworks divides the brain along a unimodal (blue) to higher-order (red) axis (Figure 8, *Centre*); this clustering is replicable across data splits. We show the $t \times t$ covariance matrices for 2 example subjects, illustrating the distinct dynamic profiles of the two subnetworks. Clustering $p \times p$ covariances into $c = 3$ states yields three whole-brain connectivity states that are each distinct both from one another and from the time-averaged connectome (Figure 8, *Right*). The learned assignment of time frames to the three states is plotted for an example subject. In contrast with the most commonly used state detection methods (e.g. Allen et al. (2014)), the method we apply does not require estimating the covariance over a sliding window; it instead directly learns to assign time frames to states.

Methodological details: Time-by-time covariance (subnetwork detection) We again used the HCP dataset for the time-by-time covariance clustering experiment. Of our 12 HCP dataset splits, we selected 4 for training and 4 for evaluation. Vertex-wise BOLD data were mapped onto 400 parcel-wise time series using the 400-parcel dTSC parcellation that we trained in our first experiment. Because of the hemispheric tether regularisation that we used when learning the parcellation, we were able to impose a soft inter-hemispheric symmetry constraint on the subnetwork detection problem.

We used a batch size of 20 and selected a random window of 800 time frames from each parcellated time series when sampling a batch. Further preprocessing steps included imputation, filtering, and denoising, each implemented as described in *Denoising* above. Denoising was performed using a 36-confound model with demonstrated efficacy at removing structured artefact from fMRI data (Satterthwaite et al., 2013; Ciric et al., 2017; Parkes et al., 2018). The model was trained for 600 epochs, each of which consisted of 15 training steps, using SGD. Training required approximately 3 hours per replicate on a single RTX 2080 Ti GPU on a machine running Ubuntu Linux.

The objective of the clustering is to learn a $c \times p$ covariance parameter Θ: an assignment of p parcels to c clusters, which we interpret as subnetworks of the brain. For this proof of concept, we selected as our learning objective the minimal nontrivial clustering into

$c = 2$ subnetworks. During each forward pass through the covariance layer, a softmax mapped the 2-dimensional cluster assignment of each parcel to the set of Bernoulli distributions. Next, we computed the two time-by-time covariances parameterised by the matrices $\mathbf{\Theta}$ obtained by embedding the elements of each row of Θ along the main diagonal. These time-by-time covariances can be interpreted as dynamic profiles of the two detected subnetworks. Finally, the covariances were normalised to time-by-time Pearson correlations.

To perform the clustering, we used a loss function consisting of 5 terms. First, a dispersion term, equal to the negative L2 distance between the vectorised upper triangles of time-by-time matrices, promoted separation between dynamic profiles. To favour structured dynamics with larger correlations, we also imposed a symmetric L2 term that penalised distance of each entry in the correlation matrix from either -1 or 1. Entropy and equilibrium terms were used as in the parcellation experiment to promote eventual deterministic assignment to each of 2 approximately equal-sized subnetworks. As in the parcellation experiment, we started with a weak entropy term that we increased at the 500th step. The final term was a Jensen-Shannon divergence penalty placed on the distance between a parcel's subnetwork assignment distribution, and the assignment distribution of that parcel's analogue in the opposite hemisphere. This penalty resulted in a relatively symmetric subnetwork assignment.

Because of the substantial autocorrelation between temporally proximal BOLD frames, which is further inflated by the temporal filter that we apply, each time-by-time correlation matrix typically features large values near its main diagonal. To downweight the importance of these autocorrelations in the clustering, we took the Hadamard product of each time-by-time correlation matrix with a Toeplitz-structured exponential discounting matrix. Specifically, before it was passed to the loss function, each entry of the correlation matrix was scaled by the factor $1-e^{-\lambda t}$, where t denotes its offset from the main diagonal and $\lambda = 0.1$ is a discount hyperparameter. To smooth training and improve clustering repeatability, we also used stochastic weight averaging (Izmailov et al., 2019). We tuned model and training hyperparameters by repeating the analysis on our training set until we found hyperparameters that yielded a repeatable solution.

We then assessed the out-of-sample replicability of the learned subnetwork structure by training the model on a held-out evaluation set and assessing the convergence between the results across samples. Although there were subtle differences, the subnetwork assignments detected in the two samples were approximately the same after alignment (mean JS divergence over 400 parcels: 0.0052; 3 parcels' maximum assignments differed per hemisphere). In Figure 8, we show maps of each parcel's maximum subnetwork assignment, together with examples of dynamic profiles for two subjects from the training set. The dynamic profiles are windowed time-by-time correlation matrices parameterised by subnetwork assignments.

Here, we also run a further evaluation of the subnetwork detection (Figure 9, *Top*). We used the subnetworks detected in the training set as a reference, and we created 200 null subnetwork models by randomly assigning each hemisphere's parcels to subnetworks while preserving the number of parcels per subnetwork and inter-hemispheric symmetry. We selected 300 subjects from the evaluation set and computed for each subject the L2 distance between the dynamic profiles of the learned subnetworks, as well as the L2 distance between the dynamic profiles of each null model's subnetworks. Fig-

ure 9 compares the separation between dynamic profiles for the learned subnetworks (red) against null distributions (black) for each subject. In all cases, we find that the learned subnetwork assignments yield a superior separation to all null models.

Methodological details: Regional covariance (state detection) For the state detection experiment, we used small subsamples of HCP data. Each of the 12 data splits we created was further divided randomly into 5 shards, each consisting of approximately 50-80 resting-state BOLD time series. The results in the main text are for one shard with 59 time series; we found this to be sufficient for stable state detection. We show these results together with 9 replicates in Figure 9.

Preprocessing followed a standard functional connectivity pipeline. The first stage was parcellation using the 400-parcel gwMRF atlas (Schaefer et al., 2018), which we selected because its parcels are ordered to follow the brain's large-scale community structure and would thus be conducive to visualisation of differences between states. The 400 parcel-wise time series were then processed through imputation, filtering, and denoising as described in the *Denoising* section. We again denoised the time series using the proven 36-confound model, which has demonstrated efficacy in removing structured artefact due to motion and respiration from BOLD time series (Satterthwaite et al., 2013; Ciric et al., 2017; Parkes et al., 2018). The state detector model was trained for 50 epochs. Training required approximately 15 minutes per replicate on a 12-core first-generation Threadripper CPU on a machine running Ubuntu Linux.

We used k-means clustering to initialise $k = 3$ learnable state *templates*. For this proof of concept, we select $k = 3$ because it is the smallest number of clusters that does not produce only simple high-connectivity and low-connectivity states. The templates are initialised by first computing sliding-window correlations (as in Allen et al. (2014)) with a window size of 50 and a sliding step size of 25 for each of 1180 total images that constitute 4 splits of the HCP dataset. All windows are submitted as observations to the k-means clustering algorithm. We use the L2 distance to cluster in this proof of concept because it is compatible with various common implementations of k-means; however, a cosine distance might better capture differences in connectivity configurations that are not simply due to connection magnitude. Each of the k-means states that we detected using k-means clustering approximately corresponded with a single state that we later detected using our differentiable approach. The k-means initialisations additionally formed a baseline for model evaluation.

The learnable parameters of the clustering model include the templates thus initialised and instance-specific $c \times t$ covariance parameters Θ, where we chose $c = k = 3$. Each of the c parameter vectors of length t can be interpreted as the time course of a state; we interpret a value close to 0 as the absence of the state, and a value close to 1 as the presence of the state at some time point. We require an *instance-specific* parameterisation for each state because the presence or absence of a state is likely to occur at different times for each subject under the unconstrained conditions of the resting state. The use of instance-specific parameterisations is not handled natively by optimisers included with many existing deep learning software libraries. This is likely in part because of the ambiguity over how to handle buffers associated with instance-specific parameters (such as parameter-specific moments). For this proof of concept, we take the most direct approach of maintaining a separate buffer for each instance. We thus implemented new op-

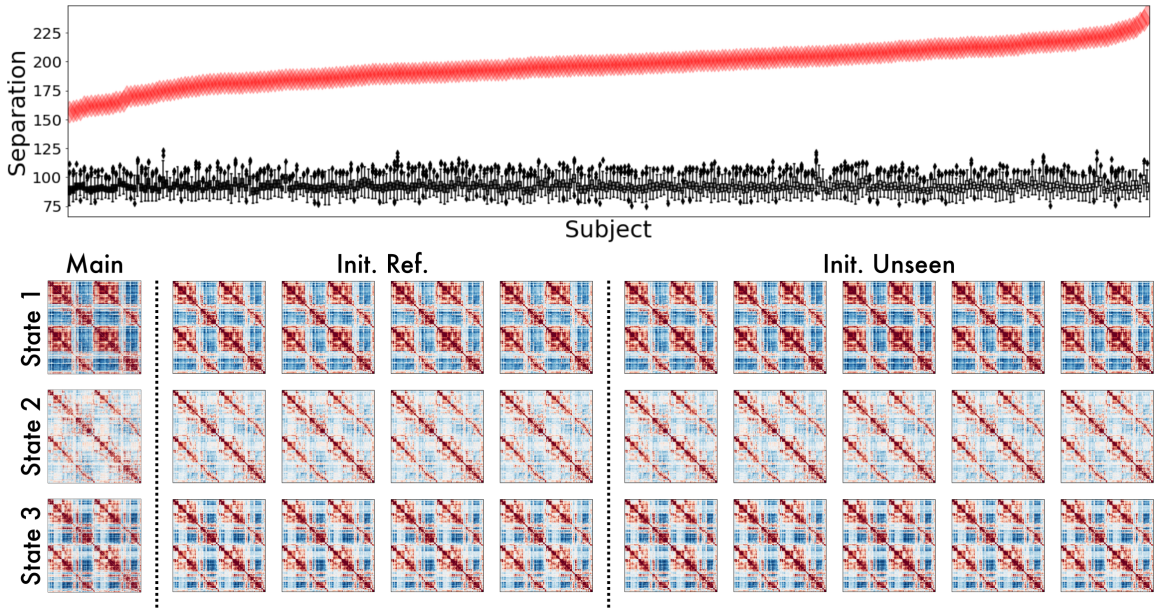


Figure 9: *Top*, Compared with random symmetric subnetwork assignments, the subnetworks detected by clustering consistently achieved greater separation of time-by-time dynamic covariance matrices. *Bottom*, The detected states were relatively consistent across both subsamples used for the initialisation and entirely unseen data.

timiser classes to handle parameter groups that exist ephemerally in memory. (This applies to the instance-specific parameters that we use here, but could also be extended to parameters specific to subgroups of the dataset–for instance, subjects with several runs.)

To train the state detection model, we used an instance-specific extension of SGD with a loss function consisting of 5 terms. First, an L2 penalty was applied to the backwards temporal difference of the instance-specific state time courses Θ in order to promote state persistence and increase the evidence required for the model to predict a transition between states. Second, an equilibrium loss was imposed to ensure that all states were represented in each instance. Third, a dispersion penalty was used to promote separation of each instance's detected states. The two remaining loss terms used the learnable templates to align detected states across instances. One term, another dispersion penalty, was used to promote separation of templates, and the final term was a penalty on the L2 distance between each instance's three detected states and the three templates. This final term encouraged the learnable templates to function as population-level representations of the states detected across instances. We did not use an entropy penalty for this clustering problem.

The main text figure shows the learned parameters Θ and the learned templates, and compares the L2 distance between the states detected by the k-means initialisation and the averages (across instances) of analogous states detected by the differentiable approach. We also repeated the state detection experiment in 4 additional subsamples of the HCP dataset, each comprising between 50 and 80 instances, which were used in computing the k-means initialisation, and 5 similar subsamples that were not used in computing

the initialisation. We qualitatively observed consistent detection of three analogous states across all subsamples (Figure 9, *Bottom*).

Appendix D. Community detection

Dataset and preprocessing We use the Midnight Scan Club dataset (MSC) for the community detection analysis. The MSC dataset includes a total 10 subjects each densely scanned over 10 sessions, both at rest and performing a number of directed cognitive tasks (Gordon et al., 2017). MSC data are openly available and were acquired in accordance with the guidelines of the host institution's IRB. In this proof of concept, we limit our analysis to resting-state data. We use the first three scans from each subject for the community detection analysis.

Data were parcellated using the 400-parcel gwMRF parcellation (Schaefer et al., 2018), filtered, and denoised using a 36-confound model with demonstrated efficacy at removing structured artefact associated with motion and respiration from BOLD data (Satterthwaite et al., 2013; Ciric et al., 2017; Parkes et al., 2018). The denoised, parcellated BOLD time series were inputs to the community detection model.

The learnable parameters of the model were (i) the $p \times c$ community affiliation matrix $\mathbf{C}$ of p parcels to c communities and (ii) the instance-specific $c \times t$ time courses Θ of each community. The community affiliation of each parcel was mapped through a softmax function that ensured it was a valid probability distribution, and each community time course was mapped through a sigmoid that constrained it to $(0, 1)$. Each community time course could thus be interpreted as the extent to which the corresponding community was modularised during each time frame of the scan. Each of these community time courses thereafter parameterised a separate covariance matrix.

Model details We thus computed $c = 10$ parameterised 400×400 covariance matrices $\mathbf{A}_{\mathcal{C}_i}$ and 1 unparameterised (standard or "time-averaged") 400×400 covariance matrix $\mathbf{A}$ among the 400 parcel-wise BOLD time series. (All covariance matrices were normalised to Pearson correlations for the purpose of further analysis.) Next, we used the community affiliation $\mathbf{C}$ to estimate the relaxed Girvan-Newman modularity (Newman and Girvan, 2004) on the unparameterised covariance matrix $\mathbf{A}$ as described in the main text. We then considered each column of $\mathbf{C}$ separately. Each column $\mathbf{C}_i$ of $\mathbf{C}$ corresponds to a single community $\mathcal{C}_i$, and the entries of the rank-one outer product $\mathbf{C}_i\mathbf{C}_i^\intercal$ indicate the extent to which each edge of the connectome graph connects two vertices in $\mathcal{C}_i$. We used each of these rank-one outer products as coaffiliation matrices $\mathbf{H}_{\mathcal{C}_i}$ to compute the relaxed Girvan-Newman modularity on the corresponding parameterised covariance $\mathbf{A}_{\mathcal{C}_i}$.

We thus sought to optimise the $c+1$ modularities by jointly learning the affiliation of each parcel to a community and the extent to which the community was modularised during each time frame. To achieve this objective, we trained the model using a loss function with four main terms. Two multipliers controlled the balance between static communities parameterised only by $\mathbf{C}$ and dynamic communities parameterised by $\mathbf{C}$ and Θ. The remaining two multipliers regularised the learned parameters: one modulated an L2 penalty on the minimum distance between each entry in Θ and either 0 or 1, thereby steering the model to make a binary decision for each time point as to whether a community was modularised or not, and the other modulated a backward difference L2 penalty on Θ that promoted persistence of modularisation and demodularisation states for each community. The model was trained for 500 epochs using the Adam optimiser

(Kingma and Ba, 2017). Training required under 1 hour on a 12-core, 24-thread CPU.

Evaluation To evaluate the validity of the learned community time courses Θ, we created, for each instance, a distribution of 500 null community time courses by randomly shifting that instance's learned time courses. Time frames that the shift displaced over the end of each time course were wrapped back to its start. We then computed, for each of the null covariances $\widetilde{\mathbf{A}}_{\mathcal{C}_i}$ parameterised by a randomly shifted time course, the corresponding modularity. We created the main text plot of learned and null modularities for each community by subtracting each of the 500 null modularities computed for each instance from the learned modularity for that instance. We generally found that the learned modularity outperformed random-shift nulls. (We remark that small random shifts, and even random shifts of zero, are likely to occur because 500 random shifts are selected for each subject, and each time course is 814 frames long. Due to factors including the persistence imposed on each community's modularisation state and the substantial autocorrelation of filtered BOLD time series, these small random shifts will likely have modularity very close to that of the learned model.)

Appendix E. Selected advances in differentiable programming

(i) The data throughput problem, rebatching, and local accumulator nodes

The parcellation module in general operates on vertex- or voxel-wise time series data, whose dimension v typically numbers in the tens of thousands. Because the parcellation module is a dimension-reducing map from vertices to parcels, any downstream workflow steps can operate in a much lower-dimensional space (in practice, not greater than the order of 1000). Loss functions on the parcellation module can have a space complexity that scales linearly or quadratically in v, and the capacity of hardware accelerators vis-à-vis the parcellation module consequently bottlenecks the overall data throughput of the differentiable workflow. In practical terms, these expensive loss functions set an effective ceiling on the maximum batch size of the training procedure. In our experiments on cortical HCP data in the vertex-wise fsLR coordinate space, we were frequently limited to a batch size not exceeding 5 BOLD time series.

This is not a problem when the workflow is optimised using block coordinate-style training loops that limit learning to one module at a time, as in the current manuscript. However, it becomes a significant challenge in the context of end-to-end connectome workflow optimisation. This is because the QC-FC loss function used to train the denoising module relies on stable estimation of correlations across the batch dimension, which is clearly untenable for a batch size of 5 or less. (Recall that the QC-FC loss function is simply the correlation between a QC measure such as gross subject motion and edge-wise estimates of functional connectivity. This correlation is computed across the batch dimension.)

In summary, a data throughput problem arises because the parcellation module requires a small batch size, while the denoising module requires a large batch size. We resolve this mismatch between data throughput demands by implementing a local backward pass accumulator node in the computational graph. This accumulator node wraps a throughput-limited operation (here, parcellation) to (in essence) support rebatching the data. It requires explicitly defining (i) a parameterised vector-Jacobian product (VJP)

rule corresponding to the wrapped operation, which is accumulated over (in our case, the parcellation step), and (ii) an update procedure for this VJP rule that is called on each forward pass through the accumulator node. For our use case, the accumulator node is configured to progressively update a running mean of the parameters of the VJP rule for matrix multiplication. When the backward pass is run on the graph, this VJP rule is applied, and the accumulator's local parameterisation is cleared.

We remark that research centres with access to a large number of hardware accelerators can resolve this problem without resort to an accumulator, by simply calling an appropriate map-reduce that parallelises the relevant part of the program to execute over different input datasets.

(ii) Serial mapping for memory-intensive operations

Yet another related challenge also arises as a consequence of the memory constraints of hardware accelerators vis-à-vis loss functions for parcellation. As mentioned, the space complexity of these loss functions scales with the voxel or vertex dimension of the input time series in a linear or quadratic fashion. As a result of this scaling, the full computation of these loss functions cannot feasibly fit into accelerator memory. Fortunately, the same loss functions can be partitioned over data chunks without changing the results of computation, as long as the relation $f(A : B, *) = f(A, *) : f(B, *)$ is satisfied for forward and backward passes f, arbitrary chunks A and B, any additional non-chunked arguments *, and the infix : denoting the concatenation operation. We accordingly implement functionality to perform the forward and backward passes through these chunked computations and immediately free the computational graphs in order to satisfy the memory ceiling imposed by the hardware accelerator.

In principle, a single general solution (which we call a "serialisation map") might be implemented for both this problem and the throughput/rebatching problem (i) described above, since the rebatching problem is essentially an instance of this problem where the chunking is performed over the batch axis. Our implementations are currently not sufficiently general to accomplish this: our solution to problem (i) requires explicit specification of a parameterised VJP and a parameter update rule, which can be extremely tedious to derive for complicated models, while our solution to (ii) obligatorily deletes the entire computational graph for each chunk, leaving it impractical for many use cases. A general solution might be tractable using differentiable programming frameworks that more directly expose Jacobian and VJP computations.

(iii) Incorporating state-of-the-art domain knowledge as parameter initialisations

Because a differentiable workflow is designed to interpolate existing workflow configurations, each existing configuration corresponds to some parameterisation of the differentiable workflow. We demonstrate this in our introduction, where we instantiate 18 pre-existing workflow configurations as differently parameterised neural networks.

This feature of the differentiable workflow paradigm implies that state-of-the-art methods and domain knowledge can be operationalised as initialisation schemes. This enables the learning process to theoretically begin in a relatively "good" neighbourhood of the parameterisation space. These initialisation schemes can also be combined with noise and dropout sources to perturb the pipeline slightly away from an existing so-

lution, which is useful for instance when the existing solution corresponds to a region with weak gradients.

Appendix F. Hyperparameter and training summary tables

Name	Value	Notes
Learning rate	0.05	Weakened by a factor of 0.45 at epochs 500 and 1000, and then reset at SWA begin
Max step	6000	
Batch size	3	
Num. subjects	618	
Num. images	2457	
Data interval	5	To reduce the overhead of reading from disk, a new batch of data is loaded every 5 steps.
Window length	500	Number of time frames in random window applied to data during training.
SWA begin	1600	First epoch of SWAPR model
$\nu_{\text{dispersion}}$	10	Annealed to 0.5 by step 300
$\nu_{\text{determinant}}$	.005	Annealed to 0.0001 by step 400
$\nu_{\text{equilibrium}}$	1e6	Scheduled to alternately spike and decay in synchrony with entropy cascade steps. Maximum value 3e7 at step 1500, final value 1e5 at step 6000.
ν_{entropy}	0.1	Cascaded upward progressively. 0.5 at 1500, 1 at 2000, 1.5 at 2500, 2 at 3000, 2.5 at 3500, 3 at 4000, 3.5 at 4500.
$\nu_{\text{compactness}}$	2	Strengthened to 20 at step 600 and 30 at step 2500.
ν_{tether}	0.2	
$\nu_{\text{2nd moment}}$	5	Multiplier alternated between 10, 7, and 5: increased in synchrony with each entropy cascade step, and then decayed back to 5.
Parameter revolutions	6	Parameters revolved from the SWA model to the main model 6 times, at steps 2050, 2550, 3050, 3550, 4050, 4550.

Table 3: **Table of hyperparameters for parcellation experiments.** We use the designation $\nu_{\mathcal{L}}$ to denote the loss multiplier for loss function $\mathcal{L}$. Some details have been elided; consult the training script for information about the workflow structure.

Name	Value	Notes
Learning rate	0.05	
Max epoch	200	
Batch size	100	
Weight decay	0.001	
Num. subjects	356	
Num. images	1393	
Window length	500	Number of time frames in random window applied to data during training.

Table 4: **Table of hyperparameters for denoising experiments.** The denoising experiments used only a single loss function, the QC-FC, with a multiplier of 1. Some details have been elided; consult the training script for information about the workflow structure.

Name	Value	Notes
Learning rate	0.02	
Max epoch	500	
Batch size	30	Trained using transformed batch gradient descent
Num. subjects	10	
Num. images	30	First 3 sessions from MSC dataset
Num. communities	10	
$\nu_{\text{modularity}}$	10	
ν_{dynamic}	0.002	Dynamic analogue of modularity
$\nu_{\text{smoothness}}$	.2	Penalises transitions
ν_{bimodal}	1	Penalises trajectory with value other than 0 or 1
γ	5	Community resolution parameter

Table 5: **Table of hyperparameters for community detection experiments.** We use the designation $\nu_{\mathcal{L}}$ to denote the loss multiplier for loss function $\mathcal{L}$. Some details have been elided; consult the training script for information about the workflow structure.