Learning Macroscopic Brain Connectomes via Group-Sparse Factorization

Farzane Aminmansour[1], Andrew Patterson[1], Lei Le[2], Yisu Peng[3], Daniel Mitchell[1], Franco Pestilli[4,5,6,7,8], Cesar Caiafa[9,10], Russell Greiner[1] and Martha White[1]

[1]Department of Computing Science, University of Alberta, Edmonton, Alberta, Canada
[2]Department of Computer Science, Indiana University, Bloomington, Indiana, USA
[3]Department of Computer Science, Northeastern University, Boston, Massachusetts, USA
[4]Department of Psychological and Brain Sciences, Indiana University, Bloomington, Indiana, USA
[5]Program in Neuroscience, Indiana University, Bloomington, Indiana, USA
[6]Program in Cognitive Science, Indiana University, Bloomington, Indiana, USA
[7]School of Optometry, Indiana University, Bloomington, Indiana, USA
[8]Department of Computer Science and Intelligent Systems Engineering, School of Informatics, Indiana University, Bloomington, Indiana, USA
[9]Instituto Argentino de Radioastronomía- CCT La Plata, CONICET / CIC-PBA, (1894) V. Elisa, Argentina
[10]Tensor Learning Unit- Center for Advanced Intelligence Project, RIKEN, (103-0027) Tokyo, Japan
{aminmans, ap3, daniel7, rgreiner, whitem}@ualberta.ca
{leile}@iu.edu
{peng.yis}@husky.neu.edu
{franpest}@indiana.edu
{ccaiafa}@gmail.com

Abstract

Mapping structural brain connectomes for living human brains typically requires expert analysis and rule-based models on diffusion-weighted magnetic resonance imaging. A data-driven approach, however, could overcome limitations in such rule-based approaches and improve precision mappings for individuals. In this work, we explore a framework that facilitates applying learning algorithms to automatically extract brain connectomes. Using a tensor encoding, we design an objective with a group-regularizer that prefers biologically plausible fascicle structure. We show that the objective is convex and has unique solutions, ensuring identifiable connectomes for an individual. We develop an efficient optimization strategy for this extremely high-dimensional sparse problem, by reducing the number of parameters using a greedy algorithm designed specifically for the problem. We show that this greedy algorithm significantly improves on a standard greedy algorithm, called Orthogonal Matching Pursuit. We conclude with an analysis of the solutions found by our method, showing we can accurately reconstruct the diffusion information while maintaining contiguous fascicles with smooth direction changes.

1 Introduction

Diffusion-weighted magnetic resonance imaging (dMRI) combined with fiber tractography is the only method available to map structural brain connectomes in living human brains [3, 27, 21]. This method has revolutionized our understanding of the network structure of the human brain and the role of white matter in health and disease. Standard practice in mapping connectomes comprises of several steps by which a dMRI is acquired (Fig 1A), a model is fit to the signal in each brain voxel

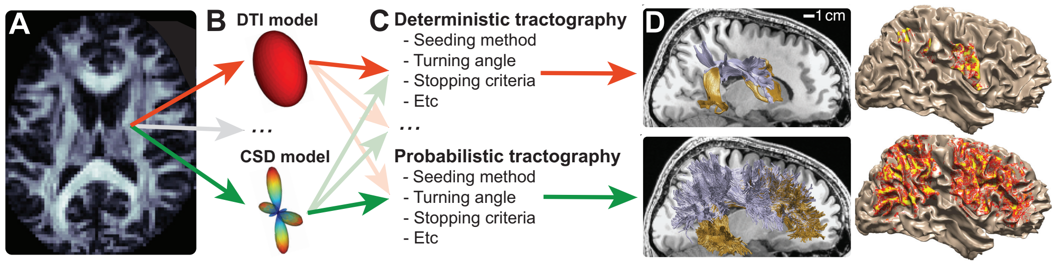


Figure 1: **A**: Measurements of white matter using diffusion-weighted magnetic resonance imaging (dMRI). **B**: Multiple models can describe the dMRI signal in each brain voxel. For example, the diffusion-tensor model (DTI; top, [2]) and the constrained-spherical deconvolution model (CSD, bottom; [26]) are commonly used. **C**: Multiple tractography methods integrate model fits across voxels to estimate long-range brain connections. There are many tractography algorithms exist, each with multiple parameters, for both deterministic and probabilistic methods [25]. In principle several combinations of methods and parameters are used by investigators. **D**: **Left**: Two major white matter tracts, the arcuate fasciculus in gold and superior lateral fasciculus in lilac, reconstructed in a single brain using deterministic (top) and probabilistic (bottom) tractography. **Right**: Cortical termination of the superior lateral fasciculus in the same brain estimated with deterministic (top) and probabilistic (bottom) tractography. Arrows show multiple possible choices of model and parameters to generate connectome estimates (D) from dMRI data (A).

(Fig. 1B) and a tractography algorithm is used to estimate long range brain connections (Fig. 1C). Multiple models can be used at each one of these steps and each model allows multiple parameters to be set. Currently, best practice in the field is to choose one model and pick a single set of parameters using heuristics such as recommendations by experts or previous publications. This rule-based approach has several limitations. For example, different combinations of models and parameters generate different solutions (Fig 1D). Figure 1 exemplifies how from a single dMRI data set collected in a brain, choosing a single model and parameters set (Fig. 1A-C) can generate vastly different connectome mapping results (Fig 1D; adapted from [18]). In the figure, we show that both estimates of white matter tracts (Fig 1D left) and cortical connections (Fig. 1D right) vary substantially even within a single brain.

There have been some supervised learning approaches proposed for tractography. These supervised methods, however, such as those using random forests [16] and neural networks [20, 5] require labelled data. This means tractography solutions must first be given for training, limiting the models mainly to mimic expert solutions rather than learn structures beyond them. A few methods have used regularized learning strategies, but for different purposes, such as removing false connections in the given tractography solution [12] and using radial regularization for micro-structure [9].

This work presents a fully unsupervised learning framework for tractography. We exploit a recently introduced encoding for connectome data, called ENCODE [8], which represents dMRI (and white matter fascicles) as a tensor factorization. This factorization was previously used only to represent expert connectomes as a tensor, generated using a standard rule-based tractography process introduced in Fig. 1. We propose to instead learn this tensor using the dMRI data, to learn the structure of brain connectomes. We introduce a regularized objective that attempts to extract a tensor that reflects a biologically plausible fascicle structure while also reconstructing the diffusion information. We address two key challenges: (1) designing regularizers that adequately capture biologically plausible tract structures and (2) optimizing the resulting objective for an extremely high-dimensional and sparse tensor. We develop a group regularizer that captures both spatial and directional continuity of the white matter fascicles. We solve this extremely high-dimensional sparse problem using a greedy algorithm to screen the set of possible solutions upfront. We prove both that the objective is convex, with a unique solution, and provide approximation guarantees on the greedy algorithm. We then show that this greedy algorithm much more effectively selects possible solutions, as compared to a standard greedy algorithm called Orthogonal Matching Pursuit (OMP). We show, both quantitatively and qualitatively, that the solutions provided by our method effectively reconstruct the diffusion information in each voxel while maintaining contiguous, smooth fascicles.

2 Encoding Brain Connectomes as Tensors

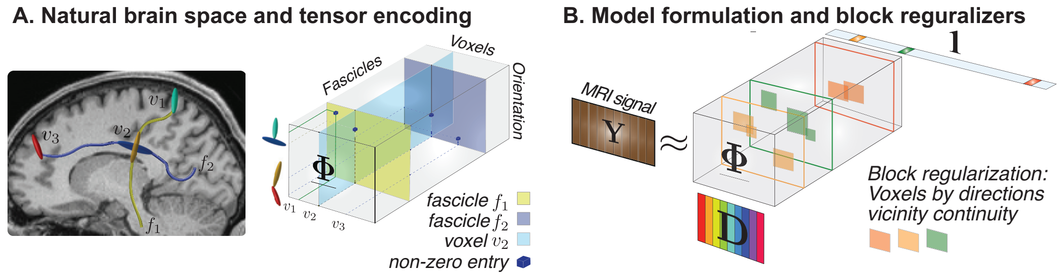


Figure 2: **A**: The ENCODE method; from natural brain space to tensor encoding. **Left**: Two example white matter fascicles (f_1 and f_2) passing through three voxels (v_1, v_2 and v_3). **Right**: Encoding of the two fascicles in a three dimensional tensor. The non-zero entries in $\underline{\mathbf{\Phi}}$ indicate fascicles orientation (1st mode), position (voxel, 2nd mode) and identity (3rd mode). **B**: Model formulation and group sparse regularization. Depiction of how ENCODE facilitates integration of dMRI signal, $\mathbf{Y}$, connectome structure, $\underline{\mathbf{\Phi}}$, and a dictionary of predictions of the dMRI signal, $\mathbf{D}$, for each fascicle orientation. The group regularizers (orange and green squares) defines pairwise groups of neighbouring voxels and similar orientations. Note that the voxels are linearized to enable $\underline{\mathbf{\Phi}}$ and the groups to be visualized. This allows us to flatten four-dimensional hyper-cubes—three dimensions for voxels and one for orientations—to squares.

ENCODE [8] maps fascicles from their natural brain space into the three dimensions of a sparse tensor $\underline{\mathbf{\Phi}} \in \mathbb{R}^{N_a \times N_v \times N_f}$ (Fig. 2A - right). The first dimension of $\underline{\mathbf{\Phi}}$ (1st mode, size N_a) encodes individual white matter fascicles orientation at each position along their path through the brain. Individual segments (nodes) in a fascicle are coded as non-zero entries in the sparse array (dark-blue cubes in Fig. 2A - right). The second dimension of $\underline{\mathbf{\Phi}}$ (2nd mode, size N_v) encodes fascicles spatial position within the voxels of dMRI data. Slices in this second dimension represent single voxels (cyan slice in Fig. 2A - right). The third dimension (3rd mode, size N_f) encodes the indices of each fascicle within the connectome. Full fascicles are encoded as $\underline{\mathbf{\Phi}}$ frontal slices (cf., yellow and blue in Fig. 2A - right). Within one tract, such as the Arcuate Fasciculus, the model we use has fine-grained orientations $N_a = 1057$, with number of fascicles $N_f = 868$ and number of voxels $N_v = 11,823$.

ENCODE facilitates the integration of measured dMRI signal with the connectome structure (Fig. 2B - right). The dMRI signal is represented as matrix $\mathbf{Y} \in \mathbb{R}^{N_\theta \times N_v}$. ENCODE allows factorizing the dMRI signal as product of the 3-dimensional tensor $\underline{\mathbf{\Phi}} \in \mathbb{R}^{N_a \times N_v \times N_f}$ by a dictionary of dMRI signal predictions $\mathbf{D} \in \mathbb{R}^{N_\theta \times N_a}$:$\mathbf{Y} \approx \underline{\mathbf{\Phi}} \times_1 \mathbf{D} \times_3 \mathbf{1}$. The notation "$\times_n$" is the tensor-by-matrix product in mode-n (see [14]). The dot product with $\mathbf{1} \in \mathbb{R}^{N_f}$ sums over the fascicle dimension.[1]

3 A Tractography Objective for Learning Brain Connectomes

The original work on ENCODE assumed the tensor $\underline{\mathbf{\Phi}}$ was obtained from a tractography algorithm. In this section, we instead use this encoding to design an objective to learn $\underline{\mathbf{\Phi}}$ directly from dMRI data. First consider the problem of estimating tensor $\underline{\mathbf{\Phi}}$ to best predict $\mathbf{Y}$, for a given $\mathbf{D} \in \mathbb{R}^{N_\theta \times N_a}$. We can use a standard maximum likelihood approach (see Appendix A for the derivation), to get the following reconstruction objective

$$\hat{\underline{\mathbf{\Phi}}} = \underset{\underline{\mathbf{\Phi}} \in \mathbb{R}^{N_a \times N_v \times N_f}}{\operatorname{argmin}} \|\mathbf{Y} - \underline{\mathbf{\Phi}} \times_1 \mathbf{D} \times_3 \mathbf{1}\|_F^2, \tag{1}$$

where $\|\cdot\|_F$ is the Frobenius norm that sums up the squared entries of the given matrix. This objective prefers $\underline{\mathbf{\Phi}}$ that can accurately recreate the diffusion information in $\mathbf{Y}$. This optimization, however, is highly under-constrained, with many possible (dense) solutions.

In particular, this objective alone does not enforce a biologically plausible fascicle structure in $\underline{\mathbf{\Phi}}$. The tensor $\underline{\mathbf{\Phi}}$ should be highly sparse, because each voxel is expected to have only a small number

[1]The original encoding uses a set of fascicles weights $\mathbf{w} \in \mathbb{R}^{N_f}$, to get $\mathbf{Y} \approx \underline{\mathbf{\Phi}} \times_1 \mathbf{D} \times_3 \mathbf{w}$. For a fixed $\underline{\mathbf{\Phi}}$, $\mathbf{w}$ was learned to adjust the magnitude of each fascicle dimension. We do not require this additional vector, because these magnitudes can be incorporated into $\underline{\mathbf{\Phi}}$ and implicitly learned when $\underline{\mathbf{\Phi}}$ is learned.

of fascicles and orientations [18]. For example, for the Arcuate Fasciculus, we expect at most an activation level in $\underline{\mathbf{\Phi}}$ of $(N_v \times 10 \times 10/(N_a \times N_v \times N_f) \approx 1e{-}6$, using a conservative upper bound of 10 fascicles and 10 orientations on average per voxel. Additionally, the fascicles should be contiguous and should not sharply change orientation.

We design a group regularizer to encode these properties. Anatomical consistency of fascicles is enforced locally within groups of neighboring voxels and orientations. Overlapping groups are used to encourage this local consistency to result in global consistency. Group regularization prefers to zero all coefficients for a group. This zeroing has the effect of clustering non-zero coefficients in local regions within the tensor, ensuring similar fascicles and orientations are active based on spatial proximity. Further, overlapping groups encourages neighbouring groups to either both be active or inactive for a fascicle and direction. This promotes contiguous fascicles and smooth direction changes. These groups are depicted in Figure 2B, with groups defined separately for each fascicle (slice). We describe the group regularizer more formally in the remainder of this section.

Assume we have groups of voxels $\mathcal{G}_\mathcal{V} \in \mathcal{V}$ based on spatial coordinates and groups of orientations $\mathcal{G}_\mathcal{A} \in \mathcal{A}$ based on orientation similarity. For example, each $\mathcal{G}_\mathcal{V}$ could be a set of 27 voxels in a local cube; these cubes of voxels can overlap between groups, such as $\{(1,1,1),(1,1,2),\ldots,(3,3,3)\} \in \mathcal{V}$ and $\{(2,1,1),(2,1,2),\ldots,(4,3,3)\} \in \mathcal{V}$. Each $\mathcal{G}_\mathcal{A}$ can be defined by selecting one atom (one orientation) and including all orientations in the group that have a small angle to that central atom, i.e., an angle that is below a chosen threshold. Consider one orientation, voxel, fascicle triple (a, v, f). Assume a voxel has a non-zero coefficient for a fascicle: $\underline{\mathbf{\Phi}}_{a,v,f}$ is not zero for some a. A voxel within the same group $\mathcal{G}_\mathcal{V}$ is likely to have the same fascicle with a similar orientation. A distant voxel, on the other hand, is highly unlikely to share the same fascicle. The goal is to encourage as many pairwise groups $(\mathcal{G}_\mathcal{V}, \mathcal{G}_\mathcal{A})$ to be inactive—have all zero coefficients for a fascicle—and concentrate activation in $\underline{\mathbf{\Phi}}$ within groups.

We can enforce this group sparsity by adding a regularizer to (1). Let $x_{\mathcal{G}_\mathcal{A},v,f} \in \mathbb{R}$ indicate whether a fascicle f is active for voxel v, for any orientation $a \in \mathcal{G}_\mathcal{A}$. Let $\mathbf{x}_{\mathcal{G}_\mathcal{A},\mathcal{G}_\mathcal{V},f}$ be the vector composed of these identifiers for each $v \in \mathcal{G}_\mathcal{V}$. Either we want the entire vector $\mathbf{x}_{\mathcal{G}_\mathcal{A},\mathcal{G}_\mathcal{V},f}$ to be zero, meaning the fascicle is not active in any of the voxels $v \in \mathcal{G}_\mathcal{V}$ for the orientations $a \in \mathcal{G}_\mathcal{A}$. Or, we want more than one non-zero entry in this vector, meaning multiple nearby voxels share the same fascicle. This second criterion is largely enforced by encouraging as many blocks to be zero as possible, because each voxel will prefer to activate fascicles and orientations in already active pairs $(\mathcal{G}_\mathcal{V}, \mathcal{G}_\mathcal{A})$. As with many sparse approaches, we use an ℓ_1 regularizer to set entire blocks to zero. In particular, as has been previously done for block sparsity [24], we can use an ℓ_1 across the blocks $\mathbf{x}_{\mathcal{G}_\mathcal{A},\mathcal{G}_\mathcal{V},f}$

$$\sum_{f\in\mathcal{F}} \sum_{\mathcal{G}_\mathcal{V}\in\mathcal{V}} \sum_{\mathcal{G}_\mathcal{A}\in\mathcal{A}} \|\mathbf{x}_{\mathcal{G}_\mathcal{A},\mathcal{G}_\mathcal{V},f}\|_2. \tag{2}$$

The outer sums can be seen as an ℓ_1 norm across the vector of norm values containing $\|\mathbf{x}_{\mathcal{G}_\mathcal{A},\mathcal{G}_\mathcal{V},f}\|_2$. This encourages $\|\mathbf{x}_{\mathcal{G}_\mathcal{A},\mathcal{G}_\mathcal{V},f}\|_2 = 0$, which is only possible if $\mathbf{x}_{\mathcal{G}_\mathcal{A},\mathcal{G}_\mathcal{V},f} = \mathbf{0}$.

Finally, we need to define a continuous indicator variable $\mathbf{x}_{\mathcal{G}_\mathcal{A},\mathcal{G}_\mathcal{V},f}$ to simplify the optimization. A 0-1 indicator is discontinuous, and would be difficult to optimize. Instead, we use the following continuous indicator

$$\mathbf{x}_{\mathcal{G}_\mathcal{A},\mathcal{G}_\mathcal{V},f} = [\|\underline{\mathbf{\Phi}}_{\mathcal{G}_\mathcal{A},v_1,f}\|_1,\ldots,\|\underline{\mathbf{\Phi}}_{\mathcal{G}_\mathcal{A},v_n,f}\|_1] \quad \text{for each } v_i \in \mathcal{G}_\mathcal{V} \tag{3}$$

An entry in $\mathbf{x}_{\mathcal{G}_\mathcal{A},\mathcal{G}_\mathcal{V},f}$ is 0 if fascicle f is not active for $(\mathcal{G}_\mathcal{V}, \mathcal{G}_\mathcal{A})$. Otherwise, the entry is proportional to the sum of the absolute coefficient values for that fascicle for orientations in $\mathcal{G}_\mathcal{A}$.

Our proposed group regularizer is

$$R(\underline{\mathbf{\Phi}}) = \sum_{f\in\mathcal{F}} \sum_{\mathcal{G}_\mathcal{V}\in\mathcal{V}} \sum_{\mathcal{G}_\mathcal{A}\in\mathcal{A}} \|\mathbf{x}_{\mathcal{G}_\mathcal{A},\mathcal{G}_\mathcal{V},f}\|_2 = \sum_{f\in\mathcal{F}} \sum_{\mathcal{G}_\mathcal{V}\in\mathcal{V}} \sum_{\mathcal{G}_\mathcal{A}\in\mathcal{A}} \sqrt{\sum_{v\in\mathcal{G}_\mathcal{V}} \left(\sum_{a\in\mathcal{G}_\mathcal{A}} |\underline{\mathbf{\Phi}}_{a,v,f}|\right)^2}, \tag{4}$$

which, combined with equation (1), gives our proposed objective

$$\min_{\underline{\mathbf{\Phi}}\in\mathbb{R}^{N_a\times N_v\times N_f}} \|\mathbf{Y} - \underline{\mathbf{\Phi}} \times_1 \mathbf{D} \times_3 \mathbf{1}\|_F^2 + \lambda R(\underline{\mathbf{\Phi}}) \tag{5}$$

for regularization weight $\lambda > 0$. This objective balances between reconstructing diffusion data and constraints on the structure in $\underline{\mathbf{\Phi}}$. Crucially, this objective is convex in $\underline{\mathbf{\Phi}}$ and has a unique solution, which we show in Theorem 1 in Appendix B. Uniqueness ensures identifiable tractography solutions and convexity facilitates obtaining optimal solutions.

4 An Efficient Algorithm for the Tractography Objective

Standard gradient descent algorithms can be used directly on (5) to find the optimal solution. Unfortunately, the number of parameters in the optimization is very large: $N_v \times N_f \times N_a$ is billions even for just one tract. At the same time, the number of active coefficients at the end of the optimization is much smaller, only on the order of N_v, because there are only handful of fascicles and orientations per voxel. Even when initializing $\underline{\mathbf{\Phi}}$ to zero, the gradient descent optimization might make all of $\underline{\mathbf{\Phi}}$ active during the optimization. Screening algorithms have been developed to prune entries for sparse problems [28, 6]. These generic methods, however, still have too many active coefficients to make this optimization tractable for wide application, as we have verified empirically.

Instead, we can design a screening algorithm specialized to our objective. Orientations can largely be selected independently for each voxel, based solely on diffusion information. We can infer the likely orientations of fascicles in a voxel that could plausibly explain the diffusion information, without knowing precisely which fascicles are in that voxel. If we can select a plausible set of orientations for each voxel before optimizing the objective, we can significantly reduce the number of parameters. For example, 20 orientations is a large superset, but would reduce the number of parameters by a factor of 10,000 because the whole $N_a = 120,000$.

One strategy is to generate these orientations greedily, such as with a method like Orthogonal Matching Pursuit (OMP). This differs from most screening approaches, which usually iteratively prune starting from the full set. Generating orientations starting from an empty set, rather than pruning, is a more natural strategy for such an extremely sparse solution, where only 0.017% of the items are used. Consider how OMP might generate orientations. For a given voxel v, the next best orientation is greedily selected based on how much it reduces the residual error for the diffusion. On the first step, it adds the single best orientation for predicting the $N_\theta = 96$ dimensional diffusion vector for voxel v. It generates up to a maximum of k orientations greedily and then stops. Then only coefficients for this set of orientations will be considered for voxel v in the optimization of the tractography objective. This procedure is executed for each voxel, and is very fast.

Though a greedy strategy for generating orientations is promising, the criterion used by OMP is not suitable for this problem. Using residual errors for the criterion prefers orthogonal or dissimilar orientations, to provide a basis with which to easily reconstruct the signal. The orientations in voxels, however, are unlikely to be orthogonal. Instead, it is more likely that there are multiple fascicles with similar orientations in a voxel, with some fascicles overlapping in a different—but not necessarily orthogonal—direction. We must modify the selection criterion to select a number of similar orientations to reconstruct the diffusion information in a voxel.

To do so, we rely on the more general algorithmic framework for subselecting items from a set, of which OMP is a special case. We need to define a criterion which evaluates the quality of subsets S from the full set of items $\mathcal{S}$. In our setting, $\mathcal{S}$ is the full set of orientations and S a subset of those orientations. Our goal is to find $S \subset \mathcal{S}$ with $|S| \leq k$ such that $\bar{g}(S)$ is maximal. If we can guarantee this criterion $\bar{g} : \mathcal{P}(\mathcal{S}) \to \mathbb{R}$ is (approximately) submodular, then we can rely on a wealth of literature showing the effectiveness of greedy algorithms for picking S to maximize $\bar{g}$.

We use a simple modification on the criterion for OMP, the $g(S) =$ the squared multiple correlation [13]. We propose a simple yet effective modification, and define the Orientation Greedy criterion as

$$\bar{g}(S) \overset{\text{def}}{=} g(S) + \sum_{s \in S} g(\{s\})$$

This objective balances between preferring a set S with high multiple correlation, and ensuring that each orientation itself is useful. Each orientation likely explains a large proportion of the diffusion for a voxel. This objective will likely prefer to pick two orientations that are similar that recreate the diffusion in the voxel well. This contrasts two orthogonal orientations, that can be linearly combined to produce those two orientation but that themselves do not well explain the diffusion information. This modification is conceptually simple, yet now has a very different meaning. The simplicity of the modification is also useful for the optimization, since a linear sum of submodular functions is itself again submodular. We provide approximation guarantees for this submodular maximization in Appendix D, using results for the multiple correlation [13].

The full algorithm consists of two key steps. The first step is to screen the orientations, using Orientation Greedy in Algorithm 1. We then use subgradient descent to optimize the Tractography

Objective using this much reduced set of parameters. The second step prunes this superset of possible orientations further, often to only a couple of orientations. The resulting solution only has a small number of active fascicles and orientations for each voxel. We provide a detailed derivation and description of the algorithm in Appendix C.

The optimization given the screened orientations remains convex. The main approximation in the algorithm is introduced from the greedy selection of orientations. We provide approximation guarantees for how effectively the greedy algorithm maximizes the criterion $\bar{g}$. But, this does not characterize whether the criterion itself is a suitable strategy for screening. In the next section, we focus our empirical study on the efficacy of this greedy algorithm, which is critical for obtaining efficient solutions for the tractography objective.

5 Empirical results: Reconstructing the anatomical structure of tracts

We investigate the properties of the proposed objective on two major structures in the brain. The first is the Arcuate Fasciculus, hereafter Arcuate. The other is the Arcuate combined with one branch of the Superior Longitudinal Fasciculus, SLF1, hereafter ARC-SLF. Due to space constraints, we relegate additional empirical results on ARC-SLF to Appendix E.5. We learn on data generated by an expert connectome solution within the ENCODE model (Appendix E.1). This allows us to objectively investigate the efficacy of the objective and greedy optimization strategy, because we have access to the ground truth $\underline{\mathbf{\Phi}}$ that generated the data. To the best of our knowledge, this is the first complete data driven approach for extracting brain connectomes. We, therefore, focus primarily on understanding the properties of our learning approach for tractography.

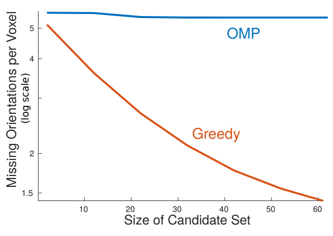
We particularly (a) investigate how effectively our Greedy algorithm selects orientations, (b) investigate how accurately the group regularized objective with this screening approach can reconstruct the diffusion information, and (c) visualize the plausibility of the solutions produced by our method, particularly in terms of smoothness of the fascicles. Even with screening, this optimization when learning over all fascicles and voxels, is prohibitively expensive for running thorough experiments. We therefore focus first on evaluating the model given the assignment of fascicles to voxels. Because the largest approximation in the algorithm is the greedy selection of orientations, this is the most important step to understand first. For a given set of (greedily chosen) orientations, the objective remains convex with a unique solution. We know, therefore, that further optimizing over fascicles as well would only reduce the reconstruction error.

5.1 Screening

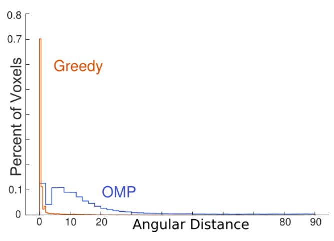
We define two error metrics to demonstrate the utility of GreedyOrientation over OMP for this task. The first is the total number of orientations present in $\underline{\mathbf{\Phi}}$-expert that are not present in $\underline{\mathbf{\Phi}}$ generated by the screening approach, measuring the exactness of the solution. The second metric is the minimum possible angular distance between each of the orientations in $\underline{\mathbf{\Phi}}$-expert with any arbitrary set of orientations in the corresponding voxel of $\underline{\mathbf{\Phi}}$ generated by the screening approach, so that the set would provide the best possible approximation of that orientation. The details of algorithm can be found in Appendix E.4.

We demonstrate the screening method's performance using both error metrics in Figure 3. In Figure 3a, we show the effect of increasing the size of our candidate set of orientations on the number of missing orientations compared to the ground truth. GreedyOrientation's advantage is likely because OMP continually adds dissimilar orientations, thus is less likely to add the exactly correct orientations because these are too similar to orientations already in the candidate set. Figure 3b shows the minimum angular distance given a linear combination of orientations in the candidate set compared to the ground truth. GreedyOrientation has high probability mass near zero, showing that it generates appropriate candidate sets. Finally, Figure 3c shows that the angular distances between the orientations weighted with the optimized weights and ground truth for each iteration of optimization.

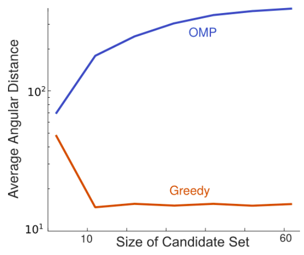
We additionally demonstrate the effects of each screen method on final reconstruction error after optimization. Figure 4a shows the distribution of reconstruction error over voxels. Starting the optimization with GreedyOrientation leads to much lower bias in the final optimization result than OMP, as demonstrated by the shift of these distributions away from the Ground Truth distribution. In Figure 4b, we show the reconstruction error on each step of optimization. The reconstruction



(a)

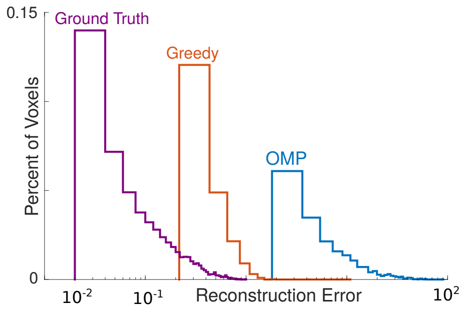


(b)

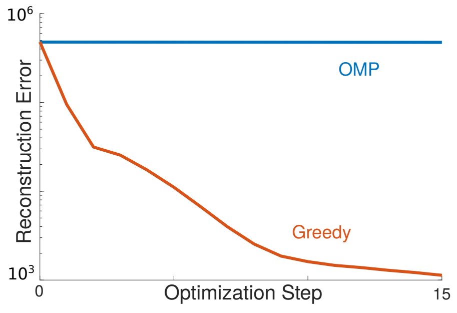


(c)

Figure 3: **(a):** Average number of missing orientations per voxel in candidate sets of increasing size. **(b):** The distribution of angular distances from the ground truth of OMP and GreedyOrientation after global optimization procedure. The angular distance is the minimum possible distance given some weighted combination of selected orientations. **(c):** Average angular distance between the weighted sum of predicted node orientations and the ground truth in each voxel for candidate sets of increasing size.



(a)



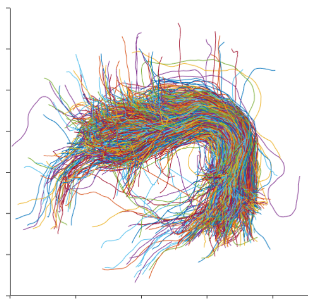
(b)

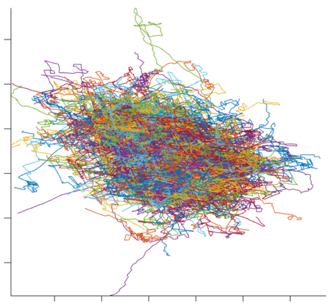
Figure 4: **(a):** Comparing the distribution of reconstruction error for ground truth, OMP, and GreedyOrientation over voxels after optimization. **(b):** The improvement of reconstruction error during the steps of gradient decent shows that the objective is not able to improve the OMP selected orientation sets while it is improving the GreedySelection choices constantly.

error when initialized with orientations generated by OMP is decreasing at a rate several orders of magnitude slower than GreedyOrientation.

5.2 Group Sparse Optimization

After $\underline{\mathbf{\Phi}}$ has been initialized with one of the locally greedy screening algorithms, we learn the appropriate weighting of $\underline{\mathbf{\Phi}}$ by optimizing the global objective. We applied batch gradient decent


(a) Ground truth


(b) OMP after optimization

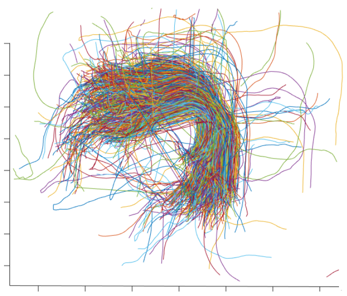
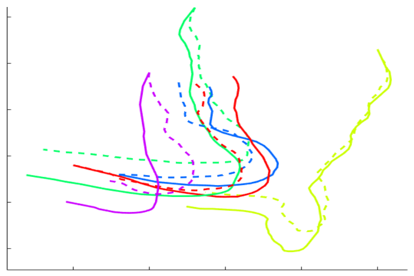
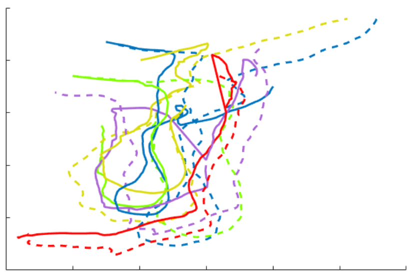

(c) Greedy after optimization

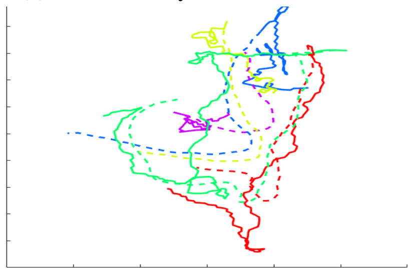
Figure 5: Solutions learned after the group sparse optimization for both screening strategies, compared to ground truth .



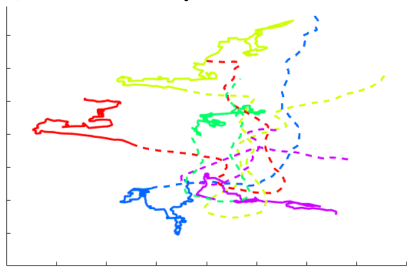
(a) Best 5 GreedyOrientation Fascicles



(b) Worst 5 GreedyOrientation Fascicles



(c) Best 5 OMP Fascicles



(d) Worst 5 OMP Fascicles

Figure 6: Top five best and worst fascicles for OMP and GreedyOrientation after optimization according to reconstruction error. Solid lines show the predicted $\underline{\mathbf{\Phi}}$ and dashed lines ground truth.

with 15 iterations and a dynamic step-size value which started from 1e-5 and decreased each time the algorithm could not improve the total objective error. The ℓ_1 and group regularizer coefficients were chosen to be 10 for most of the experiments. For ℓ_1 regularizer, we applied a proximal operator to truncate weights less than the threshold of 0.001. The derivation of the gradient and optimization procedure can be found in Appendices C.2 and E.2, respectively. The visualization algorithm, for a given $\underline{\mathbf{\Phi}}$, is given in Appendix E.3.

Figure 5 visualizes the results of $\underline{\mathbf{\Phi}}$ after optimization with both OMP and GreedyOrientation initialization strategies. Comparing the GreedyOrientation predicted $\underline{\mathbf{\Phi}}$ with expert $\underline{\mathbf{\Phi}}$ shows that the group regularizer performed well in regenerating macrostructure of the Arcuate. Figure 5b shows that the OMP initialization strategy for $\underline{\mathbf{\Phi}}$ is not appropriate for this setting, and prevents the global optimization procedure from generating the desired macrostructure.

To get a better sense for the generated fascicles, we illustrate the best and the worst fascicles for $\underline{\mathbf{\Phi}}$ initialized with GreedyOrientation and OMP in Figure 6. GreedyOrientation produces plausible fascicles in terms of orientation, in some cases seemingly even more so than the ground truth which was obtained with a tractography algorithm. In the best case, in Figure 6a the reconstruction is highly accurate. In the worst case, in Figure 6b (a), GreedyOrientation produces fascicles with sharply changing direction. Looking closer, the worst reconstructed fascicles tend to be long winding fascicles with abrupt direction changes. Because the objective attempts to minimize these features during optimization, these tracts are very difficult to reconstruct. Fascicles such as these are unlikely to occur in the brain, and are likely a result of imperfect tractography methods that were used for creating the ground truth data for this experiment. Solutions with OMP are generally poor.

6 Discussion

In this work, we considered the problem of learning macroscopic brain connectomes from dMRI data. This involves inferring locations and orientations of fascicles given local measurements of diffusion of water molecules within the white-matter tissue. We proposed a new way to formulate this learning problem, using a tensor encoding. Our proposed group sparse objective facilitates the use of optimization algorithms to automatically extract brain structure, without relying on expert tractography solutions. We proposed an efficient greedy screening algorithm for this objective, and proved approximation guarantees for the algorithm. We finally demonstrated that our specialized screening algorithm resulted in a much better orientations than a generic greedy subselection algorithm, called

OMP. The solutions with our group sparse objective, in conjunction with these selected orientations, resulted in smooth fascicles and low reconstruction error of the diffusion data. We also highlighted some failures of the solution, and that more needs to be done to get fully plausible solutions.

Our tractography learning formulation has the potential to open new avenues for learning-based approaches for obtaining brain connectomes. This preliminary work was necessarily limited, focused on providing a sound formulation and providing an initial empirical investigation into the efficacy of the approximations. The next step is to demonstrate the real utility of a full tractography solution using this formulation. This will involve learning solutions across brain datasets; understanding strengths and weaknesses compared to current tractography approaches; potentially incorporating new regularizers and algorithms; and even incorporating different types of data. All of this can build on the foundational idea introduced in this work: using a factorization encoding to automatically learn brain structure from data.

Acknowledgments

This research was funded by NSERC, Amii and CIFAR. F.P. was supported by NSF IIS-1636893, NSF BCS-1734853, NSF AOC 1916518, NIH NCATS UL1TR002529, a Microsoft Research Award, Google Cloud Platform, and the Indiana University Areas of Emergent Research initiative âĂIJLearning: Brains, Machines, Children.

References

[1] G Allen. Sparse higher-order principal components analysis. In *International Conference on Artificial Intelligence and Statistics*, 2012.

[2] P J Basser, S Pajevic, C Pierpaoli, J Duda, and A Aldroubi. In vivo fiber tractography using DT-MRI data. *Magnetic Resonance in Medicine*, 44(4):625–632, October 2000.

[3] Danielle S Bassett and Olaf Sporns. Network neuroscience. *Nature Neuroscience*, 20(3):353–364, February 2017.

[4] T E J Behrens, M W Woolrich, M Jenkinson, H Johansen-Berg, R G Nunes, S Clare, P M Matthews, J M Brady, and S M Smith. Characterization and propagation of uncertainty in diffusion-weighted MR imaging. *Magnetic resonance in medicine*, 2003.

[5] Itay Benou and Tammy Riklin-Raviv. Deeptract: A probabilistic deep learning framework for white matter fiber tractography. *arXiv preprint arXiv:1812.05129*, 2018.

[6] Antoine Bonnefoy, Valentin Emiya, Liva Ralaivola, and Rémi Gribonval. Dynamic Screening: Accelerating First-Order Algorithms for the Lasso and Group-Lasso. *IEEE Transactions on Signal Processing*, 2015.

[7] Cesar F Caiafa and Andrzej Cichocki. Computing sparse representations of multidimensional signals using Kronecker bases. *Neural Computation*, 2013.

[8] Cesar F Caiafa, Olaf Sporns, Andrew Saykin, and Franco Pestilli. Unified representation of tractography and diffusion-weighted mri data using sparse multidimensional arrays. In *Advances in neural information processing systems*, pages 4340–4351, 2017.

[9] Emmanuel Caruyer and Rachid Deriche. Diffusion MRI signal reconstruction with continuity constraint and optimal regularization. *Medical image analysis*, 2012.

[10] A Cichocki, R Zdunek, A H Phan, and S Amari. *Nonnegative Matrix and Tensor Factorizations: Applications to Exploratory Multi-way Data Analysis and Blind Source Separation*. Wiley, 2009.

[11] Andrzej Cichocki, Danilo P Mandic, Lieven De Lathauwer, Guoxu Zhou, Qibin Zhao, Cesar F Caiafa, and Anh Huy Phan. Tensor Decompositions for Signal Processing Applications: From two-way to multiway component analysis. *IEEE Signal Process. Mag. ()*, 2015.

[12] Alessandro Daducci, Muhamed Barakovic, Gabriel Girard, Maxime Descoteaux, and Jean-Philippe Thiran. Reducing false positives in tractography with microstructural and anatomical priors. Technical report, 2018.

[13] Abhimanyu Das and David Kempe. Submodular meets Spectral: Greedy Algorithms for Subset Selection, Sparse Approximation and Dictionary Selection. In *International Conference on Machine Learning*, 2011.

[14] TG Kolda and BW Bader. Tensor decompositions and applications. *SIAM Review*, 51(3):455–500, 2009.

[15] Morten Mørup, Lars Kai Hansen, and Sidse Marie Arnfred. Algorithms for Sparse Nonnegative Tucker Decompositions. *Neural Computation*, 2008.

[16] Peter F Neher, Michael Götz, Tobias Norajitra, Christian Weber, and Klaus H Maier-Hein. A machine learning based approach to fiber tractography using classifier voting. In *International Conference on Medical Image Computing and Computer-Assisted Intervention*, pages 45–52. Springer, 2015.

[17] Martin Ohlson, M Rauf Ahmad, and Dietrich von Rosen. The multilinear normal distribution: Introduction and some basic properties. *J. Multivariate Analysis ()*, 2013.

[18] Franco Pestilli, Jason D Yeatman, Ariel Rokem, Kendrick N Kay, and Brian A Wandell. Evaluation and statistical inference for human connectomes. *Nature Methods*, 11(10):1058–1063, September 2014.

[19] Franco Pestilli, Jason D Yeatman, Ariel Rokem, Kendrick N Kay, and Brian A Wandell. Evaluation and statistical inference for human connectomes. *Nature methods*, 11(10):1058, 2014.

[20] Philippe Poulin, Marc-Alexandre Cote, Jean-Christophe Houde, Laurent Petit, Peter F Neher, Klaus H Maier-Hein, Hugo Larochelle, and Maxime Descoteaux. Learn to track: Deep learning for tractography. In *International Conference on Medical Image Computing and Computer-Assisted Intervention*, pages 540–547. Springer, 2017.

[21] Ariel Rokem, Jason D Yeatman, Franco Pestilli, Kendrick N Kay, Aviv Mezer, Stefan van der Walt, and Brian A Wandell. Evaluating the accuracy of diffusion MRI models in white matter. *PLoS ONE*, 10(4):e0123272, April 2015.

[22] R Rubinstein, M Zibulevsky, and M Elad. Efficient implementation of the K-SVD algorithm using batch orthogonal matching pursuit. Technical report, 2008.

[23] Sara Soltani, Misha Elena Kilmer, and Per Christian Hansen. A Tensor-Based Dictionary Learning Approach to Tomographic Image Reconstruction. *CoRR abs/1506.04954*, 2015.

[24] Grzegorz Swirszcz, Naoki Abe, and Aurelie C Lozano. Grouped Orthogonal Matching Pursuit for Variable Selection and Prediction. *Advances in Neural Information Processing Systems*, 2009.

[25] J-Donald Tournier, Fernando Calamante, and Alan Connelly. MRtrix: Diffusion tractography in crossing fiber regions. *International Journal of Imaging Systems and Technology*, 22(1):53–66, February 2012.

[26] J-Donald Tournier, Fernando Calamante, David G Gadian, and Alan Connelly. Direct estimation of the fiber orientation density function from diffusion-weighted MRI data using spherical deconvolution. *NeuroImage*, 23(3):1176–1185, November 2004.

[27] Brian A Wandell. Clarifying Human White Matter. *Annual Review of Neuroscience*, 39(1):103–128, July 2016.

[28] Zhen James Xiang, Hao Xu, and Peter J Ramadge. Learning Sparse Representations of High Dimensional Data on Large Scale Dictionaries. *Advances in Neural Information Processing Systems*, 2011.

[29] Yangyang Xu and Wotao Yin. A block coordinate descent method for regularized multiconvex optimization with applications to nonnegative tensor factorization and completion. *SIAM J. Imaging Sciences*, 2013.

[30] Zemin Zhang and Shuchin Aeron. Denoising and Completion of 3D Data via Multidimensional Dictionary Learning. *CoRR abs/1202.6504*, 2015.

[31] Syed Zubair and Wenwu Wang. Tensor dictionary learning with sparse TUCKER decomposition. *DSP*, 2013.

A Maximum likelihood formulation

We first consider a maximum likelihood formulation for the reconstruction loss, paralleling the losses considered for non-matrix data. This approach involves making distributional assumptions on the matrix $\mathbf{Y}$; we begin with the standard normal, though this could be generalized to other distributions—as is commonly done for generalized linear models—without eschewing convexity. Using the matrix normal distribution [17], we can formulate the loss between $\mathbf{Y}$ and the factorization by parameterizing the matrix normal using the factorized variables. Assume $\mathbf{Y} \sim \mathcal{MN}(\mathbf{M}, \mathbf{U}, \mathbf{V})$ where $\mathbf{M}$ is the mean matrix, $\mathbf{U}$ is the row variance and $\mathbf{V}$ is the column variance. As a common simplification, we will take $\mathbf{U} = \sigma_u^2 \mathbf{I}$ and $\mathbf{V} = \sigma_v^2 \mathbf{I}$. Then the pdf of $\mathbf{Y}$ is

$$\frac{\exp\left(-1/2\text{tr}(\mathbf{V}^{-1}(\mathbf{Y}-\mathbf{M})^\top \mathbf{U}^{-1}(\mathbf{Y}-\mathbf{M}))\right)}{(2\pi)^{N_\theta N_v/2}|\mathbf{V}|^{N_\theta/2}|\mathbf{U}|^{N_v/2}} = \frac{\exp\left(-1/2\sigma_v\sigma_u\text{tr}((\mathbf{Y}-\mathbf{M})^\top(\mathbf{Y}-\mathbf{M}))\right)}{(2\pi)^{N_\theta N_v/2}\sigma_v^{N_\theta^2}\sigma_u^{N_v^2}}$$

because $|\mathbf{U}| = |\sigma_u^2 \mathbf{I}| = \sigma_u^{2N_v}$. This type of modeling approach assumes zero-mean, independent noise across entries in $\mathbf{Y}$, which is a common assumption [4]. Taking the negative of the log likelihood, and dropping constants which do not affect the minimum, we obtain the optimization

$$\underset{\mathbf{M}}{\operatorname{argmin}}\, \text{tr}((\mathbf{Y}-\mathbf{M})^\top(\mathbf{Y}-\mathbf{M})) = \underset{\mathbf{M}}{\operatorname{argmin}}\, \|\mathbf{Y}-\mathbf{M}\|_F^2$$

B Theoretical Properties of the Tractography Objective

An important property of this objective is that it is convex and has a unique solution for $\underline{\mathbf{\Phi}}$ (up to permutation), as we show in the below theorem. The convexity of the objective ensures that gradient descent can obtain optimal solutions, which is critical for both improving the objective and ensuring that accurate tractography solutions are extracted. The uniqueness is further important, because it provides an identifiable solution. For tractography, we would like to identify the fascicle structure for an individual; for an objective with multiple equivalent solutions, it is not clear which solution to select, and reflects an impreciseness in the objective. Any solution for $\underline{\mathbf{\Phi}}$ will always be equivalent, up to permutations of the fascicles (frontal slices, see Fig 2A - left), but should not change which fascicles are shared by which voxels.

Theorem 1. *The objective in* (5) *is convex in* $\underline{\mathbf{\Phi}}$. *Further, if the defined blocks* $\mathcal{A}$ *and* $\mathcal{V}$ *cover all possible orientations and voxels in the sense that every* v *is included in at least one group* $\mathcal{G}_\mathcal{V}$ *and every orientation* a *is included in at least one group* $\mathcal{G}_\mathcal{A}$, *then* (5) *has a unique solution (up to permutation).*

Proof. Because the sum of convex functions is convex, to show that (5) is convex, we simply need to show that each term in the objective is convex.

The first term $\|\mathbf{Y} - \underline{\mathbf{\Phi}} \times_1 \mathbf{D} \times_3 \mathbf{1}\|_F^2$ is convex in $\underline{\mathbf{\Phi}}$ because $\|\mathbf{Y}-\mathbf{M}\|_F^2$ is convex in $\mathbf{M}$, and $\mathbf{M} = \underline{\mathbf{\Phi}} \times_1 \mathbf{D} \times_3 \mathbf{1}$ is an affine transformation of $\underline{\mathbf{\Phi}}$. The composition of an affine function and a convex function is convex.

The second term is the sum of several functions of $\underline{\mathbf{\Phi}}$, which only consider subparts of $\underline{\mathbf{\Phi}}$. If each of these functions in the group regularizer is convex, then the regularizer is composed of the sum of convex functions and so is itself convex. Let $R_{\mathcal{G}_\mathcal{A},\mathcal{G}_\mathcal{V},f}(\underline{\mathbf{\Phi}}) = \|\mathbf{x}_{\mathcal{G}_\mathcal{A},\mathcal{G}_\mathcal{V},f}\|_2$. This function only changes when elements in $\underline{\mathbf{\Phi}}$ related to $\mathcal{G}_\mathcal{A}, \mathcal{G}_\mathcal{V}, f$ change, and is otherwise constant. However, since a constant function is convex, $R_{\mathcal{G}_\mathcal{A},\mathcal{G}_\mathcal{V},f}$ is convex in the entries of $\underline{\mathbf{\Phi}}$ that are ignored. Let $\underline{\mathbf{\Phi}}_{\mathcal{G}_\mathcal{A},\mathcal{G}_\mathcal{V},f}$ be the entries in $\underline{\mathbf{\Phi}}$ that give $\mathbf{x}_{\mathcal{G}_\mathcal{A},\mathcal{G}_\mathcal{V},f} = [\|\underline{\mathbf{\Phi}}_{\mathcal{G}_\mathcal{A},v_1,f}\|_1, \ldots, \|\underline{\mathbf{\Phi}}_{\mathcal{G}_\mathcal{A},v_n,f}\|_1]$ for $v_i \in \mathcal{G}_\mathcal{V}$. We can consider $\|\mathbf{x}_{\mathcal{G}_\mathcal{A},\mathcal{G}_\mathcal{V},f}\|_2$ as a vector composition, of $g : \mathbb{R}^k \to \mathbb{R}^n$ and $h : \mathbb{R}^n \to \mathbb{R}$, where $g(\underline{\mathbf{\Phi}}_{\mathcal{G}_\mathcal{A},\mathcal{G}_\mathcal{V},f}) = \mathbf{x}_{\mathcal{G}_\mathcal{A},\mathcal{G}_\mathcal{V},f}$ and $h(\mathbf{x}) = \|\mathbf{x}\|_2$. The resulting composition is $h(g(\underline{\mathbf{\Phi}}_{\mathcal{G}_\mathcal{A},\mathcal{G}_\mathcal{V},f})) = \|\mathbf{x}_{\mathcal{G}_\mathcal{A},\mathcal{G}_\mathcal{V},f}\|_2$. Each g_i of the vector-valued function g is convex in $\underline{\mathbf{\Phi}}_{\mathcal{G}_\mathcal{A},\mathcal{G}_\mathcal{V},f}$ because it applies an ℓ_1 norm—which is convex—on a subset of $\underline{\mathbf{\Phi}}_{\mathcal{G}_\mathcal{A},\mathcal{G}_\mathcal{V},f}$. Further, h is convex in $\mathbf{x}$, and non-decreasing in each $\mathbf{x}_i = g_i(\underline{\mathbf{\Phi}}_{\mathcal{G}_\mathcal{A},\mathcal{G}_\mathcal{V},f})$, because h is a norm. Therefore, the composition $h(g(\cdot))$ is convex. Therefore, because $R_{\mathcal{G}_\mathcal{A},\mathcal{G}_\mathcal{V},f}$ is convex w.r.t. $\underline{\mathbf{\Phi}}_{\mathcal{G}_\mathcal{A},\mathcal{G}_\mathcal{V},f}$ and constant w.r.t. all other values in $\underline{\mathbf{\Phi}}$, we know that $R_{\mathcal{G}_\mathcal{A},\mathcal{G}_\mathcal{V},f}$ is convex in $\underline{\mathbf{\Phi}}$. Since $R(\underline{\mathbf{\Phi}}) = \sum_{f\in\mathcal{F}}\sum_{\mathcal{G}_\mathcal{V}\in\mathcal{V}}\sum_{\mathcal{G}_\mathcal{A}\in\mathcal{A}} R_{\mathcal{G}_\mathcal{A},\mathcal{G}_\mathcal{V},f}(\underline{\mathbf{\Phi}})$ is a sum of convex function, it is convex.

To show uniqueness, we need to show that the regularizer is strictly convex. Because the sum of a strictly convex and convex function is again strictly convex, the resulting objective is itself strictly convex and so must have a unique minimum. Each component of the regularizer only considers a subset of $\underline{\mathbf{\Phi}}$; however, as long as each possible entry in $\underline{\mathbf{\Phi}}$ is considered at least once in one of these blocks, then that component of $\underline{\mathbf{\Phi}}$ has a strictly convex regularizer on it in the objective, because norms are strictly convex. $\square$

C Sparse tensor factorization algorithm

In this section, we develop an algorithm to optimize our extremely sparse, high-dimensional objective. A common strategy for sparse optimization problems is to first perform screening on the coefficients—which corresponds to all of $\underline{\mathbf{\Phi}}$ in this setting—to avoid modifying coefficients that will remain zero. A number of generalized screening approaches have been developed for general sparse problems, either with a static screening before the start of the optimization [] or with a dynamic screening that adjust the set of feasible coefficients during the optimization []. We propose a specialized static screening, where we first select a set of feasible orientations for each voxel. This static screening on the entries in $\underline{\mathbf{\Phi}}$ significantly reduces the cost per iteration of gradient descent and reduces the number of iterations. This significantly speeds up the optimization, without incurring much approximation error, because of the approximation guarantees of the static screening approach. We first highlight why standard matrix and tensor factorization algorithms are not suitable for this problem, and then derive our specialized solver.

C.1 Issues with using standard matrix or tensor factorization algorithms

A natural community to turn to for solutions to obtain $\underline{\mathbf{\Phi}}$ is tensor factorization. The goal in this work is to factorize a matrix $\mathbf{Y}$ in a tensor $\underline{\mathbf{\Phi}}$, for a given dictionary $\mathbf{D}$, such that $\mathbf{Y} = \underline{\mathbf{\Phi}} \times_1 \mathbf{D} \times_3 \mathbf{1}$. Much of the work in tensor factorization, however, has focused on decomposing tensors into a set of matrices, with a small core tensor with the Tucker decomposition focused on low rank tensor factorizations (see [10] for a thorough overview). A few of these works have examined how to obtain a sparse core tensor, but towards the aim of either enforcing uniqueness [15] or to obtain core tensors that are more efficient to store and use [31, 7, 23, 11]. There has been some work on factorizing large sparse tensors, for tensor-SVD [1, 30]; again, however, their goal is to factorize a sparse tensor, which differs from our goal to factorize a dense matrix into an extremely sparse tensor with a particular structure. The most closely related algorithm is derived for low-rank regularizers for non-negative tensor factorization [29], but it is not designed for large sparse tensors.

C.2 Computing the subgradient of the objective

Once these orientations are set per voxel, we can much more efficiently compute the gradient for the objective, because the sum over groups significantly reduces,

$$\sum_{\mathcal{G}_\mathcal{V} \in \mathcal{V}} \sum_{\mathcal{G}_\mathcal{A} \in \mathcal{A}} \|\mathbf{x}_{\mathcal{G}_\mathcal{A}, \mathcal{G}_\mathcal{V}, f}\|_2 = \sum_{\mathcal{G}_\mathcal{V} \in \mathcal{V}} \sum_{\mathcal{G}_\mathcal{A} \in \mathcal{A}(\mathcal{G}_\mathcal{V})} \|\mathbf{x}_{\mathcal{G}_\mathcal{A}, \mathcal{G}_\mathcal{V}, f}\|_2$$

where $\mathcal{A}(\mathcal{G}_\mathcal{V}) = \{\mathcal{G}_\mathcal{A} \in \mathcal{A} \mid \mathcal{G}_\mathcal{A} \cap S(v) \neq \emptyset, v \in \mathcal{G}_\mathcal{V}\}$. The set $\mathcal{A}(\mathcal{G}_\mathcal{V})$ only includes groups with orientations that are active for at least one voxel in $\mathcal{G}_\mathcal{V}$.

Let there be N_{G_a} atom groups and N_{G_v} voxel groups. We use $\mathbf{G}_A \in \{0,1\}^{N_a \times N_{G_a}}$ and $\mathbf{G}_V \in \{0,1\}^{N_v \times N_{G_v}}$ to denote if an atom or a voxel belongs to a group. Specifically, if $\mathbf{G}_A(a, g) = 1$, then atom a belongs to group g; otherwise it does not. The subderivative of the group regularizer w.r.t. $\underline{\mathbf{\Phi}}(a, v, f)$ is

$$\lambda_g \sum_{a_g}^{N_{G_a}} \mathbf{G}_A(a, a_g) \sum_{v_g}^{N_{G_v}} \mathbf{G}_V(v, v_g) \frac{\sum\limits_{a_i \in \mathcal{G}_{a_g}} |\underline{\mathbf{\Phi}}(a_i, v, f)|}{\underline{\mathbf{A}}(a_g, v_g, f)} \operatorname{sign}(\underline{\mathbf{\Phi}}(a, v, f))$$

where $\underline{\mathbf{A}} = \sqrt{\left(|\underline{\mathbf{\Phi}}| \times_1 \mathbf{G}_A^\top\right)^2 \times_2 \mathbf{G}_V^\top}$. Additionally, we include a standard ℓ_1 regularizer on all of $\underline{\mathbf{\Phi}}$ to further promote sparsity. The full subgradient of the objective w.r.t. $\underline{\mathbf{\Phi}}$ is

$$\begin{aligned}\nabla_{\underline{\mathbf{\Phi}}} =& \mathbf{D}^\top(\underline{\mathbf{\Phi}} \times_1 \mathbf{D} \times_3 \mathbf{1} - \mathbf{Y})\mathbf{1}^\top \\ &+ \lambda_g \left(\left\{ \left(|\underline{\mathbf{\Phi}}| \times_1 \mathbf{G}_A^\top\right) \circ \left(\frac{1}{\underline{\mathbf{A}}} \times_2 \mathbf{G}_V \right) \right\} \times_1 \mathbf{G}_A \right) \circ \operatorname{sign}(\underline{\mathbf{\Phi}}) \\ &+ \lambda_1 \operatorname{sign}(\underline{\mathbf{\Phi}})\end{aligned} \tag{6}$$

On each step, we step in the direction of the negative of the gradient, with a fixed stepsize which we set to $\eta = 1e-3$, until the objective improvement is below a threshold $1e-4$ or until a maximum number of iterations is reached. In addition to the initial screening, we can obtain some speed improvements on each step by only computing the gradient for currently active elements in $\underline{\mathbf{\Phi}}$. For any zeroed elements in $\underline{\mathbf{\Phi}}$, the gradient of the regularizer would be zero, as the regularizer prefers each element be zero.

C.3 Derivation of Forward Selection for Orientations

To derive an efficient forward selection for the orientations, we need to make each greedy step efficient. For a fixed voxel v, the greedy algorithm selects the atom a which most increases $\bar{g}$:

$$\max_{a \notin S} \bar{g}(S \cup \{a\}) = \max_{a \notin S} g(S \cup \{a\}) + \sum_{s \in S} g(\{s\}) + g(\{a\}).$$

To compute $g(\{a\})$, we can compute this upfront for each a and store it before doing the full greedy optimization. For the greedy optimization, the most naive solution would be to compute the full regression solution for each new subset $S \cup \{a\}$, to obtain $g(S \cup \{a\})$. Unfortunately, this brute-force approach is too expensive. Because of the structure of $\bar{g}$, however, we can take advantage of the solution on the previous step, to compute the solution on this step.

We provide the recursive update mechanism in Lemma 1. Let $\mathbf{y} \in \mathbb{R}^{N_\theta}$ be the diffusion information for one voxel. For given orientations S with $|S| = k$, let $\mathbf{D}_S \in \mathbb{R}^{N_\theta \times k}$ be the subset of columns in $\mathbf{D}$ corresponding to orientations in S. Using similar subscript notation, with $a \notin S$ a new atom not yet chosen in S, let

$$\begin{aligned}\mathbf{C}_S &= \mathbf{D}_S^\top \mathbf{D}_S \\ \mathbf{b}_S &= \mathbf{D}_S^\top \mathbf{y} \\ C_a &= \mathbf{D}_a^\top \mathbf{D}_a \\ b_a &= \mathbf{D}_a^\top \mathbf{y} \\ \mathbf{c}_{S,a} &= \mathbf{D}_S^\top \mathbf{D}_a\end{aligned}$$

The squared multiple correlation is

$$g(S) = \mathbf{b}_S^\top \mathbf{C}_S^{-1} \mathbf{b}_S$$

and $g(\{a\}) = C_a^{-1} b_a^2$. We provide the following lemma to obtain an efficient mechanism to compute $g(S \cup \{a\})$ for each a. These recursive updates are similar to the updates given by [22], for OMP.

Lemma 1. *Given $\mathbf{C}_S^{-1} \in \mathbb{R}^{k \times k}$, $\mathbf{b}_S$ and $g(S)$, for*

$$\begin{aligned}\mathbf{c} &= \mathbf{C}_S^{-1} \mathbf{c}_{S,a} \\ \nu &= (C_a - \mathbf{c}_{S,a}^\top \mathbf{c})^{-1}\end{aligned}$$

we get that

$$g(S \cup \{a\}) = g(S) + \nu(\mathbf{b}_S^\top \mathbf{c} - b_a)^2$$

Further

$$\bar{g}(S \cup \{a\}) = \bar{g}(S) + \nu(\mathbf{b}_S^\top \mathbf{c} - b_a)^2 + C_a^{-1} b_a^2$$

Proof. We know that $g(S \cup \{a\}) = \mathbf{b}_{S\cup\{a\}}^\top \mathbf{C}_{S\cup\{a\}}^{-1} \mathbf{b}_{S\cup\{a\}}$. We need to compute the inverse of $\mathbf{C}_{S\cup\{a\}}$ using the inverse of $\mathbf{C}_S$. We use the general block matrix inversion formula

$$\begin{aligned}
\mathbf{C}_{S\cup\{a\}}^{-1} &= \begin{bmatrix} \mathbf{C}_S & \mathbf{c}_{S,a} \\ \mathbf{c}_{S,a}^\top & C_a \end{bmatrix}^{-1} \\
&= \begin{bmatrix} \mathbf{C}_S^{-1} + \nu \mathbf{C}_S^{-1}\mathbf{c}_{S,a}\mathbf{c}_{S,a}^\top \mathbf{C}_S^{-1} & -\nu \mathbf{C}_S^{-1}\mathbf{c}_{S,a} \\ -\nu \mathbf{c}_{S,a}^\top \mathbf{C}_S^{-1} & \nu \end{bmatrix}
\end{aligned}$$

Therefore,

$$\begin{aligned}
g(S \cup \{a\}) &= \mathbf{b}_{S\cup\{a\}}^\top \mathbf{C}_{S\cup\{a\}}^{-1} \mathbf{b}_{S\cup\{a\}} \\
&= \mathbf{b}_S^\top \mathbf{C}_S^{-1} \mathbf{b}_S + \nu \mathbf{b}_S^\top \mathbf{C}_S^{-1} \mathbf{c}_{S,a}\mathbf{c}_{S,a}^\top \mathbf{C}_S^{-1} \mathbf{b}_S - 2\nu \mathbf{b}_S^\top \mathbf{C}_S^{-1}\mathbf{c}_{S,a} b_a + \nu b_a^2 \\
&= g(S) + \nu(\mathbf{b}_S^\top \mathbf{c})^2 - 2\nu b_a \mathbf{b}_S^\top \mathbf{c} + \nu b_a^2 \\
&= g(S) + \nu(\mathbf{b}_S^\top \mathbf{c} - \nu b_a)
\end{aligned}$$

Using this, we can see that

$$\begin{aligned}
\bar{g}(S \cup \{a\}) &= g(S \cup \{a\}) + \sum_{s \in S\cup\{a\}} g(\{s\}) \\
&= g(S) + \nu(\mathbf{b}_S^\top \mathbf{c} - \nu b_a)^2 + \sum_{s\in S} g(\{s\}) + g(\{a\}) \\
&= \bar{g}(S) + \nu(\mathbf{b}_S^\top \mathbf{c} - \nu b_a)^2 + g(\{a\})
\end{aligned}$$

completing the proof, because $g(\{a\}) = C_a^{-1} b_a^2$. □

Given this result, the computation of $g(S \cup \{a\})$ for one atom a costs $O(kN_\theta + k^2) = O(kN_\theta)$, since $N_\theta > k$. To compute g for each a, therefore, costs a total of $O(kN_\theta N_a)$. We summarize the greedy algorithm for computing the directions for a voxel in Algorithm 1. A new point is added to greedily maximize $\bar{g}(S)$, until S has k directions.

Algorithm 1 GreedyOrientation: greedy algorithm to select directions for each voxel

```
 1: Input dictionary $\mathbf{D}$, maximum number of directions $k$
 2: // Compute $g(\{a\})$ for each $a$, $\mathbf{g} = \mathrm{diag}(\mathbf{D}^\top\mathbf{D})^{-1}(\mathbf{D}^\top\mathbf{y})^2$
 3: $\mathbf{c} \leftarrow \mathbf{D}^\top\mathbf{D}$
 4: $\mathbf{b} \leftarrow \mathbf{D}^\top\mathbf{y}$
 5: $\mathbf{g} \leftarrow \mathbf{0}$
 6: $a_{\max} \leftarrow -1, g_{\max} \leftarrow 0$
 7: for $a = 1, \dots, N_a$ do                                          ▷ $O(N_\theta N_a)$
 8:     $\mathbf{g}(a) \leftarrow (\mathbf{b}(a))^2/\mathbf{c}(a,a)$
 9:     if $g_{\max} < \mathbf{g}(a)$ then
10:         $g_{\max} \leftarrow \mathbf{g}(a)$, $a_{\max} \leftarrow a$
11: $S \leftarrow a_{\max}$
12: $\mathbf{C}^{-1} \leftarrow 1/\mathbf{c}(a,a)$
13: for $i = 2, \dots, k$ do
14:     // Compute $g(S \cup \{a\})$ for every $a$
15:     $\mathbf{g}_S, \nu \leftarrow$ ComputeGain($S, \mathbf{C}^{-1}, \mathbf{c}, \mathbf{b}$)                ▷ $O(kN_\theta)$
16:     $\bar{\mathbf{g}} \leftarrow \mathbf{g}_S + \mathbf{g}$                                        ▷ $\bar{\mathbf{g}} \in \mathbb{R}_a^N$
17:     $a_{\max} \leftarrow \mathrm{argmax}_{a \notin S}\, \bar{\mathbf{g}}(a)$                          ▷ $O(N_a)$
18:     $\mathbf{C}^{-1} \leftarrow \begin{bmatrix} \mathbf{C}^{-1} + \nu(a_{\max})\mathbf{C}^{-1}\mathbf{c}(S,a_{\max})\mathbf{c}(S,a_{\max})^\top \mathbf{C}^{-1} & -\nu(a_{\max})\mathbf{C}^{-1}\mathbf{c}(S,a_{\max}) \\ -\nu(a_{\max})\mathbf{c}_{S,a}^\top \mathbf{C}_S^{-1} & \nu(a_{\max}) \end{bmatrix}$
19:     $S \leftarrow S \cup \{a_{\max}\}$
```

D Theoretical guarantees of the greedy screening strategy

We can obtain approximation guarantees from the fact that the approximately submodular function for GreedyDictionarySelection has at least as good a submodularity ratio as the typical forward selection

Algorithm 2 ComputeGain($S, \mathbf{C}^{-1}, \mathbf{c}, \mathbf{b}$)

```
1: for $a = 1, \ldots, N_a$ do
2:     $\tilde{\mathbf{c}} \leftarrow \mathbf{C}^{-1}\mathbf{c}(S, a)$
3:     $\nu(a) \leftarrow (\mathbf{c}(a, a) - \mathbf{c}(S, a)^\top \tilde{\mathbf{c}})^{-1}$
4:     $\mathbf{g}_S(a) \leftarrow \nu(a)(\mathbf{b}(S)^\top \tilde{\mathbf{c}} - \mathbf{b}(a))^2$
```

function g. The submodularity ratio is defined as

$$\gamma(g) \overset{\text{def}}{=} \min_{S,L \in \mathcal{S}, S \cap L = \emptyset} \frac{\sum_{y \in S} g(L \cup \{y\}) - g(L)}{g(L \cup S) - g(L)} \tag{7}$$

for non-negative functions $g : \mathcal{P}(\mathcal{S}) \to \mathbb{R}^+$. If g is a monotone function and $\gamma(g) \geq 1$, then g is submodular. Otherwise, for $\gamma(g) < 1$, the function is not submodular and is instead called approximately submodular for $\gamma(g)$ close to 1. The closer $\gamma(g)$ is to 1, the better the approximation guarantees of greedy algorithms on these functions, with the best approximation guarantees for $\gamma(g) \geq 1$.

In the following theorem, we show that our GreedyDictionarySelection function $\bar{g}$ has a submodularity ratio that is no worse than ForwardSelection. The proof highlights that in fact the ratio is likely strictly better.

Theorem 2. *For $g : \mathcal{P}(\mathcal{S}) \to \mathbb{R}^+$ a monotone function, and*

$$\bar{g}(S) \overset{\text{def}}{=} g(S) + \sum_{s \in S} g(\{s\})$$

then

$$\gamma(\bar{g}) \geq \gamma(g).$$

Proof. For clarity, we introduce notation for the numerator and denominator of the submodularity ratio:

$$\gamma_N(g, L, S) \overset{\text{def}}{=} \sum_{y \in S} g(L \cup \{y\}) - g(L)$$

$$\gamma_D(g, L, S) \overset{\text{def}}{=} g(L \cup S) - g(L)$$

Notice that

$$\bar{g}(L) = g(L) + \sum_{x \in L} g(\{x\})$$

$$\bar{g}(L \cup \{y\}) = g(L) + \sum_{x \in L} g(\{x\}) + g(\{y\})$$

$$\bar{g}(L \cup S) = g(L \cup S) + \sum_{x \in L} g(\{x\}) + \sum_{y \in S} g(\{y\})$$

giving

$$\begin{aligned} \gamma_N(\bar{g}, L, S) &= \sum_{y \in S} \bar{g}(L \cup \{y\}) - \bar{g}(L) \\ &= \sum_{y \in S} g(L \cup \{y\}) + g(\{y\}) - g(L) \\ &= \gamma_N(g, L, S) + \sum_{y \in S} g(\{y\}) \end{aligned}$$

and

$$\begin{aligned} \gamma_D(\bar{g}, L, S) &= \bar{g}(L \cup S) - \bar{g}(L) \\ &= g(L \cup S) + \sum_{y \in S} g(\{y\}) - g(L) \\ &= \gamma_D(g, L, S) + \sum_{y \in S} g(\{y\}). \end{aligned}$$

If we let $a_S \overset{\text{def}}{=} \sum_{y \in S} g(\{y\}) \geq 0$, then we get that

$$\begin{aligned}
\gamma(\bar{g}) &= \min_{S,L \in \mathcal{S}, S \cap L = \emptyset} \frac{\gamma_N(\bar{g}, L, S)}{\gamma_D(\bar{g}, L, S)} \\
&= \min_{S,L \in \mathcal{S}, S \cap L = \emptyset} \frac{\gamma_N(g, L, S) + a_S}{\gamma_D(g, L, S) + a_S} \\
&\geq \min_{S,L \in \mathcal{S}, S \cap L = \emptyset} \frac{\gamma_N(g, L, S)}{\gamma_D(g, L, S)} && \triangleright \text{ because } a_S > 0 \\
&= \gamma(g)
\end{aligned}$$

completing the proof.

□

The following result now easily follows, from Theorem 4.2 [13] for approximately submodular functions.

Corollary 1. *The set of orientations S chosen by GreedyDictionarySelection satisfies*

$$\bar{g}(S) \geq \left(1 - e^{-\gamma(\bar{g})}\right) \mathit{OPT}$$

where OPT $= \bar{g}(S^*)$ for the optimal selection S^* such that $|S^*| = k$.

E Experiments and Results

E.1 Generating data

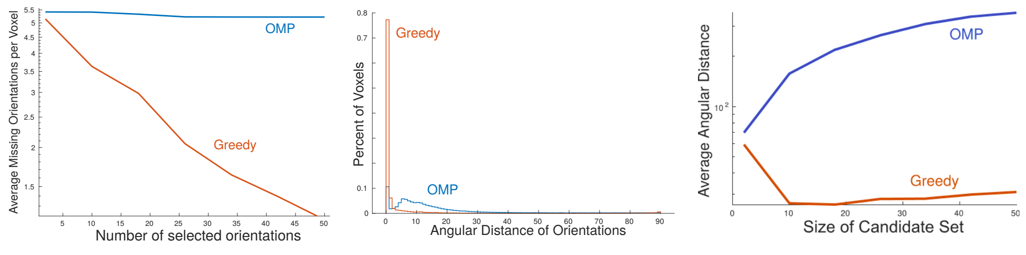
We used one subject from the public Stanford dataset [19] to fit the ENCODE model [8]. [2] The ENCODE model generated the predicted signal using the two major structures that we consider in this paper, the Arcuate Fasciculus and the ARC-SLF; which is the Arcuate combined with SLF1.

[2]The dataset can be downloaded from https://brainlife.io

E.2 Optimization algorithm for mapping connectomes of the brain

Algorithm 3 Brain: Mapping Brain Connectomes

```
Input: dMRI signal $\mathbf{Y}$, expert three dimensional sparse tensor $\underline{\mathbf{\Phi}}_{\boldsymbol{e}}$, dictionary $\mathbf{D}$, weights or fascicles' contribution $\mathbf{w}$, voxels vicinity $\mathbf{Vv}$ matrix (group information $\mathcal{G}_{\mathcal{V}}$) and atoms vicinity $\mathbf{Av}$ matrix (group information $\mathcal{G}_{\mathcal{A}}$), maximum iteration max_iter, step size step_size, termination condition tolerance
Ensure: $\|\mathbf{Y} - \underline{\mathbf{\Phi}} \times_1 \mathbf{D} \times_3 \mathbf{1}\|_F^2 + \lambda \sum_{f\in\mathcal{F}} \sum_{\mathcal{G}_{\mathcal{V}}\in\mathcal{V}} \sum_{\mathcal{G}_{\mathcal{A}}\in\mathcal{A}} \|\underline{\mathbf{\Phi}}_{\mathcal{G}_{\mathcal{A}},\mathcal{G}_{\mathcal{V}},f}\|_2$ is minimum
 1: for $f = 1, \ldots, N_f$ do
 2:     $\underline{\mathbf{\Phi}}_{\boldsymbol{e}}(:,:,f) = \underline{\mathbf{\Phi}}_{\boldsymbol{e}}(:,:,f) * w(f)$    ▷ Fold $\mathbf{w}$ in $\underline{\mathbf{\Phi}}_{\boldsymbol{e}}$
 3: voxels ← non-zero($\underline{\mathbf{\Phi}}_{\boldsymbol{e}}$, 2) ▷ Find necessary voxels from $\underline{\mathbf{\Phi}}_{\boldsymbol{e}}$ (Non-zero elements after summing up the other two dimensions)
 4: atoms ← non-zero($\underline{\mathbf{\Phi}}_{\boldsymbol{e}}$, 1) ▷ Find necessary atoms(orientations) from $\underline{\mathbf{\Phi}}_{\boldsymbol{e}}$ (Non-zero elements after summing up the other two dimensions)
 5: $\mathbf{Y}$ = $\mathbf{Y}$(:, voxels)    ▷ Remove unnecessary voxels of $\mathbf{Y}$
 6: fascicles ← fascicles(non-zero(w)) ▷ Remove unnecessary fascicles where contribution (weight) is 0
 7: $N_a, N_v, N_f$ ← size(atoms), size(voxels), size(fascicles)
 8: for $v = 1, \ldots, N_v$ do
 9:     $\mathbf{aA}$ ← GreedyOrientation($\mathbf{D}$, $k$)    ▷ Find indices of active atoms $\mathbf{aF}$ with Algorithm 1
10:     $\mathbf{aF}$ ← non-zero($\underline{\mathbf{\Phi}}_{\boldsymbol{e}}$, 3)    ▷ Find indices of active fascicles $\mathbf{aF}$ from $\underline{\mathbf{\Phi}}_{\boldsymbol{e}}$
11:     $\underline{\mathbf{\Phi}}(\mathbf{aA}, v, \mathbf{aF}) \leftarrow Initialization()$    ▷ Initialize $\underline{\mathbf{\Phi}}$ with non-zero values where atoms and fascicles are active
12: $\mathbf{G}_V$ ← find($\mathbf{Vv}$)   ▷ $\mathbf{G}_V(i,j) \in \{0,1\}$ Denotes that if voxel i is in the neighborhood of voxel j
13: $\mathbf{G}_A$ ← find($\mathbf{Av}$)   ▷ $\mathbf{G}_A(i,j) \in \{0,1\}$ Denotes that if atom i is in the neighborhood of atom j
14: $\mathbf{Emask} \leftarrow (\underline{\mathbf{\Phi}} \times_2 \mathbf{G}_V) \times_1 \mathbf{G}_A$    ▷ Entry mask tensor
15: $\mathbf{Fmask} \leftarrow \mathbf{Emask} \times_1 \mathbf{1}$    ▷ Fascicles Mask matrix
16: $\mathbf{Amask} \leftarrow \mathbf{Emask} \times_3 \mathbf{1}$    ▷ Atoms Mask matrix
17: $\mathbf{Fscreen} \leftarrow \underline{\mathbf{\Phi}} \times_1 \mathbf{1}$    ▷ Fascicles Screen matrix. Unlike $\mathbf{Fmask}$, this screen matrix does not contain group information
18: $\mathbf{Ascreen} \leftarrow \underline{\mathbf{\Phi}} \times_3 \mathbf{1}$    ▷ Atoms Screen matrix. Unlike $\mathbf{Amask}$, this screen matrix does not contain group information
19: $\mathbf{Y}_{\mathbf{diff}} \leftarrow \mathbf{Y} - \underline{\mathbf{\Phi}} \times_1 \mathbf{D} \times_3 \mathbf{1}$
20: $R(\underline{\mathbf{\Phi}}) \leftarrow \sum_{f\in\mathcal{F}} \sum_{\mathcal{G}_{\mathcal{V}}\in\mathcal{V}} \sum_{\mathcal{G}_{\mathcal{A}}\in\mathcal{A}} \|\underline{\mathbf{\Phi}}_{\mathcal{G}_{\mathcal{A}},\mathcal{G}_{\mathcal{V}},f}\|_2$
21: lnew ← $\|\mathbf{Y}_{\mathbf{diff}}\|_F^2 + \lambda R(\underline{\mathbf{\Phi}})$
22: niter ← 1
23: repeat
24:     lold ← lnew
25:     grad_p1_x ← $\mathbf{D}^\top(\underline{\mathbf{\Phi}} \times_1 \mathbf{D} \times_3 \mathbf{1} - \mathbf{Y})\mathbf{1}^\top$
26:     grad_g1_x1 ← $|\underline{\mathbf{\Phi}}| \times_1 \mathbf{G}_A^\top$    ▷ O(number of nonzero elements in $\underline{\mathbf{\Phi}}\times$ number of nonzero elements in $\mathbf{G}_A$)
27:     $\underline{\mathbf{A}} \leftarrow \sqrt{\text{grad\_g1\_x1}^2 \times_2 \mathbf{G}_V^\top}$
28:     grad_g1_x3 ← $\frac{1}{\underline{\mathbf{A}}} \times_2 \mathbf{G}_V$    ▷ O($N_v\times$ number of nonzero elements in $\underline{\mathbf{A}}$)
29:     grad_g1_x4 ← grad_g1_x3 $\circ$ grad_g1_x1
30:     grad_g1_v ← (grad_g1_x4 $\times_1 \mathbf{G}_A$) $\circ$ sign($\underline{\mathbf{\Phi}}$)
31:     g ← grad_p1_x +$\lambda_g$ grad_g1_v + $\lambda_1$ sign($\underline{\mathbf{\Phi}}$)
32:     Mask or Screen elements in g with $\mathbf{Fmask}$, $\mathbf{Amask}$ or $\mathbf{Fscreen}$, $\mathbf{Ascreen}$
33:     $\underline{\mathbf{\Phi}} \leftarrow \underline{\mathbf{\Phi}}$ − step_size $* g$
34:     $\mathbf{Y}_{\mathbf{diff}} \leftarrow \mathbf{Y} - \underline{\mathbf{\Phi}} \times_1 \mathbf{D} \times_3 \mathbf{1}$
35:     $R(\underline{\mathbf{\Phi}}) \leftarrow \sum_{f\in\mathcal{F}} \sum_{\mathcal{G}_{\mathcal{V}}\in\mathcal{V}} \sum_{\mathcal{G}_{\mathcal{A}}\in\mathcal{A}} \|\underline{\mathbf{\Phi}}_{\mathcal{G}_{\mathcal{A}},\mathcal{G}_{\mathcal{V}},f}\|_2$
36:     lnew ← $\|\mathbf{Y}_{\mathbf{diff}}\|_F^2 + \lambda R(\underline{\mathbf{\Phi}})$
37:     niter ← niter + 1
38: until lold - lnew < lold * tolerance || niter > max_iter
39: set small values in $\underline{\mathbf{\Phi}}$ zeros
40: Output: $\underline{\mathbf{\Phi}}$
```



(a) (b) (c)

Figure 7: Results using the larger **ARC-SLF** datset. **(a)**: Number of missing orientations in the candidate set generated by each screening algorithm, averaged over all voxels. **(b)**: Anuglar distance between orientations in the candidate set and the ground truth by voxel. **(c)**: Average of the possible minimum angular distance per voxel given some linear combination of orientations in candidate sets versus the ground truth.

E.3 Visualization algorithm

This section explains the visualization algorithm of the brain connectomes based on the sparse tensor $\underline{\mathbf{\Phi}}$. Each entry of this sparse tensor represents one node, each having an orientation, being located in a voxel, and belonging to a fascicle. There is no structure in Phi to indicate what order to put nodes in. The nodes do not contain any positional information of the space and this leads to an ambiguity in the accuracy of their order. The only positional information in hand is the coordinates of the voxels. Therefore, displaying an accurate $\underline{\mathbf{\Phi}}$ is itself a challenging problem due to many possible permutations of nodes in a voxel for each fascicle.

Our goal is to go over fascicles one by one and try to plot each at a time. We approach this by selecting one voxel that the fascicle passes through and has the fewest number of neighbouring voxels which also containing the same fascicle. A voxel with these properties should be at one end of the fascicle. Then the algorithm examines all surrounding voxels and chooses the pair of nodes with the smallest angular distance between them with one in the first voxel and one in the neighbours. We plot the nodes in the first voxel so that the last node in that voxel is the one chosen. Then we move forward through each voxel plotting first the node chosen in the previous pairing followed by the rest of the nodes in the voxel greedily chosen based on angular distance from the last plotted node.

E.4 Angular distance evaluation measurement

The goal here is to measure a more precise metric for the angular differences of the nodes in $\underline{\mathbf{\Phi}}$-predict and $\underline{\mathbf{\Phi}}$-expert. It is not a trivial task to measure this metric since a more precise measurement requires finding a one-by-one relationship between the nodes in $\underline{\mathbf{\Phi}}$-predict and $\underline{\mathbf{\Phi}}$-expert.

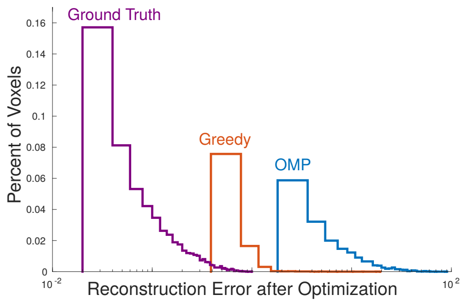
E.5 Evaluation results on ARC-SLF

Algorithm 4 Visualize $\mathbf{\Phi}$, the structure of connectomes

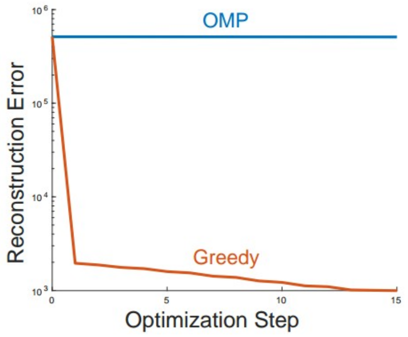
```
Input: Any three dimensional sparse tensor $\mathbf{\Phi}$, matrix $\mathbf{A}$ to map the indices of atoms in $\mathbf{\Phi}$ to the
    Cartesian components of the direction vector of that atom, and matrix $\mathbf{V}$ to map the indices of
    voxels in $\mathbf{\Phi}$ to the Cartesian coordinates of that voxel
 1: for $f = 1, \dots, N_f$ do
 2:     seen_v ← $\emptyset$      ▷ Initialize set to keep track of which voxels have been visited for fascicle $f$
 3:     $\mathbf{v}$ ← get_voxels($\mathbf{\Phi}$, $f$)      ▷ Get all voxels that fascicle $f$ passes through
 4:     vc ← GetFirstVoxel(seen_v, $\mathbf{v}$)      ▷ Select a voxel that has not been seen for fascicle $f$
 5:     seen_v ← seen_v $\cup$ {vc}
 6:     an ← PlotFirstVoxelNodes(vc, seen_v)  ▷ Plot the nodes of the current voxel and return the
    next voxel
 7:     repeat
 8:         an ← PlotNodes(an, seen_v)
 9:     until an is null
10: procedure PLOTFIRSTVOXELNODES(vc, seen_v)
11:     anset ← AllNeighbouringNodes(vc, seen_v)      ▷ Get all active nodes in the neighbouring
    active to vc
12:     $\mathbf{acset}$ ← non-zero($\mathbf{\Phi}$(:,f, vc))      ▷ Get all active nodes in the current voxel
13:     ac, an ← $\text{argmin}_{\text{an}' \in \mathbf{anset}, \text{ac}' \in \mathbf{acset}}$(AngularDistance(an′, ac′))      ▷ Finds the closest nodes
    between $\mathbf{acset}$ and $\mathbf{anset}$
14:     astack ← empty stack
15:     while $\mathbf{acset} \neq \emptyset$ do
16:         push ac onto astack
17:         $\mathbf{acset}$ ← $\mathbf{acset}$ − ac      ▷ Remove ac from $\mathbf{acset}$
18:         ac ← $\text{argmin}_{\text{ac}' \in \mathbf{acset}}$(AngularDistance(ac, ac′))      ▷ Find the next closest node to the
    previous node in the current voxel
19:     while astack not empty do
20:         ac ← pop astack
21:         Plot(ac)      ▷ Pop the nodes from astack and plot them
        Output: an, The closest node to the last plotted node
22: procedure PLOTNODES(ac, seen_v)
23:     vc ← voxel containing ac
24:     $\mathbf{acset}$ ← non-zero($\mathbf{\Phi}$(:,f, vc))      ▷ Get all active nodes in the current voxel
25:     while $\mathbf{acset} \neq \emptyset$ do
26:         $\mathbf{acset}$ ← $\mathbf{acset}$ − ac      ▷ Remove ac from $\mathbf{acset}$
27:         Plot(ac)      ▷ Plot the current node
28:         ac ← $\text{argmin}_{\text{ac}' \in \mathbf{acset}}$(AngularDistance(ac, ac′))      ▷ Find the next closest node to the
    previous node in the current voxel
29:     anset ← AllNeighbouringNodes(vc, seen_v)      ▷ Get all active nodes in the neighbouring
    active to vc
30:     an ← $\text{argmin}_{\text{an}' \in \mathbf{anset}}$(AngularDistance(an′, ac))  ▷ Finds the closest node in $\mathbf{anset}$ to the
    last plotted node Output: an, The closest node to the last plotted node
```

Algorithm 5 Minimum angular distance metric

```
Input: Two three dimensional sparse tensors $\mathbf{\Phi}_{\text{exp}}$ and $\mathbf{\Phi}_{\text{pred}}$, the expert and predicted brain structures.
1:  total_dist $\leftarrow 0$
2:  for $v_p = 1, \dots, N_v$ do
3:      dist_per_voxel $\leftarrow 0$
4:      ap_set $\leftarrow \mathcal{P}(\{a | a \in \mathbf{\Phi}_{\text{pred}}(:, v_p, :)\})$ ▷ The power set of all orientations active in the current voxel
5:      for all $a_{\text{exp}} \in \mathbf{\Phi}_{\text{exp}}(:, v_p, :)$ do
6:          vec_exp $\leftarrow a_{\text{exp}} \times \sum_{f_i \in \mathbf{\Phi}_{\text{exp}}(a_{\text{exp}}, v_p, :)} \mathbf{\Phi}_{\text{exp}}(a_{\text{exp}}, v_p, f_i)$
7:          min_dist $\leftarrow \infty$
8:          for all $s \in$ ap_set do
9:              vec_pred $\leftarrow \sum_{a_i \in s} a_i \times \sum_{f_i \in \mathbf{\Phi}_{\text{pred}}(a_i, v_p, :)} \mathbf{\Phi}_{\text{pred}}(a_i, v_p, f_i)$
10:             distance $\leftarrow$ Angular_Distance(vec_exp, vec_pred)
11:             if distance < min_dist then
12:                 min_dist $\leftarrow$ distance
13:         dist_per_voxel $\leftarrow$ dist_per_voxel + min_dist
14:     total_dist $\leftarrow$ total_dist + dist_per_voxel
Output: total_dist, The total angular distance
```

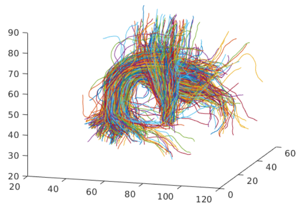


(a)

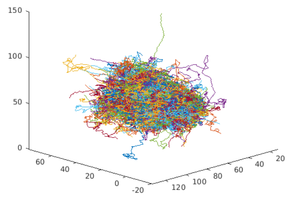


(b)

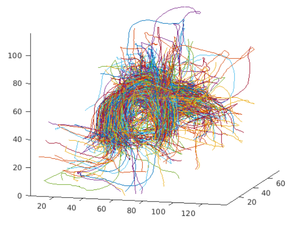
Figure 8: **(a):** Distribution of reconstruction error over voxels for each initialization strategy for orientations on the **ARC-SLF** dataset. **(b):** The improvement of reconstruction error during the steps of gradient decent shows that the objective is not able to improve the OMP selected orientation sets while it is improving the GreedySelection choices constantly.



(a) Ground truth

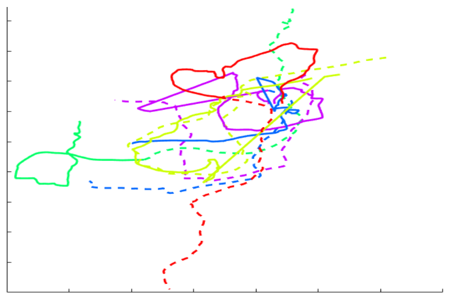


(b) OMP

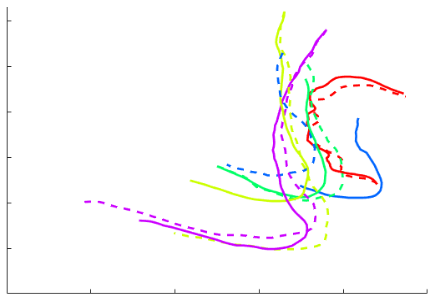


(c) Greedy

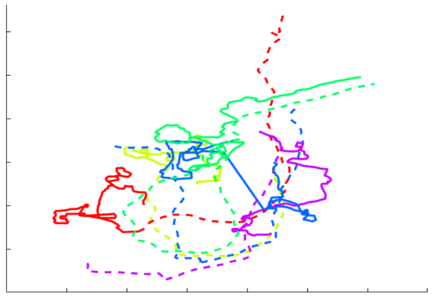
Figure 9: The quality of orientation sets selected in screening stage comparing to the ground truth orientations on the **ARC-SLF** dataset. **(a)**: Initializing $\underline{\mathbf{\Phi}}$ with expert $\underline{\mathbf{\Phi}}$, **(b)**: Initializing $\underline{\mathbf{\Phi}}$ with OMP, **(c)**: Initializing $\underline{\mathbf{\Phi}}$ with GreedyOrientation.



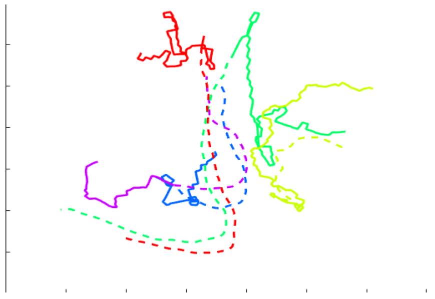
(a) Worst-GreedyOrientation-initialization



(b) best-GreedyOrientation-initialization



(c) Worst-OMP-initialization



(d) best-OMP-initialization

Figure 10: Top five best and worst fascicles for OMP and GreedyOrientation after optimization according to reconstruction error. Solid lines show the predicted $\underline{\mathbf{\Phi}}$ while dashed lines are the ground truth. The predicted $\underline{\mathbf{\Phi}}$ are of a different scale than the ground truth, making direct comparison difficult; however, the structure and shape of the fascicles in **b** clearly align closely with their ground truth counter parts.