V–Bundles: Clustering Fiber Trajectories from Diffusion MRI in Linear Time

Andre Reichenbach[1], Mathias Goldau[1],
Christian Heine[2], and Mario Hlawitschka[3]

[1] Image and Signal Processing Group, Computer Sience Institute, Leipzig University, Germany [reichenbach,math]@informatik.uni-leipzig.de
[2] Visual Computing Group, Department of Computer Science, TU Chemnitz, Germany christian.heine@informatik.tu-chemnitz.de
[3] Scientific Visualization Group, Computer Sience Institute, Leipzig University, Germany hlawitschka@informatik.uni-leipzig.de

Abstract. Fiber clustering algorithms are employed to find patterns in the structural connections of the human brain as traced by tractography algorithms. Current clustering algorithms often require the calculation of large similarity matrices and thus do not scale well for datasets beyond 100,000 streamlines. We extended and adapted the 2D vector field k–means algorithm of Ferreira et al. to find bundles in 3D tractography data from diffusion MRI (dMRI) data. The resulting algorithm is linear in the number of line segments in the fiber data and can cluster large datasets without the use of random sampling or complex multipass procedures. It copes with interrupted streamlines and allows multisubject comparisons.

Keywords: diffusion MRI · fiber clustering · vector field

1 Introduction

Tractography allows estimating the physical paths of neuronal connections in the human brain from diffusion–weighted magnetic resonance imaging (dMRI) data. Existing methods typically generate up to millions of streamlines for a single subject and then group them into bundles. These bundles represent a macro–scale wiring scheme of the brain and thus play a big role in generating atlases of the human white matter. They are also of general interest to neuroscience, in particular the study how structural as well as functional bundle characteristics relate to brain development, aging, and diseases.

To find fiber bundles, many popular methods use similarity measures and employ clustering, e.g., hierarchical clustering [4, 9]. This typically requires the computation and storage of pairwise similarities, which grow quadratically in the number of streamlines and thus quickly limit the dataset sizes that can be processed. General strategies to reduce this complexity include random sampling [5, 7], culling [10], or preclustering using a faster algorithm. Methods using similarity measures often require the data to be resampled to a fixed number of points or fixed segment length, and thus may loose information. Most

similarity measures are not very resistent to interrupted streamlines typically arising from fiber tracers. Among other subquadratic algorithms are greedy approaches (e.g. [3]), procedures based on stochastic processes (e.g. [8, 9]), and multiprocedure schemes including voxel–based clustering such as [4]. In contrast to these predominantly data–driven approaches, which are important for inferring structural connectivity without bias, there also are approaches that incorporate anatomical priors, as well as those working directly on the diffusion model omitting the tractography; a thorough review can be found in [6]. Although anatomical priors are designed to help to identify plausible white matter tracts, only little is known on white matter variability, hence essential subject differences may not be regarded.

Determining correspondence between identified clusters among subjects or image acquisitions allows population–averaged atlases or comparing diffusion indices along bundles. In prior work, this has been attempted by means of clustering subjects in unison in a common space [5], by bundle similarity measures [9,10], by using priors learned from training datasets [8,9], or by clustering an inter–subject similarity matrix [1].

In this paper, we adapt the vector field k–means algorithm by Ferreira et al. [2]. It is designed to find movement patterns in 2D directed trajectories by learning "latent" vector fields; we extend it to 3D and undirected streamlines. The result has a runtime complexity linear in the number of input points and naturally copes with interrupted streamlines. The learned vector fields can be used to classify new data or subjects.

2 Methods and Material

We first provide an overview of the vector k–means algorithm by Ferreira et al. [2] and then discuss our changes to make it applicable to streamline data.

2.1 Vector Field k–Means

The input to the algorithm is a set of 2D trajectories, given as lists of points and time stamps. The output is each trajectory's cluster and optionally the regular rectilinear vector fields which serve as cluster descriptors and can be used to classify trajectories not in the training set. The algorithm has three parameters: the vector field resolution R_v, i.e., the number of grid points along each axis, the number of clusters k, and the regularization strength λ to avoid overfitting. The algorithm first splits each trajectory i into a list of line segments $s_1^{(i)}, \ldots, s_{m^i}^{(i)}$, each lying in exactly one grid cell and having its own duration $w_l^{(i)}$.

The algorithm then proceeds iteratively, alternating between assigning the best vector field to each trajectory and fitting each vector field to its currently assigned trajectories. The vector fields $V^{(1)}, \ldots, V^{(k)}$ are represented as $R_v^2 \times 2$ matrices, each line coding the x, y velocity at one grid position. To find the best vector field for a given trajectory the algorithm computes, in terms of squared

error, the average match of its line segments with each vector field as follows:

$$e(S^{(i)}, V^{(j)}) = \sum_{l=1}^{m^{(i)}} w_l^{(i)} (C_l^{(i)} V^{(j)} - B_l^{(i)})^2 \ , \tag{1}$$

where $C_l^{(i)}$ and $B_l^{(i)}$ are suitably constructed matrices for each segment that take within–cell interpolation into account. We refer to Ferreira et al. [2] for details.

To fit a vector field $V^{(j)}$ to its currently assigned trajectories, the algorithm computes the coefficients of $V^{(j)}$ that minimize:

$$c(V^{(j)}) = \lambda \|LV\|^2 + \frac{1-\lambda}{W} \sum_{i \in \{i | \phi_i = j\}} e(S^{(i)}, V^{(j)}) \ , \tag{2}$$

where W is the total time length over all line segments and L denotes the Laplacian of the grid. Its purpose is to prefer smooth vector fields. The algorithm is initialized by fitting each cluster to a single trajectory in turn. The first cluster is fitted to a random trajectory and each next cluster is fitted to the trajectory maximizing $e(S^{(i)}, V^{(j)})$ for all clusters already fitted. The algorithm finishes as soon as the assignment of trajectories to clusters no longer changes or a fixed number of iterations is reached. The computational complexity of the algorithm is linear in the total number of line segments, clusters, and grid size.

2.2 V–Bundles

To adapt vector field k–means to streamline data, we first need to extend the algorithm to 3D. Changing the dimension of the matrices suffices, but the size of the grid changes from $O(R_v^2)$ to $O(R_v^3)$, affecting computational complexity. We avoid unnecessary computations due to empty grid cells by initializing the grid using a voxel size parameter r_v instead of the number of points. Second, streamlines lack time information. We therefore replace time the parameter by arc length parameterization in the original trajectory–to–field match cost, implemented by replacing each line segment's duration $w_l^{(i)}$ by its length.

The lack of a direction in streamlines causes a more substantial change. Replacing vector fields by orientation fields is difficult because the latter lack a suitable and fast interpolation scheme. Instead, to determine the match cost between a streamline and a vector field, we treat each streamline as being directed but take the smaller value for both possible directions. The flip is implemented by negating $B_l^{(i)}$ in Equation 1. When fitting the vector field, we originally allowed streamlines to change their direction. Since we observed that only a few, typically outlying, streamlines change their direction and cause only marginal difference in cost, we decided to assign each streamline its direction in each assignment step and keep it fixed during fitting steps. This allows us to use the same technique as Ferreira et al., to stack the matrices $C_l^{(i)}$ and $B_l^{(i)}$ to $C^{(i)}$ and $B^{(i)}$ in the fitting step and end up with a sparse linear system suitable for efficient solution by conjugate gradient. We also do not use the squared Laplacian, thus we solve the equation system $(L + {C^{(i)}}^T C^{(i)}) V^{(i)} = {C^{(i)}}^T B^{(i)}$ to compute

our vector fields. In order to keep the algorithm deterministic, we use the longest streamline to initialize $V^{(0)}$ instead of a random one.

Finally, we observed that spatially distant bundles were represented by the same vector field due to Laplacian smoothing. To avoid them being added to the same cluster, we employ additional scalar fields $Z^{(i)}$ of voxel size r_z describing the streamline density of each cluster. They are calculated by counting the number of streamlines crossing each cell, normalizing by their total number, smoothing using a Gaussian kernel and then inverting by subtracting each value from 1.0. By changing the assignment cost to $(1-\gamma)e(S^{(i)}, V^{(j)}) + \gamma\|C^{(i)}Z^{(j)}\|^2$, we can choose close by streamlines over distant ones as γ gets larger.

2.3 Material

We tested our algorithm on multiple datasets. To judge the robustness regarding interrupted streamlines, we created two artificial datasets: a crossing with 150 streamlines and a fork with 100 streamlines. In both datasets, runs of successive segments were removed randomly (overall one third of all segments) from the streamlines, breaking them into multiple pieces. We employ the *FiberCup* phantom[1], which is a hardware phantom that was originally developed in order to benchmark tracking algorithms through various types of complex fiber configurations: bundles crossing, forking, and touching. To assess clustering quality on real data, HARDI data of five healthy subjects aged 24 to 31 was acquired using a 3T Siemens Trio MRI scanner, single echo spin echo EPI sequence with GRAPPA on a 32 channel coil, $128 \times 128 \times 72$ image matrix, $1.7 \times 1.7 \times 1.7\,\mathrm{mm}^3$ voxel size, 60 gradient and six non–gradient images at $b = 1000$. The datasets were corrected for motion artifacts and linearly registered to one subject using *FLIRT* (fsl[2]). Registration used fractional anisotropy maps computed from the HARDI data. We traced streamlines using the *tensor toolkit*[3](ttk, version 1.4) and its default parameters (except for $\mathrm{FA}_1 = 0.2$ and $\mathrm{FA}_2 = 0.3$) and its standard interpolation for both the *FiberCup* as well as the human brain data. Furthermore, we created a 500 000–streamline dataset using *mrtrix*[4](version 0.2.11) using its default pipeline, parameters, and interpolation scheme. We resampled this dataset to 1mm segment length (around 28×10^6 segments).

For the purpose of comparison, we implemented four clustering algorithms: Garyfallidis et al. [3] (QuickBundles), a spectral clustering by O'Donnell and Westin [5], a point–based clustering by Zhang et al. [10], and a stochastic–process–based algorithm by Wassermann et al. [9]. The implementations make use of multiple cores where applicable. If not stated otherwise, streamlines were resampled to 20 points per streamline in order to save time or because the resampling is required by the algorithm, as in the case of QuickBundles. For V–Bundles, we usually set the maximum number of k–means iterations to 3,

[1] http://www.lnao.fr/spip.php?rubrique79
[2] http://fsl.fmrib.ox.ac.uk/fsl/fslwiki/FLIRT/UserGuide
[3] https://gforge.inria.fr/projects/ttk/
[4] http://www.brain.org.au/software/mrtrix/

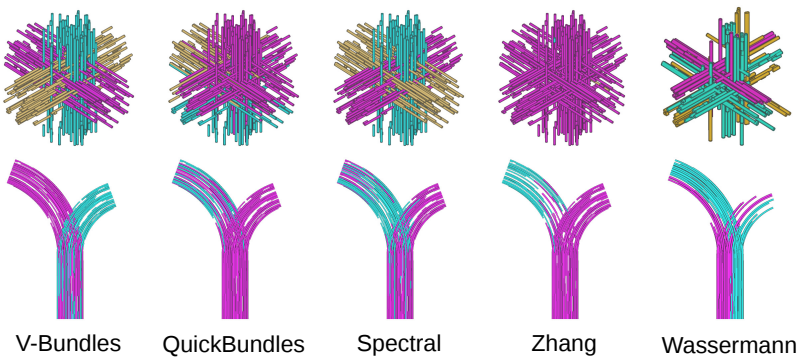

Fig. 1. The best three and two clusters for the crossing and the forking datasets respectively.

$\lambda = 0.1$, $\gamma = 0.6$, $r_v = 15mm$ and $r_z = 9mm$. All algorithms were run on a dual Intel E5-2630v3 CPU, 32GB RAM, NVIDIA GeForce GTX 980, 4GB VRAM.

3 Results

We tested the five clustering algorithms' ability to resolve clusters even in the presence of gaps and short lines using the synthetic datasets. Each parameter was tuned to give a good match with the three, respectively two, contained bundles. The results are shown in Fig. 1. V–Bundles showed the best results, perfectly discerning the bundles in both experiments, but required $\gamma = 0$ to correctly resolve the fork. Spectral clustering came close to resolving the crossing configuration, missing only a few short lines. In all other cases, we did not find parameters that provided reasonable clusters. The similarity-based methods tend to cluster lines by proximity rather than orientation and often put isolated small lines into their own cluster.

We tested the ability to resolve common fiber configurations using the *FiberCup* phantom (Fig. 2). V-Bundles is able to separate lines into meaningful clusters, while QuickBundles easily becomes confused by outliers and interrupted lines. From the dataset we selected only those lines which exactly correspond to the seven known ground truth bundles of the *FiberCup* in order to measure precision and recall (using the clusters that share the largest number of lines with the respective ground truth cluster). V-Bundles shows a perfect recall score of 1 for all ground truth clusters when computing 16 clusters for the dataset using standard parameters. Precision, however, is low due to short fibers getting added to those few clusters. QuickBundles reaches high recall and precision values of 0.97 and 0.75 when creating a large number of clusters (e.g. 73 clusters, $\theta = 230$), but then, most of those fail to capture the patterns inherent to the phantom.

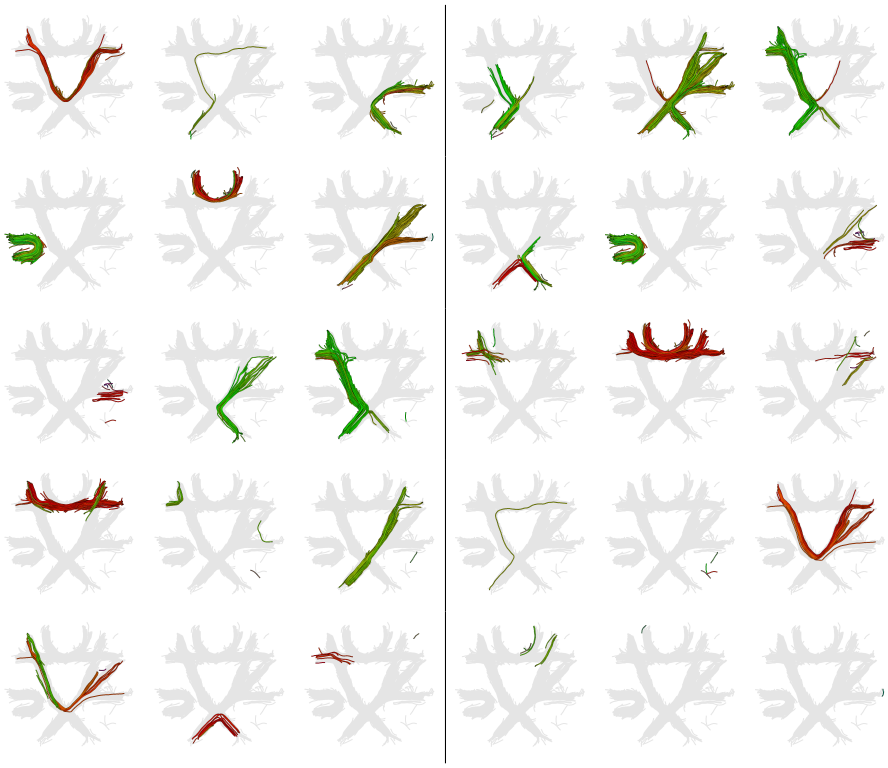

Fig. 2. Partitioning 883 lines tracked on the FiberCup dataset into 15 clusters using V–Bundles ($r_v = 15$mm, $r_z = 9$mm, $\gamma = 0.6$, $\lambda = 0.001$, *left*) and QuickBundles ($\theta = 620$, *right*), respectively. The background shows all lines for context.

Furthermore, V-Bundles was also able to reliably extract major fiber bundles from a set of full brain tractography data (approx. 88 000 lines). We used the fields output by a run of V-Bundles on a single subject to identify the corresponding clusters in three other subjects. Figure 3 shows clusters located in three well–known fiber bundles in the four subjects.

The computation times for different datasets are shown in Table 1. Note the large influence of the vector grid length r_v, but V-Bundles scales linearly in the number of segments. The scalar grid resolution has little impact on computation time. The algorithms of O'Donnell et al., Zhang et al. and Wassermann et al. fail to cluster moderately large datasets due to resource limitations.

4 Discussion and Conclusion

We described an adaption of the vector field k-means algorithm for clustering neuronal pathways. The algorithm handles short and interrupted streamlines, which can make up a significant part of a tractogram, without special treatment.

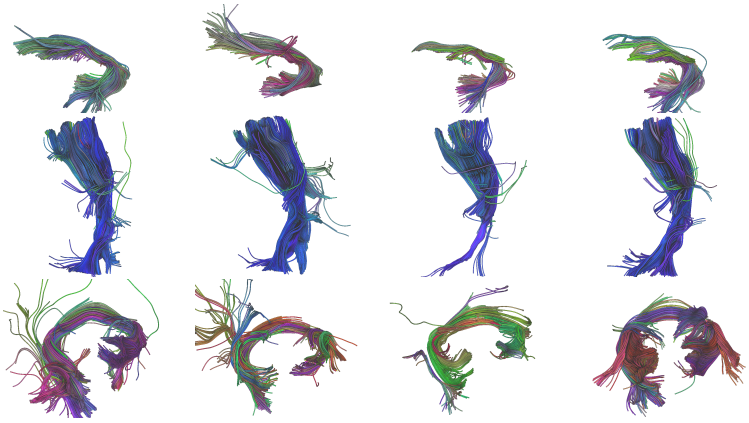

Fig. 3. Corresponding clusters calculated for the uncinate fasciculus (top row), parts of the cortico–spinal tract (mid row), and forceps major over four subjects. Each row has the same camera position. Parameters were $r_v = 15$mm, $r_z = 9$mm, $\lambda = 0.1$, $\gamma = 0.6$, 500 clusters. lines were resampled to 3 mm per segment to improve speed.

Table 1. Computation times, a single run each. The number of lines is denoted in brackets. When an algorithm exceeded 32GB of RAM, it was aborted (marked by –).

	V-Bundles	QuickB.	O'Donnell	Zhang	Wassermann
FiberCup (441)	2.5s	0.025s	0.31s	1.11s	0.67s
FiberCup (883)	3.3s	0.043s	1.08s	1.19s	3.03s
Human Brain (11k)	113s	2.23s	1770s	20.0s	531s
Human Brain (22k)	205s	4.88s	15500s	96.8s	–
Human Brain (44k)	368s	10.8s	108600s	428s	–
Human Brain (88k)	681s	17.0s	–	2260s	–
Human Brain (500k)	2290s	75.3s	–	–	–

This avoids generating a large number of "outlier"–clusters which need to be removed from the clustering afterwards. However, this property can also lead to bundles that share a common path for a significant part of their lengths to be erroneously clustered into the same cluster. Both increasing the locality parameter γ and reducing the number of clusters aggravates the problem.

As demonstrated, computation time and memory consumption scale linearly in the overall number of streamline segments, whis makes the algorithm applicable to dense sets of 100 000 or more streamlines while keeping computation times reasonably short. It may thus prove to be a useful tool in clinical settings or group studies. Computation time can be significantly reduced by resampling the streamlines to have longer segments. However, the grid resolution currently also has a high impact on performance; thus, in future work, we plan to address this problem by finding a sparser representation of the vector field which could be computed more quickly.

V–Bundles is also capable of finding corresponding clusters in different subjects in a common space, which is done by initializing the algorithm with the output fields of one subject (or possibly the result of combined tractograms of multiple subjects). It would also be conceivable to initialize the algorithm with a clustering created by an expert.

Also, we plan to evaluate different strategies for picking the first streamline in the initialization step. The source code of our reference implementation will be made available at *www.openwalnut.org*.

Acknowledgements. We thank Marcus Stuber for his initial coding effort, Marc Tittgemeyer for providing T1 and HARDI data of human subjects, and the anonymous reviewers for their excellent feedback.

References

1. Dodero, L., Vascon, S., Murino, V., Bifone, A., Gozzi, A., Sona, D.: Automated multi-subject fiber clustering of mouse brain using dominant sets. Frontiers in Neuroinformatics 8, 87 (2015)
2. Ferreira, N., Klosowski, J.T., Scheidegger, C.E., Silva, C.T.: Vector field k–means: Clustering trajectories by fitting multiple vector fields. Computer Graphics Forum 32(3), 201–210 (2013)
3. Garyfallidis, E., Brett, M., Correia, M.M., Williams, G.B., Nimmo-Smith, I.: Quickbundles, a method for tractography simplification. Front. Neurosci. 6 (2012)
4. Guevara, P., Poupon, C., Rivière, D., Cointepas, Y., Marrakchi, L., Descoteaux, M., Fillard, P., Thirion, B., Mangin, J.F.: Inference of a hardi fiber bundle atlas using a two-level clustering strategy. In: Jiang, T., Navab, N., Pluim, J., Viergever, M. (eds.) MICCAI 2010, LNCS, vol. 6361, pp. 550–557. Springer Berlin Heidelberg (2010)
5. O'Donnell, L., Westin, C.F.: White matter tract clustering and correspondence in populations. In: Duncan, J., Gerig, G. (eds.) MICCAI 2005, LNCS, vol. 3749, pp. 140–147. Springer Berlin Heidelberg (2005)
6. O'Donnell, L.J., Golby, A.J., Westin, C.F.: Fiber clustering versus the parcellation-based connectome. NeuroImage 80, 283–289 (2013)
7. Visser, E., Nijhuis, E.H., Buitelaar, J.K., Zwiers, M.P.: Partition-based mass clustering of tractography streamlines. NeuroImage 54(1), 303–312 (2011)
8. Wang, X., Grimson, W.E.L., Westin, C.F.: Tractography segmentation using a hierarchical dirichlet processes mixture model. NeuroImage 54(1), 290–302 (2011)
9. Wassermann, D., Bloy, L., Kanterakis, E., Verma, R., Deriche, R.: Unsupervised white matter fiber clustering and tract probability map generation: Applications of a gaussian process framework for white matter fibers. NeuroImage 51(1), 228–241 (2010)
10. Zhang, S., Correia, S., Laidlaw, D.H.: Identifying white-matter fiber bundles in dti data using an automated proximity-based fiber-clustering method. IEEE Transactions on Visualization and Computer Graphics 14(5), 1044–1053 (2008)