5239

# Recent advances in Diffusion Imaging in Python (Dipy)

Eleftherios Garyfallidis[1], Ariel Rokem[2], Bagrat Amirbekian[3], Stefan van der Walt[4], Mauro Zucchelli[5], Jesus-Omar Ocegueda-Gonzalez[6], Samuel St-Jean[7], Gabriel Girard[7], Michael Paquette[7], Ian Nimmo-Smith[8], Matthew Brett[9], and Maxime Descoteaux[10]

*[1]University of Sherbrooke, Sherbrooke, Quebec, Canada, [2]Stanford University, CA, United States, [3]University of California, San Francisco, CA, United States, [4]Stellenbosch University, Stellenbosch, South Africa, [5]University of Verona, Italy, [6]Centro de Investigacion en Matematicas, Mexico, [7]University of Sherbrooke, Quebec, Canada, [8]MRC Cognition & Brain Sciences Unit, Cambrige, United Kingdom, [9]University of California, Berkeley, CA, United States, [10]University of Sherbrooke, Sherbrooke, Canada*

**Target Audience/Purpose**

This paper is targeted to everyone who uses or develops software for diffusion MRI (dMRI) analysis. The purpose of this abstract is to inform the researchers in dMRI about the recent and new algorithms implemented in Diffusion Imaging in Python (Dipy).

**Methods**

Diffusion Imaging in Python (Dipy) is a *free and open source* software project for the analysis of data from dMRI experiments. Dipy gathers implementations of many different methods in dMRI, including: diffusion signal pre-processing; reconstruction of diffusion distributions in individual voxels; fiber tractography and streamline post-processing, analysis and visualization. In contrast to many other scientific software projects, Dipy is not being developed by a single research group. Rather, it is an open project that encourages contributions from any scientist/developer through GitHub and open discussions on the project mailing list. Consequently, Dipy today has an international team of contributors, spanning seven academic institutions in five countries and three continents and is still growing. Dipy has a modular structure with the main modules depicted in fig.1. It also has a few hard dependencies shown with yellow colour in the same figure and some optional dependencies shown with orange colour. In Dipy we implement state-of-the-art methods in every level of the processing pipeline. *Pre-processing:* we have implemented novel adaptive denoising methods[3], tools to create well distributed spheres (electrostatic repulsion), automatic SNR estimation, brain extraction and simulations (see bottom right figure). *Voxel Reconstruction:* a great range of methods that can be used for any type of q-space acquisition (single-shell, multi-shell and DSI-like). Here we reference briefly some of them: CSD[1], SDT[4], DSI[7], DSI Deconvolution[8], CSA[5], SHORE[6], Tensors with least squares, weighted least squares and RESTORE[10]. We also support parallelization of reconstructions and peak finding for systems with multiple CPUs. *Fiber tracking*: we have implemented a fast deterministic algorithm (EuDX)[11] and a modular method for Probabilistic tracking. *Fiber analysis:* we can calculate metrics (e.g. length, curvature) for streamlines and also check if streamlines intersect with ROIs and create connectivity matrices. Furthermore, we have an efficient method to cluster streamlines called Quickbundles[9] (see fig.2) that can be used to simplify large datasets. *Data Access:* we provide access to publicly available datasets and datasets for simulations. We can also read many different formats (e.g. Nifti1, Trackvis, HDF5).

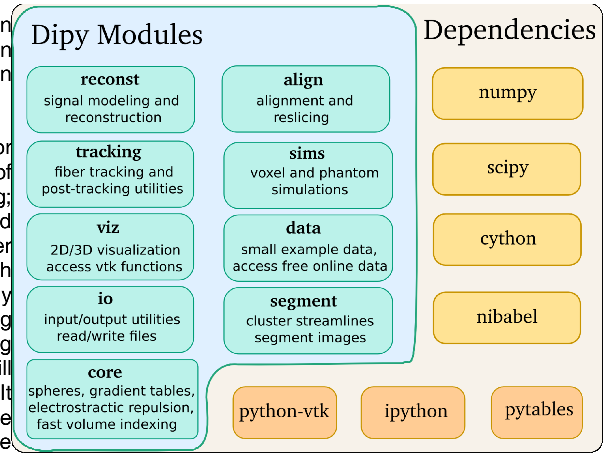


*Figure 1: Modules and dependencies*

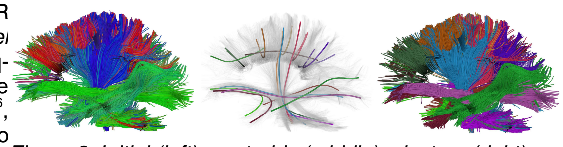

*Figure 3: Initial (left), centroids (middle), clusters (right)*

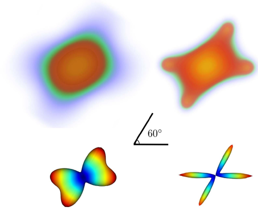


*Figure 2: Simulations*

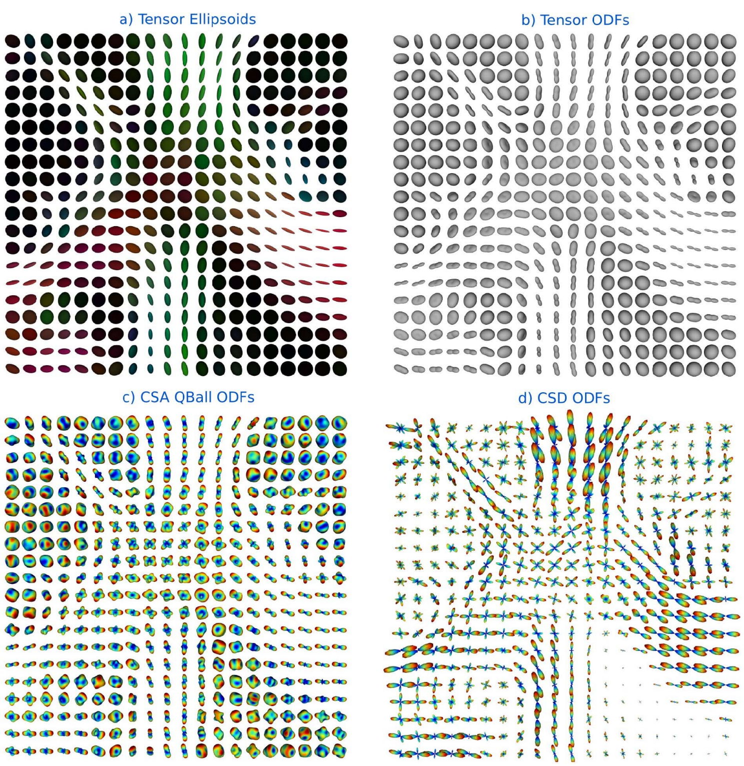


*Figure 4: Local reconstructions*

**Results/Discussion/Conclusion**

In fig.2 we show the result of QuickBundles applied in the initial streamlines (left) and showing the centroids of the clusters(middle) and the final clusters (right). In fig.3 we show ellipsoids and ODFs for the same ROI of the same dataset with different reconstruction methods implemented in Dipy. In fig.4 we show at the left column the PDF(upper) and ODF(lower) from simulated datasets of a 60 degrees crossing using DSI[7], and with DSI with deconvolution[8] on the right column. More examples are available in the project's website[2]. Dipy welcomes everyone to use or contribute in the project and give us feedback and suggestions.

**References**

**[1]** Tournier et al. Non-negativity constrained super-resolved spherical deconvolution, Neuroimage (2007). **[2]** http://dipy.org **[3]** Coupe et al. Adaptive multi-resolution non-local means filter, IETIP (2012), **[4]** Descoteaux et al., Deterministic and probabilistic tractography based on complex fibre orientation distributions, IEEE TMI (2009). **[5]** Aganj et al, Reconstruction of the orientation distribution function in single and multiple shell q-ball imaging with the constant solid angle, MRM (2010). **[6]** Merlet et al, Continuous diffusion signal, EAP and ODF estimation via Compressed Sensing in diffusion MRI, MIA (2013). **[7]** Wedeen, Diffusion spectrum imaging of crossing fibers, Neuroimage. (2009) **[8]** Canales-Rodriguez, Deconvolution in diffusion spectrum imaging, Neuroimage 2010. **[9]** Garyfallidis et al., QuickBundles: a method for tractography simplification, Frontiers (2012). **[10]** Chang et al., RESTORE: robust estimation of Tensors, MRM (2005). **[11]** Garyfallids, PhD thesis, (2012).