Oriented Tensor Reconstruction: Tracing Neural Pathways from Diffusion Tensor MRI

Leonid Zhukov Alan H. Barr

Department of Computer Science, California Institute of Technology

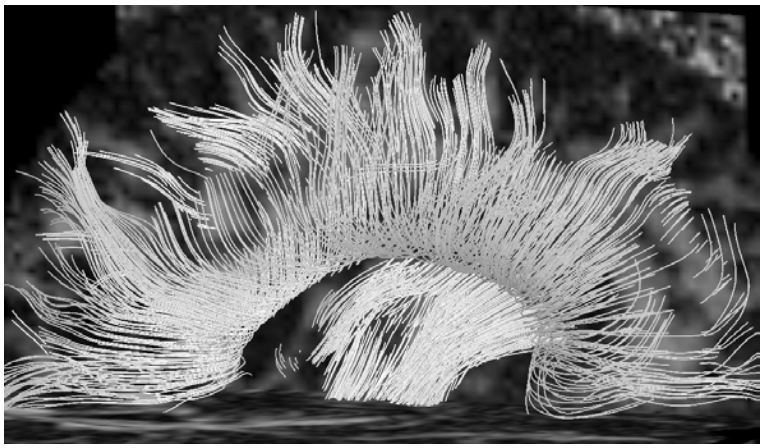

Figure 1: Human brain pathways recovered from DT-MRI data using the oriented tensor reconstruction algorithm

ABSTRACT

In this paper we develop a new technique for tracing anatomical fibers from 3D tensor fields. The technique extracts salient tensor features using a local regularization technique that allows the algorithm to cross noisy regions and bridge gaps in the data. We applied the method to human brain DT-MRI data and recovered identifiable anatomical structures that correspond to the white matter brain-fiber pathways. The images in this paper are derived from a dataset having 121x88x60 resolution. We were able to recover fibers with less than the voxel size resolution by applying the regularization technique, i.e., using a priori assumptions about fiber smoothness. The regularization procedure is done through a moving least squares filter directly incorporated in the tracing algorithm.

CR Categories: I.3.8 [Computing Methodologies]: Computer Graphics—Applications, I.6 [Computing Methodologies]: Simulation and Modeling, J.3 [Computer Application]: Life and Medical Sciences;

Keywords: Diffusion tensors, adaptive filtering, moving least squares, streamlines, fiber tracing, pathways, salient features

1 INTRODUCTION

Directional tracking through vector fields has been a widely explored topic in visualization and computer graphics [5, 18, 19]. The standard streamline technique advects massless particles through the vector field and traces their location as a function of time. Analogously, a hyper-streamlines approach has been proposed to trace changes through tensor fields, following the dominant eigenvector direction [7]. These methods work best on very "clean" datasets, which are usually produced as a result of simulations; these methods typically do not handle raw experimental data very well, due to noise and resolution issues.

Recently, attention has been given to the visualization of 2D [12] and 3D [10] diffusion tensor fields from DT-MRI data. Although these methods provide nice visual cues, they do not attempt to recover the underlying anatomical structures, which are the white matter fiber tracts (bundles of axons) found within the brain[1].

Several previous endeavors have been made for recovering the underlying structure by extracting fibers through the application of modified streamline algorithms. Examples include tensor-lines [20] and stream-tubes [23, 6]. Direct fiber tractography method has been developed in [2]. Other work suggests separate regularizaton of eigenvalues and eigenvectors in the tensor fields before fiber tracing [14]. These algorithms have had some success in recovering the underlying structures. Some problems still remain due to the complexity of the tensor field, voxelization effects and the significant amount of noise that is omnipresent in experimental data. The most recent work concentrated on deriving a continuous tensor field approximation [15] and using signal processing techniques (for example, Kalman filtering [8]) for cleaning up the data.

The goal of this paper is to develop more stable tensor tracing techniques which allow the extraction of the underlying continuous anatomical structures from experimental diffusion tensor data. The proposed technique uses a moving local regularizing filter that allows the tracing algorithm to cross noisy regions and gaps in the data while preserving directional consistency.

[1] The white matter constitutes the "wiring" of the brain; the gray matter constitutes the computational components of the brain.

2 METHOD

2.1 Data

Diffusion tensor magnetic resonance imaging (DT-MRI) [1] is a technique used to measure the anisotropic diffusion properties of the water molecules found within biological tissues as a function of the spatial position within the sample. Due to differing cell shape and cell membrane properties, the diffusion rates of the water molecules are different in different directions and locations.

For instance, neural fibers are comprised mostly of bundles of long cylindrical cells that are filled with fluid and are bounded by less-water-permeable cell membranes. The average diffusion rate (at a spatial location) is fastest in the three-dimensional axis direction along the length of the neuron cells, since more of the water molecules are free to move in this direction. The average diffusion rate is slowest in the two transverse directions, where the cell membrane interferes, reducing and slowing down the movement of the water molecules.

Other parts of the brain are primarily comprised of fluid without cell membranes, such as the ventricles. Here the average diffusion rate is larger and more uniform (almost the same in all directions).

The diffusion properties can be represented with a symmetric second order tensor - 3x3 matrix:

$$\mathbf{T} = \begin{pmatrix} T^{xx} & T^{xy} & T^{xz} \\ T^{yx} & T^{yy} & T^{yz} \\ T^{zx} & T^{zy} & T^{zz} \end{pmatrix}. \quad (1)$$

The 6 independent values (the tensor is symmetric) of the tensor elements vary continuously with spatial location.

The 3-D local axis direction of the neuron fibers will correspond to the dominant eigenvector of the tensor. There should be one large eigenvalue, and two small eigenvalues. This can be seen from the physical interpretation of the diffusion tensor, which can be thought of as a vector-valued function whose input is the local 3-D concentration gradient and whose output is the 3-D directional vector flux[2] of the water molecules. The function is evaluated by multiplying the 3x3 matrix by the 3x1 concentration gradient, producing the 3x1 vector flux of the water molecules. Water will diffuse fastest in the direction along the axis of the neurons and slowest in the two transverse directions.

For the ventricles, a dominant eigenvector should not exist: the three eigenvalues of the tensor should have roughly the same value. Water will diffuse roughly at the same speed in all directions. Hence, we can use the diffusion tensor to distinguish tissues with a primary diffusion axis from parts that do not.

In this paper, the experimental dataset contains sampled values of the diffusion tensor on a regularly spaced grid of 121x88x60 (cubic) voxels. We will denote these given tensor values as $\mathbf{T}^{\alpha\beta}_{ijk}$, where α and β are the three dimensional tensor components $\{xx, xy, .., zz\}$, and i, j, k are traditional integer indexes into the regular grid volume. Also, when no upper indexes are provided, the operations are assumed to be performed on the entire tensor component-wise $\mathbf{T} \equiv \mathbf{T}^{\alpha\beta}$, i.e., on each of the six independent values of the tensor.

2.2 Tensor Classification

Geometrically, a diffusion tensor can be thought of as an ellipsoid with its three axes oriented along the tensor's three perpendicular eigenvectors, with the three semi-axis lengths proportional to the square root of eigenvalues of the tensor - mean diffusion distances [1].

[2]Vector flux measures a quantity per unit area per time, in the direction perpendicular to the area

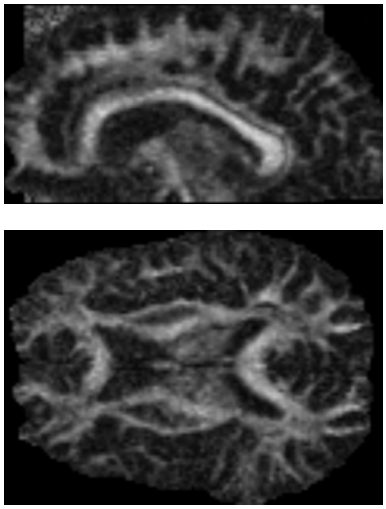

Figure 2: Sagittal and axial slices of anisotropy measure c_ℓ of the dataset. The lighter regions correspond to stronger anisotropy areas found in the white matter. See Eq. (4).

In general, eigenvalues λ and eigenvectors $\mathbf{e}$ can be found as a solution to the eigen-equation

$$\mathbf{T}\mathbf{e}_i = \lambda_i \mathbf{e}_i \quad (2)$$

Since the tensor is symmetric, its eigenvalues are always real numbers, and the eigenvectors are orthogonal and form a Cartesian vector basis $\{\mathbf{e}_1, \mathbf{e}_2, \mathbf{e}_3\}$. This basis (frame of reference) can be used to represent the tensor in diagonal form and to specify directions with respect to the "world coordinate" system

$$\mathbf{T} = \{\mathbf{e}_1, \mathbf{e}_2, \mathbf{e}_3\} \begin{pmatrix} \lambda_1 & 0 & 0 \\ 0 & \lambda_2 & 0 \\ 0 & 0 & \lambda_3 \end{pmatrix} \{\mathbf{e}_1, \mathbf{e}_2, \mathbf{e}_3\}^T \quad (3)$$

Using the ellipsoidal interpretation, one can classify the diffusion properties of tissue according to the shape of the ellipsoids, with extended ellipsoids corresponding to regions with strong linear diffusion (long, thin cells) flat ellipsoids to planar diffusion, and spherical ellipsoids to regions of isotropic media (such as fluid-filled regions like the ventricles). The quantitative classification can be done through the coefficients c_ℓ, c_p, c_s (linear, planar, spherical) proposed in [22, 21]:

$$c_\ell = \frac{\lambda_1 - \lambda_2}{\lambda_1 + \lambda_2 + \lambda_3} \quad (4)$$

$$c_p = \frac{2(\lambda_2 - \lambda_3)}{\lambda_1 + \lambda_2 + \lambda_3} \quad (5)$$

$$c_s = \frac{3\lambda_3}{\lambda_1 + \lambda_2 + \lambda_3} \quad (6)$$

These coefficients are normalized to the range of $[0..1]$ and could be interpreted as barycentric coordinates. For example, close to 1 values of c_ℓ chooses the regions with strong linear $(\lambda_1 >> \lambda_2 \approx \lambda_3)$ diffusion.

2.3 Data Interpolation

We start by reconstructing a continuous tensor field in the volume through trilinear interpolation. In this scheme the value of a tensor at any point inside the voxel is a linear combination of the 8 values at its corners and is completely determined by them. Since

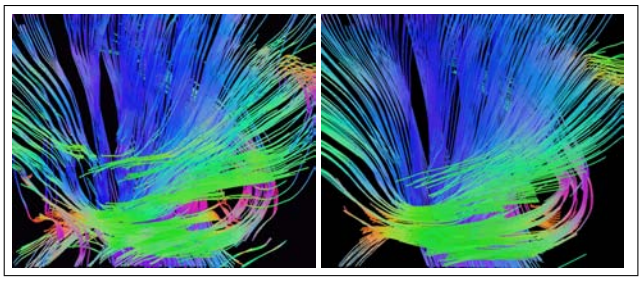

Figure 3: Comparison of non-filtered (left image) and MLS filtered (right image) fiber tracts. Note the smoother and more regular behavior of the filtered fibers

the coefficients of this linear combination are independent of the tensor indexes, the linear combination of the tensors can be done component-wise.

$$\begin{aligned}\mathbf{T}(x,y,z) = {} & \mathbf{T}_{ijk}\ (1-x)(1-y)(1-z) + \\ + {} & \mathbf{T}_{i+1,jk}\ x(1-y)(1-z) + \mathbf{T}_{i,j+1,k}\ (1-x)y(1-z) \\ + {} & \mathbf{T}_{ij,k+1}\ (1-x)(1-y)z + \mathbf{T}_{i+1,j,k+1}\ x(1-y)z \\ + {} & \mathbf{T}_{i,j+1,k+1}\ (1-x)yz + \mathbf{T}_{i+1,j+1,k}\ xy(1-z) \\ + {} & \mathbf{T}_{i+1,j+1,k+1}\ xyz\end{aligned} \tag{7}$$

We can use trilinear component-wise interpolation because symmetric tensors form a linear subspace in the tensor space: any linear combination of symmetric tensors remains a symmetric tensor, i.e., symmetric tensors are closed under linear combination (the manifold of symmetric tensors is not left). Component-wise interpolation is sufficient for our purposes; more sophisticated interpolation methods, however, would better preserve the eigenvalues along an interpolation path [14].

On the other hand, component-wise interpolation of eigenvectors and eigenvalues themselves would not lead to correct results, since a linear interpolation between two unit vectors is not a unit vector anymore – the interpolated eigenvector value would leave the manifold of unit vectors. In addition, there can be a correspondence problem in the order of the eigenvalues.

Various types of tensor interpolation are discussed, for example, in [11].

2.4 Regularization: Moving Least Squares

To perform a stable fiber tracing on experimental data, the data needs to be filtered. A simple global box or Gaussian filter will not work well, since it will blur (destroy) most of the directional information in the data. We also want the filter to be adjustable to the data and be able to put more weight on the data in the direction of the traced fiber, rather than between fibers. We also want the filter to have an adjustable local support, which can be modified according to the measure of the confidence level in the data. Finally, we want the filter to preserve sharp features where they exist (ridges), but eliminate irrelevant noise if it can. Thus, to do this, the behavior of the filter at some voxel in space will depend on the "history of the tracing", (where it came from), so the filtering needs to be tightly coupled with the fiber tracing process.

Due to the above reasons, we chose to use a moving least squares (MLS) approach. The idea behind this local regularization method is to find a low degree polynomial which best fits the data, in the least squares sense, in the small region around the point of interest. Then we replace the data value at that point by the value of the polynomial at that point $\mathbf{T}(x_p, y_p, z_p) \rightarrow \bar{\mathbf{T}}_p$. Thus, this is a

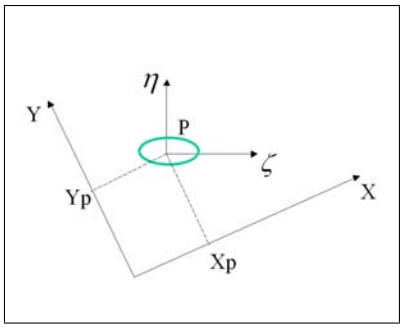

Figure 4: Coordinate systems used by linear transformation Eq. (10) to change from $\{x, y, z\}$ coordinates in Eq. (8) into $\{\zeta, \eta, \theta\}$ coordinates in Eq. (11).

data approximation rather than an interpolation method. The measure of "fitness" will depend on the filter location, orientation and the history of motion. The one dimensional MLS method was first introduced in signal processing literature [9, 16].

We start the derivation by writing a functional E that measures the quality of the polynomial fit to the data in the world coordinate system. To produce E, we integrate the squared difference between $\mathbf{F}$, which is an unknown linear combination of tensor basis functions, and $\mathbf{T}$, the known continuous trilinear interpolated version of given tensor data. We integrate over all of 3D space and use weighting function G to create a region of interest centered at chosen 3-D point $\mathbf{r}_p$ with coordinates (x_p, y_p, z_p):

$$E(\mathbf{r}_p) = \int_{-\infty}^{\infty} G(\mathbf{r} - \mathbf{r}_p; \mathbf{T}_p))[\mathbf{F}(\mathbf{r} - \mathbf{r}_p) - \mathbf{T}(\mathbf{r})]^2\ d\mathbf{r}^3 \tag{8}$$

The second argument $\mathbf{T}_p$ (value of the tensor at the center point) of the weighting function G determines the weighting function's size and orientation.

The square of the tensor difference in (8) is a scalar, defined through the double-dot (component wise) product [4, 3]:

$$\begin{aligned}(\mathbf{F} - \mathbf{T})^2 = (\mathbf{F} - \mathbf{T}) : (\mathbf{F} - \mathbf{T}) = \\ \sum_{\alpha\beta} (\mathbf{F}^{\alpha\beta} - \mathbf{T}^{\alpha\beta})(\mathbf{F}^{\beta\alpha} - \mathbf{T}^{\beta\alpha}) = \sum_{\alpha\beta} (\mathbf{F}^{\alpha\beta} - \mathbf{T}^{\alpha\beta})^2\end{aligned} \tag{9}$$

Within the functional $E()$, the tensor function $\mathbf{F}$ is a linear combination of tensor basis functions we will use to fit the data. The function G is the moving and rotating anisotropic filtering window, centered at the point $\mathbf{r}_p$ (See Figure 4.)

We felt it was more convenient to perform the computations in the local frame of reference connected to a particular filtering window, rather than in world coordinates. It is straightforward to transform equation (8) to a local frame of reference using a change of variables involving the following translation and rotation transformation:

$$\begin{pmatrix} \zeta \\ \eta \\ \theta \end{pmatrix} = \mathbf{R}_p^{-1} \begin{pmatrix} x - x_p \\ y - y_p \\ z - z_p \end{pmatrix} = \mathbf{R}_p^{-1}(\mathbf{r} - \mathbf{r}_p) \tag{10}$$

where $\mathbf{R}_p = \{\mathbf{e}_1, \mathbf{e}_2, \mathbf{e}_3\}$ is a rotation matrix formed by the eigenvectors of the tensor $\mathbf{T}_p$ at the point $\mathbf{r}_p$. Then, in the local frame of reference,$\{\zeta, \eta, \theta\}$, the equation (8) becomes

$$\begin{aligned}E = \int_V [\mathbf{F}(\mathbf{R}_p\{\zeta, \eta, \theta\}) - \mathbf{T}(\mathbf{r}_p + \mathbf{R}_p\{\zeta, \eta, \theta\})]^2 \\ G(\mathbf{R}_p\{\zeta, \eta, \theta\}; \mathbf{T}_p)\ d\zeta d\eta d\theta\end{aligned} \tag{11}$$

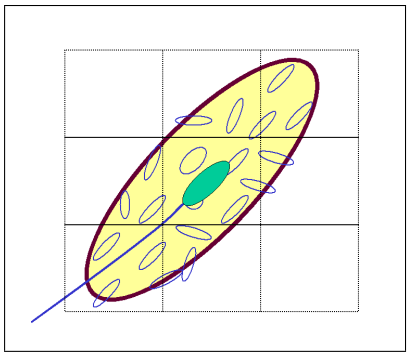

Figure 5: Oriented moving least squares (MLS) tensor filter. The smallest ellipsoids represent the interpolated tensor data; the largest ellipsoid represents the domain of the moving filter $g()$ described in Eq. (8); the dark ellipsoid represents the computed filtered tensor. The filter travels continuously along the fiber line; grid shows initial sampling of the tensor data.

Integration is performed over the parts of space where G has been chosen to be nonzero.

We now instantiate $\mathbf{F}$ and G in the local (rotated and translated) frame of reference. These can be thought of as functions of $\{\zeta, \eta, \theta\}$ only, since $\mathbf{R}_p$ is independent of the integration variables.

For $\mathbf{F}$ we will use a polynomial function of degree N in the variables ζ, η and θ:

$$\mathbf{F}(\mathbf{R}_p\ \{\zeta, \eta, \theta\}) \equiv f(\mathbf{A}; \zeta, \eta, \theta) \tag{12}$$

where

$$f(\mathbf{A}; \zeta, \eta, \theta) = \sum_{mnp} \mathbf{A}_{mnp}\ \zeta^m \eta^n \theta^p \tag{13}$$

To instantiate G, we also use a function of variables ζ, η and θ:

$$\mathbf{G}(\mathbf{R}_p\{\zeta, \eta, \theta\}; \mathbf{T}_p) \equiv g(\zeta, \eta, \theta, \mathbf{T}_p) \tag{14}$$

The function g() is clipped to zero at some low threshold value to create a finite integration volume.

Substituting expressions (13) and (14) into the equation (11) we get:

$$E = \int_V [\sum_{mnp} \mathbf{A}_{mnp}\ \zeta^m \eta^n \theta^p - \mathbf{T}(\mathbf{r}_p + \mathbf{R}_p\{\zeta, \eta, \theta\})]^2 \\ g(\zeta, \eta, \theta; \mathbf{T}_p)\ d\zeta d\eta d\theta \tag{15}$$

The least squares fitting procedure reduces to minimizing functional E with respect to tensor elements $\mathbf{A}^{\alpha\beta}_{rst}$. To do this we differentiate the expression (11) with respect to each one of the $\mathbf{A}$ coefficients, equate the result to zero, and linearly solve to find the unknown $\mathbf{A}$'s:

$$\partial E / \partial \mathbf{A}^{\alpha\beta}_{rst} = 0 \tag{16}$$

This gives us the following linear system for the unknown $\mathbf{A}$'s:

$$\sum_{mnp} \mathbf{M}_{mnp,rst} \mathbf{A}^{\alpha\beta}_{mnp} = \mathbf{B}^{\alpha\beta}_{rst} \tag{17}$$

where $\mathbf{M}_{mnp,rst}$ are elements of the matrix; $\mathbf{B}_{rst}$ are right hand side values of the equation:

$$\mathbf{M}_{mnp,rst} = \int_V \zeta^{m+r} \eta^{n+s} \theta^{p+t}\ g(\zeta, \eta, \theta; \mathbf{T}_p)\ d\zeta d\eta d\theta \tag{18}$$

$$\mathbf{B}^{\alpha\beta}_{rst} = \int_V \mathbf{T}^{\alpha\beta}(\mathbf{r}_p + \mathbf{R}_p\{\zeta, \eta, \theta\}) \zeta^r \eta^s \theta^t \\ g(\zeta, \eta, \theta;\ \mathbf{T}_p)\ d\zeta d\eta d\theta \tag{19}$$

These integrals can be computed numerically or any specific choice of $g()$.[3]

The equation (17) is just a "regular" linear system to find the $\mathbf{A}$'s. Written out component-wise for the tensors $\mathbf{A}$, $\mathbf{B}$ and $\mathbf{M}$ with contracted indexing it becomes

$$\begin{aligned} \mathbf{A}^{\alpha\beta}_{mnp} &\equiv \mathbf{A}^{\alpha+3\beta}_{m+Nn+N^2p} = \mathbf{a}^{\alpha+3\beta}_{j} \\ \mathbf{B}^{\alpha\beta}_{rst} &\equiv \mathbf{B}^{\alpha+3\beta}_{r+Ns+N^2t} = \mathbf{b}^{\alpha+3\beta}_{i} \\ \mathbf{M}_{mnp,rst} &\equiv \mathbf{M}_{m+Nn+N^2p,e+Ns+N^2t} = \mathbf{M}_{ij} \end{aligned} \tag{20}$$

This system is also known as a system of "normal equations" for the least squares optimization

$$\sum_j \mathbf{M}_{ij} \mathbf{a}^{\alpha+3\beta}_j = \mathbf{b}^{\alpha+3\beta}_i \tag{21}$$

The optimization procedure allows as to compute the polynomial coefficients for best approximation of the tensor data within a region of a chosen point by a chosen degree of polynom. Then the value at the point $\mathbf{r}_p$, which is the origin in the $\{\zeta, \eta, \theta\}$ frame of reference, can be easily calculated using (13):

$$\bar{\mathbf{T}}_p = \sum_{mnp} \mathbf{A}_{mnp}\ \zeta^m \eta^n \theta^p|_{\zeta=\eta=\theta=0} = \mathbf{A}_{000} \tag{22}$$

It is important to notice, that the value of $\mathbf{A}_{000}$ depends on the order of polynomial used for fitting.

We also notice, that using a zero-order polynomial approximation (i.e., $N = 0$) is equivalent to finding a weighted average of a tensor function within the filter volume:

$$\bar{\mathbf{T}}_p = \int_V \mathbf{T}(\mathbf{r}_p + \mathbf{R}_p\{\zeta, \eta, \theta\})\ g(\zeta, \eta, \theta; \mathbf{T}_p)\ d\zeta d\eta d\theta \tag{23}$$

The major advantage of the higher order approximation is that it better preserves the absolute magnitude of the feature in the areas which have maxima or minima, compared to simple averaging, which tends to lower the height of the maxima and the depth of the minima.

Finally, for the filter function $g()$, we have chosen an anisotropic Gaussian weighting function G with axes aligned along the eigenvector directions and ellipsoidal semi-axes (the radii) proportional to the square root of corresponding eigenvalues.

$$g(\zeta, \eta, \theta; \mathbf{T}_p) = \\ \frac{1}{V} \exp\left(-(\zeta/(\sigma a))^2 - (\eta/(\sigma b))^2 - (\theta/(\sigma c))^2\right) \tag{24}$$

with

$$a = \sqrt{\lambda_1[\mathbf{T}_p]},\quad b = \sqrt{\lambda_2[\mathbf{T}_p]},\quad c = \sqrt{\lambda_3[\mathbf{T}_p]} \tag{25}$$

The variable $lambda_1$ is the largest eigenvalue of the diffusion tensor at the location $\mathbf{r}_p$. The value σ is a parameter that can enlarge or contract all of the ellipsoid radii. It is important to notice, that since we are trying to trace fibers, i.e., to extract structures with a very strong directional information, the filter is typically much more influenced by the data points "in front" and "behind" but not on the side. Thus, usually, $a >> b, c$.

Also note that in the equation for the filter function $g()$ (25), we have a choice of values for the diffusion tensor $\mathbf{T}_p$. In our algorithm we use the filtered tensor value from the previous time step, $\bar{\mathbf{T}}_{p-1}$, to determine the weighting function ellipsoid to use for the current time step.

[3] For Gaussian filter g(), the integral in the equation (18) can be expanded over the entire domain and evaluated analytically using Gamma functions

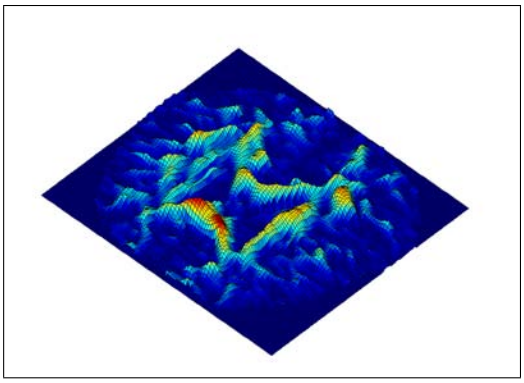

Figure 6: Height plot for anisotropy measure ("mountain" function) described in Section 2.6 for an axial slice of the data. The higher portions, red, corresponds to stronger anisotropy. See Eq. (30).

2.5 Streamline Integration

The fiber tract trajectory $\mathbf{s}(\tau)$ can be computed as a parametric 3D curve through linear integration of the filtered principal eigenvector:

$$\mathbf{s}(\tau) = \int_0^\tau \bar{\mathbf{e}}_1(t)dt \tag{26}$$

where t is a parameter of the curve and has corresponding $t = t(x, y, z)$ values and $\bar{\mathbf{e}}_1$ is the MLS filtered principal direction (unit) eigenvector as a function of position.

$$\bar{\mathbf{T}}\bar{\mathbf{e}}_1 = \bar{\lambda}_1\bar{\mathbf{e}}_1 \tag{27}$$

The discrete integration can be done numerically using explicit or implicit methods depending on the converging/diverging nature of the tensor field. The simplest approaches are a forward (for diverging fiber fields)

$$\mathbf{r}_{new} = \mathbf{r}_{old} + \bar{\mathbf{e}}_1[\bar{\mathbf{T}}(\mathbf{r}_{old})]\Delta t \tag{28}$$

or inverse Euler schemes (for converging fiber fields):

$$\mathbf{r}_{new} = \mathbf{r}_{old} + \bar{\mathbf{e}}_1[\bar{\mathbf{T}}(\mathbf{r}_{new})]\Delta t \tag{29}$$

One can easily employ higher order integration schemes, but they still should be chosen according to the local properties of the tensor field (converging or diverging) that are associated with the "stiffness" of the differential equation, bifurcations and the desired geometry.

2.6 The "Mountain" function

For the continuous tensor field, we use a anisotropy measure height function $c_\ell(x, y, z)$, defined using a continues version of Eq. 4:

$$c_\ell(x, y, z) = \frac{\lambda_1 - \lambda_2}{\lambda_1 + \lambda_2 + \lambda_3} \tag{30}$$

where λ_i are eigenvalues of $T(x, y, z)$.

Metaphorically, we call this a "mountain function" because we initiate the fibers at the high points and peaks of the mountain (the most highly directional portions of a region) and grow them following the major eigenvector directions. The metaphor continues as the anisotropy measure decreases; we let the fibers grow until they go "under water" into the lakes (corresponding to a chosen lower value for the anisotropy measure); the low anisotropy values indicate an absence of fibers.

We can also incorporate the mountain function within the filter function $g()$ itself, so that the higher regions will be given more weight in the scheme.

2.7 Fiber Tracing Algorithm

The algorithm starts with the user selecting a rectangular starting region. The fibers are traced starting from the points only where the anisotropy measure is bigger than the threshold, i.e., that are high enough on the mountainside. The initial direction will be determined by the "largest" eigenvector of locally filtered tensor field. At this point the filter is not oriented. The tracing will proceed in two opposite directions along the "largest " eigenvector.

The tracing procedure integrates forward from the provided initial point and initial direction using forward or inverse Euler method. It then computes a filtered value of the tensor at the new point using the oriented filter (orientation and width of the filter is determined from the previous position: the filter is oriented along the "largest" eigenvector and is shaped according to the eigenvalues, with largest semi-axis along the "largest" eigenvector). If the anisotropy of the new point is greater than threshold value, the point is accepted and the tracing continues, otherwise the tracing is finished. The tracing routine also chooses the direction of tracing consistent with previous steps (no turn $>$ 90 degrees is allowed).

```
1.  User inputs starting region
2.  For every starting point P in the region
{
     Tp = filter(T,P,sphere);
     cl = anisotropy(Tp);
     if (cl > eps){
         e1 = direction(Tp);
         trace1 = fibertrace(P, e1);
         trace2 = fibertrace(P,-e1);
         trace = trace1 + trace2;
     }
   }

   trace = fibertrace(P,e){
     trace->add(P);
     do {
         Pn = integrate_forward(P,e1,dt);
         Tp = filter(T,Pn,ellipsoid,e1);
         cl = anisotropy(Tp)
         if (cl>eps){
         trace->add(Pn);
         P = Pn;
         e1 = direction(Tp);
         }
        } while (cl >eps)
    return(trace);
 }
}
```

We have also incorporated some simple mechanisms to ignore very short fibers and to stop tracing when the length of the fiber exceeds an allowed limit. The starting points are usually generated on a grid within user defined regions. We use numerical integration to evaluate the integrals (18)-(19) inside the filter. We use SVD and LU factorization routines from the "Numerical Recipes" [16] to solve the linear system (21). Evaluation of the tensor function $\mathbf{T}$ at the center of the filter (origin) requires only the first coefficient of the polynomial expansion (22), so we use only a single back-substitution procedure in LU factorization.

3 Results: Brain Anatomy

The DT-MRI data set we used has 121x88x60 voxels which provides resolution of roughly $1mm^3$. We used various orders of

polynomial approximation from zero (average within the filter) up to 3rd in each dimension. We were able to trace bundles of fibers which correspond to well known anatomical structures, such as: anterior and posterior forceps, corona radiata, optic radiation, cranial nerves, U-shape fibers, (superior) longitudinal fasciculus, cingulum and corpus callosum. The results are illustrated and further explained in the figures Fig. (7) to Fig.(12).

4 Conclusions and Future Work

In this paper we developed a new technique for tracing anatomical fibers from 3D DT-MRI tensor fields, recovering identifiable anatomical structures that correspond to many of the white matter brain-fiber pathways.

We found that simple component-wise interpolation of the tensors, forming a crude continuous approximate tensor field, worked well for extracting brain fiber directions when combined with a moving least squares filtering approach.

Our plan is to extend the work to more advanced nonlinear filtering methods to better handle bifurcation points like crossings and "T" junctions. In addition, we plan to examine the benefits of detecting the local stiffness properties of the tensor field related to convergence or divergence of the fibers and automatically switch from forward to inverse integration schemes.

5 Acknowledgments

We would like to thank Yarden Livnat for the initial discussion, Gordon Kindlmann and SCI Institute for the data, David Breen for suggestions. We are also grateful to our reviewers for multiple valuable comments and suggestions.

This work was supported by National Science Foundation grants #ASC-89-20219 and #ACI-9982273, and the National Institute on Drug Abuse and the National Institute of Mental Health, as part of the Human Brain Project.

References

[1] P.J. Basser, J. Mattiello, and D. LeBihan. Mr diffusion tensor spectroscopy and imaging. *Biophysical Journal*, 66:259–267, 1994.

[2] P.J. Basser, S. Pajevic, C. Peropaoli, J. Duda, and A. Aldroubi. In vivo fiber tractography using dt-mri data. *Magnetic Resonance in Medicine*, 44:625–632, 2000.

[3] P.J. Basser and C. Pierpaoli. Microstructural and physiological features of tissues elucidated by quantitative-diffusion-tensor mri. *J. Magn. Res. Ser. B*, 111:209–219, 1996.

[4] A.I. Borisenko and I.E. Tarapov. *Vector and Tensor Analysis*. Dover, New York, 1979.

[5] B. Cabral and L.(C.) Leedom. Imaging vector fields using line integral convolution. In *Proceedings of SIGGRAPH 93*, pages 263–272, 1993.

[6] M.J. da Silva, S. Zhang, C. Demiralp, and D.H. Laidlaw. Visualizing diffusion tensor volume differences. In *IEEE Visualization 01 Proceedings, Work in Progress*, 2001.

[7] T. Delmarcelle and L. Hesselink. Visualization of second order tensor fields and matrix data. In *IEEE Visualization 92 Proceedings*, pages 316–323, 1992.

[8] C. Gossl, L. Fahrmeir, B. Putz, L.M. Auer, and D.P. Auer. Fiber tracking from dti using linear state space models: Detectability of the pyramidal tract. *NeuroImage*, 16:378–388, 2002.

[9] R.W. Hamming. *Digital Filters*. Prentice-Hall, 1983.

[10] G. Kindlmann and D. Weinstein. Hue-balls and lit-tensors for direct volume rendering of diffusion tensor field. In *IEEE Visualization 99 Proceedings*, pages 183–189, 1999.

[11] G. Kindlmann, D. Weinstein, and D. Hart. Strategies for direct volume rendering of diffusion tensor fields. *IEEE Trans. on Visualization and Computer Graphics*, pages 124–138, 2000.

[12] D.H. Laidlaw, E.T. Ahrens, D. Kremers, M. J. Avalos, R.E. Jacobs, and C. Readhead. Visualizing diffusion tensor images of the mouse spinal cord. In *IEEE Visualization 98 Proceedings*, pages 127–134, 1998.

[13] W. Lorenson and H. Cline. Marching cubes: A high resolution 3d surface construction algorithm. In *Computer Graphics (Proceedings of SIGGRAPH '87)*, volume 21, pages 163–169, 1987.

[14] O.Coulon, D.C. Alexander, and S.R. Arridge. A regularization scheme for diffusion tensor magnetic resonance images. In *IPMI'2001, XVIth International Conference on Information Processing in Medical Imaging, in Lecture Notes in Computer Science*, volume 2082, pages 92–105, 2001.

[15] S. Pajevic, A. Aldroubi, J. Duda, and P.J. Basser. A continuous tensor field approximation of discrete dt-mri data for extracting microstructural and architectural features of tissues. *Journ. Magn. Res.*, 154:85–100, 2002.

[16] W.H. Press, S.A. Teukolsky, W.T. Vetterling, and B.P Flannery. *Numerical Recipes in C*. Cambridge University Press, 1992.

[17] T.C. Pritchard and K.D. Alloway. *Medical Neuroscience*. Fence Creek Publishing, 1999.

[18] B. Steve and C. Levit. The virtual wind tunnel: An environment for the exploration of three-dimensional unsteady flows. In *IEEE Visualization 91 Proceedings*, pages 17–24, 1991.

[19] G. Turk and D. Banks. Image guided streamline placement. In *Proceedings of SIGGRAPH 96*, pages 453–460, 1996.

[20] D. Weinstein, G. Kindlmann, and E. Lundberg. Tensorlines: Advection-diffusion based propagation through diffusion tensor fields. In *IEEE Visualization 99 Proceedings*, pages 249–253, 1998.

[21] C.-F. Westin, S.E. Maier, B. Khidhir, P.Everett, F.A. Jolesz, and R.Kikinis. Image processing for diffusion tensor magnetic resonance imaging. In *Proceedings of MICCAI'99*, pages 441–452, 1999.

[22] C.-F. Westin, S. Peled, H. Gubjartsson, R. Kikinis, F.A. Jolesz, and R. Kikinis. Image processing for diffusion tensor magnetic resonance imaging. In *Proceedings of ISMRM'97*, 1997.

[23] S. Zhang, C. Curry, D. Morris, and D. Laidlaw. Streamtubes and streamsurfaces for visualizing diffusion tensor mri volume images. In *IEEE Visualization 00 Proceedings, Work in Progress*, 2000.

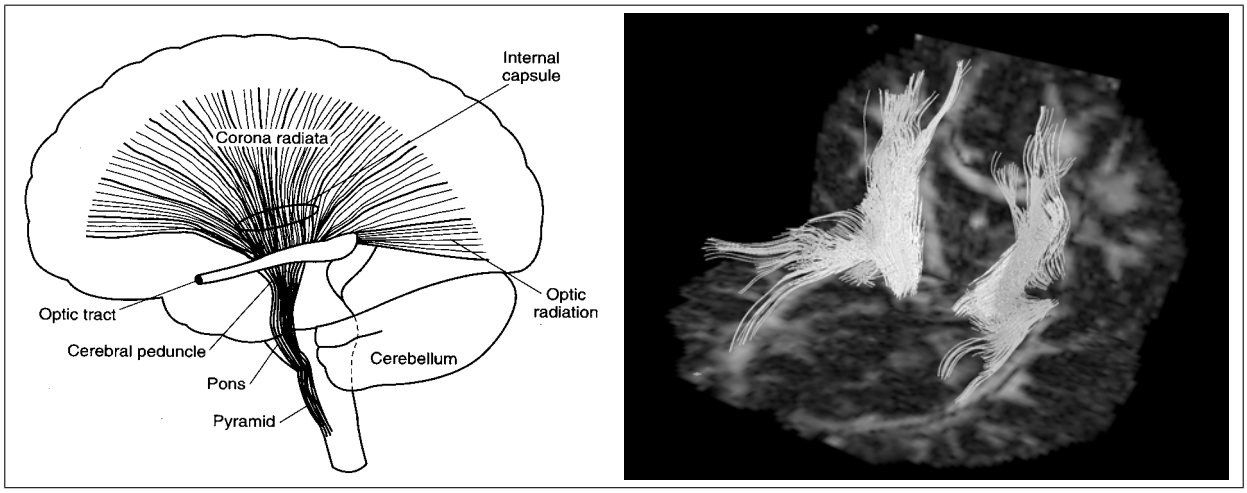

Figure 7: Brain structures: Corona radiata. The diagram on the left is from [17]. On the right the fibers are reconstructed from DT-MRI data using our oriented tensor reconstruction (OTR) algorithm. The corona radiata is visible in both hemispheres.

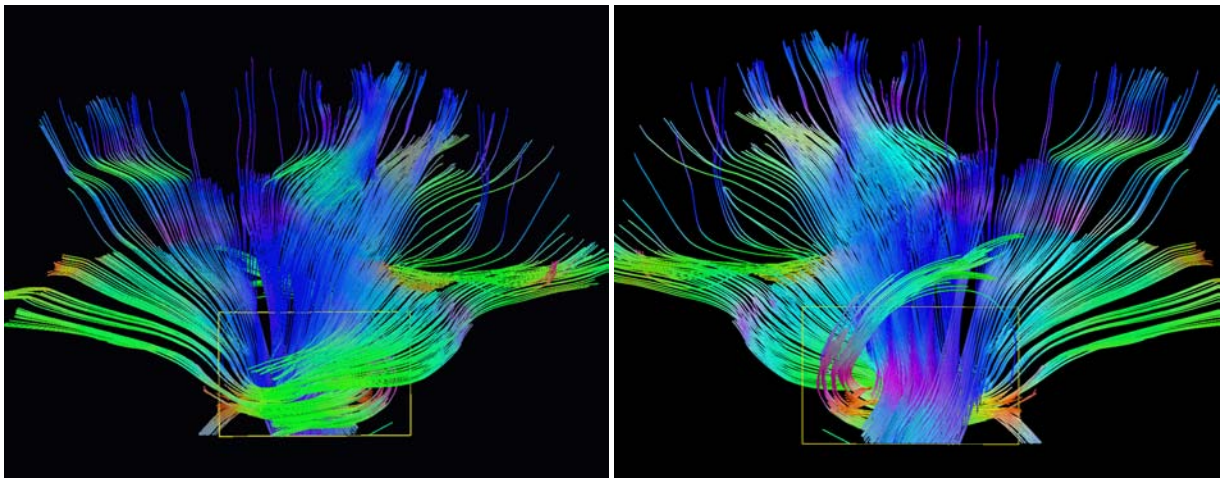

Figure 8: Right hemisphere corona radiata shown from opposite directions. The yellow boxes show the seed region for the OTR fiber tracing algorithm. Color coding indicates orthogonal directions in the amount of RGB (XYZ).

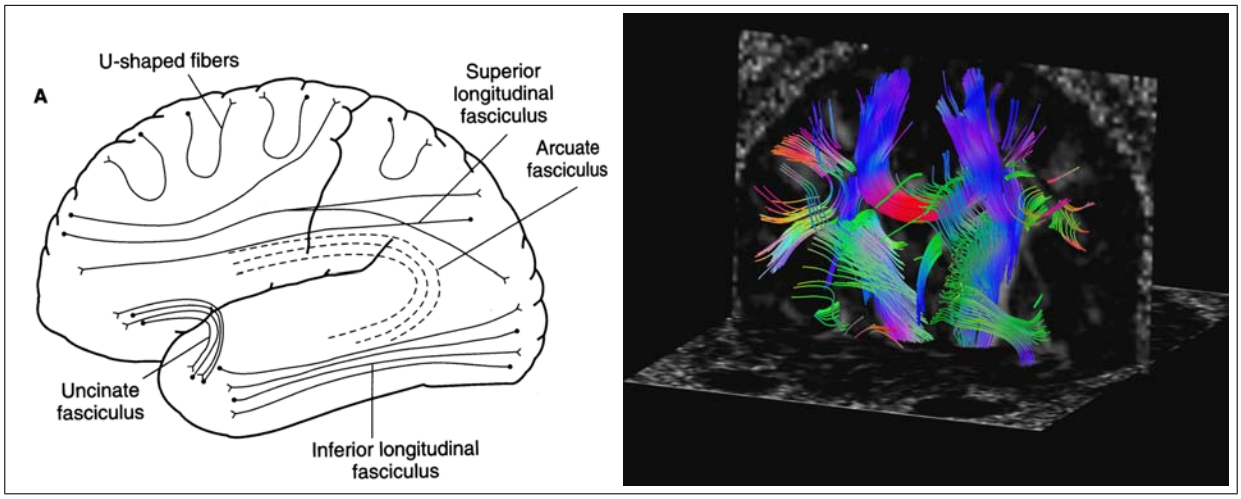

Figure 9: Brain structures: U-shaped fibers, parts of corona radiata and corpus callosum. The diagram on the left is from [17]. Note the short U-shaped fibers in the left upper part of the right image. Color coding is the same as the previous figure.

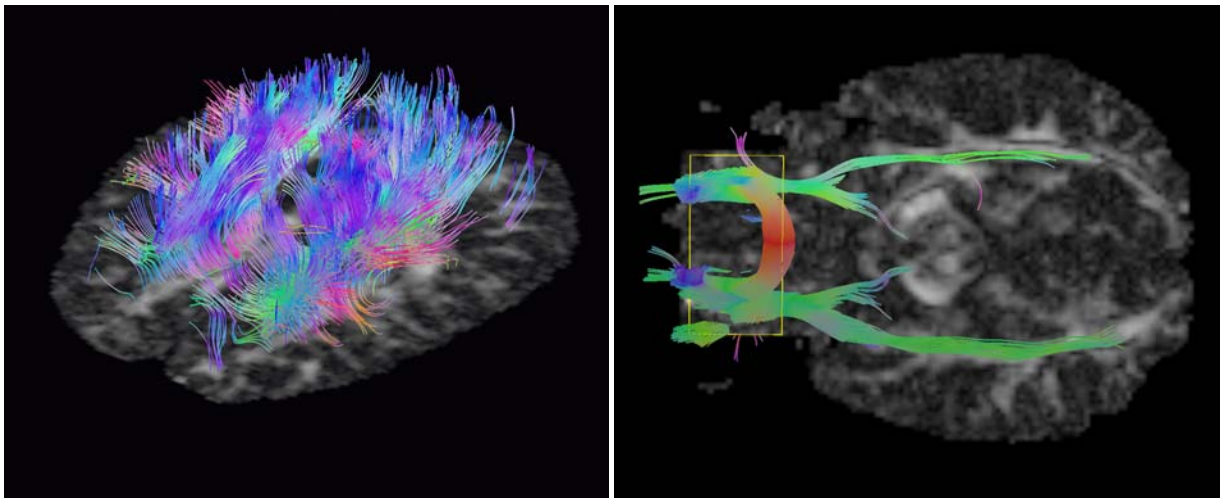

Figure 10: Brain structures: Fibers near the cortical surface and U-shaped fibers on the left; optic tract on the right.

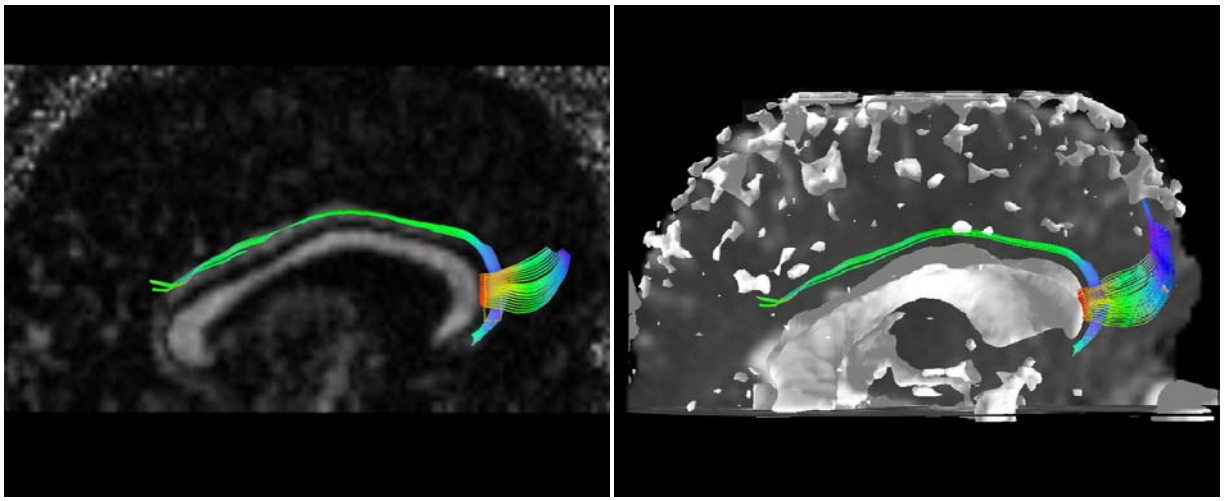

Figure 11: Brain structures: on the left is a side view of the right hemisphere cingulum bundle on the background of corresponding c_l anisotropy; on the right the same structure together with 3d models of the ventricle and CSF (cerebrospinal fluid) extracted by isosurfacing [13] on isotropic part c_s (see Eq. (6)) of the same DT-MRI dataset.

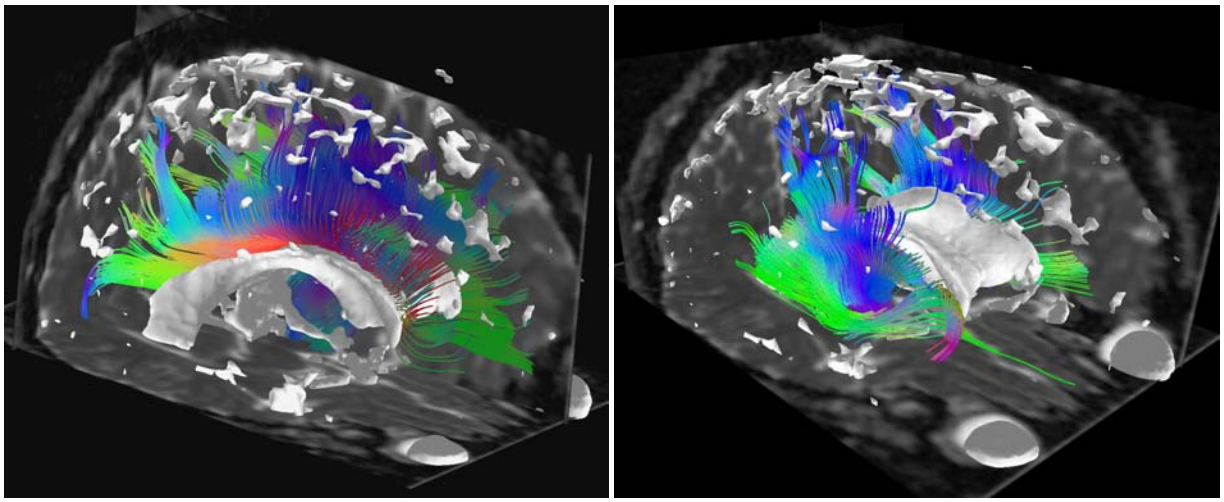

Figure 12: Brain structures: corpus callosum (left) and corona radiata (right) shown together with isotropic brain structures - ventricle, eye sockets and pockets of CSF on the top of the brain. Cutting planes show isotropic c_s values.