Diffusion Tensor Estimation Method

Robert Cox and Daniel Glen
Scientific and Statistical Computing Core
NIMH

DTI Overview

MRI imaging method that shows relative rates and directions of diffusion corresponding to flow principally along fiber tracts

Applications for diseases involving white matter including perinatal brain injury, stroke, multiple sclerosis, tumors, ….

Acquisition of MRI volume followed by at least six or more volumes (typically many more) with an additional gradient applied from non-collinear directions

$$I^{(q)} = J\, e^{-b^{(q)} \cdot D}$$

Where

$I^{(q)}$ = the image voxel intensity for each gradient q

J = the ideal image intensity without applied gradient

usually taken as $J = I^{(0)}$

$b^{(q)}$ = "b-matrix" = $\gamma^2 G_i G_j \delta^2(\Delta-\delta/3)$ for the qth encoding gradient

γ = 267.5 rad/ms·mT,

Δ= time lag between starts of gradient,

δ = duration of gradient

D = the Diffusion tensor, Diffusion in 6 principal directions,

Dxx, Dxy, Dxz, Dyy, Dyz, Dzz

$$\ln(I^{(0)} / I^{(q)}) = b^{(q)} \cdot D$$

$$D = \ln(I^{(0)} / I^{(q)}) \times b^{-1}$$

Eigenvalue calculations

$$D\,V = \lambda\,V$$

Where

- D = diffusion tensor in a symmetric, square matrix form (3x3)
- V = the eigenvector, a vector corresponding to an orientation (3x1)
- λ = the eigenvalue, a scalar constant

For a 3x3 matrix, there are 3 sets of orthogonal eigenvector and eigenvalue solutions

$$\det(D - \lambda I) = 0$$

$$(D - \lambda I)\,V = 0$$

Solved with f2c converted eispack routine in AFNI using a tridiagonal reduction followed by a QL2 solution for the eigenvalues and vectors

General Measures

- Fractional Anisotropy (FA)

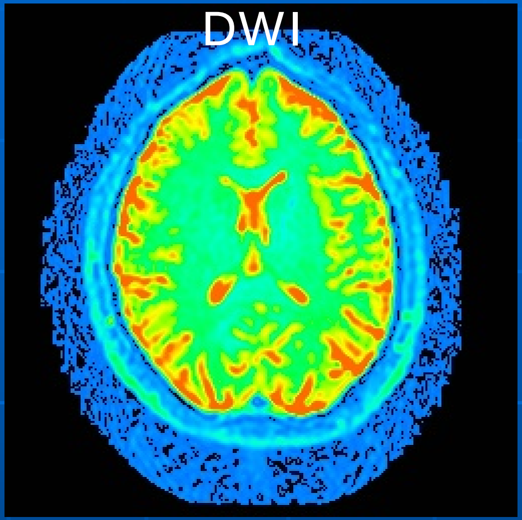
$$FA = \frac{\sqrt{(\lambda_1 - \lambda_2)^2 + (\lambda_1 - \lambda_3)^2 + (\lambda_2 - \lambda_3)^2}}{\sqrt{2}\,\sqrt{\lambda_1{}^2 + \lambda_2{}^2 + \lambda_3{}^2}}$$

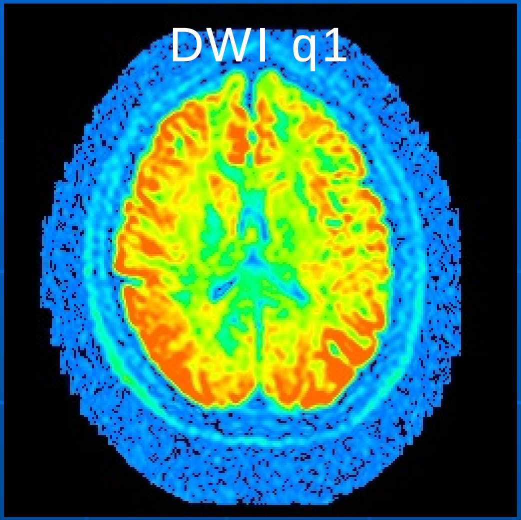
- Lattice Index (LI) = $\Sigma\, a_n LI_n \,/\, \Sigma\, a_n$

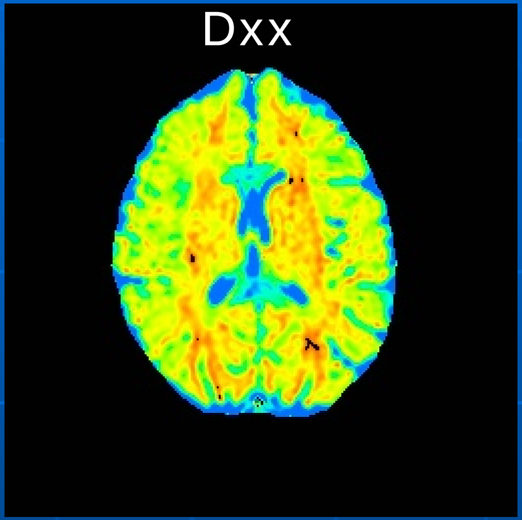
$$LI_n = \quad \Lambda_N = \frac{\sqrt{3}}{\sqrt{8}} \frac{\sqrt{D_{ref}:D_N}}{\sqrt{\mathrm{D_{ref}}:\mathrm{D_N}}} + \frac{3}{4} \frac{D_{ref}:D_N}{\sqrt{\mathrm{D_{ref}}:\mathrm{D_{ref}}}\,\sqrt{\mathrm{D_N}:\mathrm{D_N}}}$$

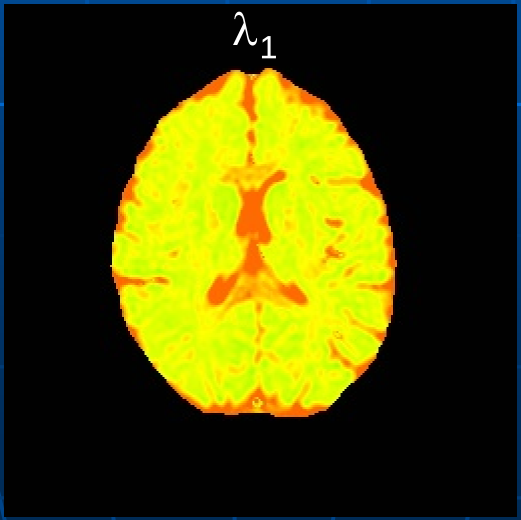
- Mean Diffusivity $\tilde{\lambda} = (\lambda_1 + \lambda_2 + \lambda_3) / 3$

DWI, DT, Eigenvalue Samples

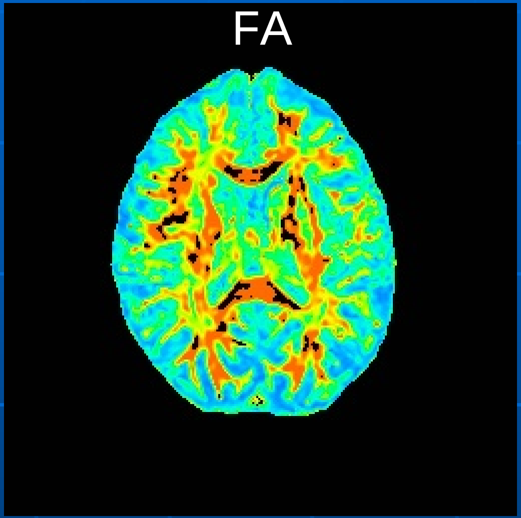




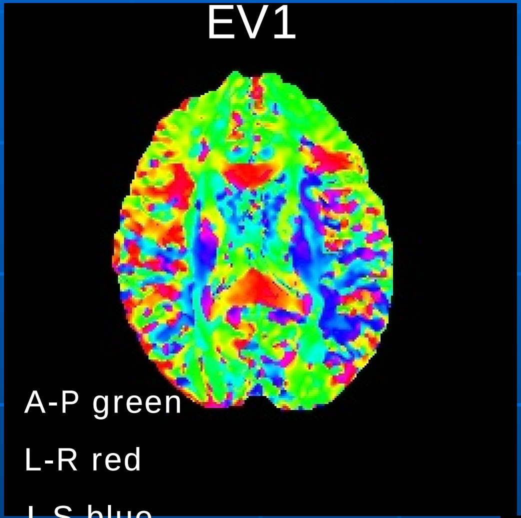




DTI images



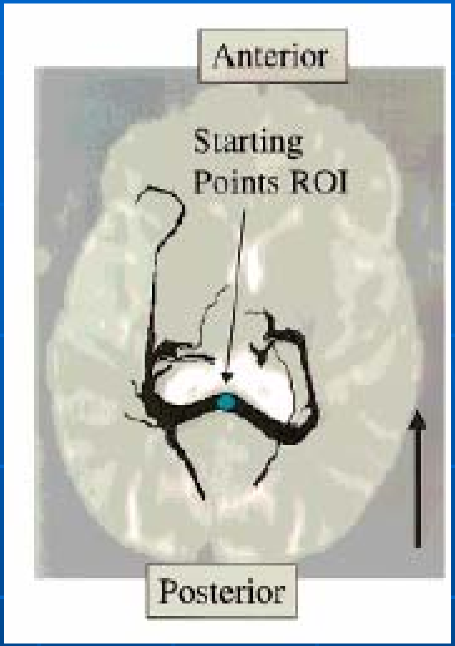
Color coded

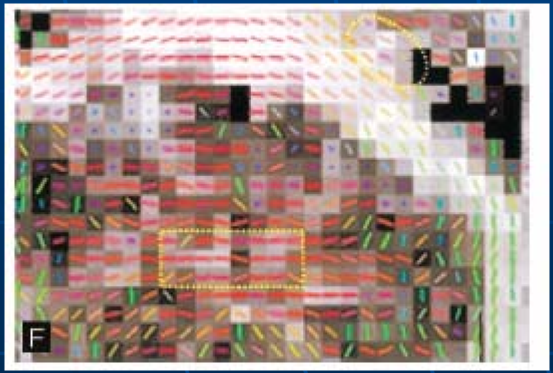


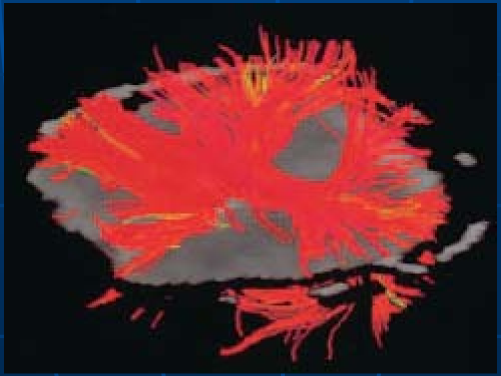
A-P green

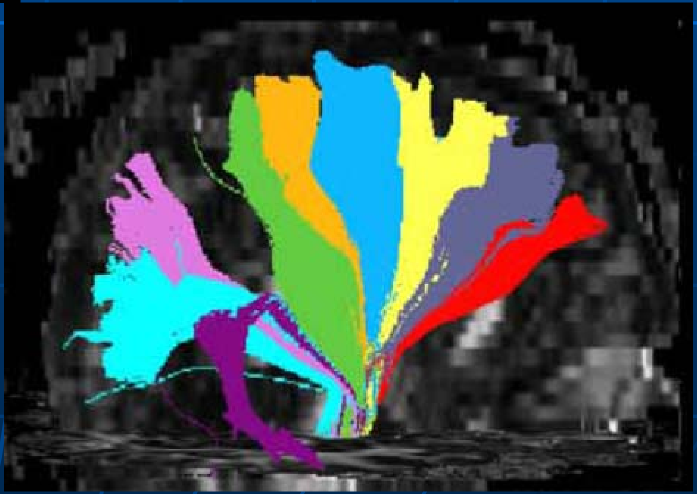
L-R red

I-S blue









Negative Eigenvalues

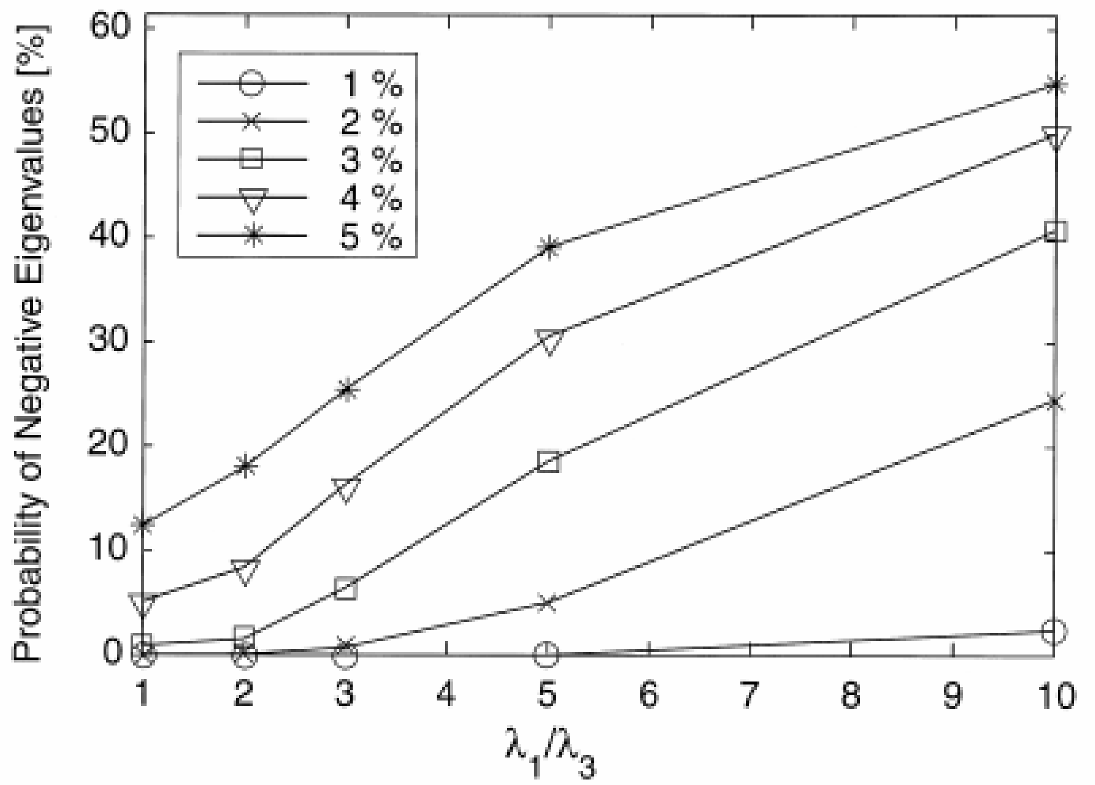


Fig. 2. The probability of obtaining negative eigenvalues for the 'rice' shaped diffusion tensor as a function of λ_1/λ_3 at different noise levels (1–5% of S_0).

Skare, et. al, Mag. Res. Imaging, 18, 2000

Negative Eigenvalue Solutions

Negative values more likely with increasing FA and with increasing noise (Skare, et al, Basser)

- Calculate FA with negative eigenvalues reset to 0 (GE, Hopkins DTI Studio)
- Use another index, e.g. Lattice Anisotropy Index (Basser)
- Spatial smoothing of DWI images (Hahn, et al)
- Temporal smoothing with repeat experiments (Skare, et al)
- Calculate D, limiting D to positive definite matrix (Tschumperlé and Deriche, Mangin et al)

Direct Least Squares for D

$$I^{(q)} = J\, e^{-b^{(q)} \cdot D} + \text{noise}$$

- Noise – Johnson noise, Eddy currents, motion including periodic beat of CSF with blood flow, partial volume effect
- Find the symmetric non-negative definite matrix D that minimizes the Error functional (cost function)

$$E(D, J) = \tfrac{1}{2} \sum_{q} w^{(q)} \left(J\, e^{-b^{(q)} \cdot D} - I^{(q)}\right)^2$$

Generally, estimate D, adjust through a gradient step to find new estimate for D until D converges. Construct gradient step to guarantee D is always positive definite (no negative eigenvalues).

Initial Estimate of D, D_0

- Estimate for D the traditional way and find eigenvalues, vectors too
- Limit eigenvalues to positive ones by setting λ_2, λ_3 eigenvalues to be at least $0.2 * \lambda_1$
- Recompute $D_0 = U \Lambda U^T$

Where $U = [u_1, u_2, u_3]$ matrix of eigenvectors

$\Lambda = \mathrm{diag}(\lambda_1, \lambda_2, \lambda_3)$

Compute J (Ideal image voxel value)

$$E(D, J) = \tfrac{1}{2} \sum w^{(q)} \left(J\, e^{-b^{(q)} \cdot D} - I^{(q)}\right)^2$$

$$\partial E/\partial J = \sum w^{(q)} \left(J\, e^{-b^{(q)} \cdot D} - I^{(q)}\right) e^{-b^{(q)} \cdot D}$$

$$J = \left(\sum w^{(q)}\, I^{(q)}\, e^{-b^{(q)} \cdot D}\right) / \left(\sum w^{(q)}\, e^{-2b^{(q)} \cdot D}\right)$$

Modified Gradient Descent

Gradient of E (error) with respect to D = F

$$F = \sum w^{(q)} \left(J\, e^{-b^{(q)} \cdot D} - I^{(q)}\right) b^{(q)}$$

We want to change D to minimize E the fastest. From definition of gradient,

$$\partial D / \partial \tau = -F \quad \text{(we don't use this)}$$

Where τ is pseudo-time in the descent.

But this doesn't prevent D from becoming non-positive definite, so instead …

Modified Gradient Descent

$$\partial D / \partial \tau = -(FD^2 + D^2F)$$

This can be shown to be the fastest descent while remaining positive definite

We will do the descent with finite steps

If $\partial D / \partial \tau = -(ND + DN)$ with N as a constant matrix, it can also be shown

$$D(\tau + \Delta\tau) = e^{(-\Delta\tau N)}\, D(\tau)\, e^{(-\Delta\tau N)}$$

Let N=FD and approximately constant over $\Delta\tau$ step

$$D(\tau + \Delta\tau) = e^{(-FD\Delta\tau)}\, D(\tau)\, e^{(-FD\Delta\tau)}$$

Modified Gradient Descent

We can replace e^{-x} with the Padé approximant (similar to a Taylor series expansion)

$$e^{-x} \sim (1 - x/2) / (1 + x/2)$$

Similarly, for a matrix exponential,

$$e^{-M} \sim (I - \tfrac{1}{2} M) (I + \tfrac{1}{2} M)^{-1}$$

If we let

$$H_{\pm} = I \pm \tfrac{1}{2} \Delta\tau \, FD$$

Then

$$D(\tau + \Delta\tau) = H_{-}(\Delta\tau) \, H_{+}(\Delta\tau)^{-1} \, D(\tau) \, H_{+}(\Delta\tau)^{-1} \, H_{-}(\Delta\tau)$$

$$D(\tau + \Delta\tau) = A(\Delta\tau) \, D(\tau) \, A(\Delta\tau)^{T}$$

where $A = H_{-} H_{+}$, which will always be symmetric and positive definite

$\Delta\tau$, pseudo-time step size calculation

Initial $\Delta\tau$ conservatively estimated

$$\Delta\tau_0 = 0.01 \sum |D_0| \ / \ \sum |G|$$

where $G = FD^2 + D^2F$

Start with initial calculation of J, E(D,J) (Cost function)

Take trial step of $\Delta\tau_0$

$$D(\tau + \Delta\tau) = A(\Delta\tau)\ D(\tau)\ A(\Delta\tau)^T$$

Recalculate J and E(D, J)

If the new E(D, J) is less than the previous E(D,J), use this time step

Modified Gradient Descent Algorithm

- Compute D traditional linear way
- Compute eigenvalues and adjust
- Compute D based on new eigenvalues
- Calculate Ed = E(D,J) error
- Compute Initial $\Delta\tau$
- Take trial steps until convergence
  - Find acceptable trial step $\Delta\tau$ that gives lower Ed by halving the initial $\Delta\tau$ up to 10 times
  - Try step size of $2\Delta\tau$, $\Delta\tau$, $½\Delta\tau$. Compute corresponding Ed, D for each and pick the $\Delta\tau$, D that gives the lowest Ed. Use $\Delta\tau$ as the initial time step in the next convergence loop
  - Test convergence (starting with second step)
    - $\Sigma\ |D_{new} - D_{old}|\ /\ \Sigma\ |D_{new}| < 10^{-4}$
- Optionally recompute convergence loop with new weight factors

Weight Factor Computation

- The initial weight factors were all set to 1
- Recompute weight factors to downweight data points (gradients) that don't fit well (outliers)
- Compute residual at each gradient level from (0..n) as

$$r_q = J\, e^{-b^{(q)} \cdot D} - I^{(q)}$$

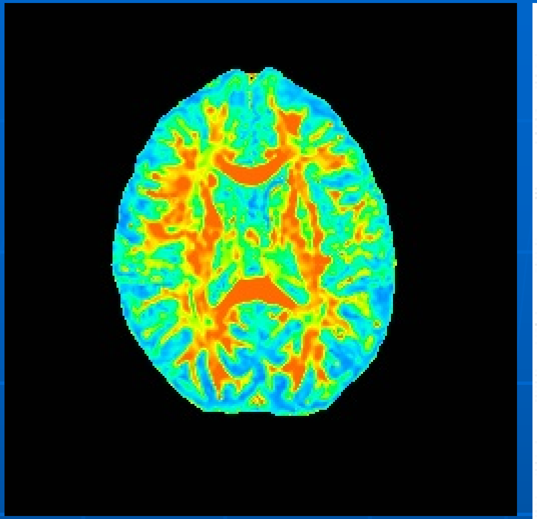
Estimate Std. Dev. as

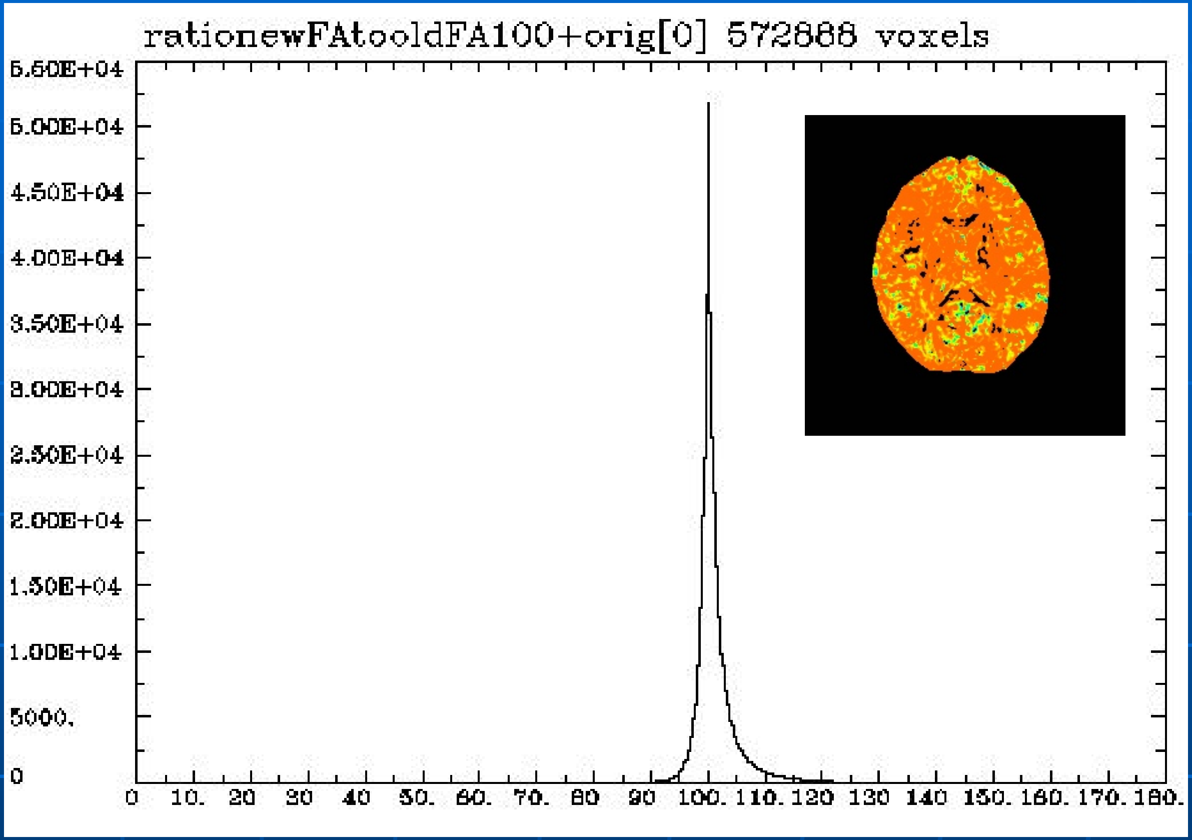
$$\sigma = [1/N_q \, \Sigma \, r_q^2]^{½}$$

$$w_q = [1 \,/\, \mathrm{sqrt}(\, 1 + (r_q /\sigma)^2)] \;*\; N_q \,/$$

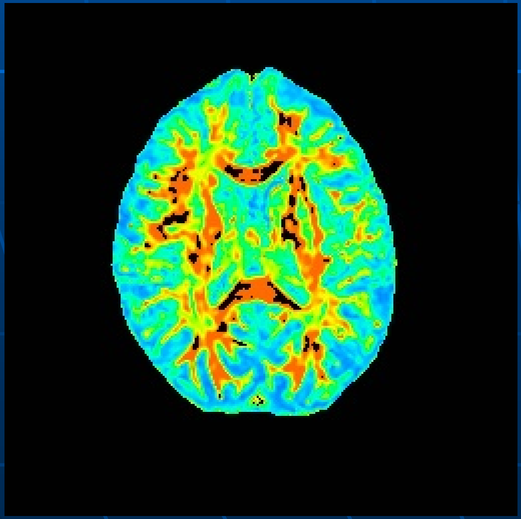
$$\Sigma \, [1 \,/\, \mathrm{sqrt}(\, 1 + (r_q /\sigma)^2)]$$

FA - non-linear

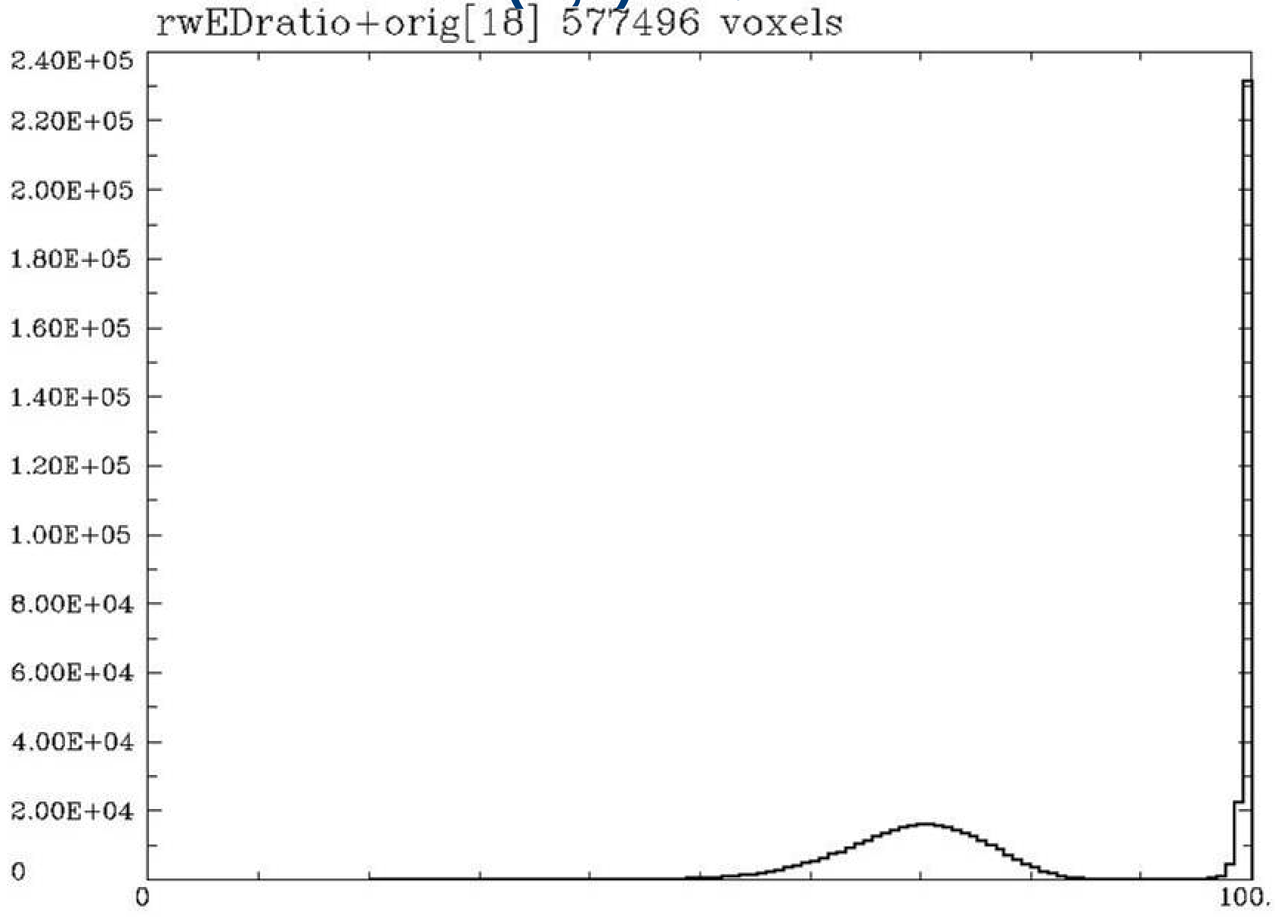




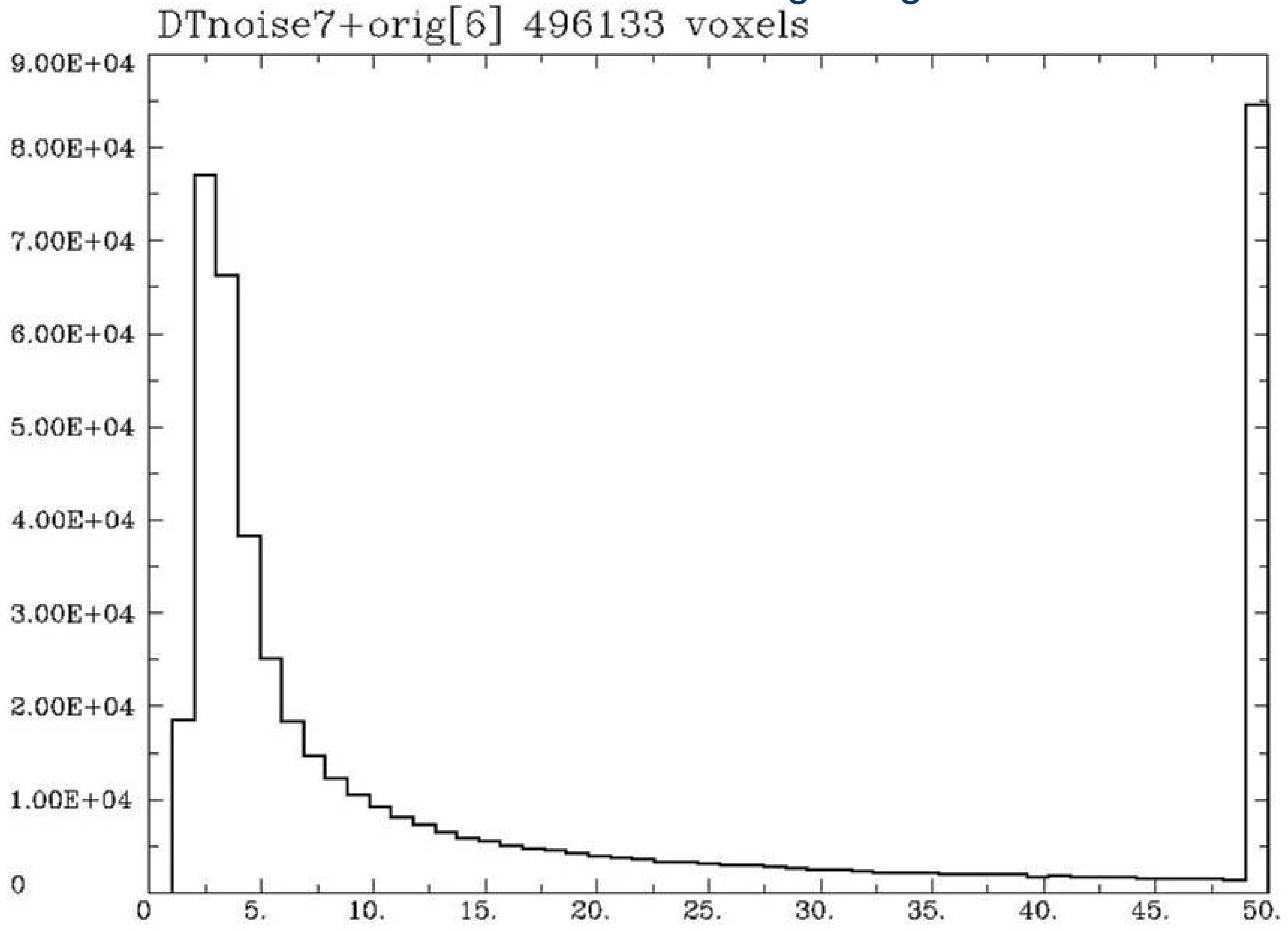
FA - linear



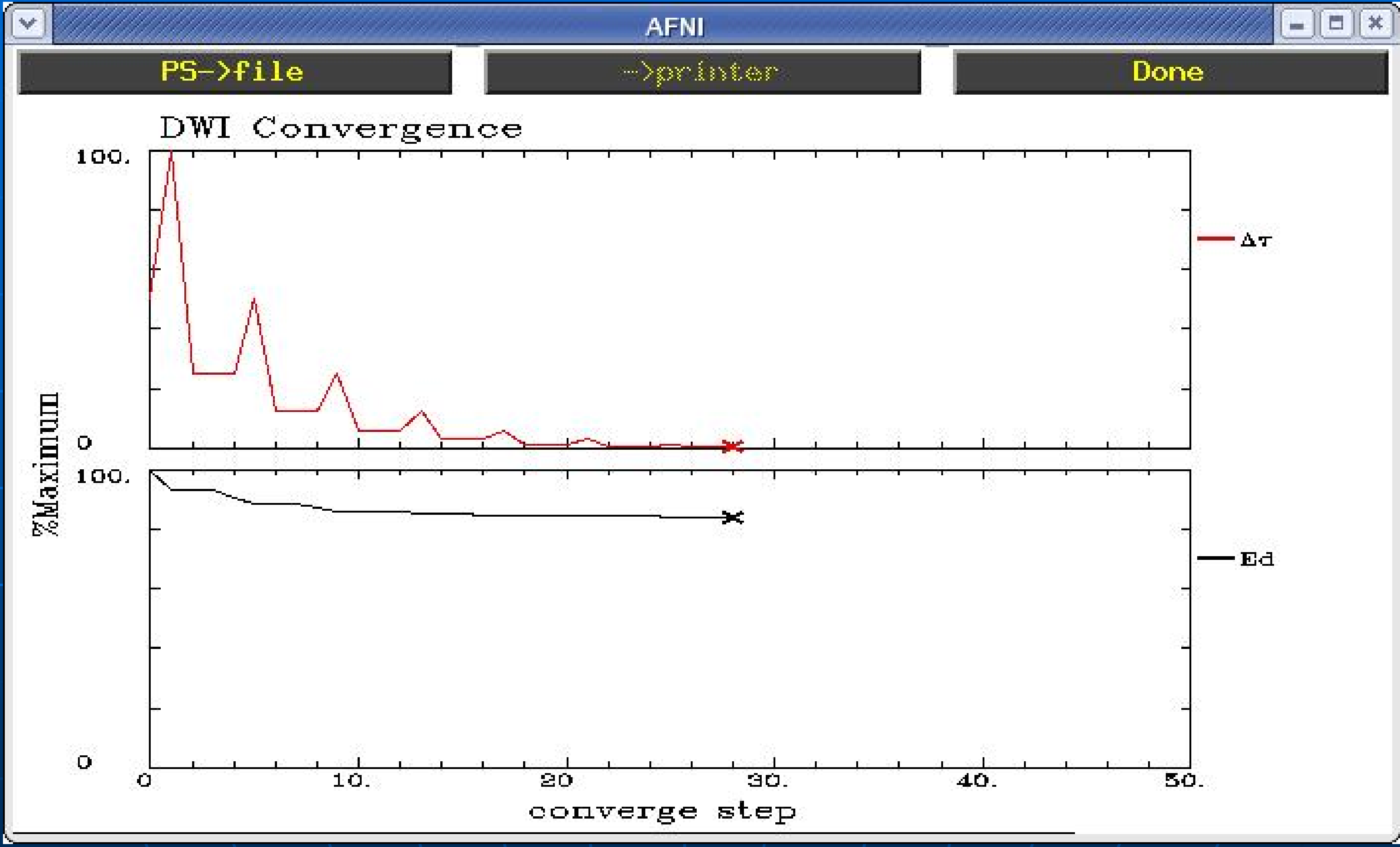
Ratio of final E(D,J) to E(D,J) at $\tau=0$



Number steps to convergence without reweighting



```
Usage: 3dDWItoDT [options] gradient-file dataset
Computes 6 principle direction tensors from multiple
      gradient vectors
 and corresponding DTI image volumes.
 The program takes two parameters as input :
    a 1D file of the gradient vectors with lines of ASCII
      floats Gxi,Gyi,Gzi.
    Only the non-zero gradient vectors are included in
      this file (no G0 line).
    a 3D bucket dataset with Np+1 sub-briks where
      the first sub-brik is the
    volume acquired with no diffusion weighting.
 Options:
   -automask =  mask dataset so that the tensors are
      computed only for
    high-intensity (presumably brain) voxels.  The
      intensity level is
    determined the same way that 3dClipLevel works.

   -nonlinear = compute iterative solution to avoid
      negative eigenvalues.
    This is the default method.

   -linear = compute simple linear solution

   -reweight = recompute weight factors at end of
      iterations and restart

   -max_iter n = maximum number of iterations for
      convergence (Default=10)
    values can range from -1 to any positive integer
      less than 101.
    A value of -1 is equivalent to the linear solution.
    A value of 0 results in only the initial estimate of
      the diffusion tensor
    solution adjusted to avoid negative eigenvalues.

-max_iter_rw n = max number of iterations after
reweighting (Default=5)
    values can range from 1 to any positive integer less
than 101.

   -eigs = compute eigenvalues, eigenvectors and
fractional anisotropy in
    sub-briks 6-18. Computed as in 3dDTeig

   -debug_briks = add sub-briks with Ed (error
functional), Ed0 (original error)
    and number of steps to convergence

   -cumulative_wts = show overall weight factors for
each gradient level
    May be useful as a quality control

   -verbose nnnnn = print convergence steps every
nnnnn voxels that survive to
    convergence loops (can be quite lengthy)

   -drive_afni = show convergence graphs every nnnnn
voxels that survive to convergence loops. AFNI must
have NIML communications on (afni -niml).

 Example:
  3dDWItoDTnoise -prefix rw01 -automask -reweight -
max_iter 10 \
            -max_iter_rw 10 tensor25.1D grad02+orig.

 The output is a 6 sub-brick bucket dataset containing
Dxx,Dxy,Dxz,Dyy,Dyz,Dzz. Additional sub-briks may be
appended with the -eigs and -debug_briks options.
 These results are appropriate as the input to the
3dDTeig program.
```



AFNI
PS->file
->printer
Done
DWI Convergence
100.
0
100.
0
%Maximum
Δτ
Ed
0
10.
20
30.
40.
50.
converge step

Other Methods Comparison

- Tschumperlé and Deriche, Variational Framework
  - Simultaneous spatial smoothing
  - Complicated cost function versus x^2

    $\Psi(\ln (I^{(0)}/I^{(q)}) - g_k^T D g_k) + \alpha \phi (|\nabla D|)$

    where $\Psi(s) = \log(1+s^2)$ and $\phi(s) = \mathrm{sqrt}(1+s^2)$

    We use $\Psi(s) = s^2$, $\alpha = 0$ (no spatial contribution)
  - Slightly more complicated gradient function
    - $G = (F+F^T)D^2 + D^2 (F+F^T)$
  - Linearized with $\ln(I^{(0)} / I^{(q)})$ versus non-linear relationship, $I^{(q)} = J e^{-b^{(q)} \cdot D} +$ noise
  - No reweighting
- Mangin, et al, Robust Tensor Estimation
  - Cost function, Geman-McLure M-estimator, made to remove outliers, $\varepsilon_i^2 / (\varepsilon_i^2+C^2)$ where $C=1.48 \,\mathrm{median}_i |\varepsilon_i|$ (we use reweighting)
  - Similar to traditional method, does not enforce positive definiteness on D

Future Directions

- Create and show fiber tracts in SUMA and AFNI
- Test model with computed DWI and artificial noise
- Add other indices (Lattice Index, Mean diffusivity,...)
- Refine model and algorithm
- Respond to AFNI user requests

Acknowledgements

- Rick Reynolds
- Rich Hammett
- Ziad Saad
- Peter Basser
- Wolfgang Gaggl, Fernanda Tovar-Moll