Tensor Invariants, their Gradients, and their Failings

Gordon Kindlmann

Scientific Computing and Imaging Institute
School of Computing
University of Utah

Rough Outline

Shape: invariants, eigenvalues, eigenvalue moments

Variety of invariants between Trace and λ_1

Geometric and visual intuition

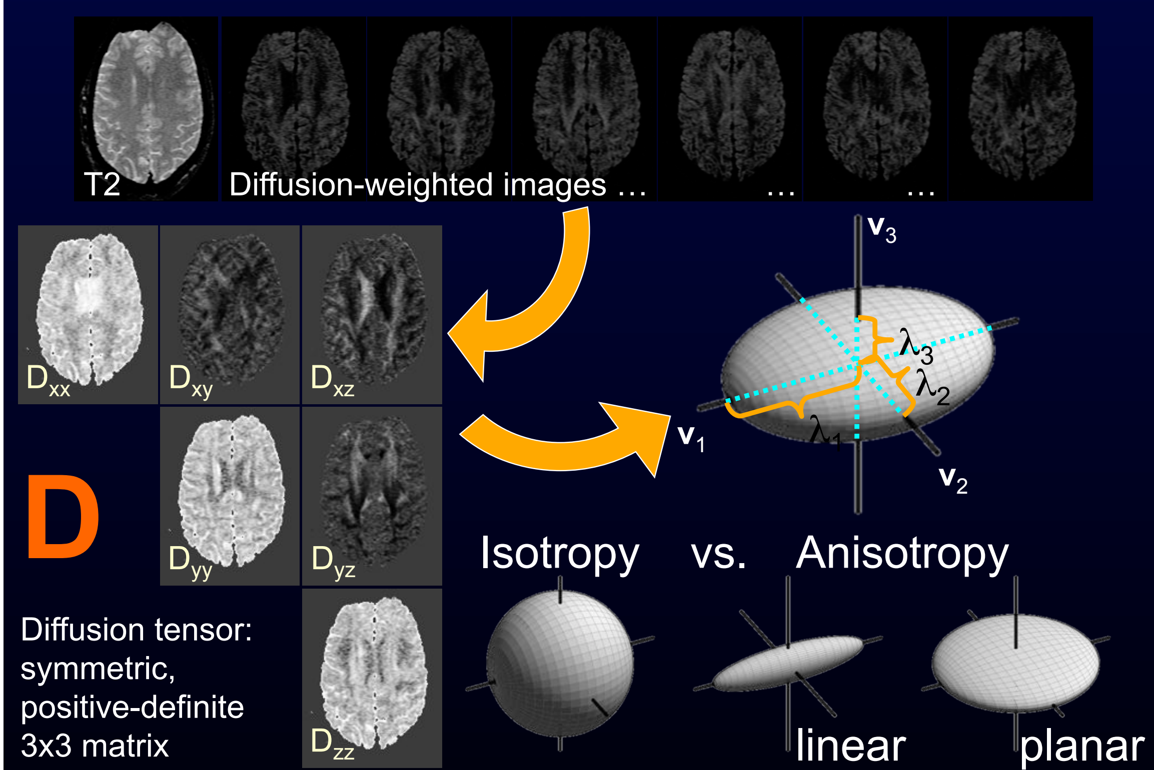
Invariant gradients

1. WRT position: Shading in rendering
2. WRT tensor components: DOF of shape

Sometimes invariant grads fail as shape DOF

Needs a fix for filtering

Diffusion Tensor MRI



Invariants

f(**D**) is invariant ⇔ f(**D**) = f(**RDR**$^{-1}$) ∀ **R**

Characteristic equation of **D**: det(**D** - λ**I**) = 0 ⇒

$$\lambda^3 - J_1\lambda^2 + J_2\lambda - J_3 = 0$$

$$\det(\mathbf{RDR}^{-1} - \lambda\mathbf{I}) = \det(\mathbf{R})\det(\mathbf{D} - \lambda\mathbf{I})\det(\mathbf{R}^{-1})$$

$$= \det(\mathbf{D} - \lambda\mathbf{I}) \Rightarrow$$

J_1, J_2, J_3 are ("principal") invariants:

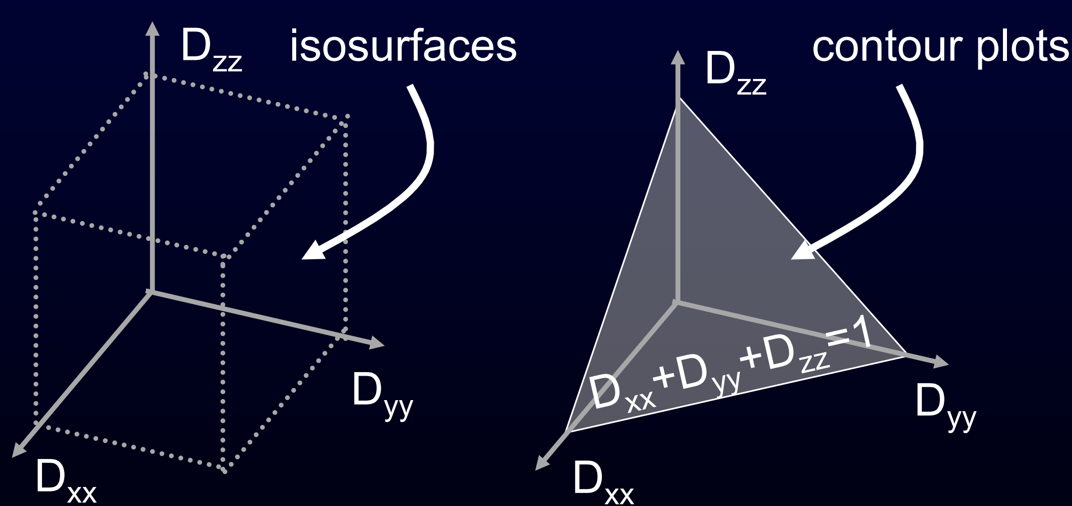
$J_1 = \mathrm{Tr}(\mathbf{D})$

$J_2 = (\mathrm{Tr}(\mathbf{D})^2 - \mathrm{Tr}(\mathbf{D}^2))/2$

$J_3 = \mathrm{Det}(\mathbf{D})$

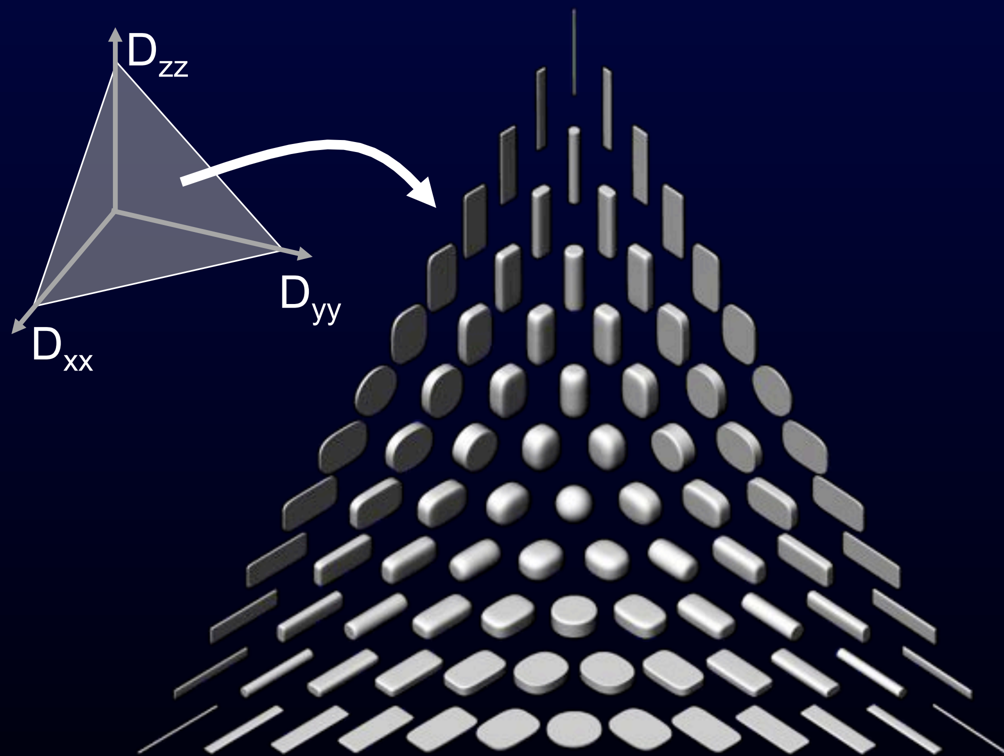
But what do invariants **look** like?

Visualize them in space of diagonal matrices



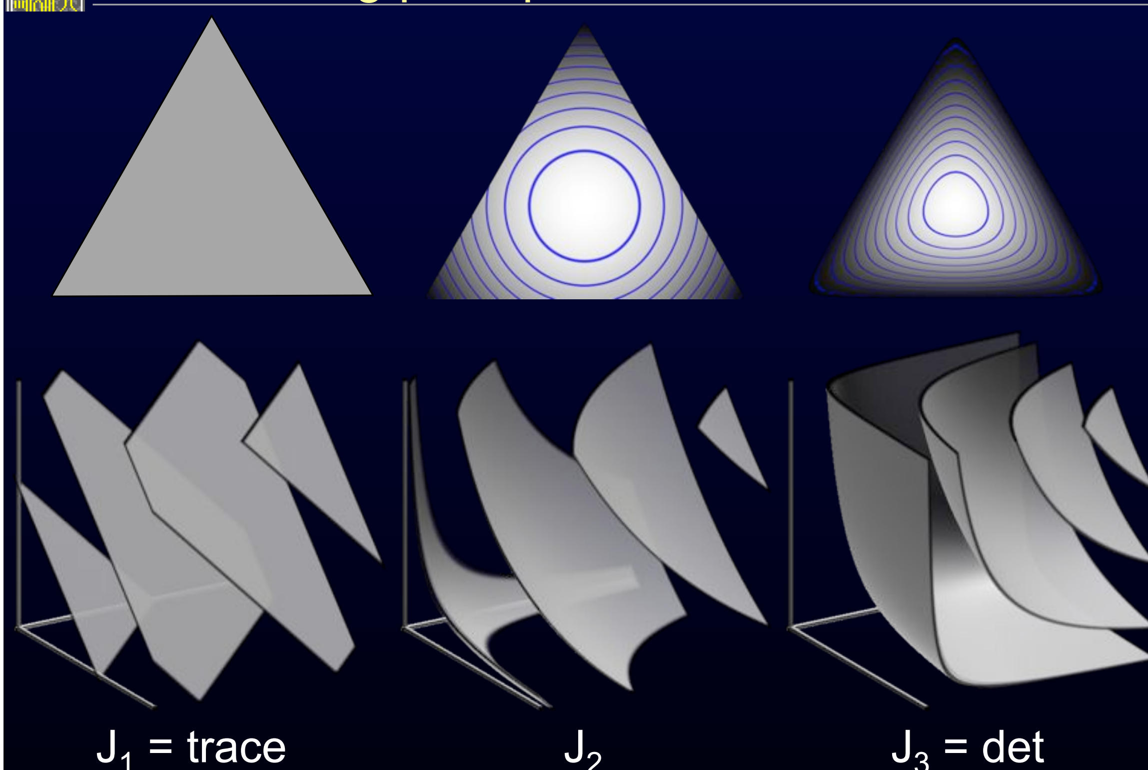
(What do glyphs look like?)

7

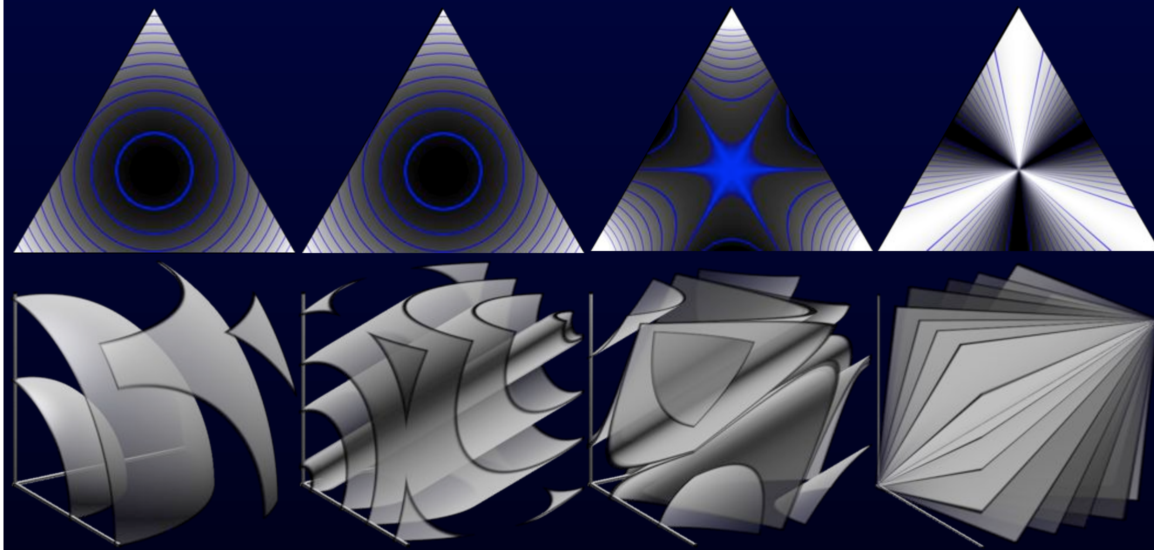


Visualizing principal invariants

8



More invariants



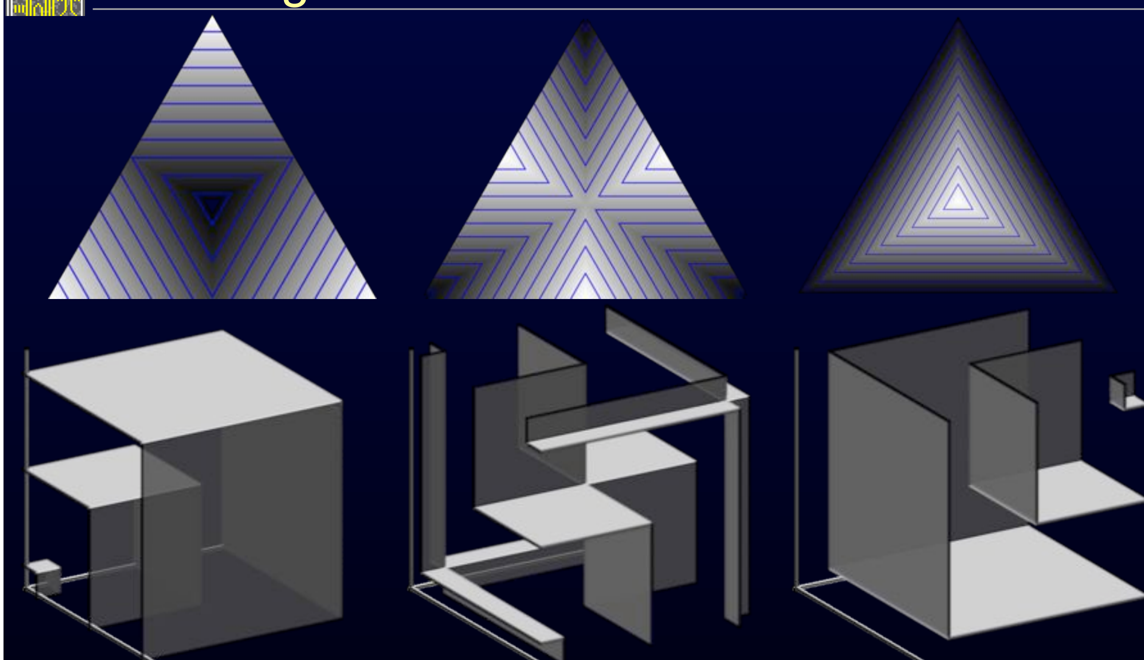
$S = |\mathbf{D}|_F^2 = J_1^2 - 2J_2$

$\mu_2 = \frac{2(S - J_2)}{9} = \mathrm{var}(\lambda)$

$\mu_3 = \frac{2J_1S + 27J_3 - 5J_1J_2}{27}$

$\mathrm{skew} = \frac{\mu_3}{\mu_2^{3/2}}$

The eigenvalues



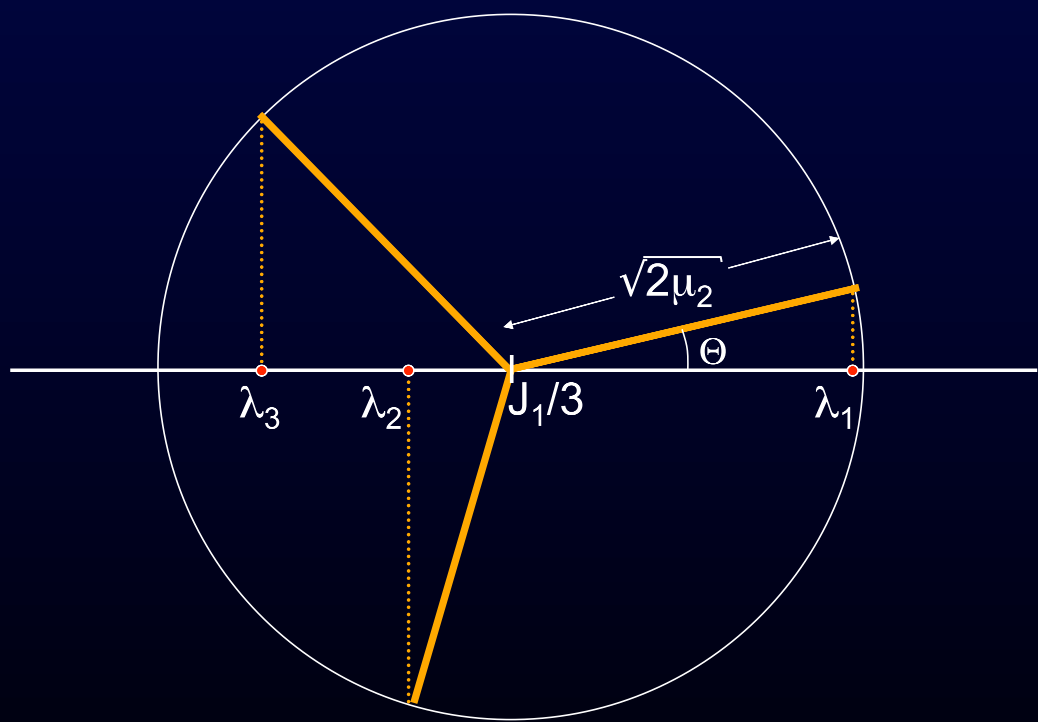
$(\Theta = \cos^{-1}(\sqrt{2}\ \mathrm{skew})/3)$

$\lambda_1 = J_1/3 + \sqrt{2\mu_2}\cos(\Theta)$

$\lambda_2 = J_1/3 + \sqrt{2\mu_2}\cos(\Theta - 2\pi/3)$

$\lambda_3 = J_1/3 + \sqrt{2\mu_2}\cos(\Theta + 2\pi/3)$

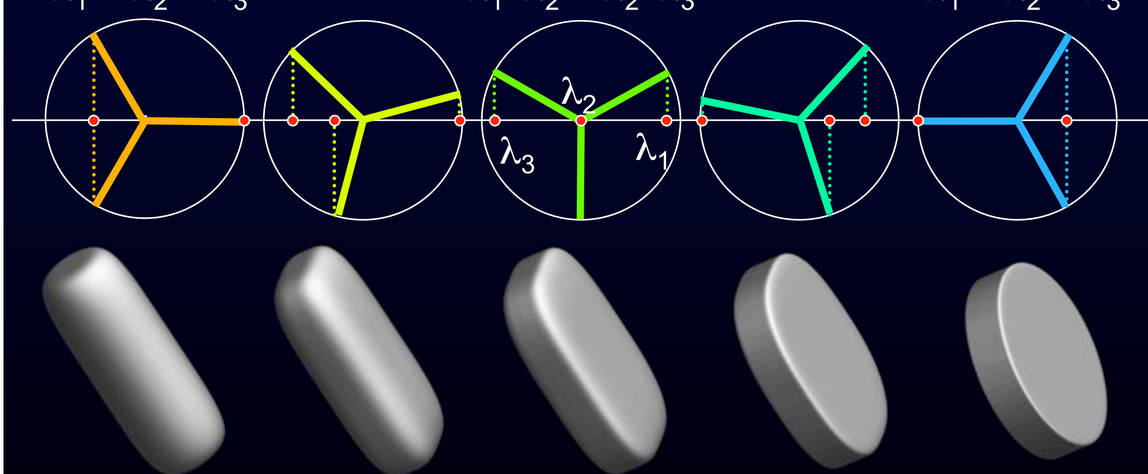
Eigenvalue wheel

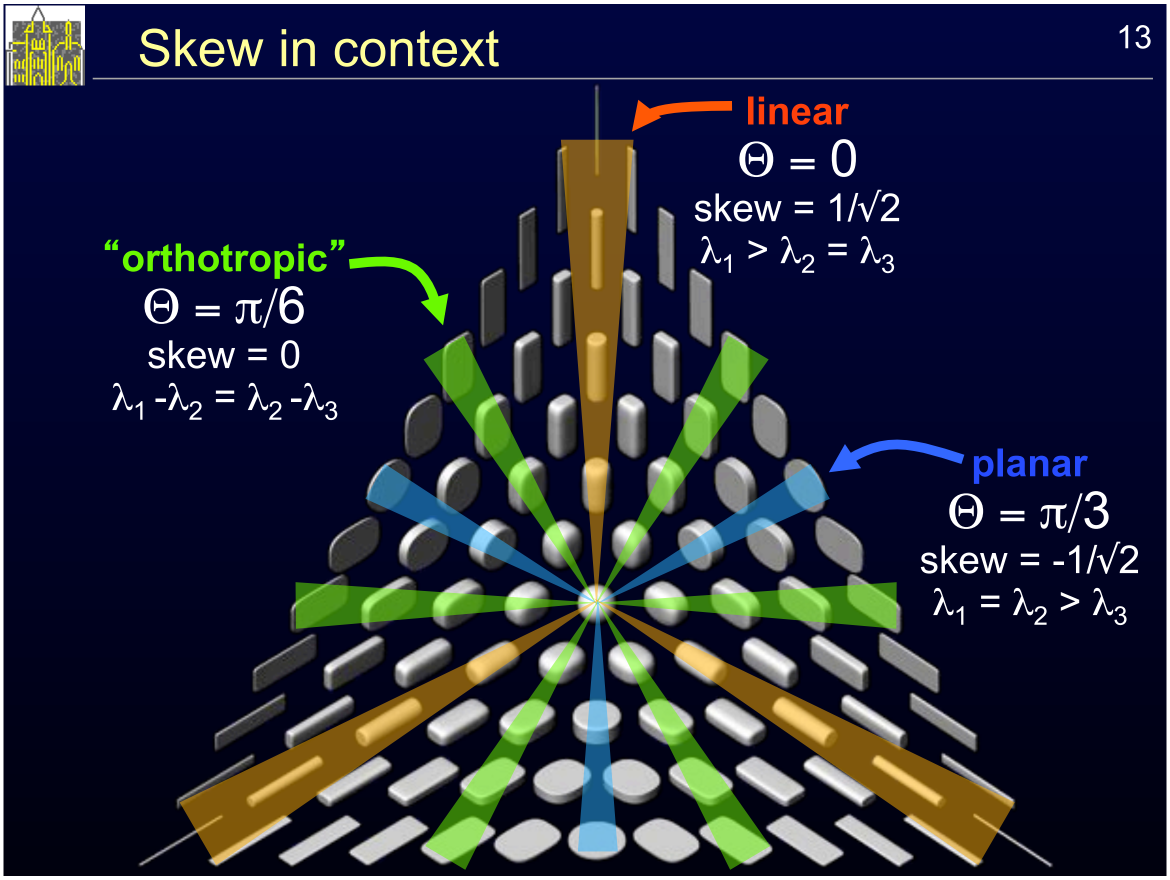


Eigenvalue "sorting", 2nd order isotropy

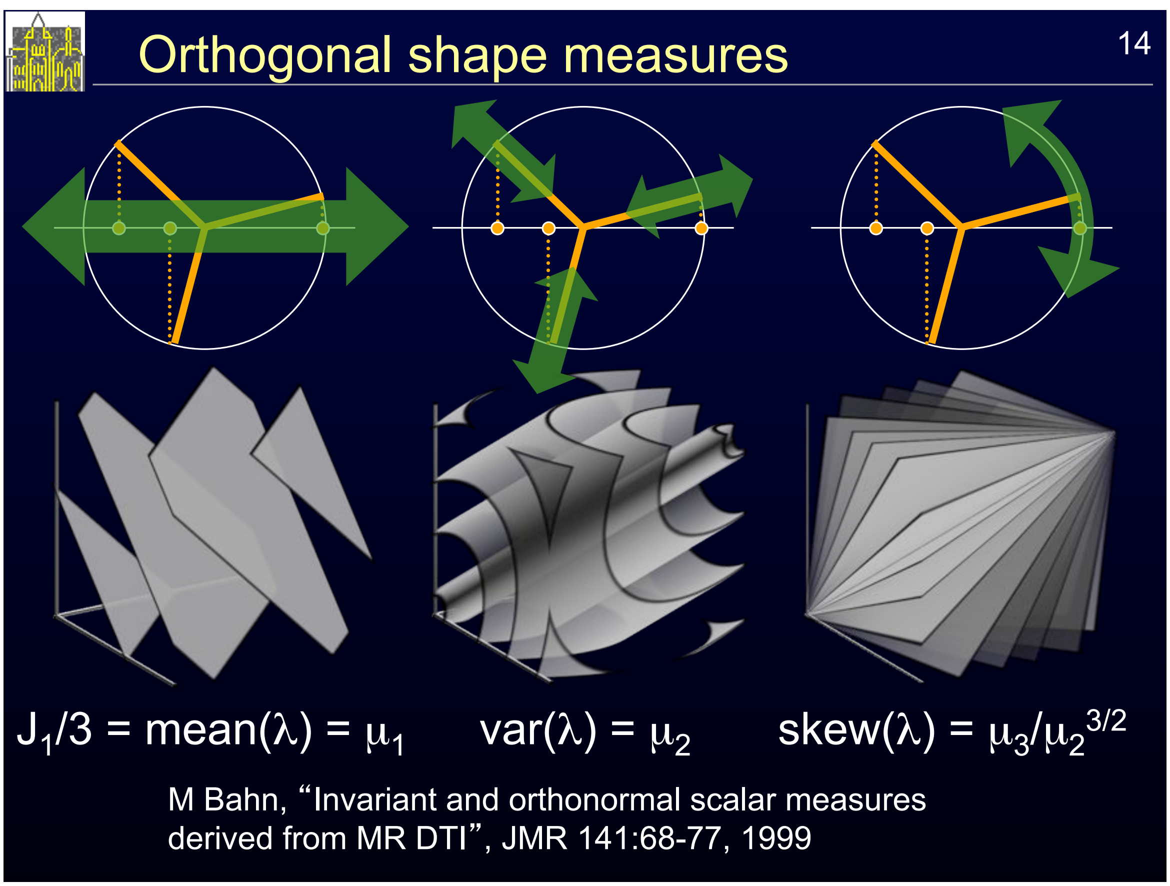
$\Theta = \cos^{-1}(\sqrt{2}\ \text{skew})/3 \ \Rightarrow \Theta \in [0, \pi/3] \ \Rightarrow \lambda_1 \geq \lambda_2 \geq \lambda_3$

linear	"orthotropic"	planar
$\Theta = 0$	$\Theta = \pi/6$	$\Theta = \pi/3$
skew = $1/\sqrt{2}$	skew = 0	skew = $-1/\sqrt{2}$
$\lambda_1 > \lambda_2 = \lambda_3$	$\lambda_1 - \lambda_2 = \lambda_2 - \lambda_3$	$\lambda_1 = \lambda_2 > \lambda_3$



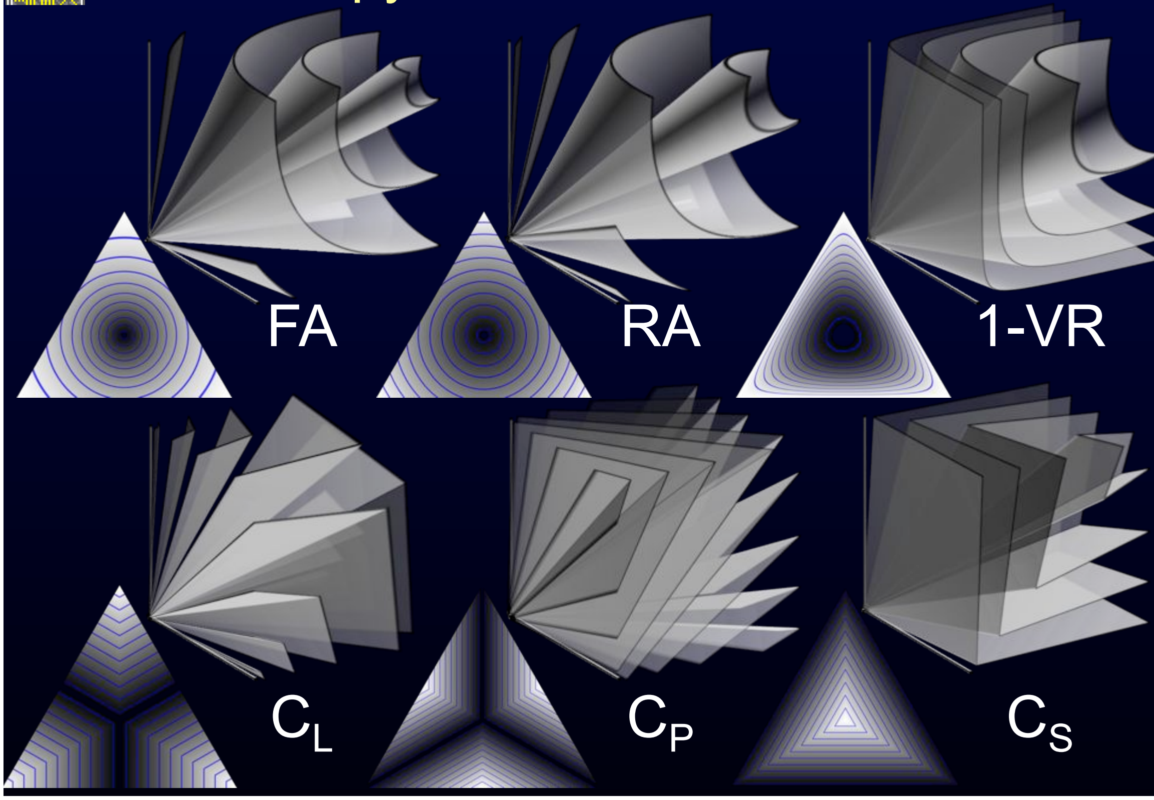


Skew in context
13
linear
Θ = 0
skew = 1/√2
λ1 > λ2 = λ3
"orthotropic"
Θ = π/6
skew = 0
λ1 -λ2 = λ2 -λ3
planar
Θ = π/3
skew = -1/√2
λ1 = λ2 > λ3

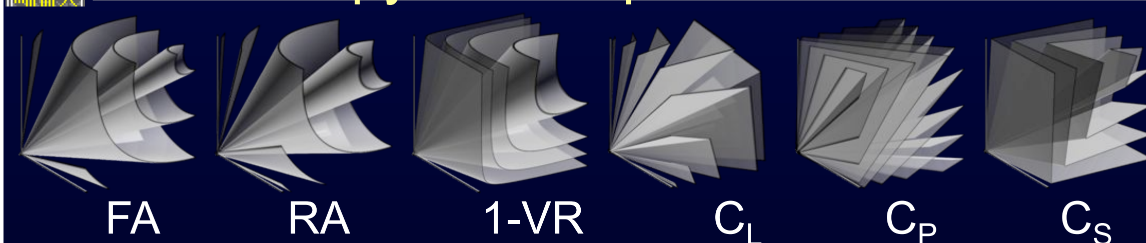


Orthogonal shape measures
14
J1/3 = mean(λ) = μ1
var(λ) = μ2
skew(λ) = μ3/μ2^3/2
M Bahn, "Invariant and orthonormal scalar measures
derived from MR DTI", JMR 141:68-77, 1999

Anisotropy metrics



Anisotropy metric questions



- How exactly does noise sensitivity arise?
- Intuitive way to trace noise through this?

$$\text{DWI} \rightarrow \mathbf{D}_{ij} \rightarrow J_{1,2,3} \begin{cases} \nearrow \text{FA, RA, 1-VR} \\ \searrow \mu_1, \mu_2, \text{skew} \rightarrow \lambda_{1,2,3} \rightarrow C_{L,P,S} \end{cases}$$

- Does differentiability of some metrics (FA) and not others (C_L) matter for noise sensitivity?
- In what contexts is differentiability an interesting or important property of an invariant?

Invariant gradients

17

If these are differentiable:

D(**p**) : tensor data as function of position

Q(**D**) : invariant as function of tensor

1) $\nabla_{P} Q(\mathbf{D}(\mathbf{p}))$: Derivative WRT position: For visualization (volume rendering) : use chain rule

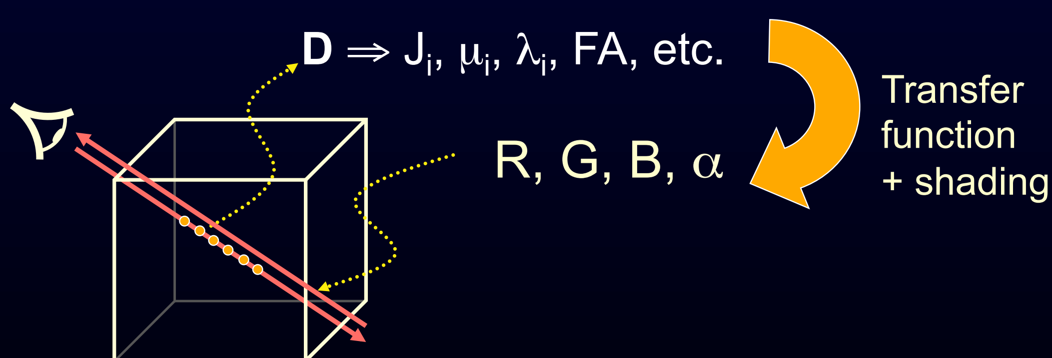
2) $\nabla_{D} Q(\mathbf{D})$: Derivative WRT tensor components: For filtering/processing

Direct Volume Rendering

18

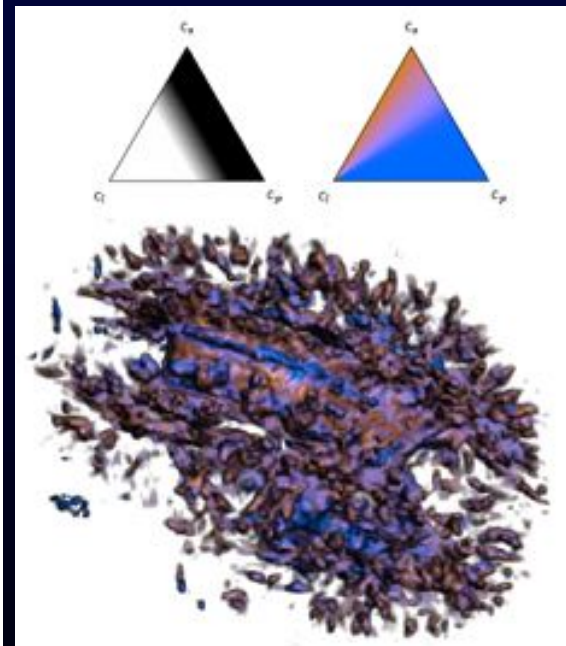
Simple algorithm

- Cast rays through volume
- Measure tensor, tensor properties
- Assign colors and opacities
- Modulate colors with shading
- Composite with “over” operator



Volume Rendering Improvements

Earlier work (IEEE Vis '99):

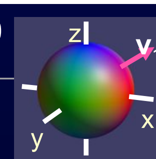


- **Trilinear** interpolation of D
- Shading by interpolation of pre-computed gradients of pre-computed opacity
- experience/enthusiasm < ε

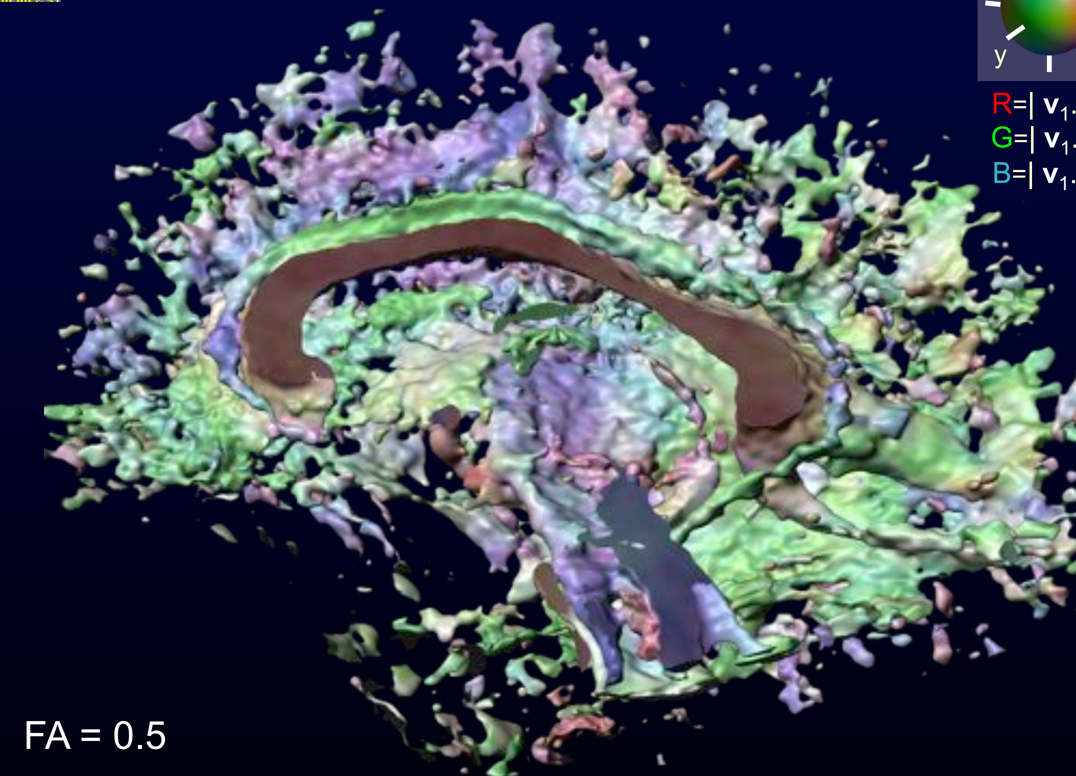
New work:

- Arbitrary kernels
- Transfer functions of differentiable invariants
- Shading by analytic spatial gradients of invariants

Volume Rendering Results

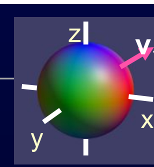


R=| $\mathbf{v}_1$.x |
G=| $\mathbf{v}_1$.y |
B=| $\mathbf{v}_1$.z |

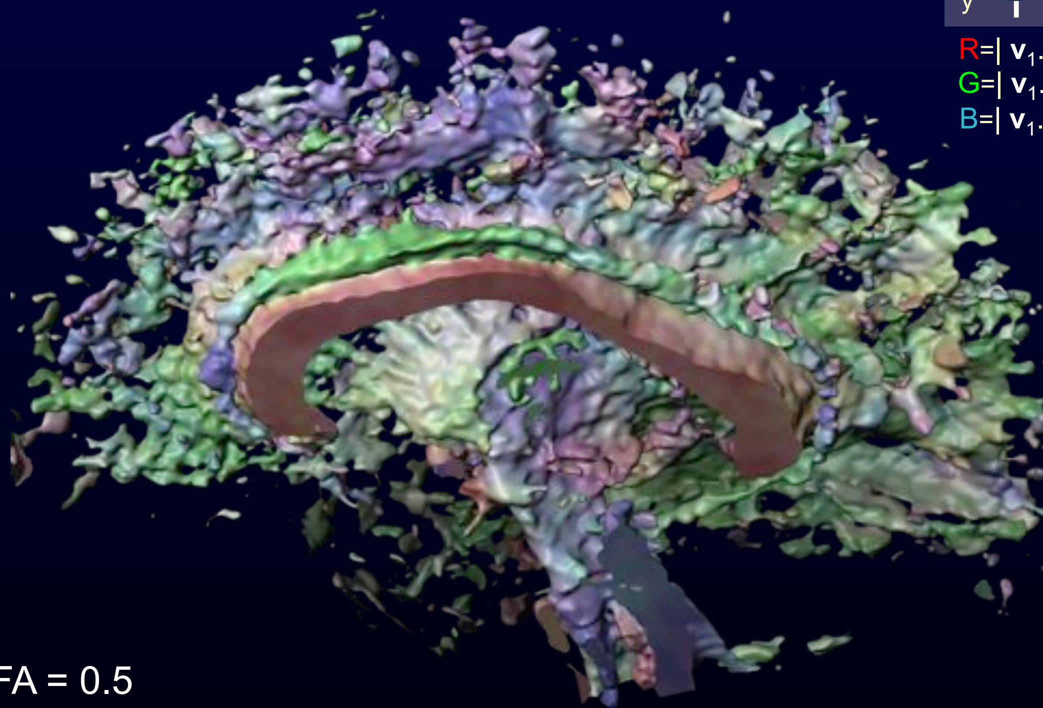


FA = 0.5

Volume Rendering Results

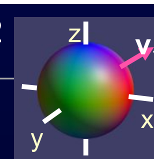


R=| v_1.x |
G=| v_1.y |
B=| v_1.z |

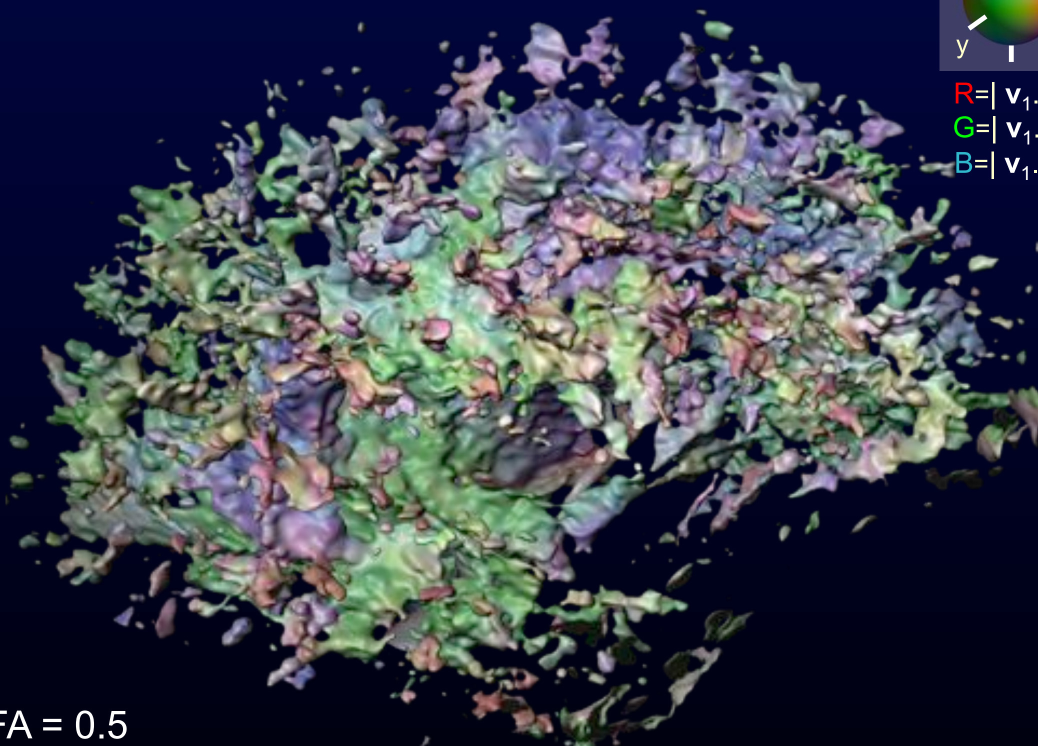


FA = 0.5

Volume Rendering Results

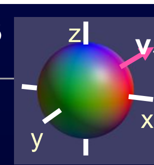


R=| v_1.x |
G=| v_1.y |
B=| v_1.z |

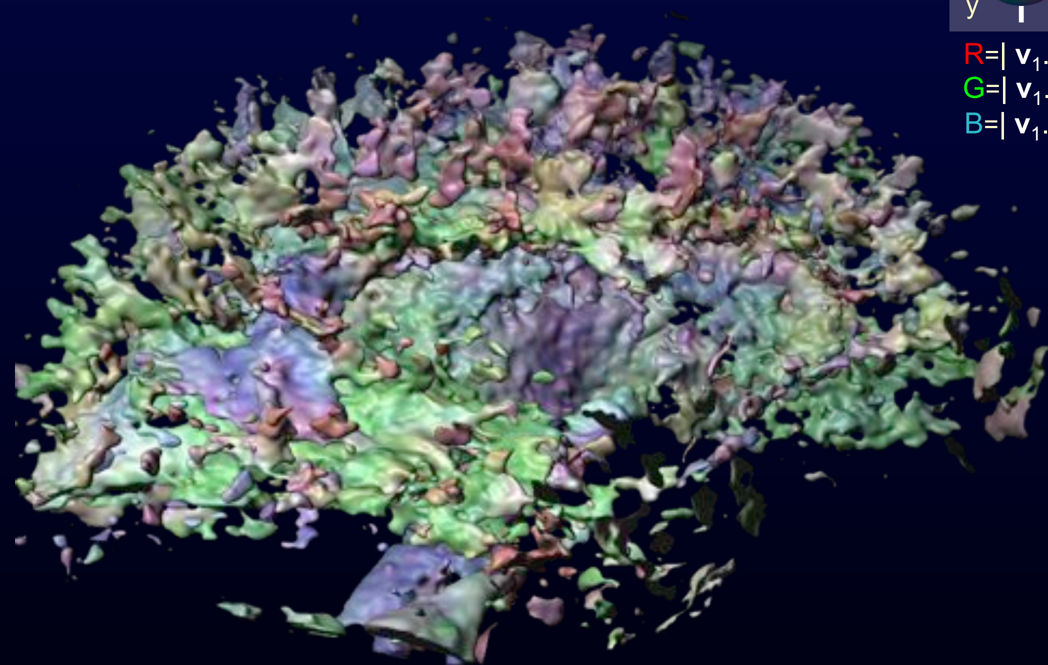


FA = 0.5

Volume Rendering Results

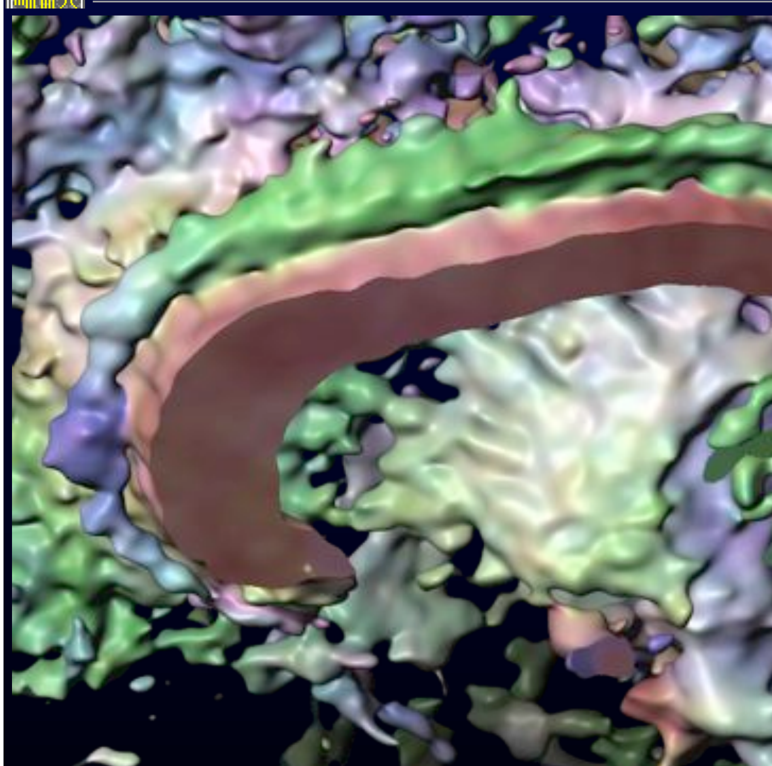


R=| v_1.x |
G=| v_1.y |
B=| v_1.z |

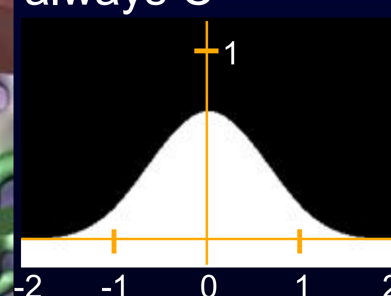


FA = 0.5

Visualizing kernel differences



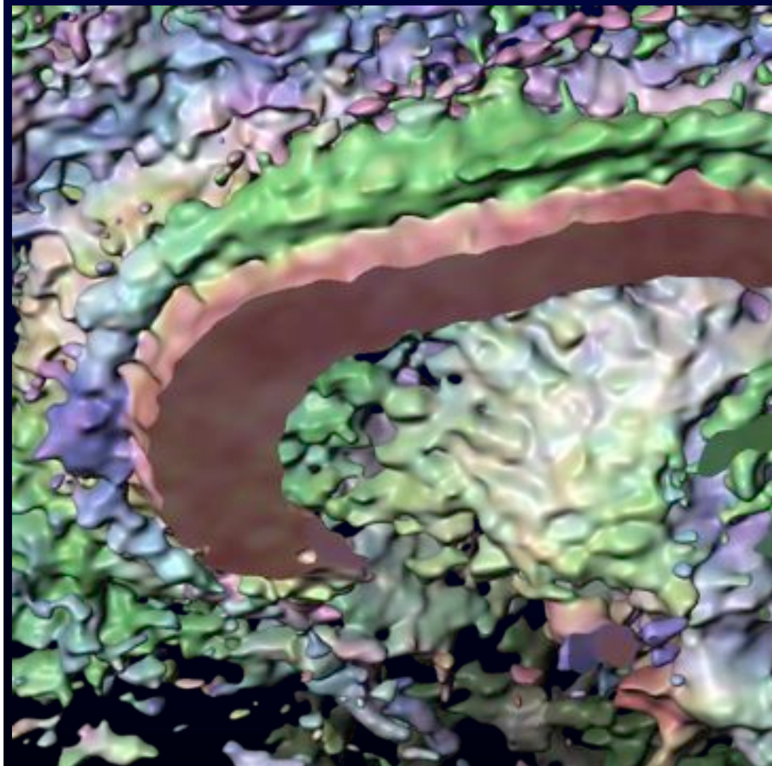
Mitchell-Netravali BC-splines: Simple, tunable, always C^1

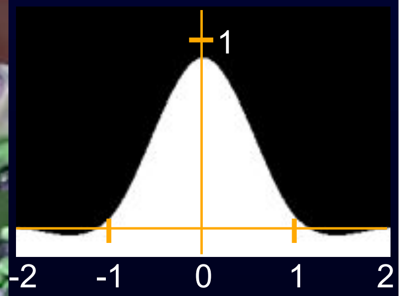


(B,C) = (1,0) Uniform cubic B-spline, also C^2

Visualizing kernel differences

25

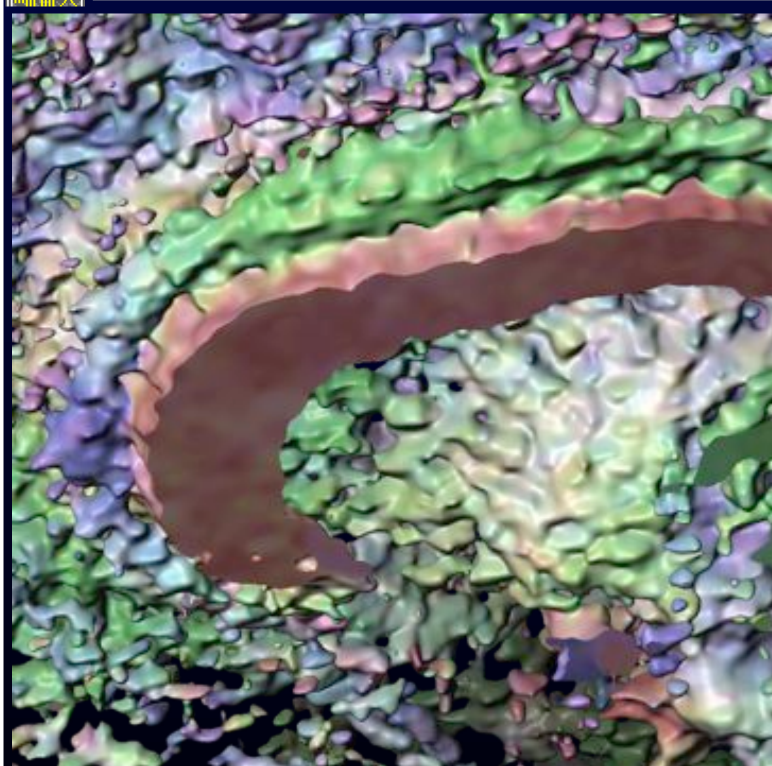


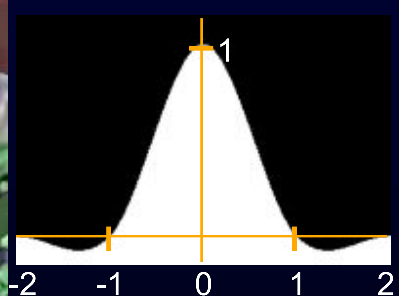


(B,C) = (1/3,1/3)
Blurs a little

Visualizing kernel differences

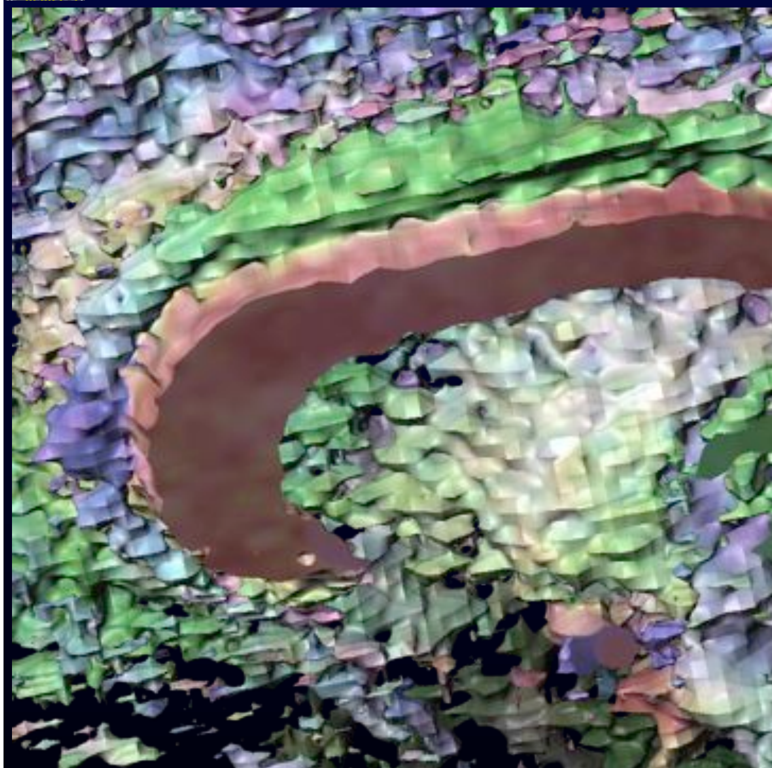
26

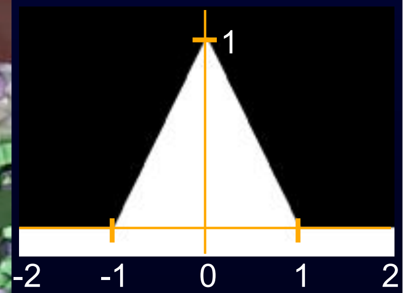




(B,C) = (0,1/2)
Catmull-Rom
Interpolates

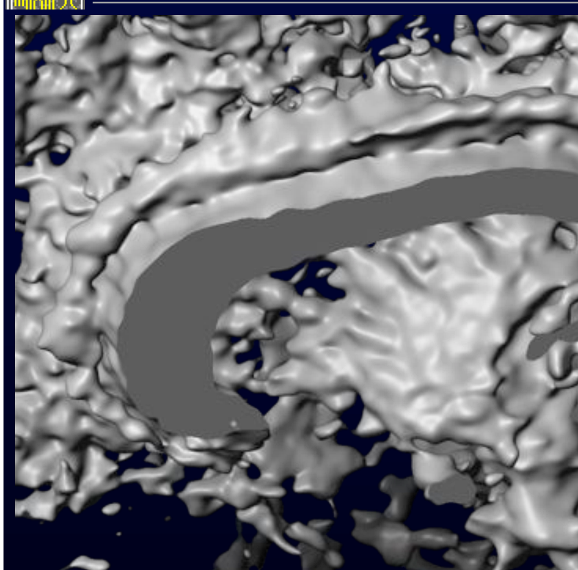
Visualizing kernel differences



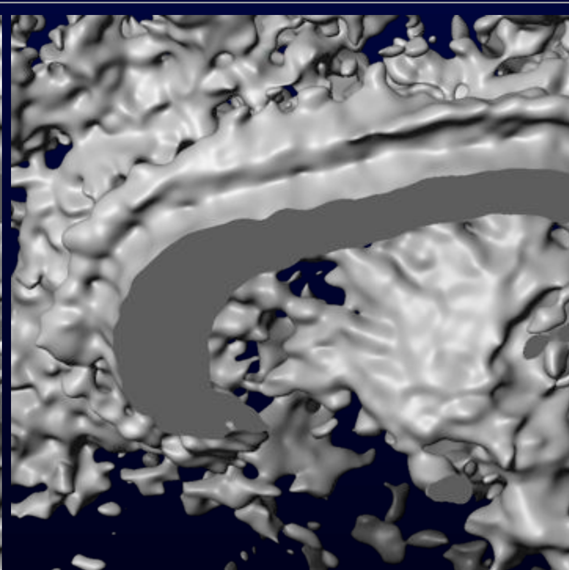


Linear : Not C^1
$\Rightarrow$ nasty edges,
but can see
each sample

Reconstruction+invariants don't commute

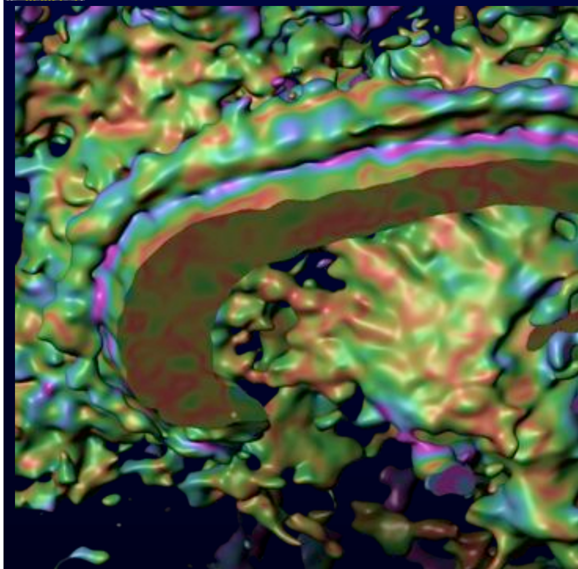


Reconstruct tensors, then
Calculate FA

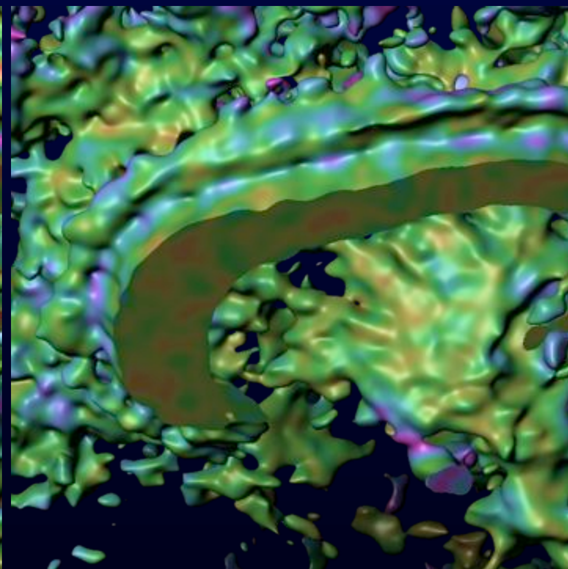


Calculate FA, then
Reconstruct FAs

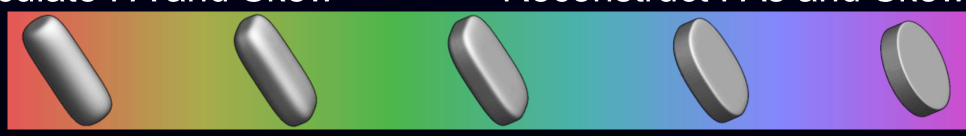
Reconstruction+invariants don’t commute



Reconstruct tensors, then Calculate FA and Skew



Calculate FA and Skew, then Reconstruct FAs and Skews



Invariant gradients

If these are differentiable:

D(**p**) : tensor data as function of position

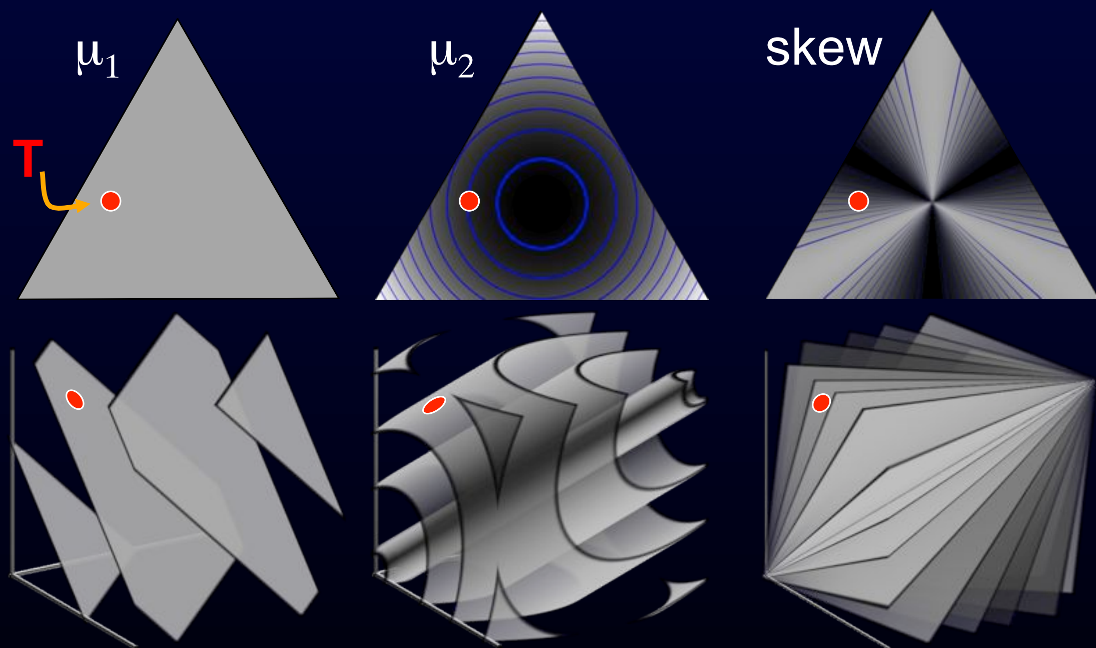
Q(**D**) : invariant as function of tensor

1) $\nabla_P Q(\mathbf{D}(\mathbf{p}))$: Derivative WRT position: For visualization (volume rendering) : use chain rule

2) $\nabla_D Q(\mathbf{D})$: Derivative WRT tensor components: For filtering/processing

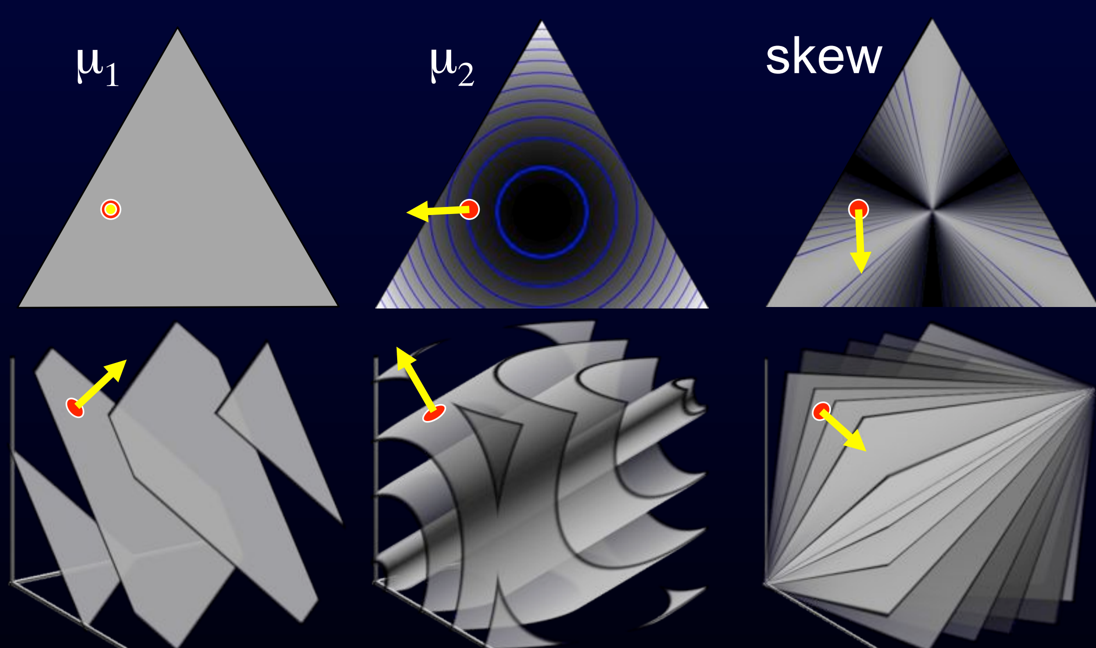
Orthonormal invariant gradients

If you're at tensor **T**, how do you change …



Orthonormal invariant gradients

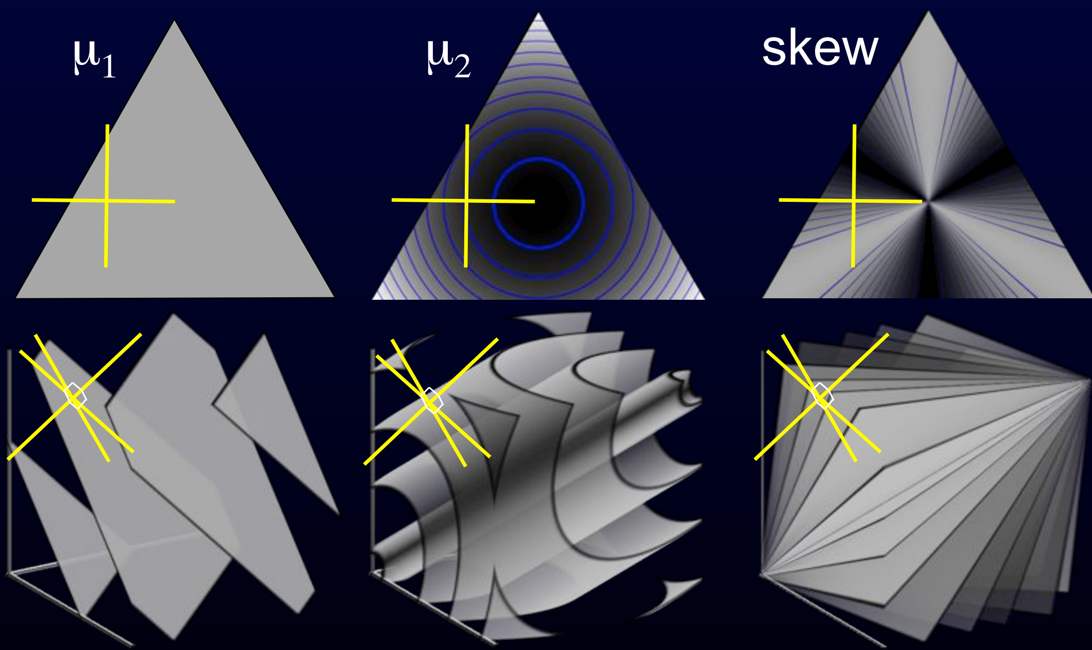
Follow the **tensor-valued** gradients



33

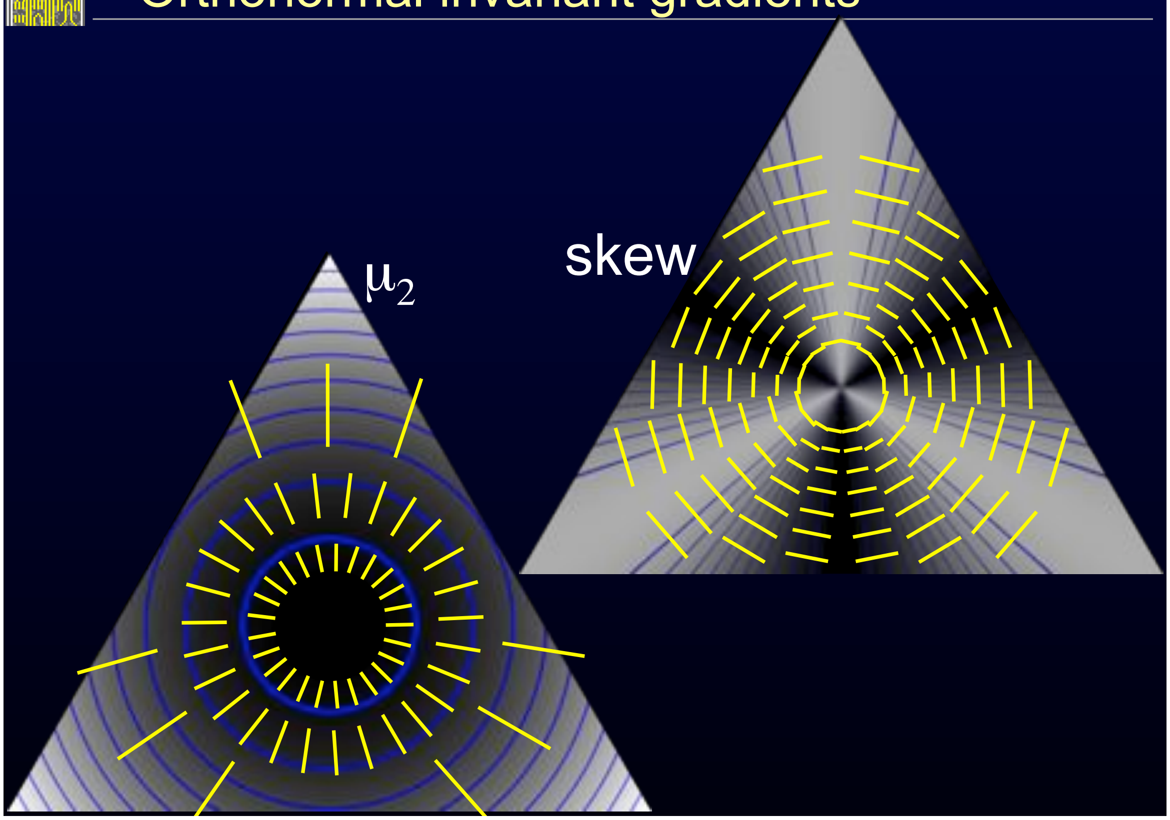
Orthonormal invariant gradients

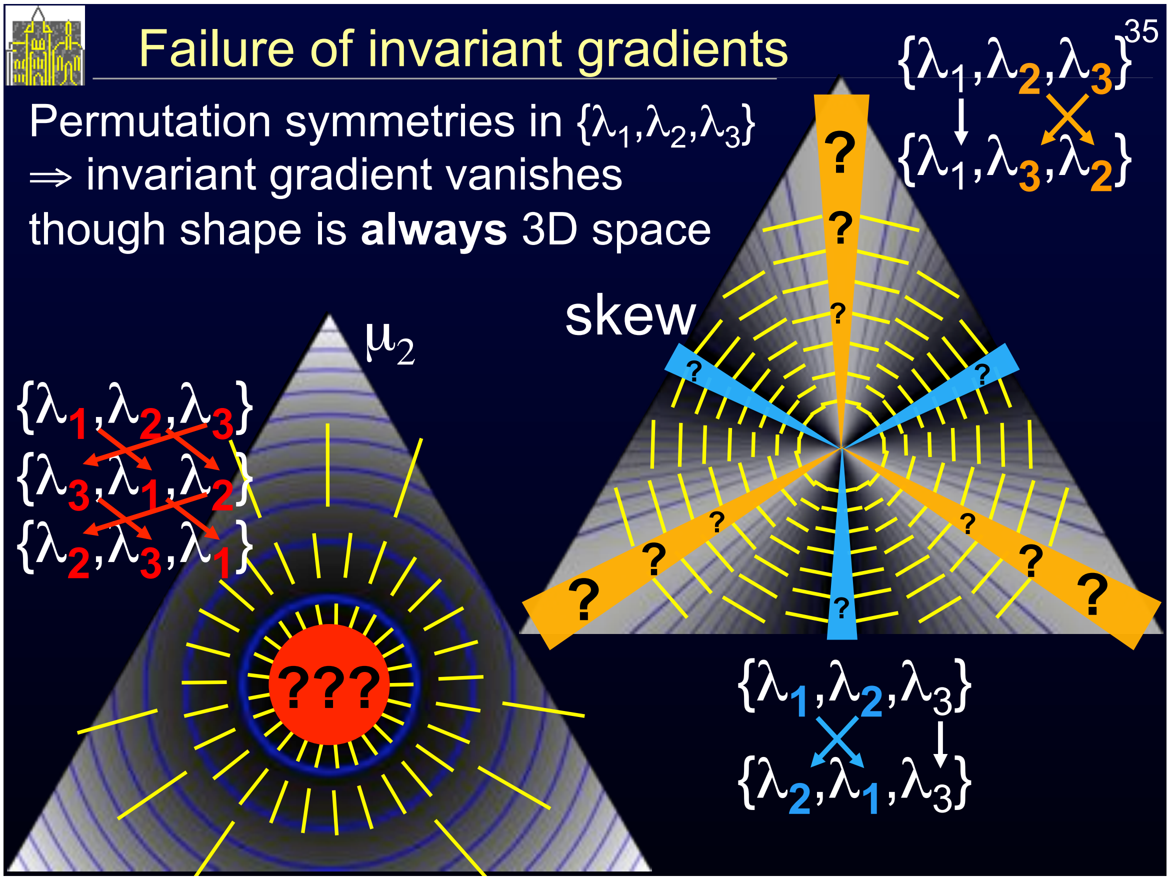
Local coordinate system (after M Bahn)

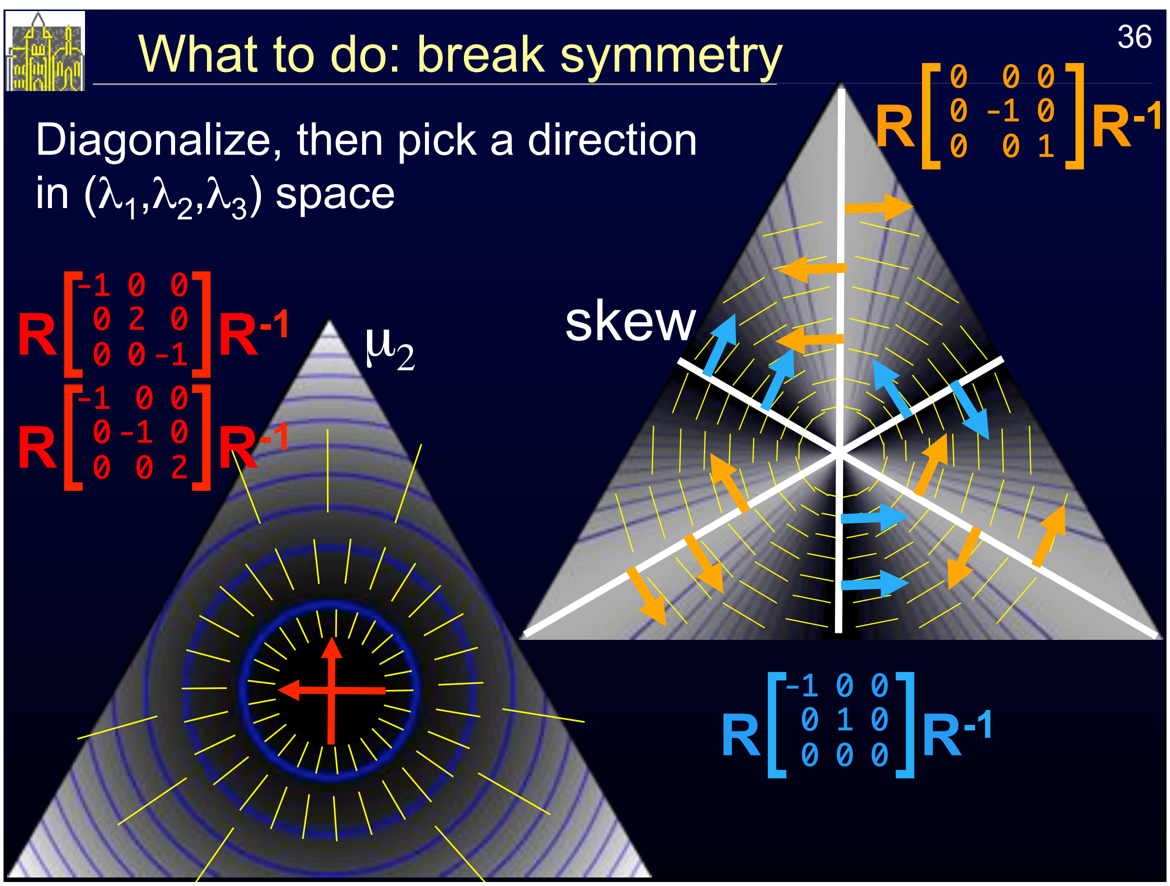


34

Orthonormal invariant gradients




Failure of invariant gradients
35
Permutation symmetries in {λ1,λ2,λ3}
⇒ invariant gradient vanishes
though shape is **always** 3D space
{λ1,λ2,λ3}
{λ1,λ3,λ2}
skew
μ2
{λ1,λ2,λ3}
{λ3,λ1,λ2}
{λ2,λ3,λ1}
???
{λ1,λ2,λ3}
{λ2,λ1,λ3}


What to do: break symmetry
36
Diagonalize, then pick a direction
in (λ1,λ2,λ3) space
R[0 0 0; 0 -1 0; 0 0 1]R-1
skew
μ2
R[-1 0 0; 0 2 0; 0 0 -1]R-1
R[-1 0 0; 0 -1 0; 0 0 2]R-1
R[-1 0 0; 0 1 0; 0 0 0]R-1

Issues

- Can't reliably pick same sign of direction (so don't depend on it)
- 2^{nd}-order isotropy not too bad
- 3^{rd}-order isotropy ugly: no skew direction (can't define how to change hue of a gray color)
- have to smoothly de-emphasize skew direction as a function of low μ_2

Conclusions

- Cubic polynomial solution is not a black box
- Invariants, convolution: good
- Good to have a local coordinate system for changes in shape

All my software's online:

http://teem.sourceforge.net

Thank you!