ANTs image registration

Brian Avants

Biogen

2016-08-11

Background and basics

Simple hints for . . .

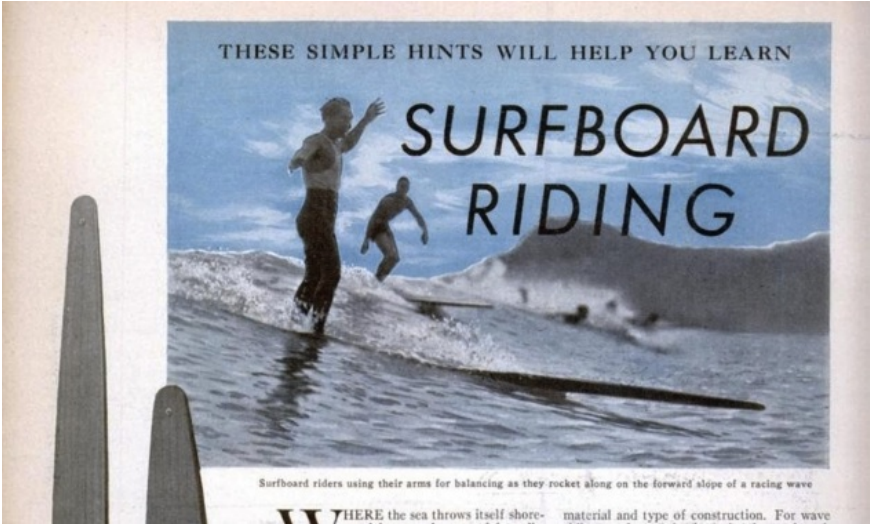


Figure 1:

What is image registration?

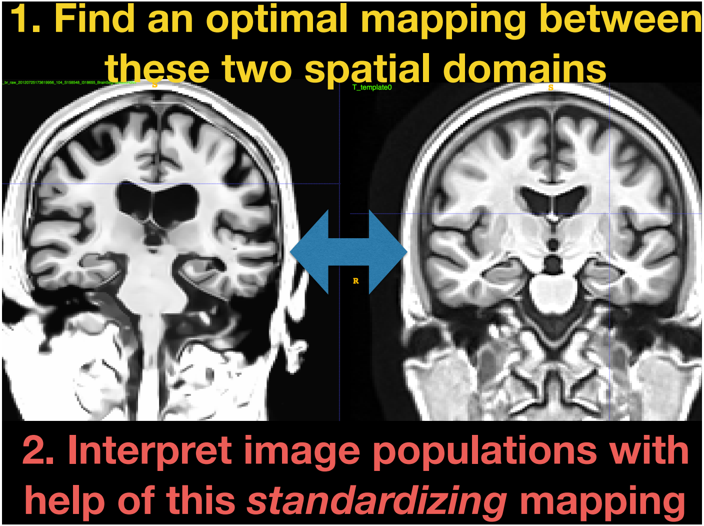


Figure 2:

Key Assumption:

Image content is “very similar”

up to geometric transformation

What is image registration?

Figure 2:

What is image registration?

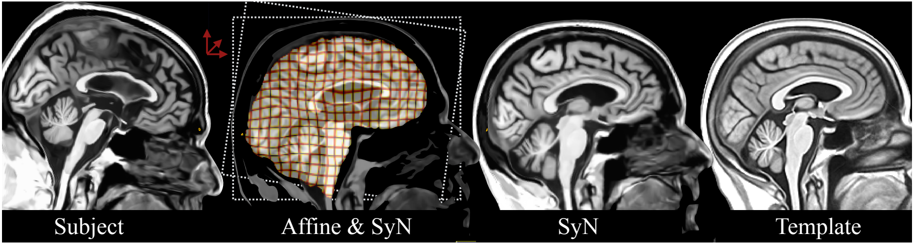


Figure 4:

What is image registration?

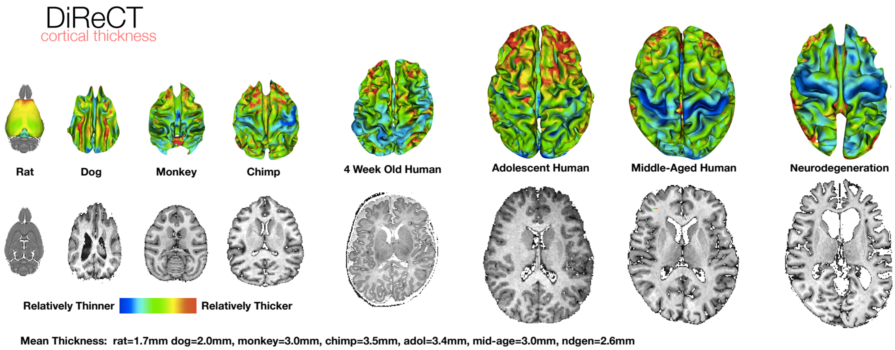
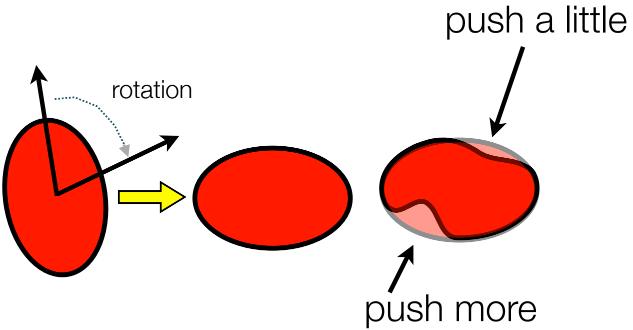


Figure 5:

Basics of the image registration algorithm



$$\int_\Omega \left(\|I(\phi(\mathbf{x})) - J\|^2 + \Pi(\phi(\mathbf{x})) \right) d\Omega$$

Data Term Prior Term

Figure 6:

Apply continuous mapping in digital space

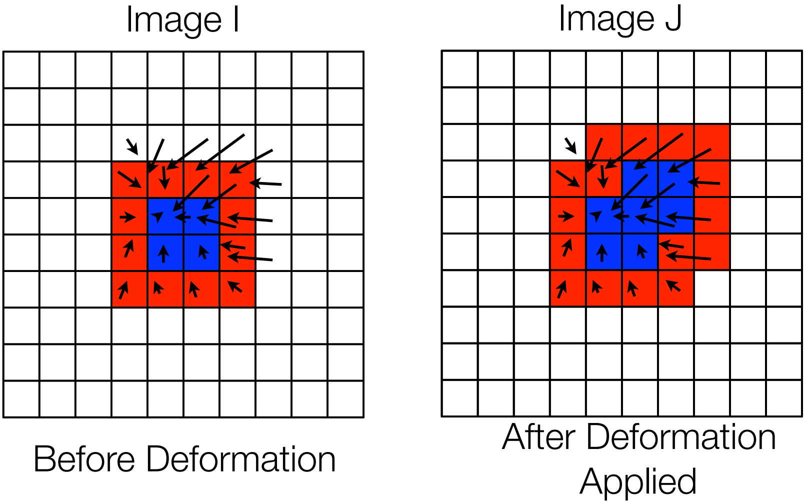


Figure 7:

History: Theory

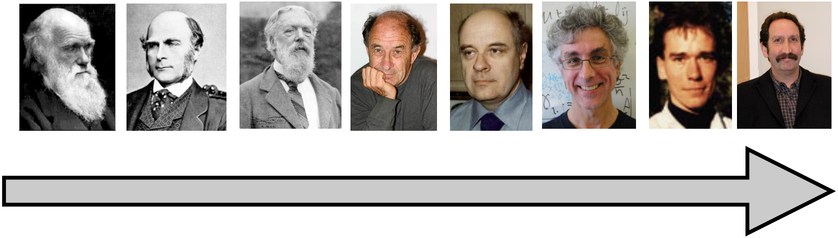
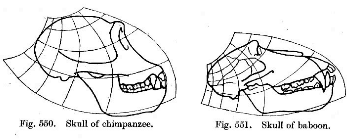


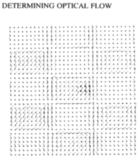
Figure 8:

History: Implementation

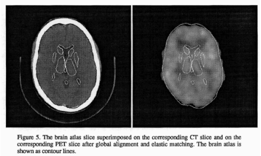
ANTs absorbs, borg-like, much of this history . . .



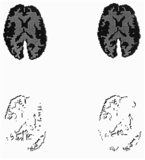
Thompson 1917



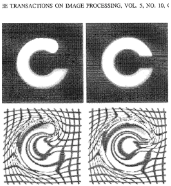
Horn 1980



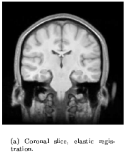
Bajcsy 1988



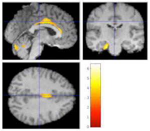
Gee 1994



Miller 1996



Guimond 1999



Ashburner 2000

Figure 9:

ANTs Design

Overview: Programs + brief description

see the ants handout

ANTs is good for scripting large-scale studies

This is a very common trope in our work . . .

```
for x in *nii.gz ; do
  exe = " antsCorticalThickness.sh -d .... "
  qsub ... $exe ...
done
```

Currently running over 20,000 images within such a framework . . .

App-like framework

- ANTs likes short bash scripts

App-like framework

- ANTs likes short bash scripts
- Longer scripts should use R, ANTsR or perl

App-like framework

- ANTs likes short bash scripts
- Longer scripts should use R, ANTsR or perl
- `#! /usr/bin/Rscript`

App-like framework

- ANTs likes short bash scripts
- Longer scripts should use R, ANTsR or perl
- `#! /usr/bin/Rscript`
- No GUI (for good reason)

Design principles

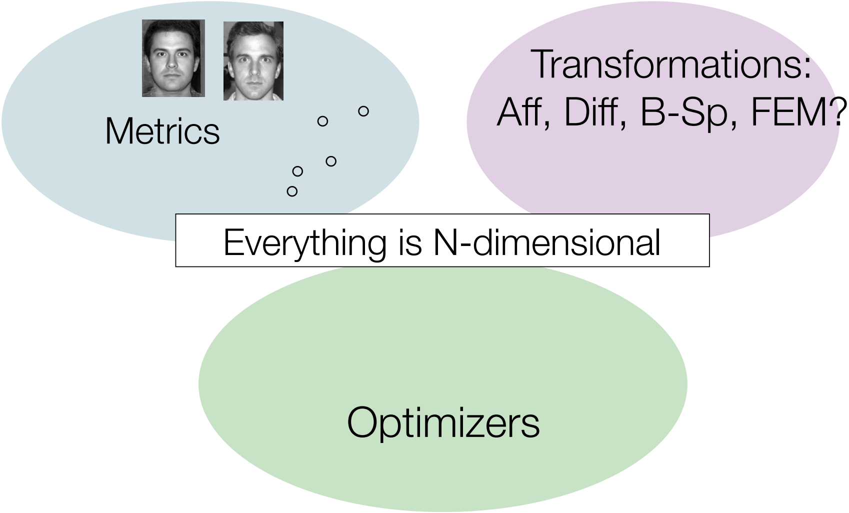


Figure 10:

Design principles: Matched to command line

$$\int_0^1 \|v(x,t)\|_L^2 dt + \|I - J(\phi(x,1))\|^2$$

subject to:

$$\phi(x,0) = A(x)$$

$$\frac{d\phi(x,t)}{dt} = v(\phi(x,t),t), t \in [0,1]$$

Design principles: Matched to command line

$$\int_0^1 \|v(x,t)\|_L^2 dt + M(I, J, \phi(x,1))$$

subject to:

$$\phi(x,0) = A(x)$$

$$\frac{d\phi(x,t)}{dt} = v(\phi(x,t),t), t \in [0,1]$$

and M is a *similarity metric*.

- Look at the command line for `antsRegistration` via the ANTs wiki

Design principles: Matched to command line

$$\int_0^1 \|v(x,t)\|_L^2 dt + M(I, J, \phi(x,1))$$

subject to:

$$\phi(x,0) = A(x)$$

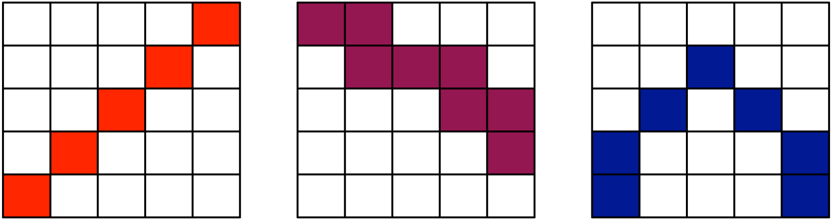
$$\frac{d\phi(x,t)}{dt} = v(\phi(x,t),t), t \in [0,1]$$

and M is a *similarity metric*.

- Look at the command line for `antsRegistration` via the ANTs wiki
- compare with the code for `antsRegistration.R`

Similarity Metrics

Design principles: Similarity metrics



$$\|I - J\|^2 \qquad \frac{\langle I, J \rangle}{\|I\|\|J\|} \qquad p(I,J) log \frac{p(I,J)}{p(I)p(J)}$$

Figure 11:

What types of images are relevant for each?

- Mean squares: very rarely do we use this except perhaps for binary images or maybe CT

What types of images are relevant for each?

- Mean squares: very rarely do we use this except perhaps for binary images or maybe CT
- Global correlation: sometimes useful in conjunction with mutual information but overall rarely used

What types of images are relevant for each?

- Mean squares: very rarely do we use this except perhaps for binary images or maybe CT
- Global correlation: sometimes useful in conjunction with mutual information but overall rarely used
- **Mutual information**: our go-to metric for low-dimensional or fast mapping

What types of images are relevant for each?

- Mean squares: very rarely do we use this except perhaps for binary images or maybe CT
- Global correlation: sometimes useful in conjunction with mutual information but overall rarely used
- **Mutual information**: our go-to metric for low-dimensional or fast mapping
- **Neighborhood correlation**: our go-to metric for high-dimensional, detailed mapping

Mean squares example 1

```
fcfn = paste( bd, "figures/templateFace.jpg", sep='' )
facet = antsImageRead( fcfn ) %>% iMath("Normalize")
fcfn = paste( bd, "figures/face3.jpg", sep='' )
face3 = antsImageRead( fcfn ) %>% iMath("Normalize")
face3t = antsRegistration( facet, face3,
        typeofTransform = "Translation")
faceDiff = facet - face3t$warpedmovout
```

Mean squares example 1

Mean squares example 1

```
## NULL
```

Mean squares example 1

```
## NULL
```

Mean squares example 2

Run a deformable registration ...

```
fcfn = paste( bd, "figures/templateFace.jpg", sep='' )
facet = antsImageRead( fcfn ) %>% iMath("Normalize")
fcfn = paste( bd, "figures/face3.jpg", sep='' )
face3 = antsImageRead( fcfn ) %>% iMath("Normalize")
face3t = antsRegistration( facet, face3, verbose=F,
        typeofTransform = "SyNLessAggro")
faceDiff = facet - face3t$warpedmovout
```

Mean squares example 2

```
## NULL
```

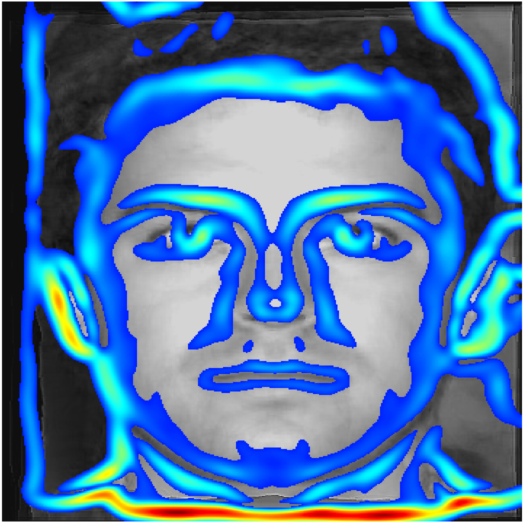
Mean squares example 2

NULL

Mean squares example 2

```
## NULL
```

Mean squares example 2



```
## NULL
```

Mean squares example 3

Run a deformable registration . . .

```
face3 = antsImageRead( fcfn ) %>% iMath("Normalize")
# make a gradient across this image
corrvals = 1:length(face3[face3 >= 0])
fcorrupt = as.antsImage(as.array(face3) * corrvals^1 )
fcorrupt = antsCopyImageInfo( face3, fcorrupt )
face3t = antsRegistration( facet, fcorrupt,
        typeofTransform = "SyNLessAggro")
```

Mean squares example 3

```
## NULL
```

CC example 1

Let's look at another image type . . .

```
ford = paste( bd, "figures/ford.jpg", sep='' )
ford = antsImageRead( ford )
beet = paste( bd, "figures/beetle.jpg", sep='' )
beet = antsImageRead( beet ) %>% iMath("Normalize")
carmap = antsRegistration( ford, beet,
        flowSigma = 1, gradStep = 0.25,
        typeofTransform = "SyNCC", totalSigma=0.5 )
```

CC example 1: Ford

```
## NULL
```

CC example 1: VW Bug

```
## NULL
```

CC example 1: Bug to Ford

```
## NULL
```

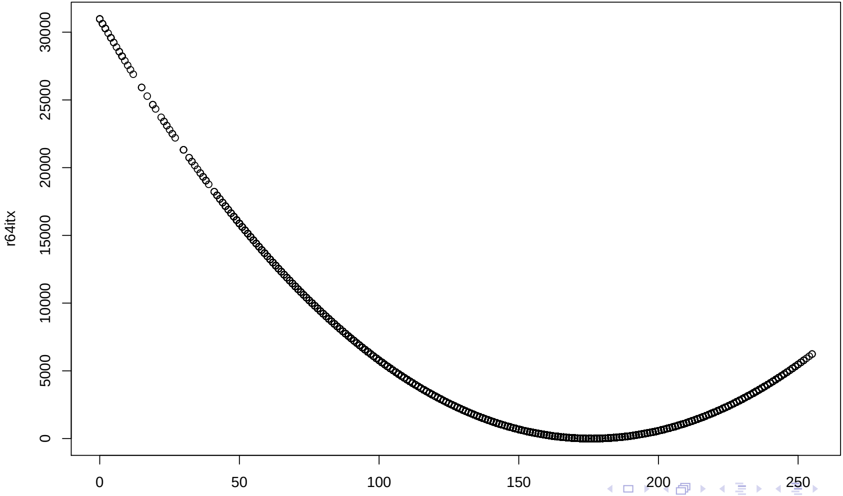
Mutual Information example

Let us transform the intensity of some brain images to mimic a situation where MI does well.

```
r16 = antsImageRead( getANTsRData( "r16" ) )
r64 = antsImageRead( getANTsRData( "r64" ) )
r64m = getMask( r64 )
r64i = r64[ r64m == 1 ]
r64itx = r64i - median( r64i )
r64itx = r64itx^2
r64itx2 = r64i * (-1)   # negation - we should try this
r64[ r64m == 1 ] = r64itx
```

Simulated intensity transformation

```
plot( r64i, r64itx )
```



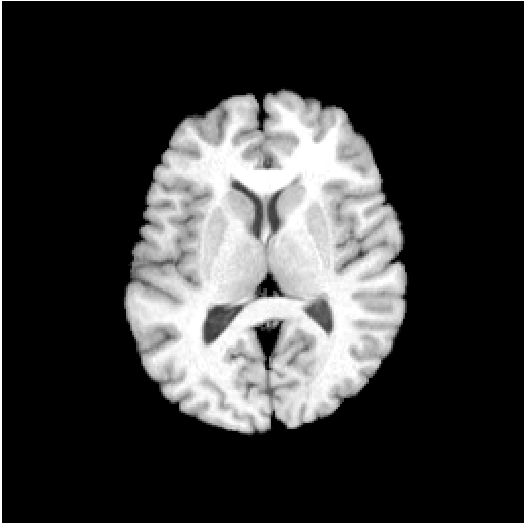
Mutual Information example

```
## NULL
```

Mutual Information example: Two different metrics

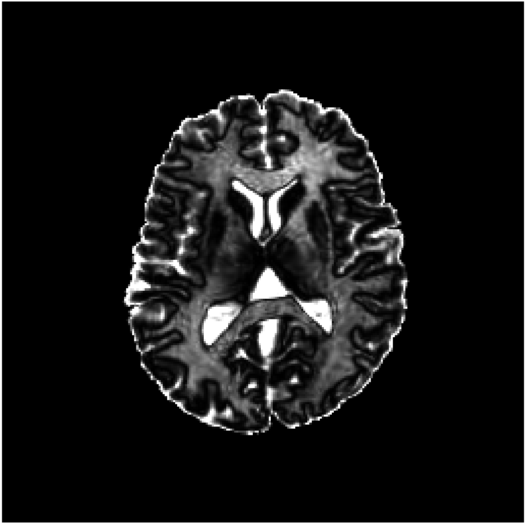
```
mitx = antsRegistration( r16, r64,
  typeofTransform = "SyN",
  flowSigma = 1, gradStep = 0.2, totalSigma=0.0 )
cctx = antsRegistration( r16, r64,
  typeofTransform = "SyNCC",
  flowSigma = 1, gradStep = 0.2, totalSigma=0.0 )
ssdtx = antsRegistration( r16, r64,
  typeofTransform = "SyNAggro",
  flowSigma = 1, gradStep = 0.2, totalSigma=0.0 )
```

Fixed image



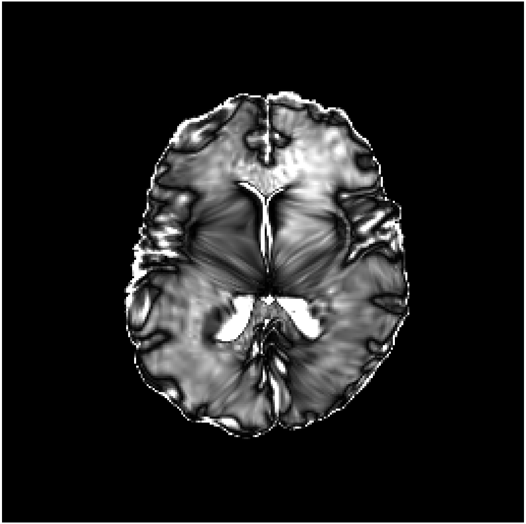
```
## NULL
```

MI result (warped moving)



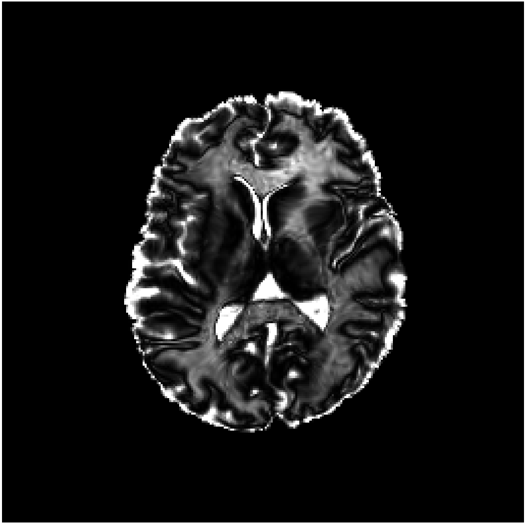
```
## NULL
```

MSQ result (warped moving)



```
## NULL
```

CC result (warped moving)



```
## NULL
```

Collect metric values

SSD, Corr and MI

```
reslist = list( ssdtx$warpedmovout, cctx$warpedmovout, mitx$wa
mydf = data.frame( SSD=rep(NA,3), Corr=rep(NA,3), MI=rep(NA,3)
ct = 1
r16mask = getMask( r16 )
for ( img in reslist ) {
  mydf$SSD[ ct ] = mean( abs( r16-img ) )
  mydf$Corr[ ct ] = cor( r16[r16mask==1], img[r16mask==1] )
  mydf$MI[ ct ] = antsImageMutualInformation( r16,img )
  ct = ct + 1
}
rownames( mydf ) = c("SSDtx","CCtx","MItx")
```

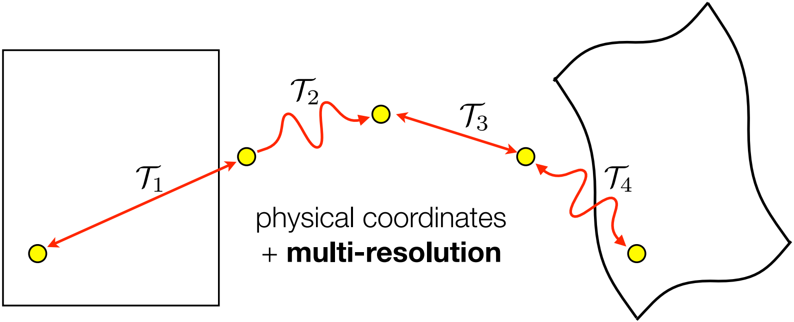
Compare metric values

```
knitr::kable( mydf )
```

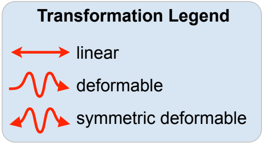
	SSD	Corr	MI
SSDtx	562.5690	0.2541564	-0.3467888
CCtx	512.5470	-0.2400976	-0.2420342
MItx	531.3453	-0.4771264	-0.2744099

Transformation spaces

Design principles: A mapping



$$\mathcal{T} = \mathcal{T}_1 \circ \mathcal{T}_2 \circ \mathcal{T}_3 \circ \mathcal{T}_4$$



Key Advance: unify high and low dimensional transformations.

Figure 12:

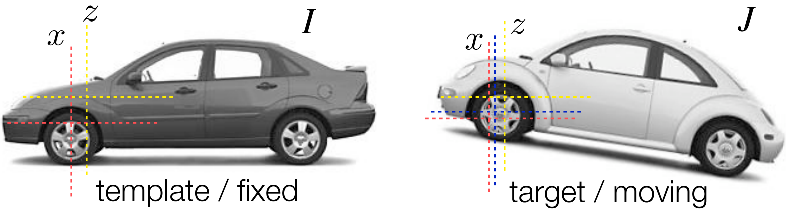
Design principles: A mapping

- Look at the command line for `antsApplyTransforms`

Design principles: A mapping

- Look at the command line for `antsApplyTransforms`
- compare with the help for `antsApplyTransforms.R`

How are coordinate systems managed?



$$I \leftrightsquigarrow \longrightarrow J$$

equivalent to:

$$I = J(A(\phi(x)))$$

Figure 13:

Affine mapping

$$A(x) = y$$

x •

y •

Figure 14:

Affine mapping - applied

$$A(x) = y \Rightarrow J(A(x))$$

x •

y •

Figure 15:

Deformable mapping

$$A(\phi(x)) = z \qquad J(A(\phi(x)))$$

x

y

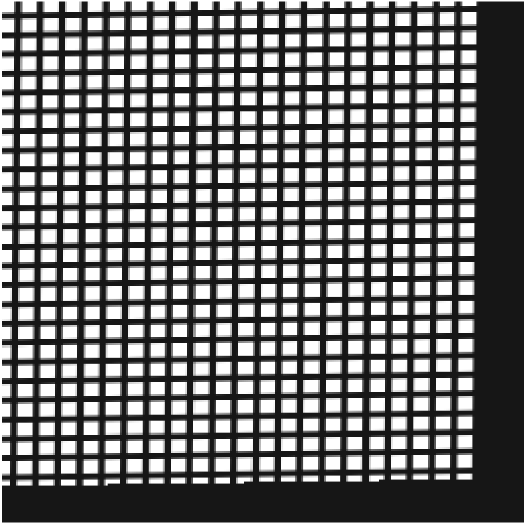
Figure 16:

ANTs uses composite transformations

- Registration: Data is in ANTsR `?getANTsRData`

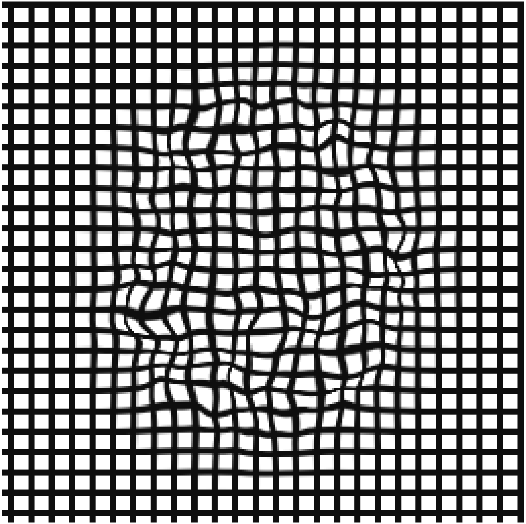
```
fi <- antsImageRead( getANTsRData("r16") )
mi <- antsImageRead( getANTsRData("r64") )
mytx <- antsRegistration(fixed=fi, moving=mi,
                         typeofTransform = c('SyN') )
mywarpedimage <- antsApplyTransforms( fixed=fi, moving=mi,
  transformlist=mytx$fwdtransforms )
mygridw = createWarpedGrid( mi, fixedReferenceImage = fi,
  transform = mytx$fwdtransforms[ 1 ])
mygrida = createWarpedGrid( mi, fixedReferenceImage = fi,
  transform = mytx$fwdtransforms[ 2 ])
mygridwa = createWarpedGrid( mi, fixedReferenceImage = fi,
  transform = mytx$fwdtransforms )
```

The Affine grid transform



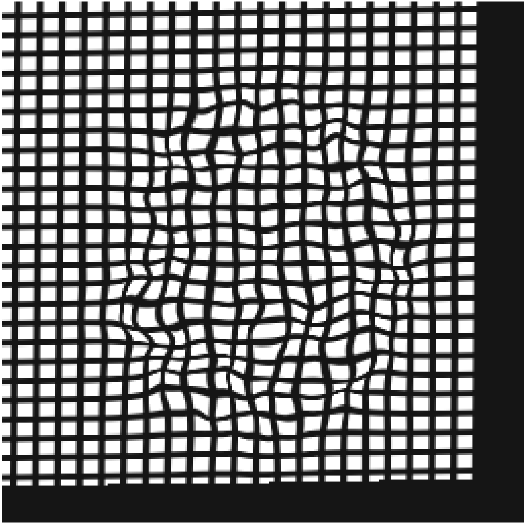
NULL

The Warp grid transform



```
## NULL
```

The composite grid transform



```
## NULL
```

ANTs composite transformations can link several domains

```
fcfn = paste( bd, "figures/templateFace.jpg", sep='' )
facet = antsImageRead( fcfn ) %>% iMath("Normalize")
fcfn = paste( bd, "figures/face3.jpg", sep='' )
face3 = antsImageRead( fcfn ) %>% iMath("Normalize")
fcfn = paste( bd, "figures/face4.jpg", sep='' )
face8 = antsImageRead( fcfn ) %>% iMath("Normalize")
face3t = antsRegistration( facet, face3, flowSigma = 6,
        typeofTransform = "SyN", totalSigma = 0.5 )
face8t = antsRegistration( facet, face8, flowSigma = 6,
        typeofTransform = "SyN", totalSigma = 0.5 )
```

Map subject 3 to subject 8

```
compositeTx = c( face3t$invtransforms, face8t$fwdtransforms )
f8to3 = antsApplyTransforms( face3, face8,
  transformlist = compositeTx,
  whichtoinvert = c(T,F,F,F) )
```

Map subject 3 to subject 8: result

```
## NULL
```

Map subject 3 to subject 8: result

```
## NULL
```

Map subject 3 to subject 8: result

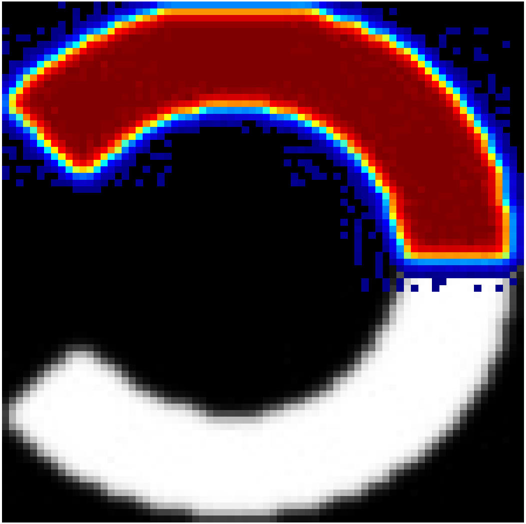
```
## NULL
```

Time varying example

Run a large deformation registration ...

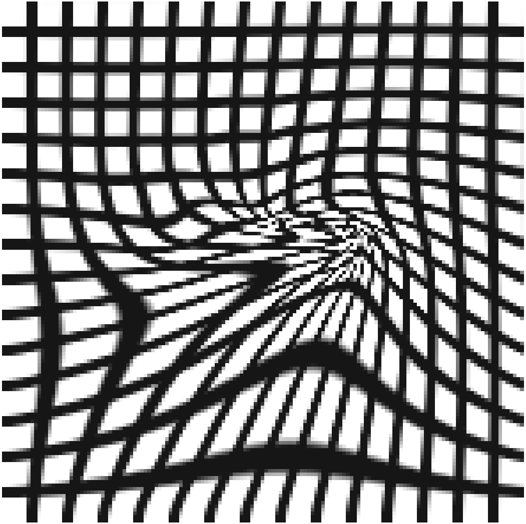
```
cfull = antsImageRead( paste( bd, "figures/c.jpg", sep='' ) )
chalf = antsImageRead( paste( bd, "figures/chalf.jpg", sep=''
cmap = antsRegistration( chalf, cfull, gradStep = 2,
        typeofTransform = "TVMSQC", verbose=F )
cgridf = createWarpedGrid( chalf, fixedReferenceImage = chalf,
      transform = cmap$fwdtransforms )
cgridi = createWarpedGrid( chalf, fixedReferenceImage = chalf,
      transform = cmap$invtransforms )
```

"C"lassic example: full to half "C"



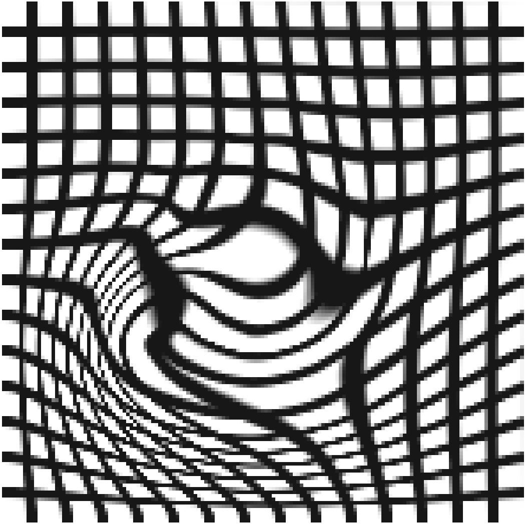
NULL

Time varying example: Forward map



```
## NULL
```

Time varying example: Inverse map



```
## NULL
```

ANTs Optimization

Notation for composite transformation / optimization

$$I \leftrightsquigarrow_{mi} \rightarrow_{ssd} \rightarrow_{cc} J$$

$$I_{MNI} \leftrightsquigarrow_{cc} \rightarrow_{ssd} \rightsquigarrow_{mi} \rightarrow_{mi} J_{BOLD}$$

Optimization problem statement for image registration

$$\text{Find mapping } \phi(x,p) \in \mathcal{T}$$

such that

$$M(I, J, \phi(x,p)) \text{ is minimized.}$$

all metrics minimize .....
transforms "update themselves"

Gradient calculation at a point

$$\frac{\partial M}{\partial p_i} = \frac{\partial M}{\partial J} \frac{\partial J(\phi(x,p))}{\partial \phi} \frac{\partial \phi}{\partial p_i}^T \Big|_x$$

Figure 19:

Conjugate gradient algorithm

$$\gamma = \frac{\|\nabla M_t - \nabla M_{t-1}\|^2}{\|\nabla M_{t-1}\|^2}$$

$$CG_t = \nabla M_t + \gamma CG_{t-1}$$

$$p_{\text{new}} = p_{\text{old}} + \text{learning rate } CG$$

regularization happens here ...

Figure 20:

"Global" optimization - multistart

Initialization is critical to image registration . . .

```
butterfly1 = antsImageRead( paste( bd, "figures/butterfly-3.jp
butterfly2 = antsImageRead( paste( bd, "figures/butterfly-15.j
```

"Global" optimization - multistart code

See the ANTsR wiki

```
nout = 25
thetas = seq( from = 0, to = 360, length.out = nout )[1:(nout-
mytx = "/tmp/best.mat"
mival<-invariantImageSimilarity( butterfly1, butterfly2,
  thetas=thetas, localSearchIterations=10, txfn=mytx )
www = antsApplyTransforms( butterfly1, butterfly2, transformli
```

“Global” optimization - butterfly2

```
## NULL
```

"Global" optimization - butterfly1

```
## NULL
```

"Global" optimization - butterfly2 to 1 result

```
## NULL
```

Case studies and conclusions

Specific case: MI + Affine

$\mathcal{T}$ is affine, M is mutual information

antsRegistration -d 3
-m mattes[$f, $m , 1 , 32, random , 0.05]
-t affine[2]
-c [1800x1000x1500x20,1.e-8,20]
-s 4x2x1x0
-f 8x4x2x1

Figure 21:

Specific case: MI + Elastic

$\mathcal{T}$ is elastic, M is image difference

antsRegistration -d 2
-m meansquares[$f , $m, 1, 1]
-t gaussiandisplacementfield[1.5,10,0.5]
-c [290x290x290x120,1.e-7,3]
-f 4x2x1 -s 2x1x0

Figure 22:

Specific case: MI + Elastic “C” example

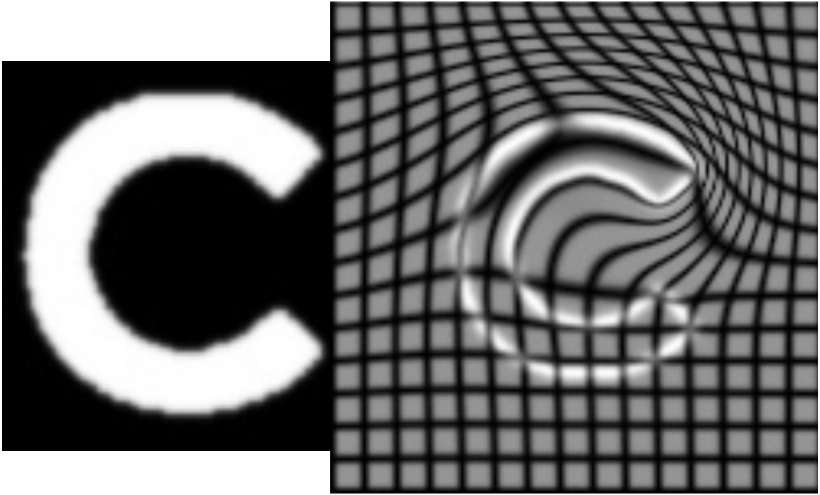


Figure 23:

Specific case: CC $+$ diffeomorphism

$\mathcal{T}$ is Diff, M is N-Hood Correlation

$$I \leftrightsquigarrow_{ncc} \longrightarrow_{mi} J$$

Figure 24:

Specific case: CC + diffeomorphism

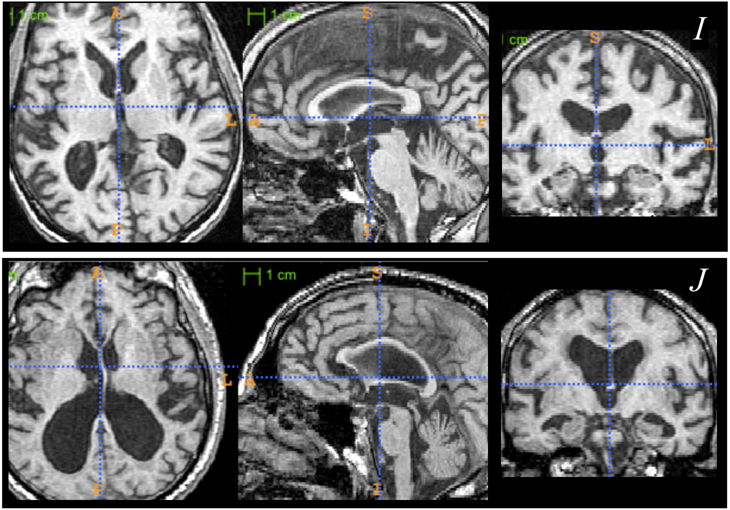


Figure 25:

Specific case: CC $+$ diffeomorphism result

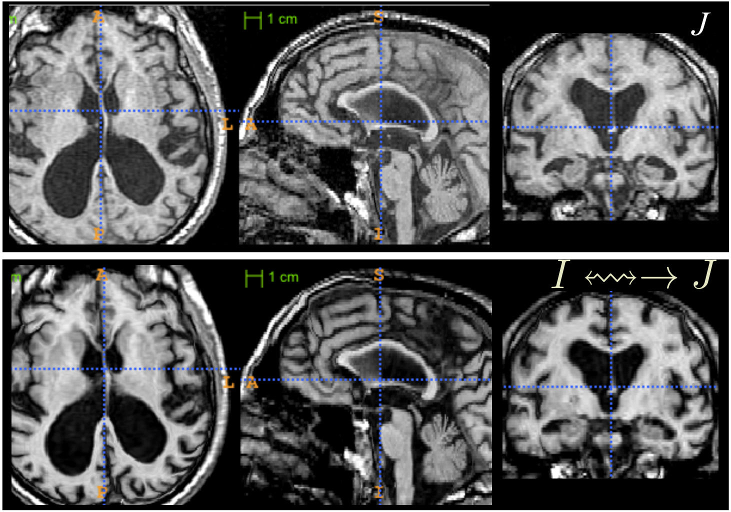


Figure 26:

Specific case: MSQ RGB + affine and elastic

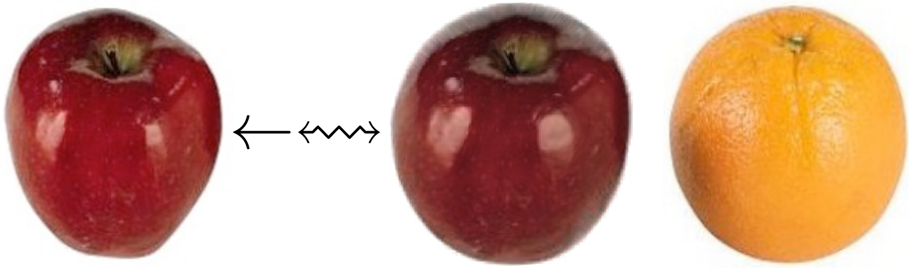


Figure 27:

Mapping two images: parameters

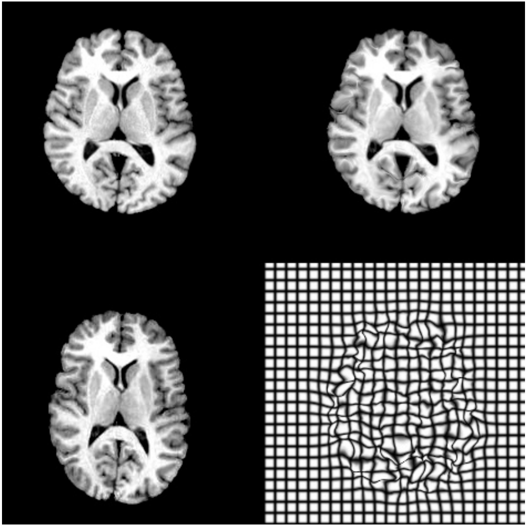


Figure 28:

Meaning of ANTs registration parameters

What happens when i vary each parameter?

- Robustness increases with regularization

Meaning of ANTs registration parameters

What happens when i vary each parameter?

- Robustness increases with regularization
- `-r Gauss[6,3]` $>$ `-r Gauss[3,0]`

Meaning of ANTs registration parameters

What happens when i vary each parameter?

- Robustness increases with regularization
- `-r Gauss[6,3]` > `-r Gauss[3,0]`
- Flexibility decreases with regularization

Meaning of ANTs registration parameters

What happens when i vary each parameter?

- Robustness increases with regularization
- `-r Gauss[6,3]` $>$ `-r Gauss[3,0]`
- Flexibility decreases with regularization
- `-r Gauss[6,3]` $<$ `-r Gauss[3,0]`

Meaning of ANTs registration parameters

What happens when i vary each parameter?

- Robustness increases with regularization
- `-r Gauss[6,3]` $>$ `-r Gauss[3,0]`
- Flexibility decreases with regularization
- `-r Gauss[6,3]` $<$ `-r Gauss[3,0]`
- Robustness increases with correlation window

Meaning of ANTs registration parameters

What happens when i vary each parameter?

- Robustness increases with regularization
- `-r Gauss[6,3]` > `-r Gauss[3,0]`
- Flexibility decreases with regularization
- `-r Gauss[6,3]` < `-r Gauss[3,0]`
- Robustness increases with correlation window
- `-m CC[ . , . , 1 , 4 ]` < `-m CC[ . , . , 1 , 6 ]`

Meaning of ANTs registration parameters

What happens when i vary each parameter?

- Robustness increases with regularization
- `-r Gauss[6,3]` > `-r Gauss[3,0]`
- Flexibility decreases with regularization
- `-r Gauss[6,3]` < `-r Gauss[3,0]`
- Robustness increases with correlation window
- `-m CC[ . , . , 1 , 4 ]` < `-m CC[ . , . , 1 , 6 ]`
- but computation time also increases

Meaning of ANTs registration parameters

What happens when i vary each parameter?

- Robustness increases with regularization
- `-r Gauss[6,3]` $>$ `-r Gauss[3,0]`
- Flexibility decreases with regularization
- `-r Gauss[6,3]` $<$ `-r Gauss[3,0]`
- Robustness increases with correlation window
- `-m CC[ . , . , 1 , 4 ]` $<$ `-m CC[ . , . , 1 , 6 ]`
- but computation time also increases
- Details matter: pre-processing, feature extraction, etc.

Meaning of ANTs registration parameters

What happens when i vary each parameter?

- Robustness increases with regularization
- `-r Gauss[6,3]` > `-r Gauss[3,0]`
- Flexibility decreases with regularization
- `-r Gauss[6,3]` < `-r Gauss[3,0]`
- Robustness increases with correlation window
- `-m CC[ . , . , 1 , 4 ]` < `-m CC[ . , . , 1 , 6 ]`
- but computation time also increases
- Details matter: pre-processing, feature extraction, etc.
- Successful affine step is essential!!

Meaning of ANTs registration parameters

What happens when i vary each parameter?

- Robustness increases with regularization
- `-r Gauss[6,3]` $>$ `-r Gauss[3,0]`
- Flexibility decreases with regularization
- `-r Gauss[6,3]` $<$ `-r Gauss[3,0]`
- Robustness increases with correlation window
- `-m CC[ . , . , 1 , 4 ]` $<$ `-m CC[ . , . , 1 , 6 ]`
- but computation time also increases
- Details matter: pre-processing, feature extraction, etc.
- Successful affine step is essential!!
- Step-size increases stability but slows convergence

Meaning of ANTs registration parameters

What happens when i vary each parameter?

- Robustness increases with regularization
- `-r Gauss[6,3]` > `-r Gauss[3,0]`
- Flexibility decreases with regularization
- `-r Gauss[6,3]` < `-r Gauss[3,0]`
- Robustness increases with correlation window
- `-m CC[ . , . , 1 , 4 ]` < `-m CC[ . , . , 1 , 6 ]`
- but computation time also increases
- Details matter: pre-processing, feature extraction, etc.
- Successful affine step is essential!!
- Step-size increases stability but slows convergence
- `SyN[ 0.1 ]` more stable than `SyN[ 0.25 ]`.

Coordinates of computation time

- 2D 256^2 pixels intensity difference (MSQ) registration $\approx$ 30 seconds

Coordinates of computation time

- 2D 256^2 pixels intensity difference (MSQ) registration $\approx$ 30 seconds
- 3D 256^3 voxels correlation-8 (CC[. , . , 1, 8]) could take 3 days if you use full-resolution and the images are very different.

Coordinates of computation time

- 2D 256^2 pixels intensity difference (MSQ) registration $\approx$ 30 seconds
- 3D 256^3 voxels correlation-8 (CC[. , . , 1, 8]) could take 3 days if you use full-resolution and the images are very different.
- or it could take 15 minutes if you use low-resolution and the images are very similar.

Multiple metrics driving registration

```
antsRegistration -d $dim -r [ $img1, $img2, 1 ] \
  -m mattes[$img1,$img2,1,32,Regular,0.25]  \
  -t Affine[0.1] \
  -c 50x40x30 \
  -f 4x2x1 -s 2x1x0 \
  -m CC[$img1,$img2,1,4]  \
  -m meansquares[$lm1,$lm2,1,4]  \
  -f 4x2x1 -s 2x1x0 \
  -t SyN[0.25,3,0] -c 50x40x30 -o $out
```

Multiple metrics driving registration

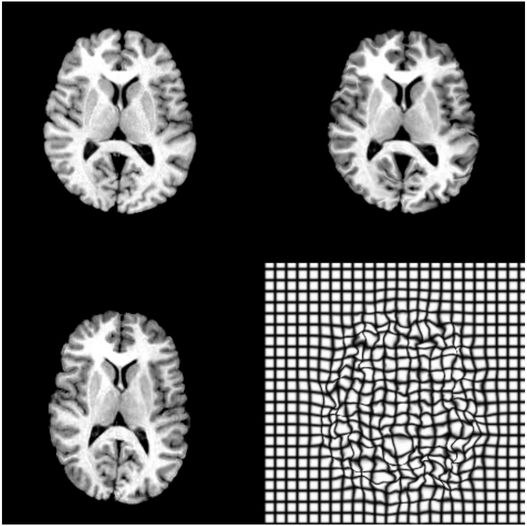


Figure 29:

Landmark-based registration 1

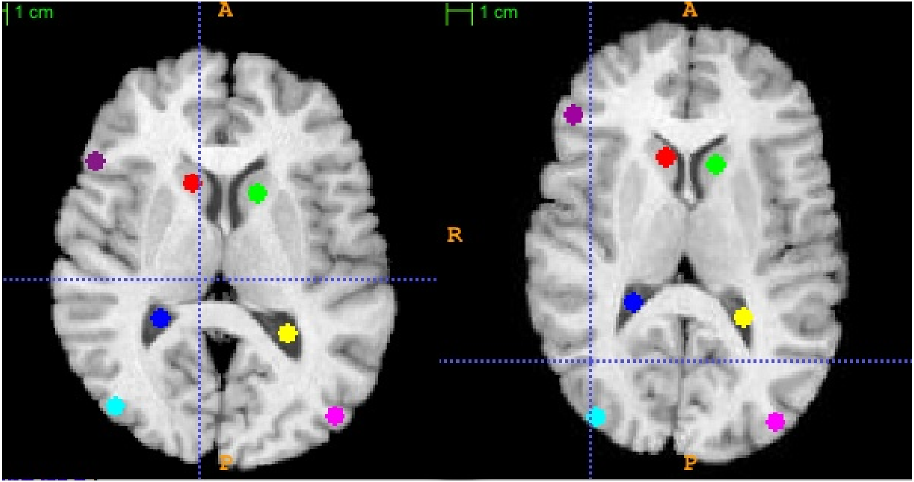


Figure 30:

Landmark-based registration 2

```
WarpImageMultiTransform $dim $lm2  ${out}lm.nii.gz \
 ${out}preAffine.txt -R $img1 --use-NN
WarpImageMultiTransform $dim $img2 ${out}img.nii.gz \
 ${out}preAffine.txt -R $img1
wt=1 ; pct=0.5 ; sig=50
 ANTS $dim  -i 55x40x30  -r Gauss[8,0] -t SyN[ 0.25 ]
 -m PSE[ $lm1 ,  ${out}lm.nii.gz  , $lm1 ,  ${out}lm.nii.gz  ,
  -m  CC[$img1,${out}img.nii.gz,1,4] -o $out -i 50x50x50 --nu
 --use-all-metrics-for-convergence 1 --continue-affine 0
```

Landmark-based registration 3

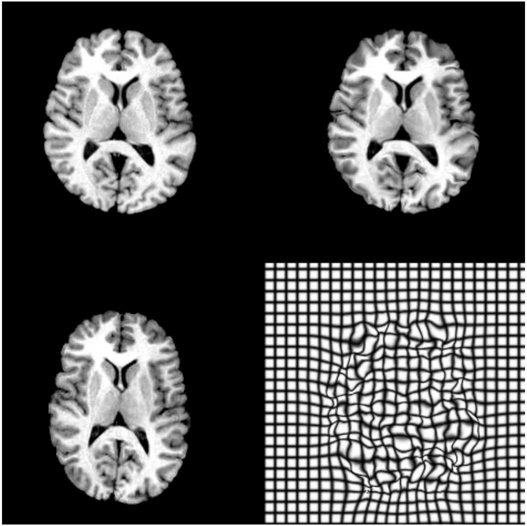


Figure 31:

Other Landmark-based registration tools

- for 3D

Other Landmark-based registration tools

- for 3D
- ANTSUseLandmarkImagesToGetAffineTransform lm1.nii.gz lm2.nii.gz affine outaffine.txt

Other Landmark-based registration tools

- for 3D
- ANTSUseLandmarkImagesToGetAffineTransform lm1.nii.gz lm2.nii.gz affine outaffine.txt
- ANTSUseLandmarkImagesToBSplineDisplacementField lm1.nii.gz lm2.nii.gz outLMWarp.nii.gz 10x10x10 6 3 0

Other Landmark-based registration tools

- for 3D
- ANTSUseLandmarkImagesToGetAffineTransform lm1.nii.gz lm2.nii.gz affine outaffine.txt
- ANTSUseLandmarkImagesToBSplineDisplacementField lm1.nii.gz lm2.nii.gz outLMWarp.nii.gz 10x10x10 6 3 0
- Then use antsApplyTransforms to apply the warp to the relevant image.

Conclusions

- Background: ANTs fuses ideas from mathematics, optimization, software engineering

Conclusions

- Background: ANTs fuses ideas from mathematics, optimization, software engineering
- Design: ANTs uses interchangeable pieces to aid flexibility

Conclusions

- Background: ANTs fuses ideas from mathematics, optimization, software engineering
- Design: ANTs uses interchangeable pieces to aid flexibility
- Metrics: We illustrated the advantages of different similarity metrics

Conclusions

- Background: ANTs fuses ideas from mathematics, optimization, software engineering
- Design: ANTs uses interchangeable pieces to aid flexibility
- Metrics: We illustrated the advantages of different similarity metrics
- Transforms: We showed both constrained and highly flexible mappings and how to use them

Conclusions

- Background: ANTs fuses ideas from mathematics, optimization, software engineering
- Design: ANTs uses interchangeable pieces to aid flexibility
- Metrics: We illustrated the advantages of different similarity metrics
- Transforms: We showed both constrained and highly flexible mappings and how to use them
- Optimization: We indicated our primary optimization techniques and showed a tool for overcoming challenging registration scenarios

Acknowledgements

- NIBIB for software development support under grant R01-EB006266-01.

Acknowledgements

- NIBIB for software development support under grant R01-EB006266-01.
- The ITK development team.

Acknowledgements

- NIBIB for software development support under grant R01-EB006266-01.
- The ITK development team.
- Dr. Murray Grossman for pushing application development.

Acknowledgements

- NIBIB for software development support under grant R01-EB006266-01.
- The ITK development team.
- Dr. Murray Grossman for pushing application development.
- The NLM for providing stimulus funding in support of ITKv4.