A Practical Tutorial for Brain Image Registration Using the ANTs Normalization Tools

Alexander Woodward
NIJC, RIKEN BSI
Agenda

1. What is ANTs?
   1. What can it be used for?

2. Steps in registration.

3. ANTs SyN nonlinear registration algorithm.
   1. (I will skip over slides with a blue star in the corner.)

4. Example walk-through.
What is ANTs

• ANTs (Advanced Normalization Tools).
  – A state-of-the-art medical image registration and segmentation toolkit.
  – Built from ITK.
  – ANTs ‘Image’ data can be 2D/3D/4D.

• We can use it to register brain image data to a common space for analysis.
• We can use it to create brain atlases.
• I will focus on the data registration component of ANTs.
Getting ANTs

• ANTs website:
  – http://stnava.github.io/ANTs/

• For *nix OSes: sudo apt-get install ants

• Make from source:
  – https://github.com/stnava/ANTs
  – Personally tested on OSX Yosemite.
Registration Required Input

1. Fixed image (brain dataset 1).

2. Moving image (brain dataset 2):
   1. Transformed to be as close as possible to the fixed image.

3. Mask image (optional).
   1. Reduce computation time.
   2. Mask should include object boundaries - a strong image feature for matching.
Registration Procedure

1. Roughly align datasets using *linear* registration:
   1. Centers align.
   2. Orientations align.
   3. Account for any scale factors.
   4. Linear enough for data from same individual, e.g. T1 to T2.

2. Fine alignment using *nonlinear* registration:
   1. Match boundaries and internal structure by warping the data.
1. **Linear:**
   1. Translation: (3 DOF).
   2. Rigid: (6 DOF).
   3. Similarity: (7 DOF).
   4. Affine: (12 DOF).

2. **Nonlinear:**
   1. Bspline, SyN, etc. (many DOF)

• At each stage specify a similarity metric:
  – A measure to compare fixed and transformed moving image.

• Multi-resolution approach: specify number of scales to match over:
  – Matching at low-resolution is used to initialize finer-resolution match.
Registration Procedure Example

• Register moving image to fixed image.
  – Orientation, scaling, and nonlinear differences.
Image Alignment

- Roughly align positions.
Similarity Transform

• Orient and scale both datasets.
Affine Transform

• Linearly match as close as possible.
  – Depending on data affine may be sufficient.
Nonlinear Transform

- Refine match using nonlinear techniques, e.g. SyN.
Nonlinear Registration: SyN

• ANTs includes a powerful nonlinear registration algorithm called SyN (Symmetric Normalization).

• Popular and a top performing algorithm:
SyN Based Warping (Ex. 1)

- Half C to full C:

![Image of deformation field]

Deformation field
SyN Based Warping (Ex. 2)

- Linear + nonlinear bidirectional mapping:

  Symmetric Diffeomorphic Mapping with ANTS
  Affine+ Diffeomorphic SyN

Figure 9: This example shows the benefit of the symmetric normalization model—invertibility, symmetry, highly deformable and accurate registration. This example may be recreated by the reader via: http://stnava.github.io/cars/
Large Deformation Diffeomorphomic Metric Mapping (LDDMM)

• SyN is based on theory of diffeomorphisms.
  – Mappings smooth and invertible.
  – Bijective (one to one mapping)
    • No overlapping points
    • Bspline approach cannot guarantee this.

• SyN is a large deformation diffeomorphic metric mapping (LDDMM) algorithm
A Number of LDDMM Formulations

- E.g. SyN:
  - Original (Geodesic) SyN.
  - Greedy SyN.
  - BSpline SyN.
  - Refer to paper “ANTS: Open-Source Tools for Normalization And Neuroanatomy” - Avants et al.

- E.g. DARTEL:
  - Focuses on estimating a static flow field $v$. 
General LDDMM Approach

- For mapping image $I$ to $J$, by the diffeomorphic mapping, $\Phi$, minimize the functional:

$$
\nu^* = \arg\min_{\nu} \left\{ \int_0^1 \|L\nu\|^2 dt + \lambda \int_\Omega \Pi(\mathcal{I}, \phi(x, 1), \mathcal{J}) d\Omega \right\}
$$

- $\Phi$ generated by integrating a smooth velocity field $\nu$.
- (Terms are described in next slides)
First Term

• Smoothness term.
  – Minimize a smooth velocity field: the geodesic path between two images.
  – $L$ is a smoothing operator.
  – $t$ is time.

\[
\int_0^1 \| L \nu \|^2 dt
\]
Second Term

• **Data term:** Compare the similarity between two images
  – $\lambda$: controls exactness in matching
  – $\Omega$: image domain
  – $x$: position

\[
\lambda \int_{\Omega} \Pi_\sim(\mathcal{I}, \phi(x, 1), \mathcal{J}) d\Omega
\]

• $\Pi_\sim$ is a similarity metric:
  – E.g. sum of squared differences (SSD), cross-correlation (CC), or mutual information (MI)
  – Allows for inexact matching due to photometric transformations or intermodal differences.
Similarity Metric Recommendations

• MI: Recommended in general.
  – Especially for attempting intermodal registration.

• CC: Can be useful for fine-scale nonlinear registration step.
  – More computationally intensive than MI.
CC and MI References

• MI formulation (Mattes MI):

• CC formulation (original SyN paper):
Symmetry in LDDMM

• Diffeomorphisms are theoretically symmetric
• SyN algorithm invented to exploit symmetry in diffeomorphic mapping:
  – Both images are treated equally in formulation.
  – Computationally efficient.
  – Calculates both forward and inverse transforms.
  – Symmetry minimizes potential interpolation errors in invertibility of the diffeomorphism.
SyN Algorithm Approach

• The functional $v^*$ is rewritten to have two velocity fields, $v_1, v_2$ (from $I$ to $J$ and $J$ to $I$).

• Data term is evaluated at the mid-point, $t = 0.5$.

$$\{v_1^*, v_2^*\} = \arg\min_{v_1, v_2} \left\{ \int_0^{0.5} \|Lv_1\|^2 dt + \int_0^{0.5} \|Lv_2\|^2 dt + \lambda \int_{\Omega} \Pi_n (I \circ \phi(x, 0.5), J \circ \phi_2(x, 0.5)) d\Omega \right\}$$
Euler-Lagrange Equations

- In a E-L formulation $v_1$ and $v_2$ form the action and first and second terms form the Lagrangian $L$.
  - Calculated E-L eq.s are the velocity fields, $v_1$, $v_2$, based on the smoothed gradient of the similarity metric w.r.t diffeomorphisms $\Phi_1$, $\Phi_2$.

- Original SyN: integrating from $t = 0$ to 0.5, gives estimate for $\Phi_1$ and $\Phi_2$. 
Original (Geodesic) SyN Formulation

For each time step calculate \( \nu_i \) as:

\[
\nu_i(x, t) = \nu_i(x, t) + \delta L \ast \nabla \Pi_i(x, t)
\]

Do numerical integration from \( t = 0 \) to 0.5 to get \( \Phi_i \):

\[
\frac{d\phi_i(x, t)}{dt} = \nu_i(\phi_i(x, t), t), \quad \phi_i(x, 0) = Id, \quad i \in \{1, 2\}
\]
Greedy SyN Formulation

• Only evaluate metric at end points of $\Phi_1$ and $\Phi_2$:

$$\nabla \Pi = \frac{\partial}{\partial \phi_i} \Pi_{\sim}(I(\phi_1^{-1}(x, 0.5)), J(\phi_2^{-1}(x, 0.5)))$$

• Calculate $\Phi_i$: gradient at midpoint is mapped back to origin of each diffeomorphism:

$$\phi_i(x, 0.5) = \phi_i(x, 0.5) + (\delta L \ast \nabla \Pi_i(x, 0.5)) \circ \phi_i(x, 0.5)$$
Enforcing Invertibility Constraint

- Explicit discrete domain enforcement of:
  \[ \phi^{-1}(\phi(x, 1)) = x \]

- Achieved by iterative inverse transform field estimation.

- Algorithm details:
Full Diffeomorphism from $\Phi_1$ and $\Phi_2$

- Full diffeomorphism, $\Phi$, and its inverse, $\Phi^{-1}$, can be calculated through the following compositions:

\[ \phi = \phi_1 \circ \phi_2^{-1}, \quad \phi^{-1} = \phi_2 \circ \phi_1^{-1} \]
SyN Based Warping (Ex. 3)

- Calculations performed in both directions from $\Phi_1(x,0)$ and $\Phi_2(x,0)$.

- **Row ‘Initialization’:** Each column shows the same starting shape.
- **Row ‘Solution’:** $I$ can be warped to $J$ and vice versa between any point along the horizontal axis.
SyN Supports Adding Landmarks and Multi-channel Data

• Multi-channel data registration via multiple metrics.
  – RGB, gradient image, etc.

• Can include landmarks to constrain registration.

\[ \int_0^1 \langle Lv(x, t), v(x, t) \rangle \, dt + w_1 SSD(I, J) + w_2 MI(I, J) + w_3 \sum_i LM_i(I, J) \]

- Diffeomorphic Regularization
- Intensity Difference
- Mutual information
- Landmark Guidance

Could be different channels, e.g. \( I_{\text{edge}}, J_{\text{Edge}} \)
Greedy SyN Pseudocode

Algorithm 1 | Greedy SyN algorithm

\[
\phi_i \leftarrow \text{Id}, \quad \phi_i^{-1} \leftarrow \text{Id} \quad \triangleright \quad i \in \{1, 2\}
\]

for all image resolution levels do

\[
\begin{align*}
\quad n & \leftarrow 1 \\
\textbf{while} \text{ not converged do} \\
\quad \nu_1^n & \leftarrow \nabla \Pi_{\sim} \left( I \circ \phi_1^{n-1}, I \circ \phi_2^{n-1} \right) \\
\quad \nu_2^n & \leftarrow \nabla \Pi_{\sim} \left( I \circ \phi_2^{n-1}, I \circ \phi_1^{n-1} \right) \\
\quad \nu_i^n & \leftarrow S_{\nu}(\nu_i^n) \quad \triangleright \quad S_{\nu} \text{ is a smoothing operation on the}
\quad \text{update transform field} \\
\quad \phi_i^n & \leftarrow S_{\phi}(\nu_i^n \circ \phi_i^{n-1}) \quad \triangleright \quad S_{\phi} \text{ is a smoothing operation on the}
\quad \text{total transform field} \\
\quad (\phi_i^n)^{-1} & \leftarrow \text{Inv} \left( \phi_i^n, \left( \phi_i^{n-1} \right)^{-1} \right) \quad \triangleright \quad \text{Inverse field}
\end{align*}
\]

estimation described in Avants et al. (2008)

\[
\begin{align*}
\quad n & \leftarrow n + 1 \\
\textbf{end while} & \\
\quad \text{upsample current } \phi_i \text{ and } \phi_i^{-1} \text{ to next resolution level} \quad \triangleright \quad i \in \{1, 2\}
\end{align*}
\]

end for

return \( \phi \leftarrow \phi_1 \circ \phi_2^{-1}, \phi^{-1} \leftarrow \phi_2 \circ \phi_1^{-1} \)
Walk-through Example

- Walk through individual MRI to MRI template registration.
- Example from ANTs website: 
  http://stnava.github.io/BasicBrainMapping/
Files Used

• From the zip file:
  – Bash script: `bbm.sh`
  – Fixed image: `./data/IXI/T_template2.nii.gz`
  – Moving image: `./data/IXI594-Guys-1089-T1.nii.gz`
  – Mask image: `./data/IXI/T_templateExtractionMask.nii.gz`

• `.nii.gz` are compressed Nifti files.
Individual MRI (viewed in 3D Slicer)
Template MRI
Register individual to template
Registration result (with mask)
Volume Slices: Template
Volume Slices: Individual
Volume Slices: Template Mask
Volume Slices: 50% Template, 50% Mask
Registration Result: Individual Mapped to Template
Video: Volume Rendering Result

- Video removed to reduce file size.
#!/bin/bash

dim=3 # image dimensionality
AP=“/home/yourself/code/ANTS/bin/bin/” # path to ANTs binaries
ITK_GLOBAL_DEFAULT_NUMBER_OF_THREADS=4 # controls multi-threading
export ITK_GLOBAL_DEFAULT_NUMBER_OF_THREADS

f=$1 ; m=$2 ; mask=$3 # fixed and moving image file names and fixed image
mas, here the fixed image is the template
if [[ ${#f} -eq 0 ]] ; then # CLI feedback when parameters are not given
correctly to the script
echo usage is
echo $0 fixed.nii.gz moving.nii.gz fixed_brain_mask.nii.gz
exit
fi
Bash Script Walkthrough

if [[ ! -s $f ]]; then echo no fixed $f; exit; fi
if [[ ! -s $mask ]]; then echo no fixed mask $mask; exit; fi
if [[ ! -s $m ]]; then echo no moving $m; exit; fi

reg=${AP}antsRegistration       # path to antsRegistration
its=10000x1111x5  #iterations per scale for affine step
percentage=0.25  #percentage of voxels sampled for evaluating the metric
syn="20x20x0,0,5"  #iterations per scale and stopping criterion
nm=BBM  #naming prefix
imgs=" $f, $m "  #variable specifying the fixed and moving images
Bash Script Walkthrough

if [[ ! -s ${nm}0GenericAffine.mat ]]; then #run if the .mat file does not exist
$reg -d $dim -r [ $imgs ,1] ¥ #initialize based on aligning centroids of voxel intensities

-m mattes[ $imgs , 1 , 32, regular, 0.05 ] ¥ #metric
-t translation[ 0.1 ] ¥ #transformation type
-c [1000,1.e-8,20] ¥ #no. of iterations and stopping criteria
-s 4vox ¥ #smoothing sigmas
-f 6 -l 1 ¥ #scale factors 6= 1/6 original size + -l estimate learning rate
-m mattes[ $imgs , 1 , 32, regular, 0.1 ] ¥
-t rigid[ 0.1 ] ¥
-c [1000x1000,1.e-8,20] ¥ #two scales used for rigid
-s 4x2vox ¥
-f 4x2 -l 1 ¥
Bash Script Walkthrough

-m mattes[ $imgs , 1 , 32, regular, 0.1 ]
-t affine[ 0.1 ]
-c [$its,1.e-8,20] #three scales used for affine
-s 4x2x1vox
-f 3x2x1 -l 1
-m mattes[ $imgs , 1 , 32 ]
-t SyN[ .20, 3, 0 ]
-c [ $syn ]
-s 1x0.5x0vox
-f 4x2x1 -I 1 -u 1 -z 1 -x $mask --float 1 #u use histogram matching, -z combine output transforms (linear/nonlinear), -x use a mask during nonlinear step
-o [${nm},${nm}_diff.nii.gz,${nm}_inv.nii.gz] #specify output prefix, forward and reverse file names

${AP}antsApplyTransforms -d $dim -i $m -r $f -n linear -t ${nm}1Warp.nii.gz -t
${nm}0GenericAffine.mat -o ${nm}_warped.nii.gz --float 1 #example of applying the calculated linear+nonlinear transforms to input data
fi
File output

• The following files are created:

  \textit{BBM\_diff.nii} moving image after registration to fixed image.

  \textit{BBM\_inv.nii} fixed image transformed to moving image, using the inverse transform.

  \textit{BBM0GenericAffine.mat} composite linear transform.

  \textit{BBM1Warp.nii.gz} nonlinear forward warp.

  \textit{BBM1InverseWarp.nii.gz} nonlinear inverse warp.
Video: Registration Result

• Video removed to reduce file size.
Recommended Reading

• ANTs documentation:
• Avants et al. “ANTS: Open-Source Tools for Normalization And Neuroanatomy”. (Could not find a citation.)
  – Describes SyN variants
  – Gives pseudocode for the Greedy SyN algorithm
  – Gives formulation of SyN using CC metric
Web Links

• ANTs:
  http://stnava.github.io/ANTs/

• Basic Brain Mapping example:
  http://stnava.github.io/BasicBrainMapping/

• 3D Slicer:
  http://www.slicer.org/
End

• Any questions?