Advanced Normalization Tools (ANTS)

Release 2.x

Brian B. Avants[1], Nick Tustison[2] and Hans Johnson[3]

June 21, 2014

University of Pennsylvania[1]
University of Virginia[2]
University of Iowa[3]

Abstract

We provide examples and highlights of Advanced Normalization Tools (ANTs), versions 2.x, that address practical problems in real data.

Contents

1 Introduction

This update to ANTs documentation was initiated April 29, 2014. *This document does not cover all of ANTs functionality — but summarizes the most frequently used components.* Originally, the ANTs framework provided open-source functionality for deformable image registration with small or large deformations, as shown in figure 1. Independent evaluation of the 0.0 version of ANTs software, applied to "control" data, placed the toolkit as a top performer amongst 14 methods [26]. Developer evaluation showed stronger differences with other methodology in neurodegenerative neuroimaging data, where large deformation is required [10]. ANTs has since grown to include N4 bias correction [37], additional evaluation of multiple modalities and organ systems [36, 38, 28], univariate or multivariate image segmentation [19, 40], tighter integration with the Insight ToolKit [35, 17], a well-evaluated cortical thickness pipeline [41] and, more recently, visualization tools and integration with *R*[42]. ANTs serves as both a core library for further algorithm development and also as a command-line application-oriented toolkit. ANTs also has a permissive software license that allows it to be employed freely by industry [31]. ANTs enables diffeomorphic normalization with a variety of transformation models, optimal template construction, multiple types of diffeomorphisms, multivariate similarity metrics, diffusion tensor processing and warping, image segmentation with and with-

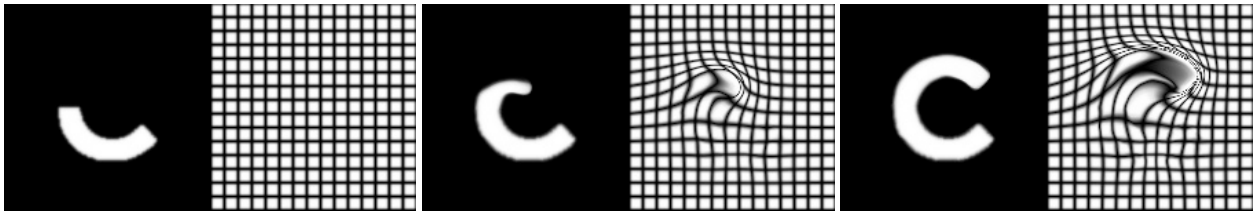


Figure 1: The original goal of ANTs was to develop public, open source large deformation image registration. This is a classic example showing the progress of deforming a half C to a full C along a geodesic diffeomorphism. The deforming grid accompanies each deformed image. See http://stnava.github.io/C/ for example data and code.

out priors and measurement of cortical thickness from probabilistic segmentations. The normalization tools, alone, provide a near limitless range of functionality and allow the user to develop customized objective functions. Objective functions in ANTs are of the form:

$$\text{Deformation Cost} + \text{Data Terms},$$

and the command line reflects this balance of two terms. As mentioned above, the data term may combine multiple different measures of similarity that are optimized in parallel, for instance, image similarity and landmark terms. This document seeks to provide a practical overview of basic functionality and some of the common use cases that users seek. Additional information is available online – see http://stnava.github.io/ANTs/. For compilation details, see: https://brianavants.wordpress.com/2012/04/13/updated-ants-compile-instructions-april-12-2012/ or section 6.1. The most important core C++-based ANTs programs are: `antsRegistration`, `antsApplyTransforms`, `Atropos` for segmentation, `N4BiasFieldCorrection` for inhomogeneity correction, `KellyKapowski` for estimating thickness, `ImageMath` image processing utilities and, finally, `sccan` for sparse dimensionality reduction. There are many other programs which are listed in https://github.com/stnava/ANTs/tree/master/Examples. The scripts in ANTs wrap the core programs and are in https://github.com/stnava/ANTs/tree/master/Scripts. Perhaps the most important are `antsCorticalThickness.sh` as a wrapper of several sub-programs that support our cortical thickness pipeline [41], `antsMultivariateTemplateConstruction2.sh` for template construction and `antsRegistrationSyN.sh` as a basic interface to a few commonly used registration approaches. [1]

1.1 Structure of this document and its examples

This document is generated with LaTeX and is version controlled at https://github.com/stnava/ANTsDoc. The examples, here, have data and example scripts stored at dedicated github repositories, to which we will refer. Other (simpler) examples will use data and scripts that are stored in the ANTsDoc git repository. These scripts are (loosely speaking) tested and should serve as reproducible examples for the reader to try. All data and code is available via ANTs-related repositories. The `compile` script builds both latex and tests the example scripts.

1.2 Example 1: Quick SyN

If you want a decent, fast registration you might run something like this:

```
1 antsRegistrationSyNQuick.sh -d 2 -f r16slice.nii.gz -m r64slice.nii.gz -o $op
```

[1] This document is a work in progress. Please check for updates with each release.

The variable $op represents the output prefix for the filename. This will be the case in many of the following examples. The -d option denotes image dimensionality, -f option denotes the "fixed" image and the -m option denotes the "moving" image. The moving image will be deformed to match the fixed image. The inverse of these maps deform the fixed to the moving image. Output is determined by -o where the output will be named with prefix=outputEx1.sh (the word output concatenated with the name of the Example 1 script) and include a prefix0GenericAffine.mat (the low-dimensional affine transform which may be inverted numerically), the prefix1Warp.nii.gz (the diffeomorphic transformation pulling the affine transformed moving image toward the fixed image), and the prefix1InverseWarp.nii.gz (the inverse diffeomorphic transformation pulling the fixed image toward the affine transformed moving image). The caveats for this "canned" approach include: 1. registration performance can always be improved by using prior knowledge [44]; 2. there are many assumptions about the data embedded in the above call and they may not be appropriate for whatever problem is at hand; 3. you must have some facility with the command line to run a shell script. It is, in general, better to understand a little bit about image registration rather than running methods blindly. To aid readers in this, we have two options: 1. github issues https://github.com/stnava/ANTs/issues; 2. sourceforge discussion or help sites http://sourceforge.net/p/advants/discussion/. Feel free to use either to ask clarifying questions. We note that many issues have been discussed previously and you might find answers by searching the archives.

1.3 The antsRegistration executable

The antsRegistration program itself is the central program encapsulating normalization/registration functionality. Its main output is an affine transform file and a deformation field, potentially with inverse. Options allow the user to navigate the similarity and transformations that are available. antsRegistration allows multiple similarity and optimization criteria as options. The program is wrapped in antsRegistrationSyN.sh for normalization with "out of the box" parameters and in antsMultivariateTemplateConstruction2.sh for computationally distributed optimal (multivariate) template construction.

Initializing antsRegistration

You can use the -r option in antsRegistration to initialize a registration with an ITK .mat format transformation matrix, with a deformable mapping or with a center of mass alignment. See the scripts for examples. The output transformation will include the initial transformation. FIXME need to check this.

1.4 The antsApplyTransforms executable

The antsApplyTransforms program applies ANTs mappings to images including scalars, tensors, timeseries and vector images. It also composes transforms together and is able to compute inverses of low-dimensional (affine, rigid) maps. antsApplyTransformsToPoints similarly works on point sets (see http://stnava.github.io/chicken/ for details). One may apply an arbitrarily long series of transformations through these programs. Thus, they enable one to compose a series of affine and deformable mappings and/or their inverses. One may therefore avoid repeated interpolations of a single image. Several different interpolation options are available and multiple image types may be transformed including: tensors, vectors, timeseries and d-dimensional scalar images where $d = 2,3,4$.

1.5 Using antsApplyTransforms or antsApplyTransformsToPoints

For example: to apply the transform to the moving image and the inverse to the fixed image:

```
1 antsRegistrationSyNQuick.sh -d 2 -f B.nii.gz -m A.nii.gz -o RegA2B
2 antsApplyTransforms -d 2 -i A.nii.gz -o ADeformed.nii.gz -r B.nii.gz -t RegA2B1Warp.
    nii.gz -t RegA2B0GenericAffine.mat
3 antsApplyTransforms -d 2 -i B.nii.gz -o BDeformed.nii.gz -r A.nii.gz -t [
    RegA2B0GenericAffine.mat,1] -t RegA2B1InverseWarp.nii.gz
```

The numbers of the transformations (here, 0 and 1 because there is only a deformation and affine mapping) relate to the order in which the transforms are computed, during optimization, and also the order in which they should be applied. There is reasonably complete discussion of this framework in [17].

The usage of antsApplyTransformsToPoints is nearly identical. *However*, it is critical to recognize that transforms that are one-to-one and onto in image space may not be in point space. Therefore, points are transformed by the inverses of the transformation that is applied to images. This is discussed in several places in the image registration literature but we first discussed this in [16].

1.6 The antsMotionCorr executable

Performs motion correction of time-series data. Control parameters are similar to antsRegistration. See the example http://stnava.github.io/fMRIANTs/. This example also shows how to run basic CompCor on fmri data. Our minimal fMRI pipeline involves running antsMotionCorr and CompCorr to factor out nuisance variables. More complex approaches require *ANTsR*.

1.7 I/O data formats in ANTs

ANTs supports 2D, 3D and, in some cases, 4D images. Since ANTs is implemented in concert with the Insight ToolKit (ITK) http://www.itk.org/, it is able to read and write the popular data formats supported through ITK. ANTs also uses the ITK view of world-coordinates which can be confusing, at times, when one is mixing software. There is much discussion of this on the web and in ANTs discussion sites/issues. Whatever is relevant for ITK world coordinates is relevant to ANTs as these two software systems agree, in their entirety, about what the voxel to physical space coordinate mapping should be. There are four basic types of data for ANTs.

Image volumes

ANTs supports 2D, 3D, 4D images, including

- Nifti (.nii, .nii.gz)
- Analyze (.hdr + .img / .img.gz)
- MetaImage (.mha)
- Other formats through itk::ImageFileWriter / itk::ImageFileWriter such as jpg, tiff, etc. See ITK documentation.

Affine transformation file

ANTs uses the itk::TransformFileReader / itk::TransformFileWriter to handle affine transformation files. These are Matlab like .mat files recording the matrix and offset for the low-dimensional transforms: translation, rigid, affine. The ordering of a matrix transformations parameters breaks down like this: if parameters are, $a\ b\ c\ d\ e\ f\ g\ h\ i\ j\ k\ l$, with origin (or FixedParameters), $M\ N\ O$, then the matrix components (rotation, scaling, shearing all composed together) are:

$a\quad b\quad c$
$d\quad e\quad f$
$g\quad h\quad i$

followed by $j\ k\ l$ (the translation) where $L\ M\ N$ is the center of rotation.

Deformation field file

ANTs writes the deformation field using image formats that can store multi-dimensional pixels, usually a nifti-image. The values of each pixel are the displacements in physical space. So, $y = u(x) + x$ where $u(x)$ is the displacement at x.

Labeled point sets—*Currently not supported by antsRegistration, only by old ANTS*

There are three formats supported

- Labeled point sets are saved as an image volume with the intensity = 1,2,3 for label = 1,2,3. For example, figure **??** shows 2D images representing a 3-class labeled point set.
- CSV Format. In this simple format, the point sets are saved as a text csv file. See the chicken example for details http://stnava.github.io/chicken/.

1.8 The ImageMath executable

This is a multi-purpose program that has the following syntax: `ImageMath ImageDimension outputfilename Operation InputFileName parameters` ... Most basic scalar image operations – and some tensor operations – may be performed with this program. Some operations output text files, some output images and some output only to the terminal. `ImageMath` allows one to multiply images together (m), to negate images (Neg), to take an image to a power (pow), to test the invertibility of transformations (InvId), to compute the fractional anisotropy of an image (TensorFA) and to compute the gradient or laplacian of an image (Gradient, Laplacian). Many other operations are available. Like all other ANTs programs, one may call `ImageMath` from the command line to see all of its options. `ImageMath` is used heavily in ANTs scripts.

```
1 ConvertToJpg r64slice.nii.gz Figures/${bn}ad1.jpg
2 ImageMath 2 ${bn}ad.nii.gz PeronaMalik r64slice.nii.gz 20 0.5
3 ConvertToJpg ${bn}ad.nii.gz Figures/${bn}ad2.jpg
4 ImageMath 2 ${bn}ad.nii.gz PeronaMalik r64slice.nii.gz 200 0.5
5 ConvertToJpg ${bn}ad.nii.gz Figures/${bn}ad3.jpg
```

Above, ImageMath's anisotropic diffusion.

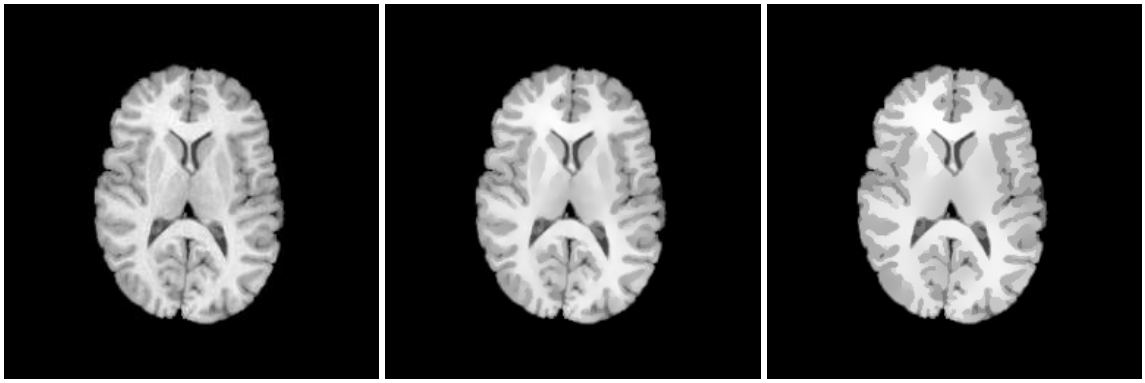


Figure 2: The original image with 2 degrees of anisotropic diffusion.

1.9 ANTs/Scripts

The ANTS/Scripts directory contains (hopefully) user-friendly wrappings of ANTs tools that enable higher-level error checking and combinations of basic ANTs functions. These scripts are called as `bash antsscriptname.sh` and provide usage when called from the command line. For instance, try `bash antsRegistrationSyN.sh`. Additionally, if you run ants in tcsh shells, you will need to call it as follows (using `antsRegistration` as an example):

```
antsRegistration -d 2 -m CC\[r16slice.nii,r64slice.nii,1,2\] ...
```

instead of

```
antsRegistration -d 2 -m CC[r16slice.nii,r64slice.nii,1,2 ] ...
```

because tcsh does no interpret the brackets as intended. Using the bash shell script interface also avoids this. The key change involves using the slash to force tcsh to interpet the bracket literally.

2 Image registration with ANTs

Quick Start: call `antsRegistrationSyNQuick.sh` (after copying the contents of ANTs/Scripts/ to your bin directory) to get usage and apply a normalization to some of your existing data. It is instructional to read the script, modify some of the parameters and re-run to witness the effect of your changes – image registration is an art as well as science. Many other "ready to go" scripts are available in ANTS/Scripts. Often, the user must set his/her ANTSPATH environment variable – which points to the location of ANTs binaries – within these scripts or export the ANTSPATH variable into the user or script environment. Note: All ANTs programs provide usage when called from the command line. Most require the image dimension to be specified as the first parameter. E.g. `ImageMath` ImageDimension where ImageDimension is 2, 3, 4.

There are two general applications of image registration. The first application is transforming labeled data from a template image space into an individual space. This strategy is important when appearance alone is not enough to locate a structure as, for example, in the case of hippocampus segmentation. The template provides a prediction of the hippocampus-amygdala boundary. Mapping multiple templates to an individual (multi-template labeling) can further improve accuracy [43, 18]. The second application operates in the "inverse" direction of the first: instead of mapping template to individual, we map individual(s) to the template. Voxel-based population studies of either functional or structural variables depend on mapping to a template space. The common coordinate system enables a statistical evaluation of the likelihood of consistent activation across a group or, in other contexts, the differences in anatomy between two groups.

The ANTs toolkit enables both types of mapping. The main challenge in image and brain mapping is defining the way in which images/anatomy are compared. There are two components to the comparison. The shape transformation space defines the range of shape variation that will be allowed in the optimization. The appearance similarity space defines the statistical assumptions that determine when one image is considered to appear similar to another.

These two components interact in a weighted optimization within a multiple resolution gradient descent framework. Each component may use either "strict" or "flexible" assumptions about the shape or appearance similarity. The selection of these models should be done in a principled way that fits the problem at hand. No single choice is appropriate for all scenarios (see the "no free lunch" theorem).

Thus, ANTs enables many operating points from both the transformation and appearance domains such that users may make choices appropriate for their problems. The ANTs command-line syntax, shown in figure 3, reflects these operating points and the various components that interact in the optimization. ANTs may be used to navigate the transformation and similarity metric space.

2.1 World coordinates: Use your header!

ANTs uses the direction/orientation, spacing and origin in its definitions of the mapping between images. That is, ANTs uses the ITK standard defintions of an image space. So, the image orientation, direction matrix, origin and spacing are important! If in doubt about, for example, which side is viewed as left or right by ANTs, then take a look at ITK-SNAP http://www.itksnap.org and label an image's known left side. Then apply an ANTs or ITK mapping to the image to see how the data is transformed. *ITK and ANTs do not use the s-form that may be stored in some Nifti images.*

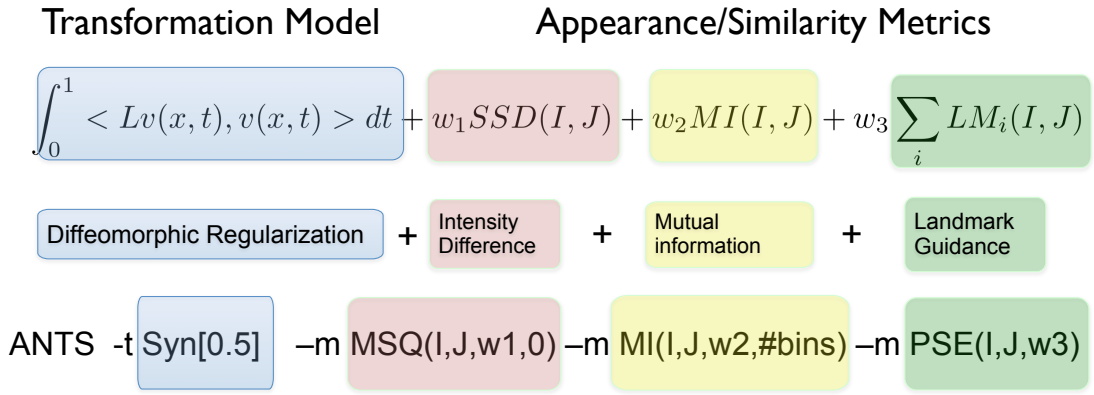


Figure 3: The relationship between the variational form that defines the optimization goals and the ANTs command line. The SyN option implements the method in [10] and evaluated in [26]. The newer version of the ants registration executable is called `antsRegistration` and uses a similar command line that is connected to the variational objective function.

DICOM, ITK and ANTs use LPS coordinate systems. In contrast, the nifti coordinate system is RPI. These coordinate system differences create much confusion especially when switching between software. Very few medical imaging software systems are internally consistent w.r.t. the treatment of world coordinates across file formats. However, ANTs and ITK tools will be consistent. ITK-SNAP will be mostly consistent with ANTs and ITK although it allows coordinates to be transformed and, in that case, all bets are off. As above, if you have a concern, please perform a straightforward experiment to test your concerns. Keep in mind that viewing images with software that does not conform to ITK world coordinates will make the images "look wrong" with respect to image orientation. This caveat may include NiPy, FSL, SPM etcetera.

2.2 ANTs transformation models

The ANTs toolkit provides a hierarchy of transformations with adjustable levels of complexity, regularization, degrees of freedom and behavior as optimizers. The simplest transformation model is the translation, followed by the rigid and/or affine transform. The most complex – and most flexible – is a symmetric diffeomorphic transformation based on optimizing and integrating a time-varying velocity field. Computation time also increases with transformation model complexity, but most normalization needs may be met with under an hour of computation. We provide an overview of some of the ANTs models below and try to communicate intuition on what one gains/loses with each choice. An overview of available models and similarity terms is in Table 1.

General Parameters: ANTs registration options include control of iterations (and, optionally, convergence criterion) via `-c 5000x5000x5000` which specifies that the affine registration uses a 3 level image pyramid with each level 5000 iterations at most. Multi-resolution options include `-s` for smoothing and `-f` for "shrink factors" i.e. downsampling rates (e.g. 8 means $1/8^{th}$ resolution). `MI[fixed,moving,1,32,Regular,0.1]` means to use Mutual Information as similarity metric with 32 bins and regularly spaced samples of 10% of the image; lower sampling rates increases speed and is useful in low-dimensional registration. `MI[fixed,moving,1,32]` means to use Mutual Information as similarity metric with 32 bins and dense

Category	Transformation, ϕ	Similarity Measures	Brief Description
Linear	Rigid†	MI, MeanSquares, GC	Rigid registration.
	Similarity†	MI, MeanSquares, GC	Rotation + uniform scaling.
	Affine†	MI, MeanSquares, GC	Affine registration.
Elastic	GaussianDisplacementField	CC, MI, MeanSquares, Demons	Demons-like algorithm.
	BSplineDisplacementField	CC, MI, MeanSquares, Demons	FFD variant.
Diffeo.	Exponential†	CC, MI, MeanSquares, Demons	min $v(\mathbf{x})$
	SyN†	CC, MI, MeanSquares, Demons	locally in time min $v(\mathbf{x},t)$
	BSplineSyN†	CC, MI, MeanSquares, Demons	locally in time min $v(\mathbf{x},t)$
	TimeVaryingVelocityField†	CC, MI, MeanSquares, Demons	min $v(\mathbf{x},t)$ over all time

Table 1: Transformations and a subset of the similarity metrics available in ANTs. Similarity metric acronyms: CC = neighborhood cross correlation (the preferred metric), MeanSquares = mean squared difference, MI = mutual information, PSE = point-set expectation (to be added soon-ish FIXME) [29]. ANTs also provides the inverse of those transformations denoted by the '†' symbol. The brief descriptions of the diffeomorphic algorithms contrast the way in which the velocity field is optimized and used to parameterize ϕ, the mapping. All ANTs Diff algorithms generate $\phi(\mathbf{x},t)$ over $t \in [0,1]$ through gradient descent.

sampling of the image and would be typically used in deformable registration. See `antsRegistration --help` for more information.

Affine and rigid registration—FIXME—this section and figures

The ANTs affine mapping syntax is shown in the affine mapping figure 4.

Affine mapping may become nontrivial when the region of interest occupies only a small portion of the image data. In such cases, ANTs enables the user to define a mask to focus the optimization effort on the region of interest. Example code for (affine) registration with a mask: http://stnava.github.io/BasicBrainMapping/. The difference between this command and a regular ANTs call is the `-x` option, which specifies the mask, defined in the template space. The mask option also affects the deformable optimization.

Rigid Registration: ANTs will also perform rigid registration. For example,

```
1 imgs=" -f r16slice.nii.gz -m r64slice.nii.gz "
2 antsRegistrationSyNQuick.sh -d 2 $imgs -o $op -t r
```

The jacobian of the resulting affine mapping should be unity. Compare the `antsRegistrationSyNQuick.sh` and `antsRegistrationSyN.sh` scripts to see how one might improve or modify registration parameters for different problems.

Deformable registration

Affine mapping is adequate when the difference between images involves only rotation, scaling or shearing. Other data requires more deformable mappings to capture shape differences and find a good alignment of image features. Figure 5 shows how deformable mapping may improve the correspondence of the deformed beetle to the ford image. Most ANTs-based applications use symmetric diffeomorphic normalization. However, ANTs also enables a simpler parameterization of a deformable mapping via a regularized vector space.

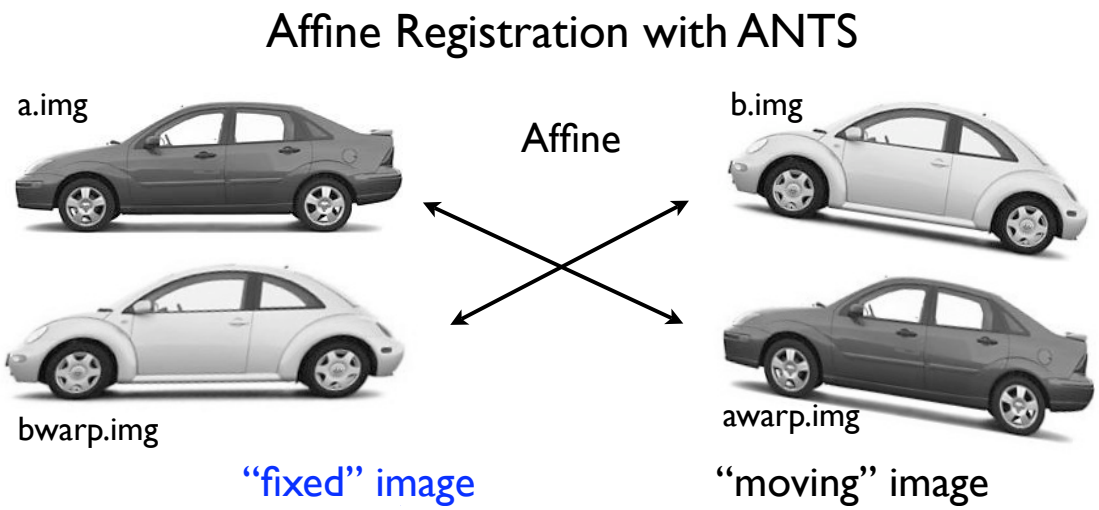


Figure 4: The anatomy of an ANTs optimization and application of the resulting warping.

Figure 5: This example shows the degree to which the beetle (b.img) may be deformed to the ford (a.img) under different transformation models. Left to right increases the degrees of freedom in the mapping and thus the registration accuracy.

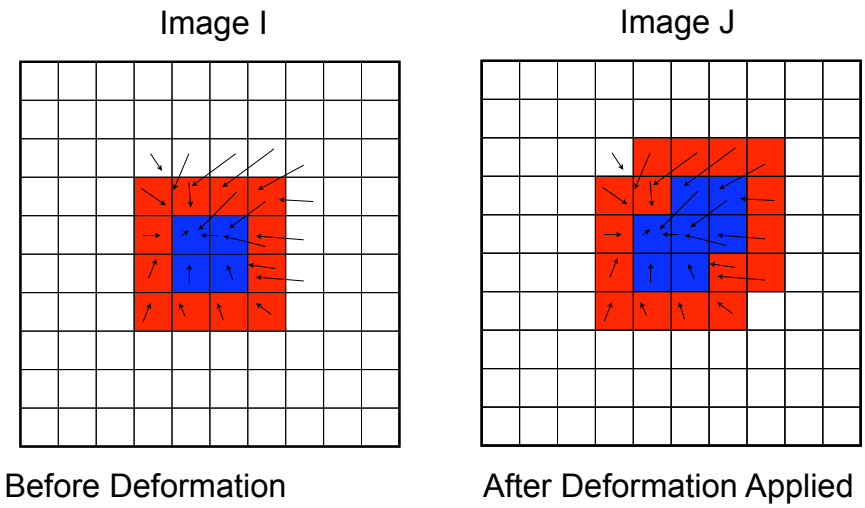


Figure 6: A deformation in a digital domain.

We term these types of transformations as "elastic". The original Demons algorithm provides a classic example of using a regularized vector space for nonlinear registration. Caveats are that a regularized vector space may not preserve the underlying topology and may also prove too inflexible to capture the shape changes one is after. Both of these shortcomings motivate the use of diffeomorphisms.

Elastic/Vector Space Transformations. If we assume no affine transformation, then an elastic transformation involves computing a mapping from image $I(\mathbf{x})$ to image $J(\mathbf{x})$ through a deformation field $\mathbf{u}(\mathbf{x})$. The deformation is defined in the physical space of the image and dictates the positional difference between corresponding features in the two images. Thus, if a feature defined at $I(\mathbf{x})$ matches a feature in J at position $\mathbf{y}$ then the deformation field at $\mathbf{x}$ should give $\mathbf{u}(\mathbf{x}) = \mathbf{y} - \mathbf{x}$. Such a deformation field may be applied to deform image J into image I by composing the mapping $J_{deformed}(\mathbf{x}) = J(\mathbf{x} + \mathbf{u}(\mathbf{x}))$. In a perfect world, then $I(\mathbf{x}) = J_{deformed}(\mathbf{x})$, though this is rarely the case. Figure 6 visualizes the deformation of an image under this standard model. Gradient descent optimization of an elastic mapping may be summarized (crudely) as:

$$\begin{aligned} \text{Compute the similarity gradient: } \nabla E &= \partial_{\mathbf{u}} \Pi(I, J(\mathbf{x} + \mathbf{u}(\mathbf{x}))). \\ \text{Update the deformation field: } \mathbf{u}(\mathbf{x}) &\leftarrow \mathbf{u}(\mathbf{x}) + \delta \nabla E. \\ \text{Regularize the deformation field: } \mathbf{u}(\mathbf{x}) &\leftarrow G_\sigma \star \mathbf{u}(\mathbf{x}), \end{aligned} \tag{1}$$

where Π is the similarity, δ is a gradient step length and G_σ is a gaussian smoother. This optimization is captured in the following ANTs command (FIXME)

```
dim=2
antsRegistration -d $dim -r [ r16slice.nii.gz , r64slice.nii.gz ,1] \
                      -m mattes[ r16slice.nii.gz , r64slice.nii.gz, 1 , 32, regular,
                          0.1 ] \
                      -t affine[ 0.1 ] \
                      -c [500x500x50,1.e-8,20] \
                      -s 4x2x1vox \
                      -f 3x2x1 -l 1 \
                      -m cc[ r16slice.nii.gz , r64slice.nii.gz, 1 , 4 ] \
                      -t GaussianDisplacementField[ .5, 3, 0.5 ] \
```

```
                       -c [ 50x50x50,0,5 ] \
                      -s 1x0.5x0vox \
                      -f 4x2x1 -l 1 -u 1 -z 1 \
                     -o [${op},${op}_diff.nii.gz,${op}_inv.nii.gz]
```

Here, we use the CC metric (a cross-correlation implementation) with window radius 4, weight 1 and gradient step length 1.5. The optimization will be performed over three resolutions with a maximum of 30 iterations at the coarsest level, 20 at the next coarsest and 10 at the full resolution. We use a Gaussian regularizer with a sigma of 3 that operates only on the deformation field and not on the similarity gradient, as 0 is passed as the first parameter to `GaussianDisplacementField` . One may see the correspondence, yet again, between the ANTs call and the optimization scheme. The optimization will stop when either the energy cannot be smoothly minimized or the maximum number of iterations is reached. BSpline regularization is also available in ANTs – see the DMFFD section below.

Warping and Invertibility. ANTs uses physical space to define mappings. The origin etc is in the coordinates of the world in which the picture (e.g. MRI) was taken. Then, warping between physical coordinates is relatively easy. Differences in bounding boxes etc present no problem – assuming you avoid inconsistent headers i.e. origins, directions, data orientation. One may use PrintHeader to check the data and run simple, fast tests (few iterations) to perform sanity checks before running through loads of data. An ANTs deformation consists of a standard naming prefix plus a standard naming extension. We usually assume nii but other file types may be used. The value of a voxel of a deformation/warp component image is the physical space displacement from that voxel. Note that an inverse warp – in the digital domain – is only approximate as shown in figure 7. A few important notes follow. (1) Deformation directionality: Warps/deformations applied to images occur in the opposite direction from warps/deformations applied to points. That is, if we map image B to A using

```
dim=2
antsRegistration -d $dim -r [ r16slice.nii.gz , r64slice.nii.gz ,1] \
                       -m mattes[ r16slice.nii.gz , r64slice.nii.gz, 1 , 32, regular,
                           0.1 ] \
                       -t affine[ 0.1 ] \
                       -c [500x500x50,1.e-8,20] \
                       -s 4x2x1vox \
                       -f 3x2x1 -l 1 \
                       -m cc[ r16slice.nii.gz , r64slice.nii.gz, 1 , 4 ] \
                       -t GaussianDisplacementField[ .5, 3, 0.5 ] \
                       -c [ 50x50x50,0,5 ] \
                      -s 1x0.5x0vox \
                      -f 4x2x1 -l 1 -u 1 -z 1 \
                     -o [${op},${op}_diff.nii.gz,${op}_inv.nii.gz]
```

then we get a deformation called `OUTPUT1Warp.nii.gz` and an affine transform called `OUTPUT0GenericAffine.mat` . This composite mapping – when applied to B – will transform B into the space of A. However, if we have points defined in B that we want to map to A, we have to use `OUTPUT1InverseWarp.nii.gz` and the inverse of `OUTPUT0GenericAffine.mat`. This is because the domain and range of the map's definition need to be dense in a way that is appropriate for the data type. Figure 7 illustrates the concept. (2) Older Image Formats: older image formats (e.g. Analyze) may not have proper origin/offset/direction support! In these cases, we recommend converting to nii and verifying that data overlays properly. (3) Transorm Composition: Composition of transforms may be

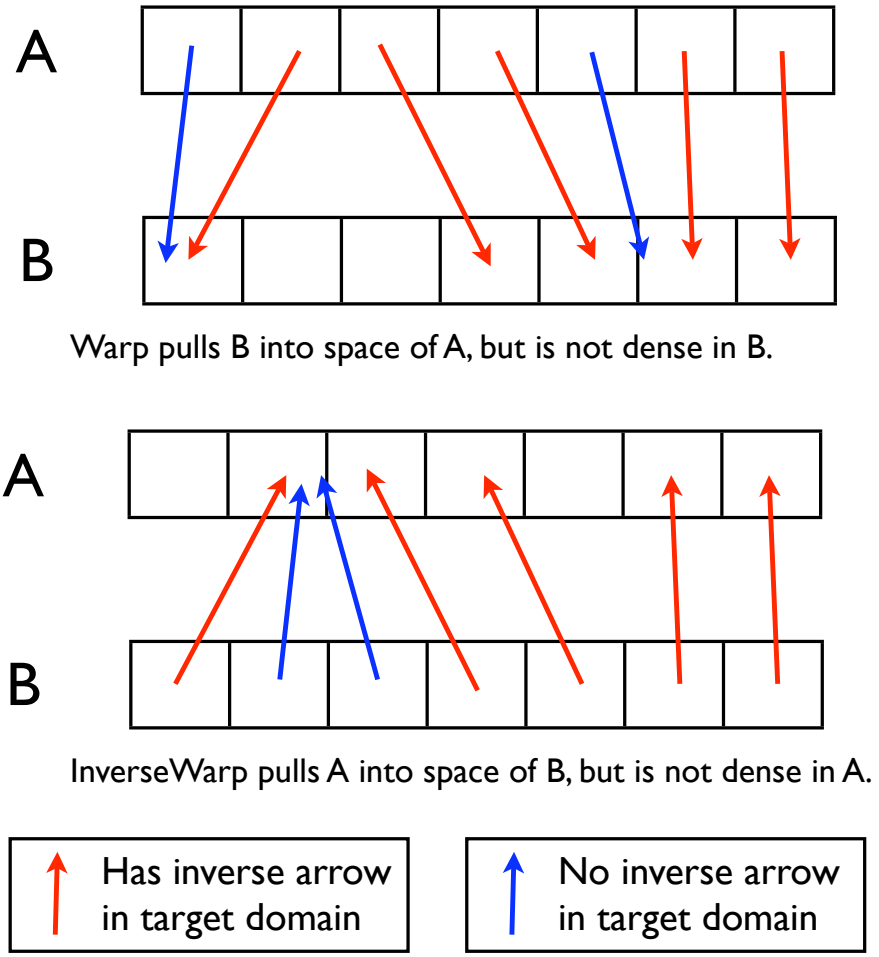


Figure 7: Digital invertibility presents some limitations. Here, we see that invertibility is not exact but is gained only by interpolation. Thus, in three-dimensional scenarios in particular, there are fundamental limits to the degree of invertibility that may be achieved. The second and third voxels – from left – in image A undergo an expansion by a factor of 2. That is, under the mapping, 2 voxels are mapped to 4. This gives the definition of the Jacobian – computed by ANTsJacobian – which is a unitless measure defined by the ratio of volumes. Thus, $\mathcal{J}(\mathbf{x}) = V(\phi(\mathbf{x}))/V(\mathbf{x})$ where V represents the volume operation and $\mathbf{x}$, here, may be a small object. Thus, if ϕ – the mapping – causes expansion, then $\mathcal{J}(\mathbf{x}) > 1$.

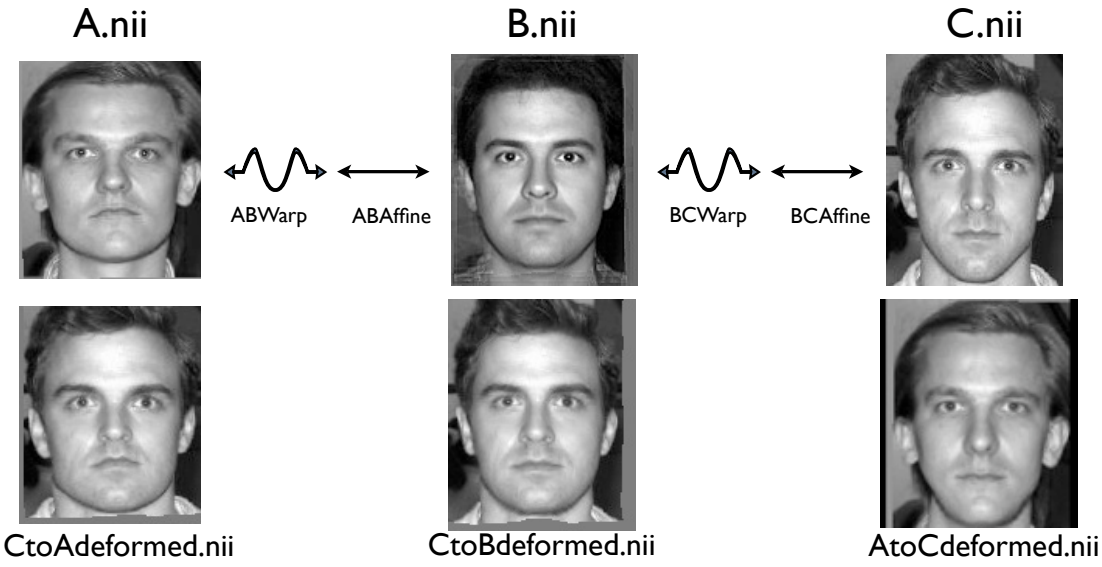


Figure 8: Here, we detail how one applies a deformation and associated inverses. We also see how antsApplyTransforms can be used to concatenate a series of transformations. In brief, read parameters left to right, then pass to `antsApplyTransforms`. To warp C to A (in short-hand): `antsApplyTransforms -t ABWarp -t ABAffine -t BCWarp -t BCAffine`. Use the inverse composite transform to warp A to C (in short-hand): `antsApplyTransforms -t [BCAffine,1] -t BCInverseWarp -t [ABAffine,1] -t ABInverseWarp` . Note, to focus on the key idea, we dropped the other parameters needed for a proper `antsApplyTransforms` call.

achieved with `antsApplyTransforms`. (4) Warping with inverses and concatenations – viable when using diffeomorphisms – are described in figure 8.

Diffeomorphic Transformations. The elastic mapping space – as indicated above – may prove inadequate for some large deformation mapping scenarios. Figures 5 illustrate the changing performance one may get in switching from affine to elastic to diffeomorphic normalization. Figure 9 shows how one may use ANTs to achieve a state-of-the-art diffeomorphic mapping. The ANTs diffeomorphic model chosen for this example – symmetric normalization [10, 25] – is invariant to the order of the input images (although the affine mapping is not). An additional advantage of the diffeomorphic model over the elastic model is that both forward and inverse transformations are computed thus allowing one to deform fixed to moving and also moving to fixed. The transformation model chosen here – `SyN[0.1,3,0.0]` – may be replaced with other diffeomorphic transformation models. The most general is global-in-time SyN via `SyN[0.1,3,0.0]` , where the time step for integration is 0.01 (lower is more accurate). A fast approximate solution may be gained through exponential mapping via `Exponential[0.1,3,0.5,10]` , where 10 integrations points are used. Update schemes for diffeomorphic optimization are similar to the elastic case and are described elsewhere [10].

2.3 ANTs similarity terms

Here is a script that will let you experiment with different similarity term and transformation model combinations. A few are listed, with parameters that are a reasonable starting point. This example implements an elastic mapping driven by neighborhood cross-correlation and initialized by mutual information affine:

```
1 dim=2
2 antsRegistration -d $dim -r [ r16slice.nii.gz , r64slice.nii.gz ,1] \
```

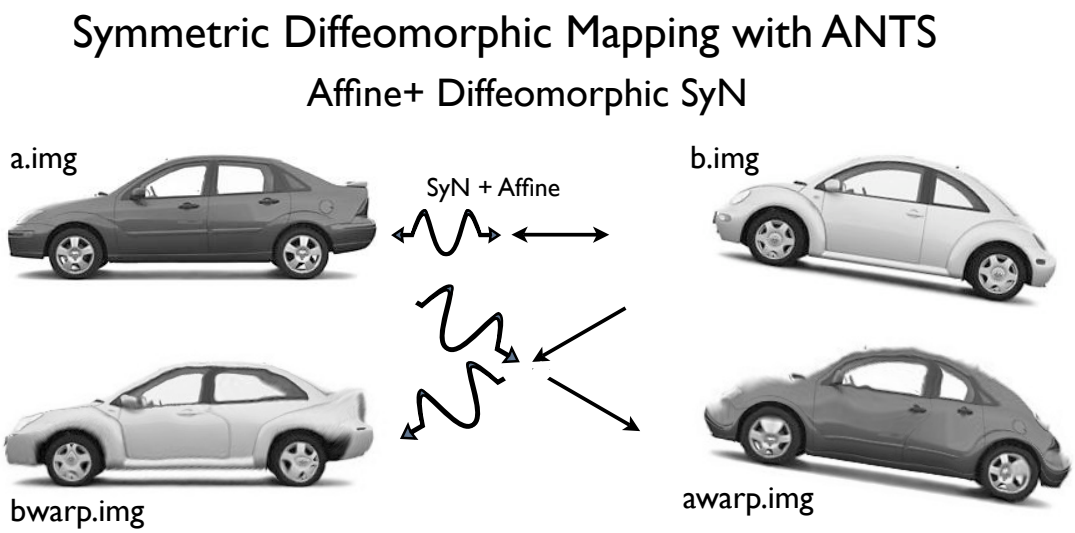


Figure 9: This example shows the benefit of the symmetric normalization model – invertibility, symmetry, highly deformable and accurate registration. This example may be recreated by the reader via: http://stnava.github.io/cars/.

```
                      -m mattes[ r16slice.nii.gz , r64slice.nii.gz, 1 , 32, regular,
                          0.1 ] \
                      -t affine[ 0.1 ] \
                      -c [500x500x50,1.e-8,20] \
                      -s 4x2x1vox \
                      -f 3x2x1 -l 1 \
                      -m cc[ r16slice.nii.gz , r64slice.nii.gz, 1 , 4 ] \
                      -t GaussianDisplacementField[ .5, 3, 0.5 ] \
                      -c [ 50x50x50,0,5 ] \
                     -s 1x0.5x0vox \
                     -f 4x2x1 -l 1 -u 1 -z 1 \
                    -o [${op},${op}_diff.nii.gz,${op}_inv.nii.gz]
```

This example implements a greedy syn mapping that includes elastic regularization and is driven by mutual information and initialized by global correlation with a similarity transform:

```
dim=2
antsRegistration -d $dim -r [ r16slice.nii.gz , r64slice.nii.gz ,1] \
                      -m gc[ r16slice.nii.gz , r64slice.nii.gz, 1 , 32, regular, 0.1
                          ] \
                      -t similarity[ 0.1 ] \
                      -c [500x500x50,1.e-8,20] \
                      -s 4x2x1vox \
                      -f 3x2x1 -l 1 \
                      -m mattes[ r16slice.nii.gz , r64slice.nii.gz, 1 , 48 ] \
                      -t syn[ .15, 3, 0.5 ] \
                      -c [ 50x50x50,0,5 ] \
                     -s 1x0.5x0vox \
                     -f 4x2x1 -l 1 -u 1 -z 1 \
                    -o [${op},${op}_diff.nii.gz,${op}_inv.nii.gz]
```

Using multiple metrics to drive registration. This example uses smoothing without downsampling (ie scale-space) and implements a greedy syn driven by two unequally weighted similarity metrics and no affine mapping:

```
dim=2
# registration is performed at a coarse 1/2 scale
antsRegistration -d $dim -r [ r16slice.nii.gz , r64slice.nii.gz ,1] \
                      -m mattes[ r16slice.nii.gz , r64slice.nii.gz, 1 , 32 ] \
                      -m cc[ r16slice.nii.gz , r64slice.nii.gz, 0.5 , 2 ] \
                      -t SyN[ .15, 3, 0 ] \
                      -c [ 50x50x50,0,5 ] \
                     -s 4x2x0vox \
                     -f 2x2x2 -l 1 -u 1 -z 1 \
                    -o [${op},${op}_diff.nii.gz,${op}_inv.nii.gz]
```

Convergence occurs under two conditions: (1) either the maximum number of iterations are reached at a given optimization level or (2) the slope of the change in the optimization objective is negative or very small.

2.4 Choosing a metric

ANTS supports both volumetric registration and point set registration. The image / point set similarity metrics in ANTS are unified in the form of a function on the images or the point sets:

$$\textbf{Similarity}[\textbf{fixedImage}, \textbf{movingImage}, \textbf{weight}, \textbf{samplingStrategy}, \textbf{parameters}].$$

The similarity type for the transformation is specified by `-m` option, which contains two parts: simarity type and the parameters inside the brackets. The possible similarity metrics for volumetric images are:

- Cross correlation estimate: `-m CC[fixedImage,movingImage,weight,radius]`. This metric works well for intra-modality image registration. For exxample, `-m CC[fixed.nii,moving.nii,1,5]` specifies:
  - the fixed image: fixed.nii
  - the moving image: moving.nii
  - weight for this metric is 1 (i.e. only this metric drives the registration).
  - the region radius for computing cross correlation is 5
- Mutual information:
  `-m MI[fixedImage,movingImage,weight,number-of-histogram-bins, samplingStrategy,samplingPercentage]`
  This metric works both well for intra-modality and inter-modality image registration. For example, the first three parameters in `-m MI[fixed.nii,moving.nii,1,32]` similar to the example above in cross correlation, except that the last parameter means that the number of bins in computing mutual information is 32. The `Mattes` metric is currently synonymous with `MI`.

- Global correlation:
  `-m GC[fixedImage,movingImage,weight,0, samplingStrategy,samplingPercentage]`
  This metric works both well for intra-modality and inter-modality image registration. For example,
  `-m GC[fixed.nii,moving.nii,1,0,Random,0.1]`
  uses 10% random sampling of the image to estimate the global correlation.

- Mean square difference:
  `-m MeanSquares[fixedImage,movingImage,weight,0]`
  This metric works for intra-modality image registration. The last parameter `0` is a padding value of no real meaning. For example, `-m MeanSquares[fixed.nii,moving.nii,1,0]`.

ANTs also support registration of point sets FIXME—only old ANTS so far ... will be added to `antsRegistration` in the near future. The supported formats for point sets can be found in I/O section. The similarity metrics for point sets are:

- Point set expectation:

```
-m PSE [fixedImage,movingImage,fixedPoints,movingPoints,weight,
pointSetPercentage,pointSetSigma,boundaryPointsOnly,
kNeighborhood,PartialMatchingIterations=100000]
```

  - `fixedImage`: defines the space domain of the fixed point set.
  - `movingImage`: defines the space domain of the moving point set.
  - `fixedPoints/Image`: defines the coordinates of the fixed point set or label image. It can be an image with discrete positive labels, a VTK format point set file, or a text file. Details can be found in I/O section (TODO).
  - `movingPoints/Image`: defines the coordinates of the moving point set or label image.
  - `weight`: weight for this metric. `1` means that only this metric drives the registration.
  - `pointSetPercentage`: the percentage of points to be randomly sampled used in the registration.
  - `pointSetSigma`: the standard deviation of the Parzen window used to estimate the expectation.
  - `boundaryPointsOnly`: `1` (or "true") means only the boundary points in the label image is used to drive registration.
  - `kNeighborhood` is a positive discrete number. The first k neighbors are used to compute the deformation during the registration.
  - `PartialMatchingIterations` controls the symmtry in the matching. This option assumes the complete labeling is in the first set of label parameters ... more iterations leads to more symmetry in the matching - 0 iterations means full asymmetry

- Jensen-Tsallis BSpline

```
-m JTB[fixedImage,movingImage,fixedPoints,movingPoints,weight,
pointSetPercentage,pointSetSigma,boundaryPointsOnly,kNeighborhood,alpha,
meshResolution,splineOrder,numberOfLevels,useAnisotropicCovariances]
```

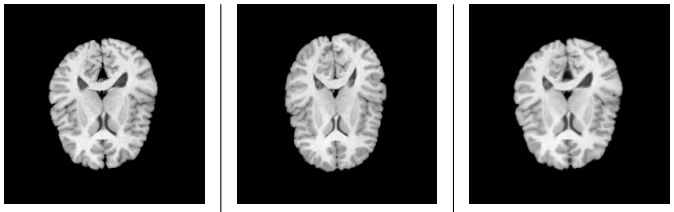


Figure 10: registration using mean square intensity difference—note: brains are upside-down, sorry!

- `fixedImage`: defines the space domain of the fixed point set.
- `movingImage`: defines the space domain of the moving point set.
- `fixedPoints`: defines the coordinates of the fixed point set. It can be an image with discrete positive labels, a VTK format point set file, or a text file. Details can be found in I/O section (TODO).
- `movingPoints`: defines the coordinates of the moving point set.
- `weight`: weight for this metric. `1` means that only this metric drives the registration.
- `pointSetPercentage`: the percentage of points to be randomly sampled used in the registration.
- `pointSetSigma`: the sigma for the Parzen window used to estimate probabilities.
- `boundaryPointsOnly`: [TODO] `1` (or "true") means only the boundary points in the point sets are used to drive registration.
- `kNeighborhood` is a positive discrete number. The first k neighbors are used to compute the deformation during the registration.
- `alpha`
- `meshResolution`
- `splineOrder`
- `numberOfLevels`
- `useAnisotropicCovariances`

Fig. 2.4 shows the registration result using intensity difference. Here is also an example script to register a pair of images using mean square intensity difference and computing the metrics of the registration image.

```
1 # use intensity difference with radius 0 -- radius no effect on intensity
    difference
2 antsRegistration -d 2 -m MeanSquares[r16slice.nii.gz,r64slice.nii.gz,1,0] -t \
3   GaussianDisplacementField[0.5,3,0.5] -c 50x50x30 -f 4x2x1 -s 0x0x0 -o ${outprefix}
      test.nii.gz
4 antsApplyTransforms -d 2 -i r64slice.nii.gz -o ${outprefix}resMSQ.nii.gz -t ${
    outprefix}test.nii.gz0Warp.nii.gz -r r16slice.nii.gz
5 MeasureImageSimilarity 2 0 r16slice.nii.gz ${outprefix}resMSQ.nii.gz ${outprefix}
    metricexamplelog.txt
6 MeasureImageSimilarity 2 1 r16slice.nii.gz ${outprefix}resMSQ.nii.gz ${outprefix}
    metricexamplelog.txt
7 MeasureImageSimilarity 2 2 r16slice.nii.gz ${outprefix}resMSQ.nii.gz ${outprefix}
    metricexamplelog.txt
8 ConvertToJpg ${outprefix}resMSQ.nii.gz ${outprefix}resMSQ.jpg
```

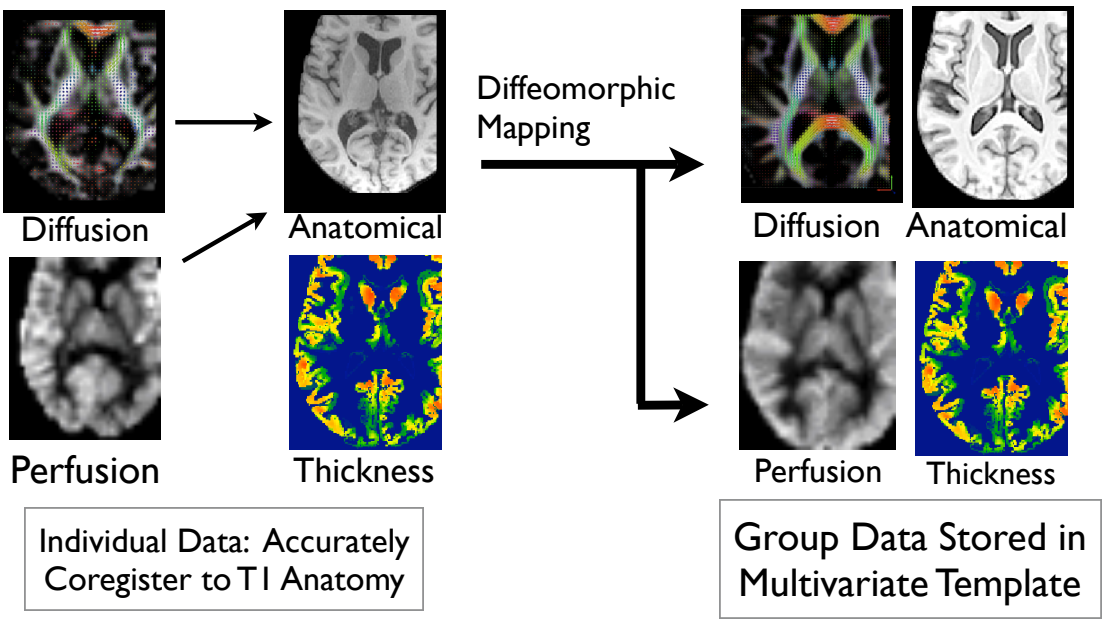


Figure 12: The current version of ANTs may perform normalization of multiple modalities by combining *intrasubject, intermodality* mappings with *intersubject, intramodality* maps to a group template. In brain imaging, the intersubject maps are usually guided by the T1 component.

2.5 Notes on basic brain mapping

There is "maximally robust" brain mapping and there is "fast" brain mapping. These two goals may, at times, be adversarial. A simple and "fast" brain registration example is here http://stnava.github.io/BasicBrainMapping/. This example shows a non-optimized use of `antsRegistration` and compares it to the case where the registration is *masked* in order to focus the registration. Such strategies may be used for brains with missing data e.g. lesions.

2.6 Normalization across different modalities: E.g. DTI

The `ImageMath` program – via TensorFA and TensorMeanDiffusion – may derive scalar images from DTI. Such images may be used to distortion correct DTI to T1 or to map DTI together. See the ANTS/Scripts program called `antsIntermodalityIntrasubject.sh` for an example of how one might implement the strategy outlined in [36]. Figure 11 shows what might happen if your tensor entries are stored in the wrong order. ANTs expects the nifti standard ordering of the DT six vector.

2.7 Multivariate normalization with ANTs

Multivariate normalization may be performed in two ways with ANTs. First, as shown in figure 12, intra-subject mappings may be used to conform all modalities into a common subject space defined by the "anchor" modality. In brain imaging, this is usually the T1 component.

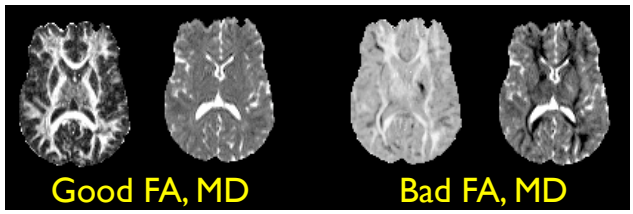


Figure 11: The FA and mean diffusion of the tensor may be corrupted, as at right, if the order of the DT entries is wrong.

A second type of multivariate normalization is able to use the T1 and DTI components directly in the optimization of the mapping. We usually assume that one has resampled data to the space of T1. In this case, dense features within white matter are gained by using the DT component in addition to the T1 component. One might also weight the T1 CC metric slightly more than the FA CC metric, 1.25 vs. 1.0. *The convergence criterion for the multivariate scenario is a slave to the* **last** *metric you pass on the ANTs command line*.

2.8 Notes on large deformation mapping

Figure 13 defines what one might expect from a high-resolution, large deformation, successful normalization. Major and many minor features are well aligned between these images to an extent that is approaching the limits allowable by biological variation.

Turning "Failures" into Successes: Below are some pointers to follow if you are unable to recreate such normalization quality. Usually, reasons for registration failure are one of a very few things:

1. The initialization is so off mark that affine registration fails – thus meaning all subsequent deformable registration will not be meaningful.
2. The information within headers is inconsistent e.g. origins are defined in different ways, the "directions" are not correct or some combination of these. The PrintHeader executable can help one in debugging this type of problem.
3. The similarity or transformation model is inappropriate or has too small a capture region for the problem.

Large deformation mapping is challenging not only because of the amount of deformation, but also because the "true" solution becomes more difficult to find as the distance between images increases. Keeping this in mind, fixing registration failures means 1. better initialization or 2. increasing the capture range of the registration. Key changes may include more multi-resolution levels, increasing the radius of the neighborhood correlation, more dense or more focused sampling and/or allowing more iterations during the optimization. Note: the coarsest levels of the computation take only about 10 percent of the time (a few minutes depending on the machine) but account for the large majority of the shape variation. Therefore, inspecting results after only low resolution registration is performed can save significant time and make parameter optimization more rapid.

2.9 Optimal template construction with ANTs

A useful script for setting up optimal template construction is available in ANTs/Scripts. This method is described in [6, 16, 25, 45, 9], where more recent publications are more descriptive and incorporate more recent features and optimization strategies. Two versions of this algorithm are available : serial and parallel. Data and a script for building an average face template is available at http://ntustison.github.io/TemplateBuildingExample/. Two key points about optimal templates: 1. The outcome stabilizes at around ten images, for most populations. That is, if we randomly select images from individuals within the same demographic distribution, we will end up with very similar average images. 2. Optimality, for the ANTs-SyN approach, is defined by the minimum shape and appearance distance image.

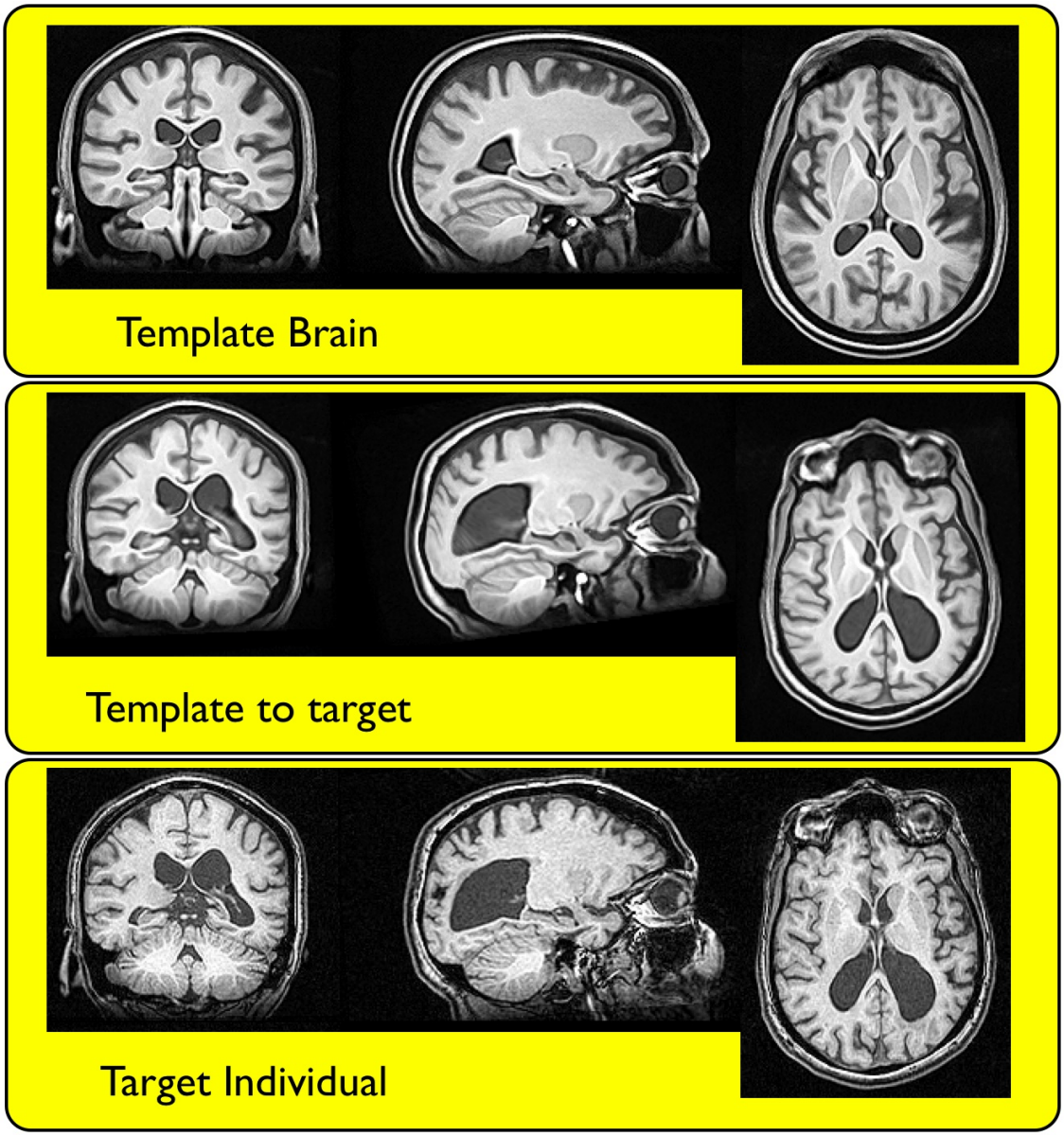


Figure 13: ANTs succeeds in a challenging normalization scenario. Comparing the template mapping to the individual shows that much of the cortex is well-aligned, as is the hippocampus, despite the relatively large difference between the initial template and the target. Here, one might note a limitation of whole brain mapping: occipital lobe sulcal variation is highly idiosyncratic and extremely difficult to capture, in particular when there is such severe atrophy nearby.

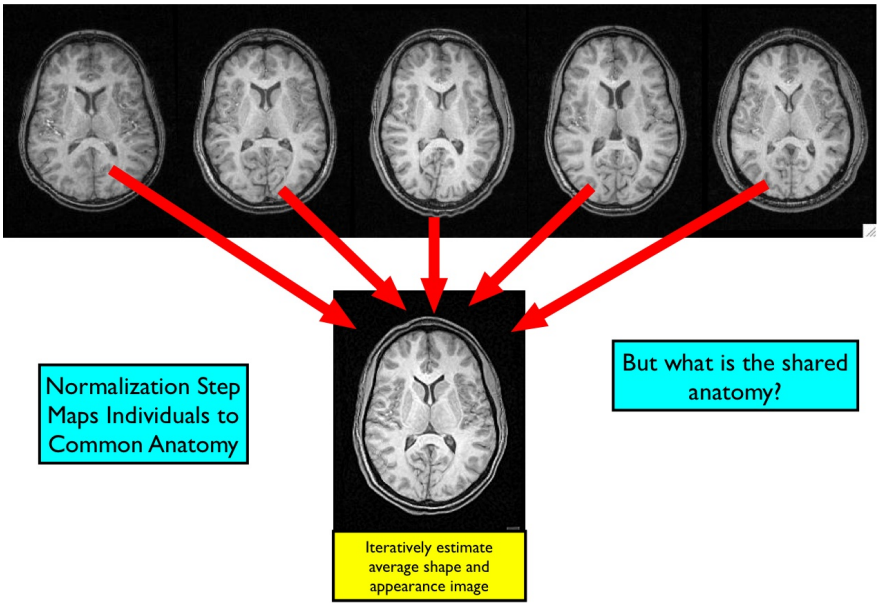


Figure 14: This example may be recreated by the user via http://ntustison.github.io/TemplateBuildingExample/. The shared anatomy – across this dataset – is recovered in the derived optimal template.

See the template page for some examples of this and users may contact ANTs developers for previously derived template images that may be useful.

The goal of template construction is to derive a "most representative" single image from a population. The steps are:

1. Make a directory for your data. Copy or link all the images into it.
2. On the command line, within that directory, run the following command: `bash antsMultivariateTemplateConstruction2.sh` to get usage.
3. Follow the example outlined in the TemplateBuildingExample above but modified for your own data and needs.
4. The script should exit and store a log of what happened.
5. If there is a problem, check the path to the programs, the fact that all the images you want to use are being listed and that the template output name is ok.

The script is fairly easy to alter so that you can send the whole thing to distributed computing. Typically, one would use voxbo or the qsub program available in the Sun Grid Engine. This is what is done in `antsMultivariateTemplateConstruction2.sh`. Figure 14 shows the data and a result similar to what may be recreated by the user.

2.10 2D to 3D registration

There has been much discussion of this on the ITK and ANTs lists. What it boils down to is this: the 2D to 3D (or vice versa) problem is inherently asymmetric and (very) ill-posed. So, we currently recommend, as

a start, running `antsRegistrationSyNQuick.sh` in both directions and seeing which works better. Both directions means pass the 2D image to the `-f` option and compare to the result of passing the 2D image to the `-m` option, noting that the resulting transforms are NOT inverses of each other.

2.11 More ANTs examples

The paper http://journal.frontiersin.org/Journal/10.3389/fninf.2014.00044/abstract shows or links to several more examples. Some of these include morphometry. Many other examples are available in the literature Google Scholar Search.

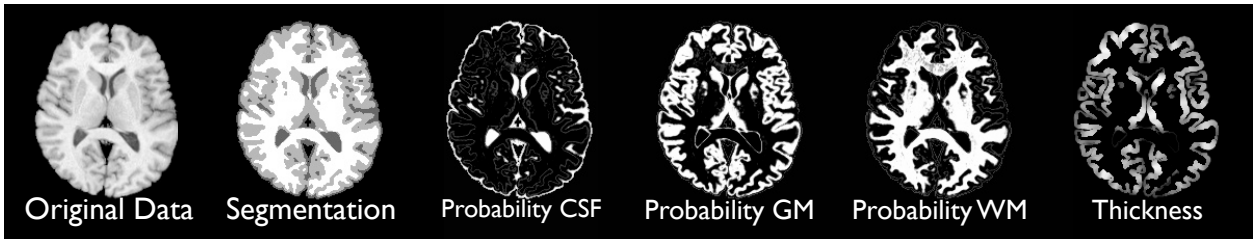


Figure 15: ANTs provides basic Markov Random Field regularized Gaussian model tissue segmentation. The ANTs program N4 or N3BiasFieldCorrection is very valuable as preprocessing for this naive approach to segmentation. The last panel – far right – shows the thickness derived from the white matter and gray matter probabilities where a prior on thickness was used to prevent thickness overestimation. Thickness was derived with DiReCT [20] which is called `KellyKapowski` in its implementation.

3 Image segmentation and labeling

ANTs has tools for both tissue based segmentation and prior-based segmentation that leverages spatial priors, usually based on a template mapping.

3.1 Basic segmentation

A suite of segmentation algorithm approaches are available by combining the `Atropos` tool with various feature images, priors and objective functions. But let's show a simple example first. We apply k-means segmentation to 2D data (the call is the same in 3D except for the dimensionality flag).

```
dim=2
img=r16slice.nii.gz
ImageMath $dim ${op}mask.nii.gz Normalize $img
ThresholdImage $dim ${op}mask.nii.gz ${op}mask.nii.gz 0.1 1
Atropos -d $dim -x ${op}mask.nii.gz -c [3,0] -m [0.1,1x1] -i kmeans[4] \
  -o ${op}_seg.nii.gz -a $img
```

The parameter `4` indicates 4 tissues plus background are sought. The `mask` identifies the region that will be segmented. There are no prior images available to guide the segmentation. Representative output is shown in figure 15 and – with priors – in figure 16.

3.2 Prior and template-based image segmentation

The same algorithm may be augmented to perform Prior-based segmentation. The parameter `4 indicates 3 tissues plus background are sought`. The `0.5` indicates we weight the priors equally as the data term and use a spatially varying set of Gaussians to estimate the segmentation. In this way, data from the priors may be used to modify and guide the segmentation in a locally varying way, accomodating for both inhomogeneity and the different imaging signature that different tissues provide. Prior images should be of the same size and dimension as the input data and should have intensities in the range $[0, 1]$.

```
dim=2
n=3
img=r16slice.nii.gz
```

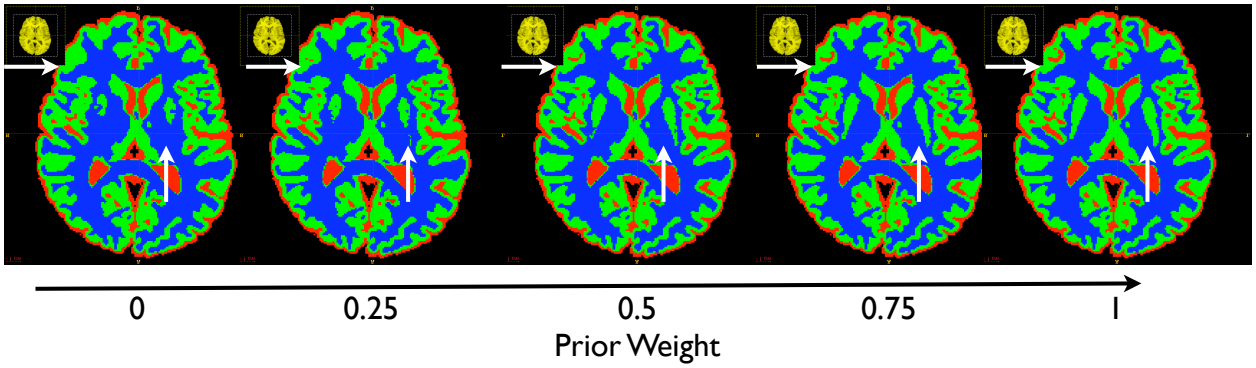


Figure 16: ANTs extends basic Markov Random Field regularized Gaussian model tissue segmentation to include priors, which allow spatially varying tissue models to determine segmentation. The ANTs program N3BiasFieldCorrection is less critical as preprocessing when using this approach to segmentation. Here, we see the improvement in segmentation as the locally varying prior models are weighted more heavily. The last panel – far right – shows the segmentation derived from data driven with an initialization founded purely on prior models based on template mapping. The prior model brings out the caudate and some CSF – highlighted by arrows – as their weight increases.

```
ImageMath $dim ${op}mask.nii.gz Normalize $img
ThresholdImage $dim ${op}mask.nii.gz ${op}mask.nii.gz 0.1 1
outvar=[${op}_segk.nii.gz,${op}_prob%0d.nii.gz]
Atropos -d $dim -x ${op}mask.nii.gz -c [3,0] -m [0.1,1x1] -i kmeans[$n] -o $outvar -
    a $img
# now prior based with 2 input features
lap=${op}lap.nii.gz
ImageMath $dim $lap Laplacian $img
inputfeatures=" -a $lap -a $img "
outvar=[${op}_seg.nii.gz,${op}_prob%0d.nii.gz]
Atropos -d $dim -x ${op}mask.nii.gz -c [3,0] -m [0.1,1x1] \
 -i priorprobabilityimages[$n,${op}_prob%0d.nii.gz,0.25] \
 -k HistogramParzenWindows -o $outvar $inputfeatures
```

This listing also shows how Atropos becomes instantaneously multivariate with more input feature images and that the likelihood model `-k` is also flexible. More detailed examples are here: https://github.com/ntustison/antsAtroposN4Example with brain extraction here: https://github.com/ntustison/antsBrainExtractionExample.

3.3 Cortical thickness

Two forms of cortical thickness estimation from probability maps are available in ANTs: first, the traditional Laplacian cortical thickness estimation and, second, the more recently developed Diffeomorphic Registration-based Cortical Thickness (DiReCT) [20]. Both methods estimate thickness of an object from probability or binary images that define object boundaries. This tool is mainly of interest in brain mapping and cardiac imaging for morphometric studies. Cortical thickness, for instance, is known to correlate with language development and IQ in adolescents. The laplacian approach to cortical thickness may be used via the program LaplacianThickness, which implements the method described in

"Three-dimensional mapping of cortical thickness using Laplace's Equation" by Jones, et al []. Super-sampling and controlling segmentation accuracy for input to this program is up to the user. Otherwise, the Laplacian method may grossly overestimate thickness in closed sulci. See the full example: https://github.com/ntustison/antsCorticalThicknessExample.

4 Data visualization with ANTs

Data visualization is important for producing figures for manuscripts, qualitative inspection of results, facilitating collaborations, and gaining insight into data and data transformations. ANTs provides three flexible programs to help with such tasks which we describe below.

4.1 Creating faux-colormapped images with `ConvertScalarImageToRGB`

For layering image data, it is often useful to map the grayscale image intensity values to distinct colormaps. We introduced such a processing framework into ITK described in [34]. The ANTs program `ConvertScalarImageToRGB` interfaces this framework and permits conversion of grayscale intensity scalar images to RGB colormapped images which can be viewed in programs such as ITK-SNAP. Converting scalar images to RGB intensities is also a preprocessing step for the next two programs described: `CreateTiledMosaic` and `antsSurf`. In addition to the built-in colormaps which are currently part of ITK, we also have several custom colormaps located in the directory `${ANTSPATH}/Examples/CustomColormaps/`. Additionally, these custom colormaps can be used as examples to build one's own set of colormaps for use with `ConvertScalarImageToRGB`. In Listing 1 we give three code examples of converting grayscale intensity images to different colormaps with and without masks.

```
# hot colormap
ConvertScalarImageToRGB 2 r16slice.nii.gz r16slice_hot.nii.gz none hot

# jet colormap (with a mask)
ThresholdImage 2 r16slice.nii.gz r16mask.nii.gz 0 0 0 1
ConvertScalarImageToRGB 2 r16slice.nii.gz r16slice_jet.nii.gz r16mask.nii.gz jet

# custom colormap (to show ITK-SNAP segmentation image)
ThresholdImage 2 r16slice.nii.gz r16mask.nii.gz 0 0 0 1
Atropos -d 2 -a r16slice.nii.gz -x r16mask.nii.gz -c [5,0] -i kmeans[3] -o r16seg.nii.gz
ConvertScalarImageToRGB 2 r16seg.nii.gz r16seg_snap.nii.gz \
  none custom ${ANTSPATH}/Examples/CustomColormaps/itkSnap.txt 0 6 0 255
```

Listing 1: Sample command calls to `ConvertScalarImageToRGB` to produce different colormaps

4.2 Figure production and large-scale data inspection using `CreateTiledMosaic`

The program `CreateTiledMosaic` in conjunction with `ConvertScalarImageToRGB` provides useful functionality for common image analysis tasks. The basic usage of `CreateTiledMosaic` is to tile a 3-D image volume slice-wise into a 2-D image. The help menu `CreateTiledMosaic --help` provides more in-depth coverage of options but some of the functionality includes:

- padding or cropping each tile element,
- alpha channel for translucent overlaid rob maps,
- flipping or permuting of each tile element,
- comprehensive slice selection, and
- tiling pattern (i.e. custom number of rows and/or columns).

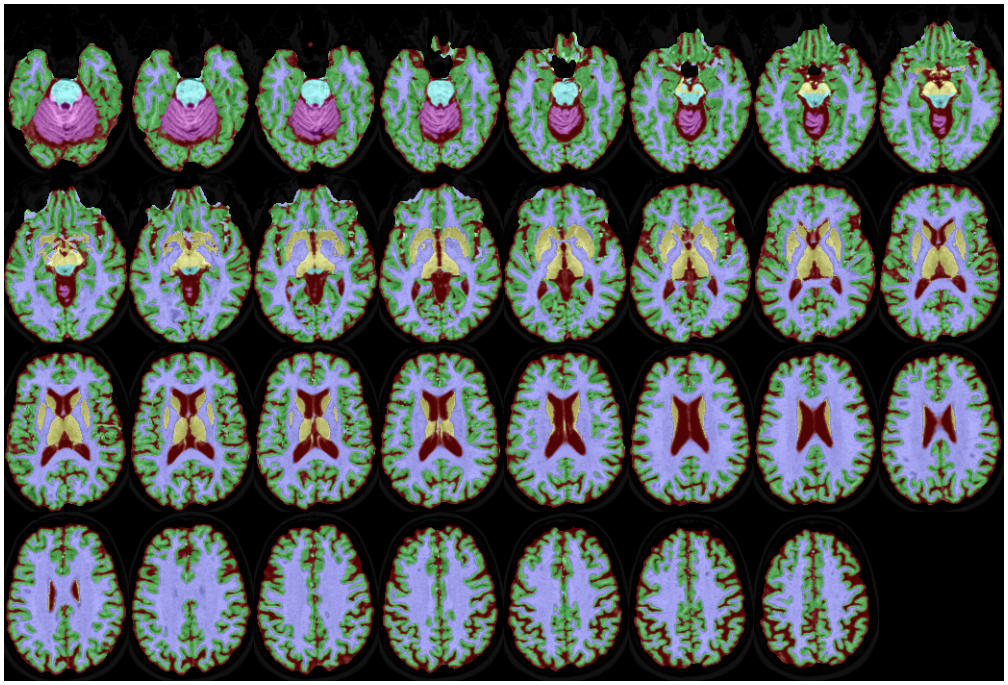


Figure 17: Tiled mosaic showing every other axial slice of a single OASIS data subject with segmentation labels overlayed ($\alpha = 0.3$).

As a use case, suppose one wants to quickly scroll through cortical thickness results from a processed cohort. CreateTiledMosaic can be used to convert each processed subject into a 2-D .png image which can be easily scanned for processing failures. For example, in Listing 2 we provide example code to process a subject OAS1_0457_MR1_mpr_n3_anon_sbj_111 from the OASIS data set[2] after it has been processed by the antsCorticalThickness.sh script.

```
##############################################################################
# Do segmentation mosaic
##############################################################################

itksnap=${ANTSPATH}/Examples/CustomColormaps/itkSnap.txt

# Convert segmentation labels to itk-snap label color scheme.  We set the min and max input
# to [0,6] since we have only included the colors for labels 1-6 in the colormap.  If we
# want to include more labels, we would need to modify the custom colormap.
ConvertScalarImageToRGB 3 OAS1_0457_MR1_mpr_n3_anon_sbj_111BrainSegmentation.nii.gz \
  OAS1_0457_MR1_mpr_n3_anon_sbj_111BrainSegmentation_snap.nii.gz none custom $itkSnap 0 6 0 255

# Create tiled segmentation mosaic.  We overlay "snap" segmentation map
CreateTiledMosaic -i OAS1_0457_MR1_mpr_n3_anon_sbj_111BrainSegmentation0N4.nii.gz \
  -r OAS1_0457_MR1_mpr_n3_anon_sbj_111BrainSegmentation_snap.nii.gz \
  -o OAS1_0457_MR1_mpr_n3_anon_sbj_111_tiledMosaic.png \
  -a 0.3 -t -1x8 -d 2 -p [-15x-50,-15x-30,0] -s [2,100,160]

##############################################################################
# Do thickness mosaic
##############################################################################

# Convert thickness map to hot color map.  We scale thickness values
#  between [0,8] to the standard RGB range of [0,255]
ConvertScalarImageToRGB 3 OAS1_0457_MR1_mpr_n3_anon_sbj_111CorticalThickness.nii.gz \
```

[2] http://www.oasis-brains.org

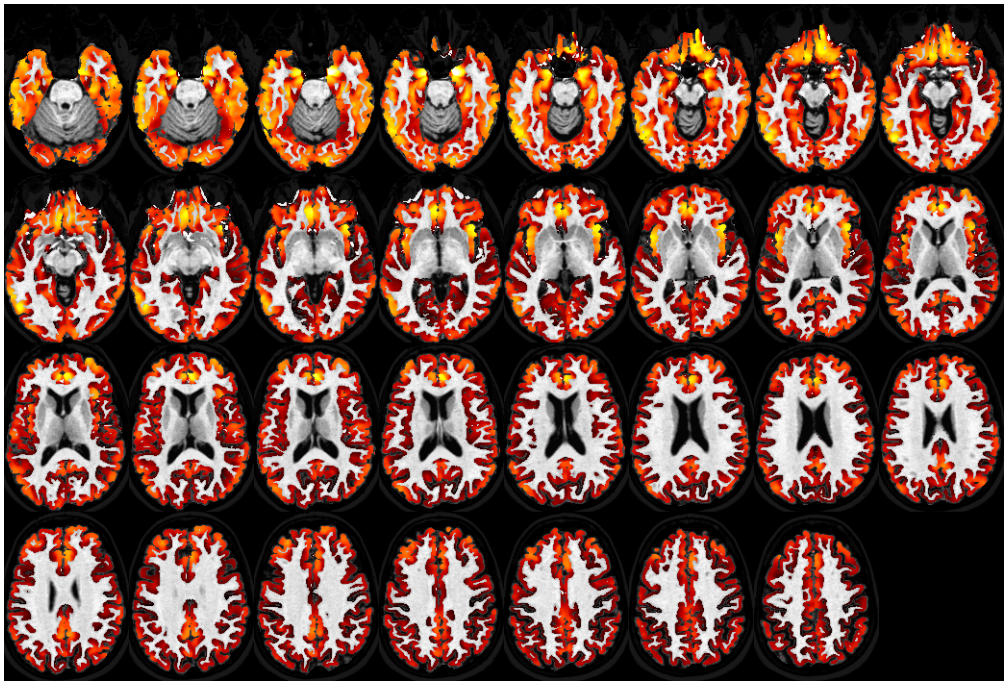


Figure 18: Tiled mosaic showing every other axial slice of a single OASIS data subject with thickness values overlayed in a "hot" colormap.

```
  OAS1_0457_MR1_mpr_n3_anon_sbj_111CorticalThickness_hot.nii.gz none hot none 0 8 0 255

# Create a mask for next call
ThresholdImage 3 OAS1_0457_MR1_mpr_n3_anon_sbj_111CorticalThickness.nii.gz \
  OAS1_0457_MR1_mpr_n3_anon_sbj_111CorticalThickness_mask.nii.gz 0 0 0 1

# Create tiled mosaic.  We overlay "hot" thickness map (but only the values within
# the mask on the bias corrected structural MR image
CreateTiledMosaic -i OAS1_0457_MR1_mpr_n3_anon_sbj_111BrainSegmentation0N4.nii.gz \
  -r OAS1_0457_MR1_mpr_n3_anon_sbj_111CorticalThickness_hot.nii.gz \
  -x OAS1_0457_MR1_mpr_n3_anon_sbj_111CorticalThickness_mask.nii.gz \
  -o OAS1_0457_MR1_mpr_n3_anon_sbj_111_tiledMosaic.png \
  -a 1.0 -t -1x8 -d 2 -p [-15x-50,-15x-30,0] -s [2,100,160]
```

Listing 2: Using `CreateTiledMosaic` to visualize axial slices for the OASIS data set subject `OAS1_0457_MR1_mpr_n3_anon_sbj_111` processed through the ANTs cortical thickness pipeline.

4.3 Volumetric visualizations with `antsSurf`

Our third program, `antsSurf`, is used to produce volumetric surface renderings from binary images with optional functional overlays. Again, the `antsSurf` help menu provides a more comprehensive description of functionality but some available options include:

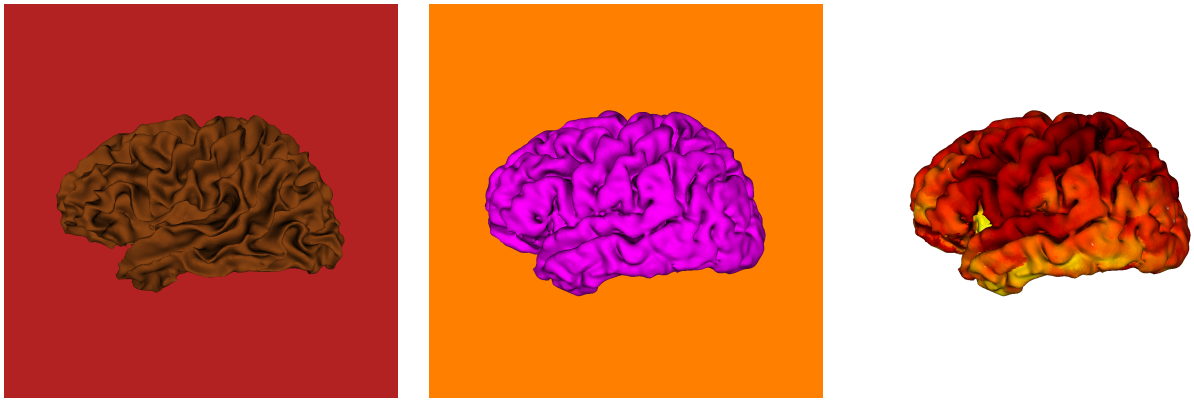
- specification of more than one functional overlay,
- painting of functional overlays based on order on the command line,
- localization of the functional overlay within a specified masked region,

- manual reorientation,
- manual adjustment of solid background and surface colors, and
- simple estimation of binary image given a mesh.

Reverting back to our OASIS cortical thickness example, we render three surfaces using antsSurf in Figure 19 which demonstrate some of the main capabilities of antsSurf and how one would incorporate other ANTs tools. The code to produce Figure 19 is located in Listing 3.

```
#############################################################################
# Do a simple volume rendering of the gray and white matters
#############################################################################

# Create gray matter binary image
ThresholdImage 3 OAS1_0457_MR1_mpr_n3_anon_sbj_111BrainSegmentation.nii.gz \
  OAS1_0457_MR1_mpr_n3_anon_sbj_111BrainSegmentationGm.nii.gz 2 2 1 0

# Volumetrically render the gray matter using a terrible color choice and reorient
antsSurf -s [OAS1_0457_MR1_mpr_n3_anon_sbj_111BrainSegmentationGm.nii.gz,255x0x255] \
  -d OAS1_0457_MR1_mpr_n3_anon_sbj_111GrayMatter.png[90x180x90,255x128x0]

# Create white matter binary image
ThresholdImage 3 OAS1_0457_MR1_mpr_n3_anon_sbj_111BrainSegmentation.nii.gz \
  OAS1_0457_MR1_mpr_n3_anon_sbj_111BrainSegmentationWm.nii.gz 3 3 1 0

# Volumetrically render the white matter using a terrible color choice and reorient
antsSurf -s [OAS1_0457_MR1_mpr_n3_anon_sbj_111BrainSegmentationWm.nii.gz,139x69x19] \
  -d  OAS1_0457_MR1_mpr_n3_anon_sbj_111WhiteMatter.png[90x180x90,178x34x34]

#############################################################################
# Render the cortical thickness map on the gray matter surface
#############################################################################

# dilate the cortical thickness values to ensure values cover the entire GM surface
ImageMath 3 OAS1_0457_MR1_mpr_n3_anon_sbj_111CorticalThicknessDilated.nii.gz GD \
  OAS1_0457_MR1_mpr_n3_anon_sbj_111CorticalThickness.nii.gz 1

# create a corresponding mask
ThresholdImage 3 OAS1_0457_MR1_mpr_n3_anon_sbj_111CorticalThicknessDilated.nii.gz \
  OAS1_0457_MR1_mpr_n3_anon_sbj_111CorticalThicknessDilated_mask.nii.gz 0 0 0 1

# convert to hot color map
ConvertScalarImageToRGB 3 OAS1_0457_MR1_mpr_n3_anon_sbj_111CorticalThicknessDilated.nii.gz \
  OAS1_0457_MR1_mpr_n3_anon_sbj_111CorticalThicknessDilated_hot.nii.gz none hot none 0 8 0 255

# Render the gray matter with superimposed cortical thickness values
antsSurf -s OAS1_0457_MR1_mpr_n3_anon_sbj_111BrainSegmentationGm.nii.gz \
  -f [OAS1_0457_MR1_mpr_n3_anon_sbj_111CorticalThicknessDilated_hot.nii.gz, \
      OAS1_0457_MR1_mpr_n3_anon_sbj_111CorticalThicknessDilated_mask.nii.gz] \
  -d  OAS1_0457_MR1_mpr_n3_anon_sbj_111CorticalThickness.png[90x180x90]
```

Listing 3: Using antsSurf to visualize volumes associated with the OASIS data set subject OAS1_0457_MR1_mpr_n3_anon_sbj_111 processed through the ANTs cortical thickness pipeline.



white matter gray matter cortical thickness

Figure 19: Volumetric renderings of `OAS1_0457_MR1_mpr_n3_anon_sbj_111` data set using `antsSurf`.

5 ANTs-based studies

We've performed several types of applied studies with ANTs. Some of my favorite include our recent work on eigenanatomy eigenanatomy-gs, eigenanatomy-pubmed and sccan sccan-gs, sccan-pubmed as well as a recent large-scale comparison with Freesurfer (if interested, google it). These all use open source tools that are freely available via ANTs and/or *ANTsR*. We are also currently validating significant resources for processing BOLD and ASL functional MRI.

5.1 Brain mapping in the presence of lesions

Many cases violate the basic assumption of a one-to-one mapping from a template image to a target image. Brain lesions caused by stroke or traumatic brain injury are a common instance of this.

The ANTs toolkit handles this situation by a constrained cost-function masking approach. In short, the mapping within an excluded region (as determined by an inclusive mask) is determined by the solution outside of the region. To achieve this with ANTs, one would use a call similar to this http://stnava.github.io/BasicBrainMapping/. The additional parameter needed for this approach is the `-x mask.nii.gz` which is the mask defined in the "fixed" (or individual) image space. Mask values above 0.5 are considered non-outlier or non-lesioned regions. The labeling may be performed with ITK-SNAP. Figure 20 illustrates the approach. Once the ANTs mapping is solved, one is able to estimate the location of the lesion in the template space by mapping the inverse lesion mask to the template. To take the negative image of the inclusive mask and warp the mask to the template :

```
ImageMath 3 lesionmask.nii.gz Neg inclusivemask.nii.gz
antsApplyTransforms -d 3 -i lesionmask.nii.gz -o lesiontotemplate.nii.gz
   -r template.nii.gz -t [output0GenericAffine.mat,1] -t output1InverseWarp.nii.gz
```

There are several examples of this approach in the literature: http://www.ncbi.nlm.nih.gov/pubmed/?term=coslett+kim+injury or relevant google-scholar search. See also: http://www.ncrrn.org/papers/methodology_papers/sp_norm_kim.pdf.

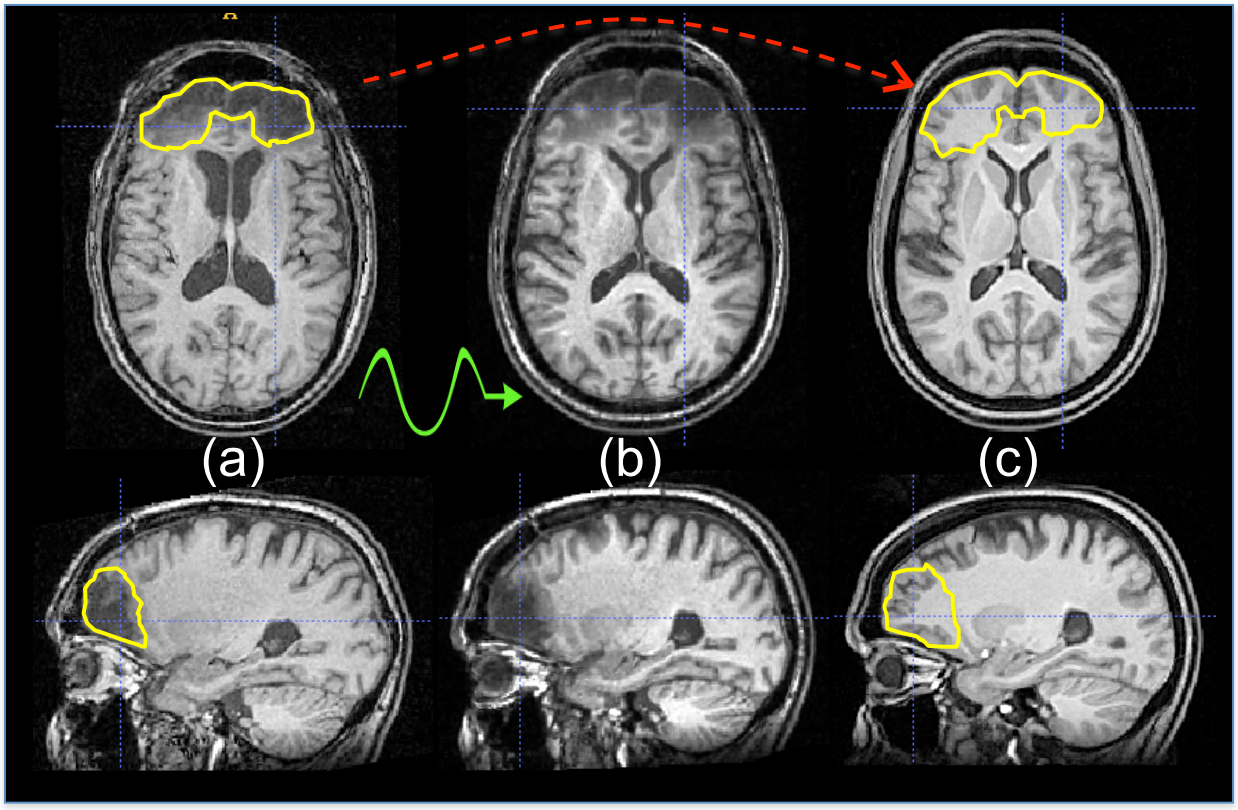


Figure 20: Here, we see a semi-automated approach for lesion analysis based on diffeomorphic normalization. The user outlines the lesion site in the subject space, as shown in (a). Diffeomorphic mapping is then used to deform the subject image in (a) into the space of the reference template. The deformed subject is shown in panel (b). Panel (c) shows the reference template space with the estimated position of the subject's lesion overlaid on the healthy tissue of the template. This approach enables one to compare the subject image against prior information stored in the template space such as anatomical labels or statistics on the appearance of lesioned and/or normal tissue.

5.2 Statistical mapping with ANTs: Morphometry, function, jacobian, thickness

ANTs has been applied in a wide array of imaging studies [5, 4, 14, 20, 26, 27, 29, 39, 45, 12, 10, 23, 25, 32, 33, 46, 1, 3, 15]. All of these studies benefit in some way from normalization whether the topic is cortical thickness, Jacobian-based morphometry, volumetric morphometry or functional studies. From a broad perspective, each of these applications requires the same steps:

1. Preprocess images – bias correction, segmentation, etc. Potentially construct an optimal template.
2. Normalize images – run `antsRegistrationSyN.sh` to map a population of images to a template and store the deformations.
3. Derive any data from the deformation that may be necessary, e.g. the log-Jacobian.
4. Apply the warp to any images one may want to analyze in template space.
5. Perform statistics in template space over the region of interest, e.g. all cortex.

We now list ANTs tools needed for basic morphometry and/or brain labeling. Typical output is in figure 21.

```
0.  antsMultivariateTemplateConstruction2.sh for template building
1.  antsRegistration
2.  antsApplyTransforms for mapping between spaces
3.  Atropos for segmentation.
4.  SmoothImage for smoothing
5.  CreateJacobianDeterminantImage for computing the jacobian
6.  antsMalfLabeling.sh | jointfusion for convenient multi-template labeling
```

See [43] to learn more about joint fusion. Figure 21 shows a more or less standard Jacobian-based approach to morphometry. Often, we use the log-Jacobian because it is symmetric about zero. Some suggest to mask with the gray matter segmentation to restrict the analysis to the cortex. The Jacobian is discussed in figure 7 and visualization of the Jacobian is shown in figure 22. This Jacobian comes from the mapping shown in figure 13. Many analyses may be performed on data that may be derived using ANTs: thickness images, functional images or fractional anisotropy images derived from the diffusion tensor modality. In all these cases, the tools listed above and throughout this document should be very helpful.

5.3 Statistics with ANTs and `R`: `ANTsR`

`R` is an open-source, cutting-edge statistical package used by statisticians world-wide. ANTs is designed to interface with `R` via the `ANTsR` package. Google this and learn more. Most statistical requirements may be met with this setup.

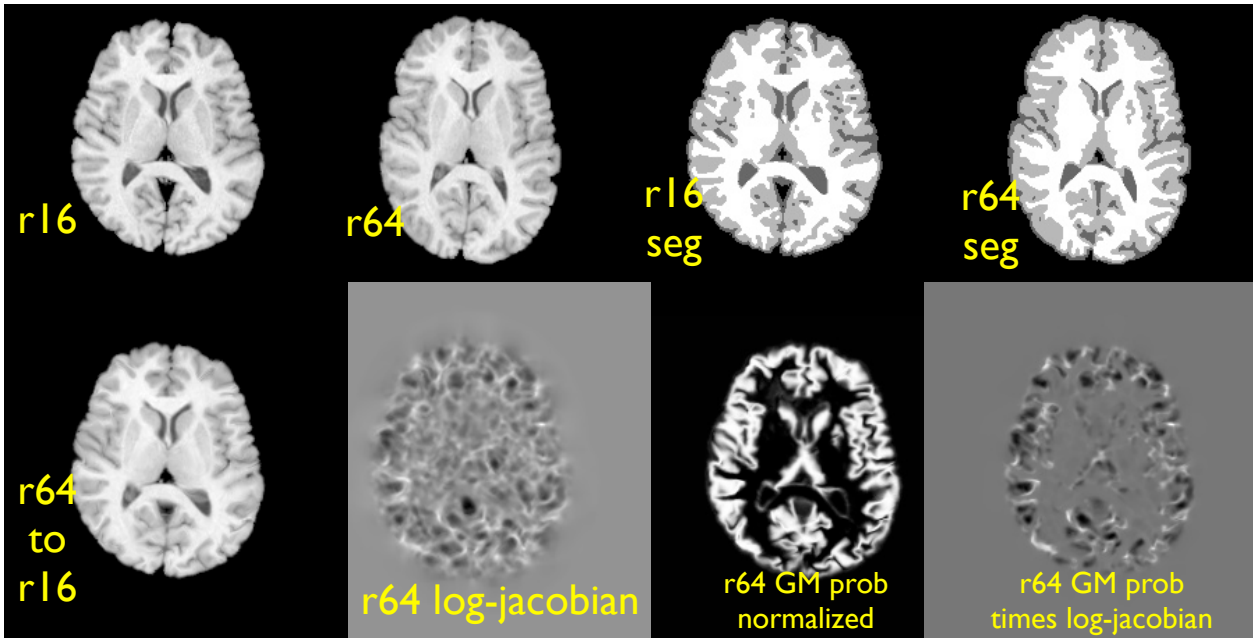


Figure 21: The output from the ANTs morphometry example.

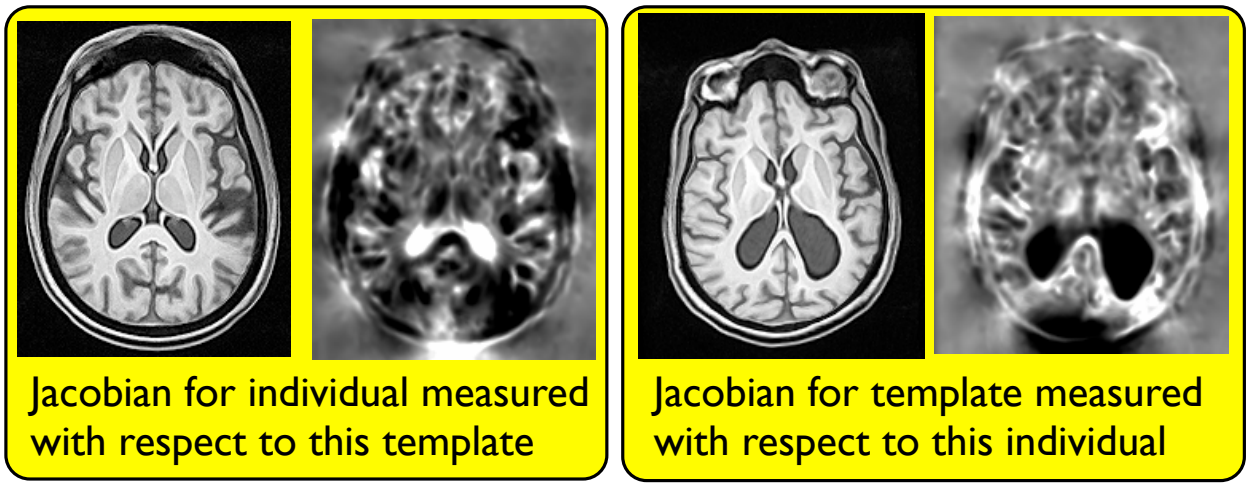


Figure 22: The Jacobian: Note that the bright values in the template image ventricles (left) indicate that the ventricles are relatively larger in the subject image. Similarly, the dark jacobian values in the individual image show that the template has smaller ventricles. This example is described on the large deformation page Large Def. Once one gains the Jacobian (or more appropriately, the log-Jacobian), then one may compute statistics across the population.

6 Dependencies and Related Software

6.1 Dependencies and Compilation

ANTs depends on the most recent version of the Insight ToolKit (ITK) – www.itk.org. However, there is no need to download ITK separately. ANTs and ITK are both compiled through CMake which automates the process. We describe this here:

```
instructions for linux / osx type installation.
windows works too ( via cmake ) but i dont have specifics.

to compile ants from the source code, you first need: git, cmake and a c++ compiler

then in a terminal, do:

>  git clone git://github.com/stnava/ANTs.git
>  mkdir antsbin

> cd antsbin

>  ccmake ../ANTs

then go into cmake and type c and then g  then exit back to the
terminal.   then:

>  make -j 1

and wait a while.  you can change 1 to n-cores if you're feeling lucky.

to update an existing ANTs install, go to the ANTs directory and type

git pull origin master

Dont forget to toggle to advanced and turn off

SuperBuild_ANTS_USE_GIT_PROTOC if behind firewall
```

We (used to) maintain a dashboard to give users an idea of what to expect from compilation:
ANTs Dashboard: http://testing.psychiatry.uiowa.edu/CDash/index.php?project=ANTS Needs to be re-booted pending funding/space etc.

6.2 Visualization and Quantification of ANTs Results

Use itk SNAP : www.itksnap.org or mricro / mricron for viewing images. The ANTs program StackSlices can be used to conveniently scan through a dataset in any of its axes. It is useful for checking both input data quality and output normalization quality. MeasureImageSimilarity will supply numerical measures of image similarity via three different metrics.

6.3 Pipelining with ANTs

Use PipeDream for automating large-scale processing:
PipeDream Homepage : https://github.com/cookpa/pipedream
PipeDream is integrated with ANTs and automates more complex studies such as multivariate diffusion tensor and T1 cortical thickness studies, longitudinal mapping and reconstruction of nifti images from Dicom.

7 Annotated Bibliography (Old)

The original statement of the symmetric normalization and template construction methodology was given in [6] with a more recent template paper here [9]. A follow up study that used landmark guidance to compare the chimpanzee cortex to the human cortex was published here [8] – this study used *in vivo* MRI and template-based normalization to confirm volumetric numbers derived from an early 20th century post-mortem study comparing one human and one chimp. This conference article has some additional detail and alternative updates to the methodology, in particular application to shape-based interpolation [7]. Network based studies were performed here [21, 22]. The main SyN paper is here [13]. Applications to neurodegeration are here [2, 11, 24, 14, 20, 45, 27]. Hippocampus focused work is here [30, 45]. The main evaluation papers include [13] and [26] for the cortex and deep brain structures whereas [30] evaluates the use of automated and semi-automated normalization for high-throughput hippocampus morphometry. An additional evaluation paper is being developed.

References

[1] Geoffrey K Aguirre, Andres M Komeromy, Artur V Cideciyan, David H Brainard, Tomas S Aleman, Alejandro J Roman, Brian B Avants, James C Gee, Marc Korczykowski, William W Hauswirth, Gregory M Acland, Gustavo D Aguirre, and Samuel G Jacobson. Canine and human visual cortex intact and responsive despite early retinal blindness from rpe65 mutation. *PLoS Med*, 4(6):e230, Jun 2007. 5.2

[2] B. Avants. *Shape optimizing diffeomorphisms for medical image analysis*. PhD thesis, University of Pennsylvania, 2005. 7

[3] B. Avants, C. Anderson, and M. Grossman. Tauopathic longitudinal gray matter atrophy predicts declining verbal fluency: A symmetric normalization study. In *Human Brain Mapping*, 2007. 5.2

[4] B. Avants, P. A. Cook, J. T. Pluta, J. nad Duda, H. Rao, J. Giannetta, H. Hurt, S. Das, and J. Gee. Follow-up on long term effects of prenatal cocaine exposure/poly-substance abuse on the young adult brain. In *Pediatric Academic Society, Annual Meeting*, 2009. 5.2

[5] B. Avants, P. A. Cook, J. T. Pluta, J. nad Duda, H. Rao, J. Giannetta, H. Hurt, S. Das, and J. Gee. Multivariate diffeomorphic analysis of longitudinal increase in white matter directionality and decrease in cortical thickness between ages 14 and 18. In *Human Brain Mapping, 15th Annual Meeting, oral presentation.*, 2009. 5.2

[6] B. Avants, C. L. Epstein, and J. C. Gee. A method for conformally mapping simply connected domains to the unit disc. in preparation, 2004. 2.9, 7

[7] B. Avants, C. L. Epstein, and J. C. Gee. Geodesic image interpolation: Parameterizing and interpolating spatiotemporal images. In *ICCV Workshop on Variational and Level Set Methods*, pages 247–258, 2005. 7

[8] B. Avants, C. L. Epstein, and J. C. Gee. Geodesic image normalization in the space of diffeomorphisms. *Mathematical Foundations of Computational Anatomy*, pages 125–133, 2006. 7

[9] B. Avants, P. Yushkevich, J. Pluta, and J. C. Gee. The optimal template effect in studies of hippocampus in diseased populations. *Neuroimage*, page in press, 2009. 2.9, 7

[10] B. B. Avants, C. L. Epstein, M. Grossman, and J. C. Gee. Symmetric diffeomorphic image registration with cross-correlation: Evaluating automated labeling of elderly and neurodegenerative brain. *Med Image Anal*, 12(1):26–41, Feb 2008. 1, 3, 2.2, 5.2

[11] B. B. Avants, C. L. Epstein, M. Grossman, and J. C. Gee. Symmetric diffeomorphic image registration with cross-correlation: Evaluating automated labeling of elderly and neurodegenerative brain. *Med Image Anal*, 12(1):26–41, Feb 2008. 7

[12] Brian Avants, Jeffrey T Duda, Junghoon Kim, Hui Zhang, John Pluta, James C Gee, and John Whyte. Multivariate analysis of structural and diffusion imaging in traumatic brain injury. *Acad Radiol*, 15(11):1360–1375, Nov 2008. 5.2

[13] Brian Avants, Jeffrey T Duda, Junghoon Kim, Hui Zhang, John Pluta, James C Gee, and John Whyte. Multivariate analysis of structural and diffusion imaging in traumatic brain injury. *Acad Radiol*, 15(11):1360–1375, Nov 2008. 7

[14] Brian Avants, Alea Khan, Leo McCluskey, Lauren Elman, and Murray Grossman. Longitudinal cortical atrophy in amyotrophic lateral sclerosis with frontotemporal dementia. *Arch Neurol*, 66(1):138–139, Jan 2009. 5.2, 7

[15] Brian B Avants, Hallam Hurt, Joan M Giannetta, Charles L Epstein, David M Shera, Hengyi Rao, Jiongjiong Wang, and James C Gee. Effects of heavy in utero cocaine exposure on adolescent caudate morphology. *Pediatr Neurol*, 37(4):275–279, Oct 2007. 5.2

[16] Brian B Avants, P. Thomas Schoenemann, and James C Gee. Lagrangian frame diffeomorphic image registration: Morphometric comparison of human and chimpanzee cortex. *Med Image Anal*, 10(3):397–412, Jun 2006. 1.5, 2.9

[17] Brian B. Avants and Nicholas J. Tustison. The itk image registration framework. *Front Neuroinform*, 7:39, 2014. 1, 1.5

[18] Brian B. Avants, Nicholas J. Tustison, Gang Song, Philip A. Cook, Arno Klein, and James C. Gee. A reproducible evaluation of ants similarity metric performance in brain image registration. *Neuroimage*, 54(3):2033–2044, Feb 2011. 2

[19] Brian B. Avants, Nicholas J. Tustison, Jue Wu, Philip A. Cook, and James C. Gee. An open source multivariate framework for n-tissue segmentation with evaluation on public data. *Neuroinformatics*, 9(4):381–400, Dec 2011. 1

[20] Sandhitsu R Das, Brian B Avants, Murray Grossman, and James C Gee. Registration based cortical thickness measurement. *Neuroimage*, 45(3):867–879, Apr 2009. 15, 3.3, 5.2, 7

[21] Jeffrey T Duda, Brian B Avants, Jane C Asmuth, Hui Zhang, Murray Grossman, and James C Gee. A fiber tractography based examination of neurodegeneration on language-network neuroanatomy. In *Workshop on Computational Diffusion MRI, Medical Image Computing and Computer-Assisted Intervention*, pages 191–198, Sep 2008. 7

[22] Jeffrey T Duda, Brian B Avants, Junghoon Kim, Hui Zhang, S Patel, John Whyte, and James C Gee. Multivariate analysis of thalamo-cortical connectivity loss in TBI. In *Computer Vision and Pattern Recognition*, pages 1–8, Los Alamitos, June 2008. IEEE Computer Society. 7

[23] M. Grossman, C. Anderson, A. Khan, B. Avants, L. Elman, and L. McCluskey. Impaired action knowledge in amyotrophic lateral sclerosis. *Neurology*, 71(18):1396–1401, 2008. 5.2

[24] M. Grossman, C. Anderson, A. Khan, B. Avants, L. Elman, and L. McCluskey. Impaired action knowledge in amyotrophic lateral sclerosis. *Neurology*, 71(18):1396–1401, Oct 2008. 7

[25] Junghoon Kim, Brian Avants, Sunil Patel, John Whyte, Branch H Coslett, John Pluta, John A Detre, and James C Gee. Structural consequences of diffuse traumatic brain injury: A large deformation tensor-based morphometry study. *Neuroimage*, 39(3):1014–1026, Feb 2008. 2.2, 2.9, 5.2

[26] Arno Klein, Jesper Andersson, Babak A Ardekani, John Ashburner, Brian Avants, Ming-Chang Chiang, Gary E Christensen, D. Louis Collins, James Gee, Pierre Hellier, Joo Hyun Song, Mark Jenkinson, Claude Lepage, Daniel Rueckert, Paul Thompson, Tom Vercauteren, Roger P Woods, J. John Mann, and Ramin V Parsey. Evaluation of 14 nonlinear deformation algorithms applied to human brain mri registration. *Neuroimage*, 46(3):786–802, Jul 2009. 1, 3, 5.2, 7

[27] Lauren Massimo, Chivon Powers, Peachie Moore, Luisa Vesely, Brian Avants, James Gee, David J Libon, and Murray Grossman. Neuroanatomy of apathy and disinhibition in frontotemporal lobar degeneration. *Dement Geriatr Cogn Disord*, 27(1):96–104, 2009. 5.2, 7

[28] Keelin Murphy, Bram van Ginneken, Joseph M. Reinhardt, Sven Kabus, Kai Ding, Xiang Deng, Kunlin Cao, Kaifang Du, Gary E. Christensen, Vincent Garcia, Tom Vercauteren, Nicholas Ayache, Olivier Commowick, Grégoire Malandain, Ben Glocker, Nikos Paragios, Nassir Navab, Vladlena Gorbunova, Jon Sporring, Marleen de Bruijne, Xiao Han, Mattias P. Heinrich, Julia A. Schnabel, Mark Jenkinson, Cristian Lorenz, Marc Modat, Jamie R. McClelland, Sébastien Ourselin, Sascha E A. Muenzing, Max A. Viergever, Dante De Nigris, D Louis Collins, Tal Arbel, Marta Peroni, Rui Li, Gregory C. Sharp, Alexander Schmidt-Richberg, Jan Ehrhardt, René Werner, Dirk Smeets, Dirk Loeckx, Gang Song, Nicholas Tustison, Brian Avants, James C. Gee, Marius Staring, Stefan Klein, Berend C. Stoel, Martin Urschler, Manuel Werlberger, Jef Vandemeulebroucke, Simon Rit, David Sarrut, and Josien P W. Pluim. Evaluation of registration methods on thoracic ct: the empire10 challenge. *IEEE Trans Med Imaging*, 30(11):1901–1920, Nov 2011. 1

[29] J. Pluta, B. Avants, P. Yushkevich, S. Glynn, S. Awate, J. Detre, D. Mechanic, and J. C. Gee. Expectation matching for incomplete label driven semi-automated hippocampus segmentation in epilepsy. *Hippocampus*, page accepted, 2009. 1, 5.2

[30] John Pluta, Brian B Avants, Simon Glynn, Suyash Awate, James C Gee, and John A Detre. Appearance and incomplete label matching for diffeomorphic template based hippocampus segmentation. *Hippocampus*, 19(6):565–571, Jun 2009. 7

[31] Lawrence Rosen. *Open source licensing*. Prentice Hall PTR, 2004. 1

[32] Tony J Simon, Zhongle Wu, Brian Avants, Hui Zhang, James C Gee, and Glenn T Stebbins. Atypical cortical connectivity and visuospatial cognitive impairments are related in children with chromosome 22q11.2 deletion syndrome. *Behav Brain Funct*, 4:25, 2008. 5.2

[33] Hui Sun, Brian B Avants, Alejandro F Frangi, Federico Sukno, James C Geel, and Paul A Yushkevich. Cardiac medial modeling and time-course heart wall thickness analysis. *Med Image Comput Comput Assist Interv Int Conf Med Image Comput Comput Assist Interv*, 11(Pt 2):766–773, 2008. 5.2

[34] N. J. Tustison, H. Zhang, G. Lehmann, P. Yushkevich, and J. C. Gee. Meeting andy warhol somewhere over the rainbow: Rgb colormapping and itk. *Insight Journal*, 2008. 4.1

[35] Nicholas J. Tustison and Brian B. Avants. Explicit b-spline regularization in diffeomorphic image registration. *Front Neuroinform*, 7:39, 2013. 1

[36] Nicholas J. Tustison, Brian B. Avants, Philip A. Cook, Junghoon Kim, John Whyte, James C. Gee, and James R. Stone. Logical circularity in voxel-based analysis: normalization strategy may induce statistical bias. *Hum Brain Mapp*, 35(3):745–759, Mar 2014. 1, 2.6

[37] Nicholas J. Tustison, Brian B. Avants, Philip A. Cook, Yuanjie Zheng, Alexander Egan, Paul A. Yushkevich, and James C. Gee. N4itk: improved n3 bias correction. *IEEE Trans Med Imaging*, 29(6):1310–1320, Jun 2010. 1

[38] Nicholas J. Tustison, Brian B. Avants, Lucia Flors, Talissa A. Altes, Eduard E. de Lange, John P Mugler, 3rd, and James C. Gee. Ventilation-based segmentation of the lungs using hyperpolarized (3)he mri. *J Magn Reson Imaging*, 34(4):831–841, Oct 2011. 1

[39] Nicholas J Tustison, Brian B Avants, and James C Gee. Directly manipulated free-form deformation image registration. *IEEE Trans Image Process*, 18(3):624–635, Mar 2009. 5.2

[40] Nicholas J. Tustison, Brian B. Avants, Marcelo Siqueira, and James C. Gee. Topological well-composedness and glamorous glue: a digital gluing algorithm for topologically constrained front propagation. *IEEE Trans Image Process*, 20(6):1756–1761, Jun 2011. 1

[41] Nicholas J. Tustison, Phillip A. Cook, Arno Klein, Gang Song, Sandhitsu R. Das, Jeffrey T. Duda, Benjamin M. Kandel, Niels van Strien, James R. Stone, James C. Gee, and Brian B. Avants. Large-scale evaluation of ANTs and FreeSurfer cortical thickness measurements. *NeuroImage*, 2014. 1

[42] Nicholas J. Tustison, K. L. Shrinhidi, Max Wintermark, Christopher R. Durst, Benjamin M. Kandel, James C. Gee, Murray C. Grossman, and Brian B. Avants. Optimal symmetric multimodal templates and concatenated random forests for supervised brain tumor segmentation (simplified) with antsr. *Neuroinformatics*, 2014. 1

[43] Hongzhi Wang, Sandhitsu R. Das, Jung Wook Suh, Murat Altinay, John Pluta, Caryne Craige, Brian Avants, Paul A. Yushkevich, and Alzheimer's Disease Neuroimaging Initiative . A learning-based wrapper method to correct systematic errors in automatic image segmentation: consistently improved performance in hippocampus, cortex and brain segmentation. *Neuroimage*, 55(3):968–985, Apr 2011. 2, 5.2

[44] D.H. Wolpert and W.G. Macready. No free lunch theorems for optimization. *IEEE Transactions on Evolutionary Computation*, 1(1):67–82, April 1997. 1.2

[45] Paul A Yushkevich, Brian B Avants, John Pluta, Sandhitsu Das, David Minkoff, Dawn Mechanic-Hamilton, Simon Glynn, Stephen Pickup, Weixia Liu, James C Gee, Murray Grossman, and John A Detre. A high-resolution computational atlas of the human hippocampus from postmortem magnetic resonance imaging at 9.4 t. *Neuroimage*, 44(2):385–398, Jan 2009. 2.9, 5.2, 7

[46] Paul A Yushkevich, Brian B Avants, John Pluta, David Minkoff, John A Detre, Murray Grossman, and James C Gee. Shape-based alignment of hippocampal subfields: evaluation in postmortem mri. *Med Image Comput Comput Assist Interv Int Conf Med Image Comput Comput Assist Interv*, 11(Pt 1):510–517, 2008. 5.2