Registration of Images

John Muschelli

2017-03-30

Overview

In this tutorial we will discuss performing co-registration of the multiple imaging modalities to the T1-weighted image. ## Loading Data

```r
library(ms.lesion)
library(neurobase)
files = get_image_filenames_list_by_subject()$training01
t1_fname = files["MPRAGE"]
t1 = readnii(t1_fname)
```

Types of Registration

- Rigid-body registration (linear) - 6 degrees of freedom (dof)
    - Co-registration (within the same person)
        - Cross-sectional between-sequences
        - Longitudinal within-sequence
        - Longitudinal between-sequences
- Affine registration – 12 dof
- Non-linear ($>$ 12 dof)
    - Usually require a prior affine registration
    - Across-subject registration
    - Registration to a template
        - There are many different templates

Types of Registration

- Rigid-body registration (linear) - 6 degrees of freedom (dof)
  - Co-registration (within the same person)
    - **Cross-sectional between-sequences**
    - **Longitudinal within-sequence**
    - **Longitudinal between-sequences**
- Affine registration – 12 dof
- Non-linear (> 12 dof)
  - Usually require a prior affine registration
  - Across-subject registration
  - Registration to a template
    - There are many different templates

Rigid Registration: The Math

For a voxel v, the rigid transformation can be written as:

$$T_{\mathrm{rigid}}(v) = Rv + t$$

where $R =$

$$\begin{bmatrix} \cos\beta\cos\gamma & \cos\alpha\sin\gamma + \sin\alpha\sin\beta\cos\gamma & \sin\alpha\sin\gamma - \cos\alpha\sin\beta\cos\gamma \\ -\cos\beta\sin\gamma & \cos\alpha\cos\gamma - \sin\alpha\sin\beta\sin\gamma & \sin\alpha\cos\gamma + \cos\alpha\sin\beta\sin\gamma \\ \sin\beta & -\sin\alpha\cos\beta & \cos\alpha\cos\beta \end{bmatrix}$$

- 6 degrees of freedom
- 3 associated with the translation vector: $t = (t_x, t_y, t_z)$
- 3 associated with the rotation parameters: $\theta = (\alpha, \beta, \gamma)$.

Image taken from
`http://cnl.web.arizona.edu/imageprops.htm`

- Pitch - Think of nodding (“yes”)
- Yaw - Think of shaking head (“no”) (SMH)
- Roll - Think of shoulder shrugging (“I don’t know”)
- x – left/right
- y – forward/backward
- z – jump up/down

Image Registration

The `registration` function from `extrantsr` can register 2 images. The main arguments are:

- `filename` - either `nifti` object or filename of image to be registered (moving)
- `template.file` - either `nifti` object or filename of target image
- `typeofTransform` - transformation of moving to fixed image (Rigid/Affine)
- `interpolator` - how are voxels averaged in `fixed` space

It can also perform bias correction if `correct = TRUE`.

Image Registration

For example, if we wanted to register the FLAIR to the T1 image, we would run:

```
library(extrantsr)
reg = registration(filename = files["FLAIR"], template.fil
                   typeofTransform = "Rigid", interpolator
```

The output in `reg` would contain the transformed image and paths to the estimated transformations.

Longitudinal Framework for Registration

Within-visit co-registration framework

First, we will register scans within a visit to the T1 of that visit:

Wrapper function to perform preprocessing

We would like to perform registration within a visit. The `extrantsr` function `preprocess_mri_within` will do the following steps:

1. Inhomogeneity correction (N3 or N4)
2. Registration of the files to the first filename (T1)
3. Skull_stripping (if desired, we will perform this after)

Registration within a Visit

The function `within_visit_registration` takes in:

- `fixed` image - the image to be registered to
- `moving` images - images to register to the `fixed`
- `typeofTransform` - transformation of moving to fixed image (Rigid/Affine)
- `interpolator` - how are voxels averaged in `fixed` space

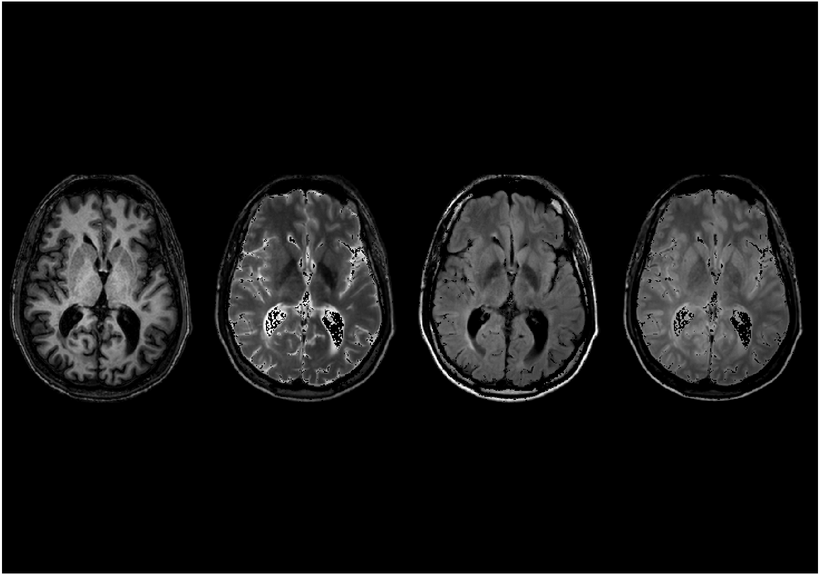
and outputs a list of transformations and output filenames

Register to the T1 image

```r
res = within_visit_registration(
  fixed = files["MPRAGE"],
  moving = files[c("T2", "FLAIR", "PD")],
  correct = TRUE, correction = "N4",
  typeofTransform = "Rigid",
  interpolator = "Linear"
)
output_imgs = lapply(res, function(x) x$outfile)
names(output_imgs) = c("T2", "FLAIR", "PD")
out = c(MPRAGE = list(t1), output_imgs)
```

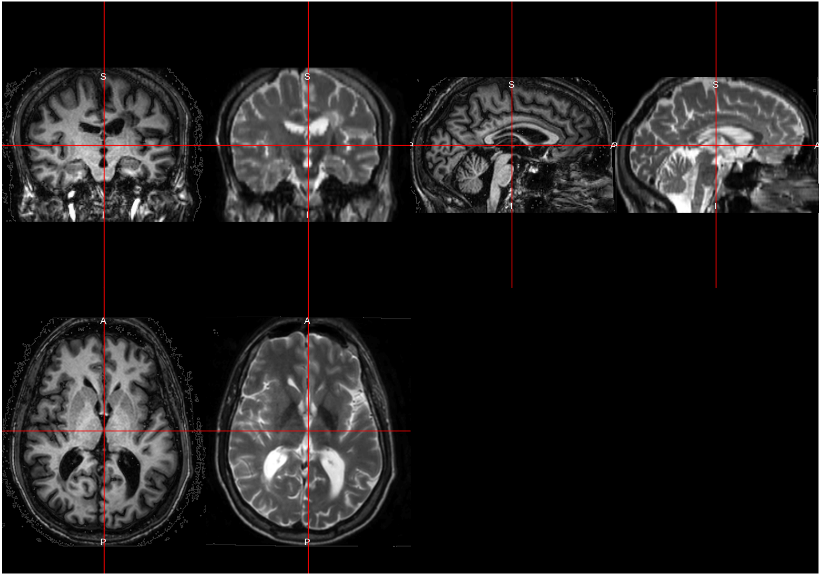
Checking Registration

```
multi_overlay(out)
```



Checking Registration

```
double_ortho(out$MPRAGE, out$T2 )
```



Coregistration within a visit results

- Overall, there seems to be good overlap after registration
- Although we ran the registration on the raw data, it is usually beneficial to do inhomogeneity correction before registration.

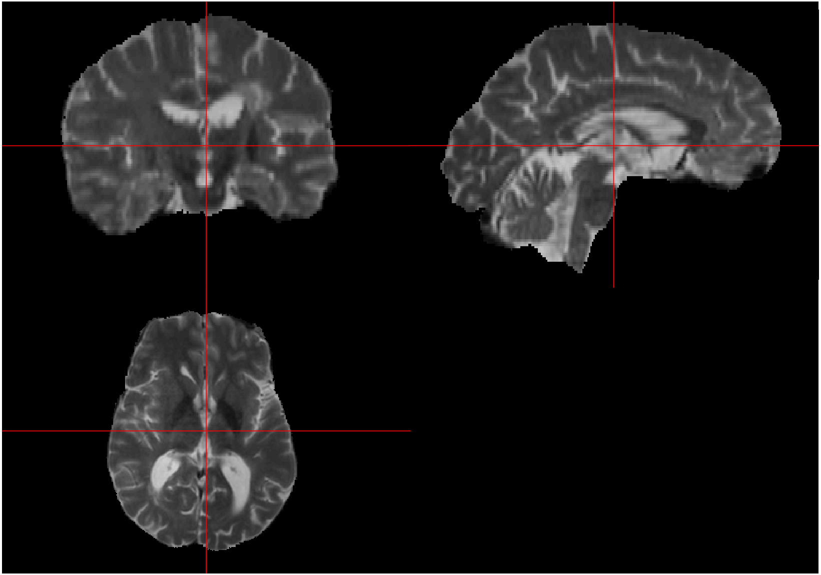
Applying a Brain mask to all registered images

Now that the images are in the same space as the T1, if we skull-strip the T1 image, we can apply this mask to those images to extract brain tissues.

```
mask = readnii("../output/training01_01_mprage_mask.nii.gz"
sub_mask = applyEmptyImageDimensions(mask, inds = dd$inds)
masked_imgs = lapply(xout, mask_img, sub_mask)
```

Result

```r
orthographic(masked_imgs[[2]])
```



Overview of Functions

- Registration within a subject can be done in R
  - `registration` wraps around the reading/writing of images and applying transformations
  - `double_ortho` and `ortho2` can provide some basic visual checks to assess registration quality
  - `preprocess_mri_within` and `preprocess_mri_across` are general wrapper functions to process MRI data
- Once images are registered in the same space, operations can be applied to all the images, such as:
  - Masking with a brain mask
  - Transforming images to new spaces with one modality

Co-registration Overview

- Co-registration requires fewer degrees of freedom (usually 6)
  - sequences from the same individual/brain are more alike than images from different subjects
- Example analyses that do not require a reference template
  - Identify location-specific longitudinal changes within an individual
  - Tissue class or structural segmentation
  - Analysis of indvidual-subject change in intensities

Population Registration

We have only done registration within a subject, but many times you want to perform a population-level analysis. This requries registration to a **template**:

- The `registration` can be done for this as well, just the `template.file` is now the template image and `filename` is the subject image.
  - other files (in the same space) can be transformed using the `other.files` and `other.outfiles` arguments. Or:
  - `ants_apply_transforms` can be used to apply this transformations to the other files