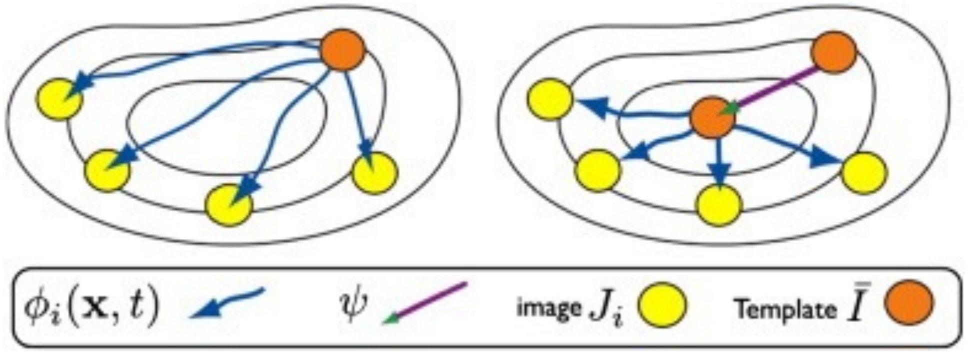
Exploring SyGN algorithm

Which type of transform should be used?

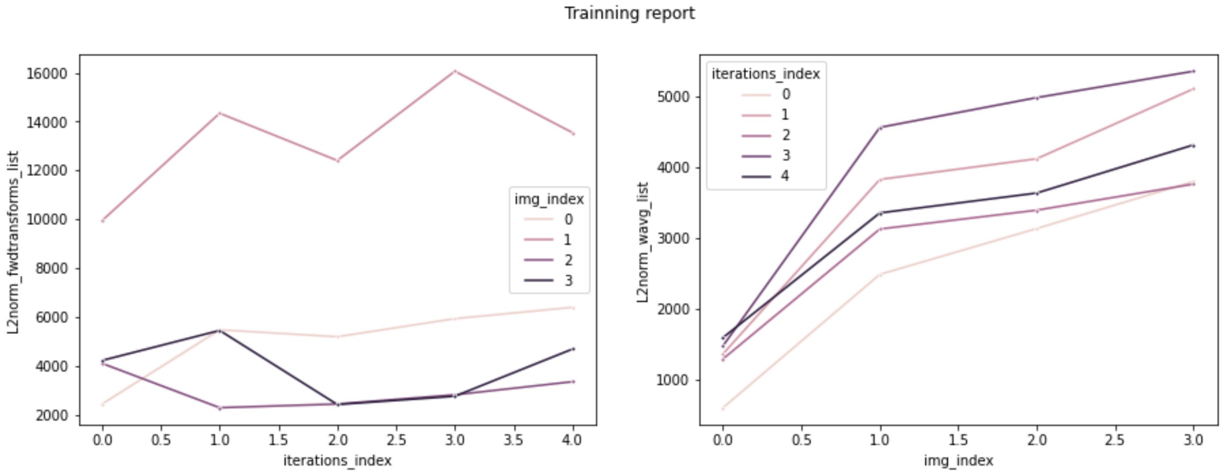


Introduction

- Knowledge Gap
  - Given 10 types of SyN transform in ANTs.build_template() API, which one is best for our NiAD/ABC-DS study?
- Objective
  - To understand how the algorithm works.
  - To testify the performance of the SyGN algoritm on dataset with different characteristic

Introduction: pseudo code of SyGN algorightm

1. Create an initial template by evenly-weighed summing all input images.
2. For the i-th iterations:
    1. For the j-th input image:
        1. Pair-wise image registration on the input image to template. (`typeofTransform` is executed at here)
        2. accumulate transforms by weighted sum.
        3. accumulate warped images by weighted-sum.
    2. Update new template by backward transforming the accumulated warped image to template space with a gradient.
3. Iterate over all j-images for i-iterations.
4. Done.



Example tanning report that showing the L2-Norm of the transform for each image and each iteration (left) and accumulated L2-Norm of the transform within each iteration (right).

Method: Algorithm comparison

- I compared the following three ` typeofTransform` option provided in ants.registration() with mutual information as optimization metric:
  - “SyNRA”: Rigid + Affine + deformable transformation.
  - "SyN": Affine + deformable transformation.
  - "SyNOnly": Only deformable transformation, no initial transformation. Assumes images are aligned by an initial transformation

Method: Image characterization

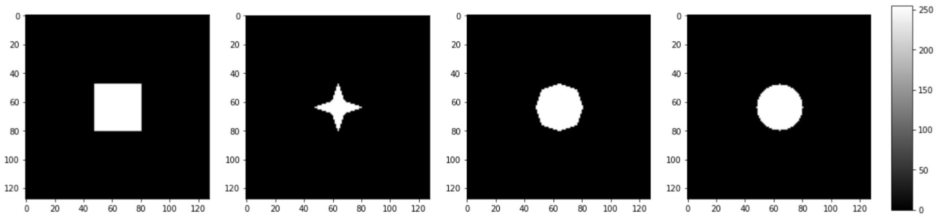
- I identify the following characteristic that is typical in the NiAD/ABC-DS dataset:
  - Three lower-level factors:
    - if_aligned: almost-aligned vs non-aligned
    - if_inside_circle: without vs with inside circle
    - if_equal_area: equal area vs unequal area
  - Two higher-level factors :
    - if_equal_intensity: intensity heterogeneity
    - (additional dataset): structural heterogeneity

- Above of all, 2^4 combinations resulted 16 datasets and one additional dataset

Method: Data Simulation

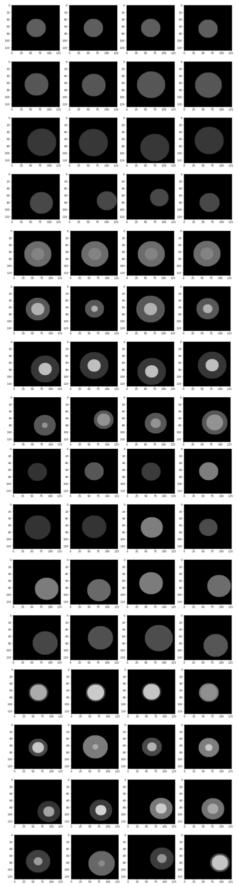
- The control experiment schema was used to generated data. (see table and figure on the right)
  - Experiment repetition = 15
  - Data sampling = 100
  - Registration iterations = 10
  - Images size = 128x128 pixel

- To test the effect of structure heterogeneity, an additional dataset was included (see figure at below)



The additional dataset including four images: square, star, octagon, circle.

	if_aligned	if_inside_circle	if_equal_area	if_equal_intensity	save_dir
0	True	False	True	True	./Dataset/Dataset_1/
1	True	False	False	True	./Dataset/Dataset_2/
2	False	False	True	True	./Dataset/Dataset_3/
3	False	False	False	True	./Dataset/Dataset_4/
4	True	True	True	True	./Dataset/Dataset_5/
5	True	True	False	True	./Dataset/Dataset_6/
6	False	True	True	True	./Dataset/Dataset_7/
7	False	True	False	True	./Dataset/Dataset_8/
8	True	False	True	False	./Dataset/Dataset_9/
9	True	False	False	False	./Dataset/Dataset_10/
10	False	False	True	False	./Dataset/Dataset_11/
11	False	False	False	False	./Dataset/Dataset_12/
12	True	True	True	False	./Dataset/Dataset_13/
13	True	True	False	False	./Dataset/Dataset_14/
14	False	True	True	False	./Dataset/Dataset_15/
15	False	True	False	False	./Dataset/Dataset_16/



Design matrix used to simulate random image data (Left) ;
4 by 16 images shown the first 4 images of 16 dataset (right).

Method : L2-norm Measurement

- Based on the locally Euclidean property of Riemannian geometry, I use the Euclidean distance (L2-norm) to quantify the distance between the original and the warped image within each registration iteration:

$$\text{L2-norm} = \sqrt{\sum_{\text{Comp}=1}^{\text{n}} \text{Deform}^2}$$

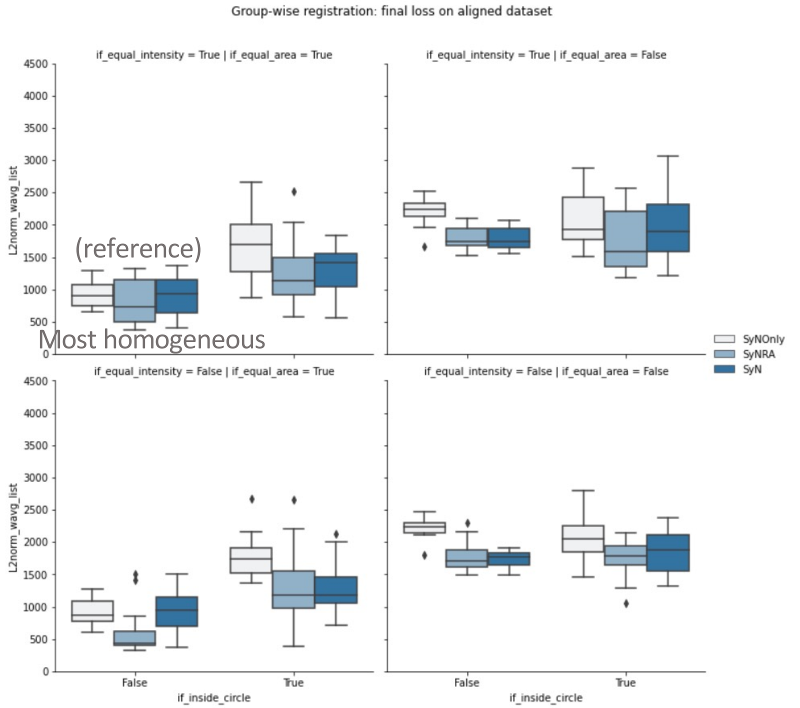
where:

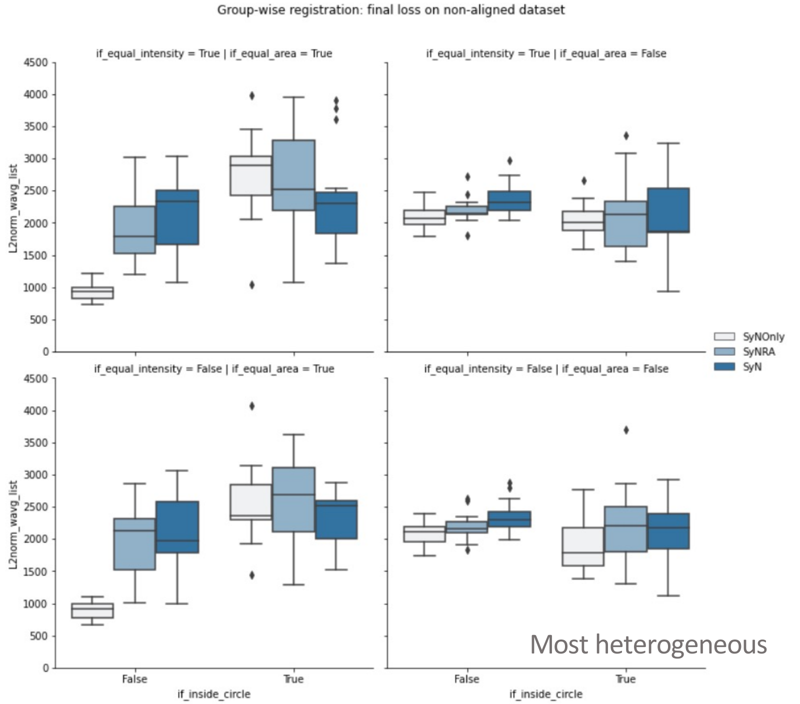
- Comp: the number of component. It is same as the dimention of the image.
- Deform: a defomation matrix for one component. It have same shape of the image.

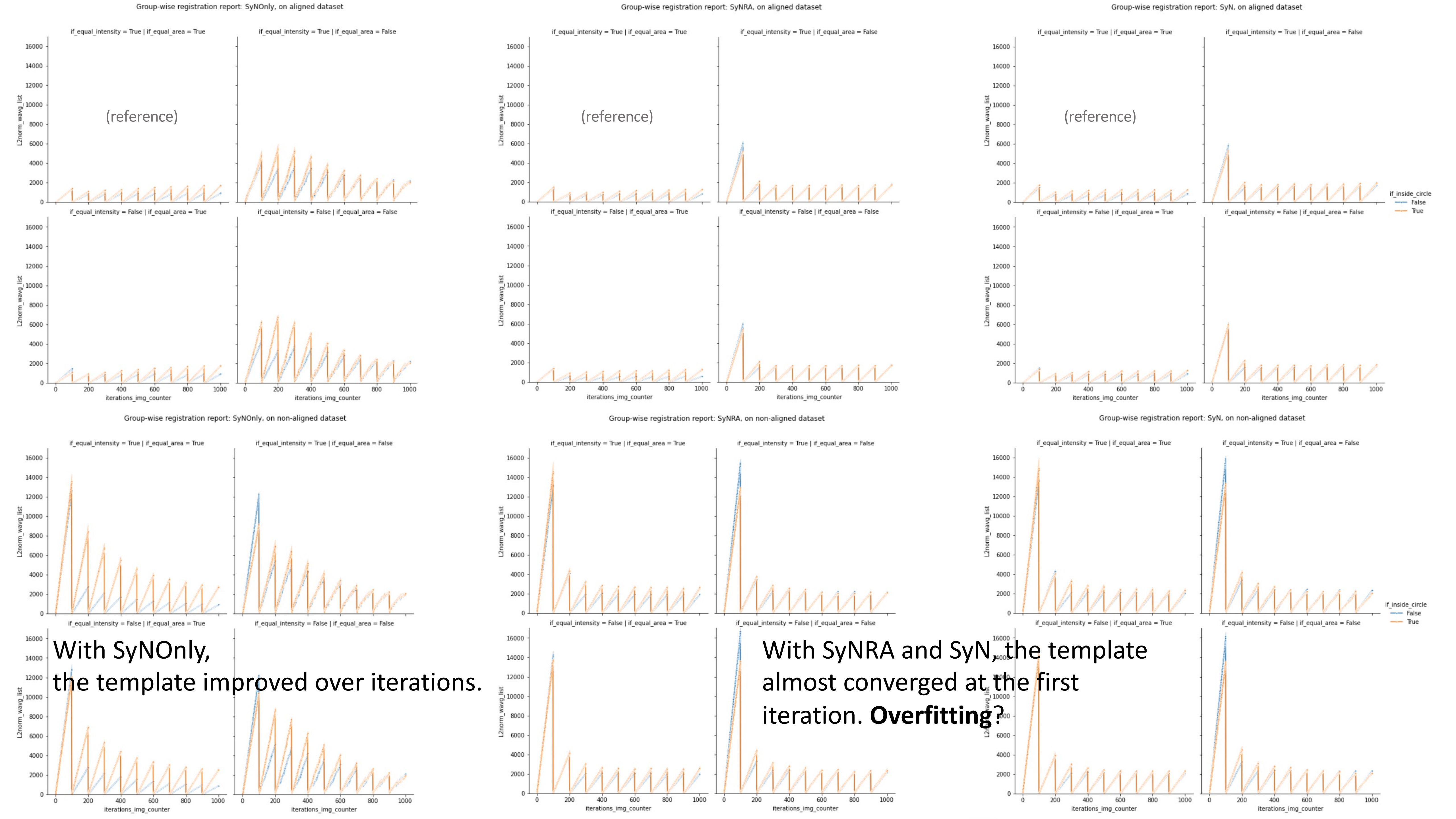
- Usage:
  - Compare final convergence between algorithm and dataset
  - Check if overfitting using the L2-norm as the learning curve

Result: Better convergence with SyN/SyNRA

- In general:
  - template converged better with SyN/SyNRA than SyNOnly.
- Noticeably for non-aligned dataset:
  - SyNOnly resulted better convergence in more homogeneous datset.
  - SyNOnly have lower variance.
- However, strong learners often cause overfitting (see the next slide)

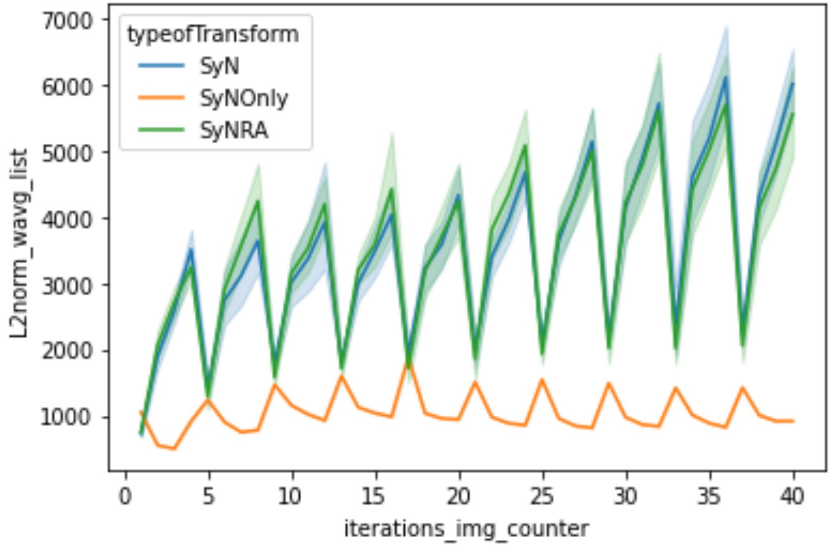




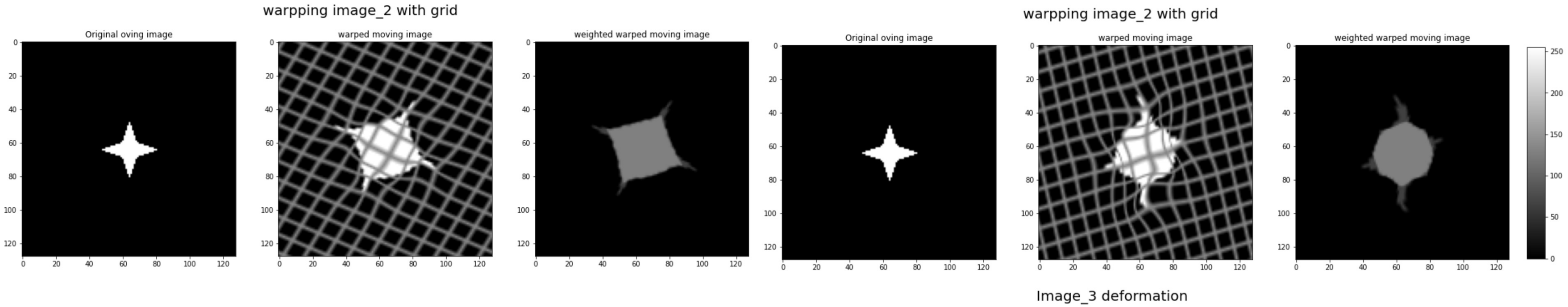

Group-wise registration report: SyNOnly, on aligned dataset
Group-wise registration report: SyNRA, on aligned dataset
Group-wise registration report: SyN, on aligned dataset
if_equal_intensity = True | if_equal_area = True
if_equal_intensity = True | if_equal_area = False
if_equal_intensity = False | if_equal_area = True
if_equal_intensity = False | if_equal_area = False
(reference)
L2norm_wavg_list
iterations_img_counter
if_inside_circle
False
True
Group-wise registration report: SyNOnly, on non-aligned dataset
Group-wise registration report: SyNRA, on non-aligned dataset
Group-wise registration report: SyN, on non-aligned dataset
With SyNOnly,
the template improved over iterations.
With SyNRA and SyN, the template
almost converged at the first
iteration. **Overfitting**?

Result: affine transform causes overfitting on structurally heterogeneous dataset

- The affine transform in SyN and SyNRA introduced randomness and overfitting.
- The randomness stacked up within each iteration.
- The affine transform is supposed to align image, but it did more than that. Noticeably, affine transformation continuedly added unexpected warping over the registration.



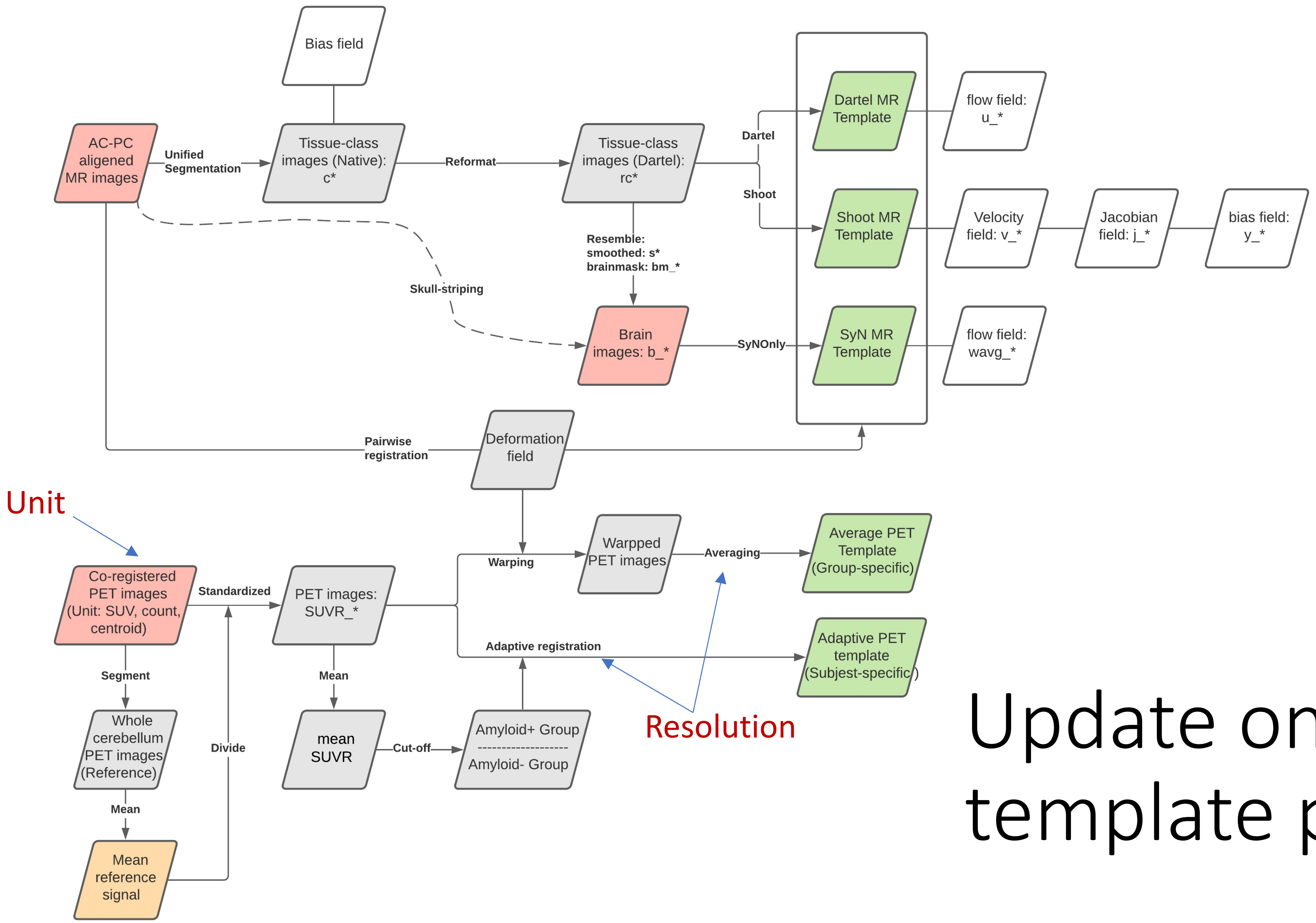
The lineplot for L2-norm of the transform as the output of 10-iteration SyGN on structurally heterogeneous dataset with 15 repetitions



The warping of an image at the 5th iteration using SyN (left) and SyNRA (right)

Conclusion

- `SyN` and `SyNRA` are not appropriate for the structurally heterogeneous dataset, due to the affine transformation at every image-to-template registrations.
- `SyNOnly` is the better option, although it require many more iterations for the template to converge.
  - It might require more than 10 iteration, although the documentation says "should be greater than 1 less than 10".



Update on PET template processing

Update on PET template processing

- Issue with different Unit
  - NiAD_Clresult_SPM8.xlsx has mean value of whole cerebellum for some scans (colname: WhlCbl_2mm). I can standardize the PET images to identical unit of SUVr.
  - Need to solve: this file did not contain some newer scans.
- Issue with different resolution:
  - I also test the function to homogenize PET images with different resolution: ants.apply_transforms.

Idea: Compare template between group-wise registration algorithms

- What can be compared?
  - SSIM: the global-level similarity between two images
  - ANOVA: the pixel-level variance of warped images between algorithm.
- What can be answer?
  - Does algorithms different?
  - Does algorithms perform perform differently by data acquisition and/or subject demographics?