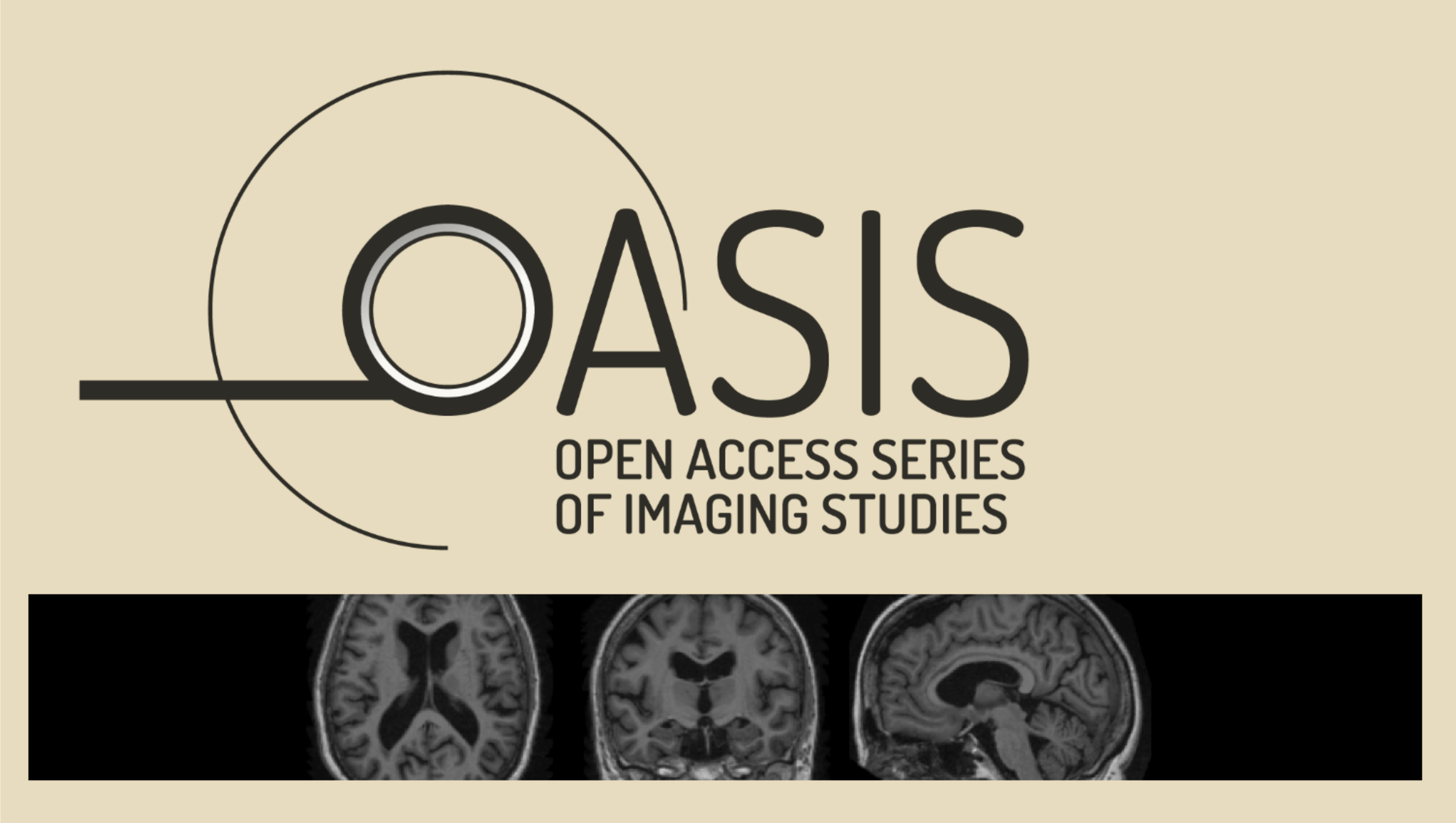
# Spatial transformations

BIOMEDICAL IMAGE ANALYSIS IN PYTHON

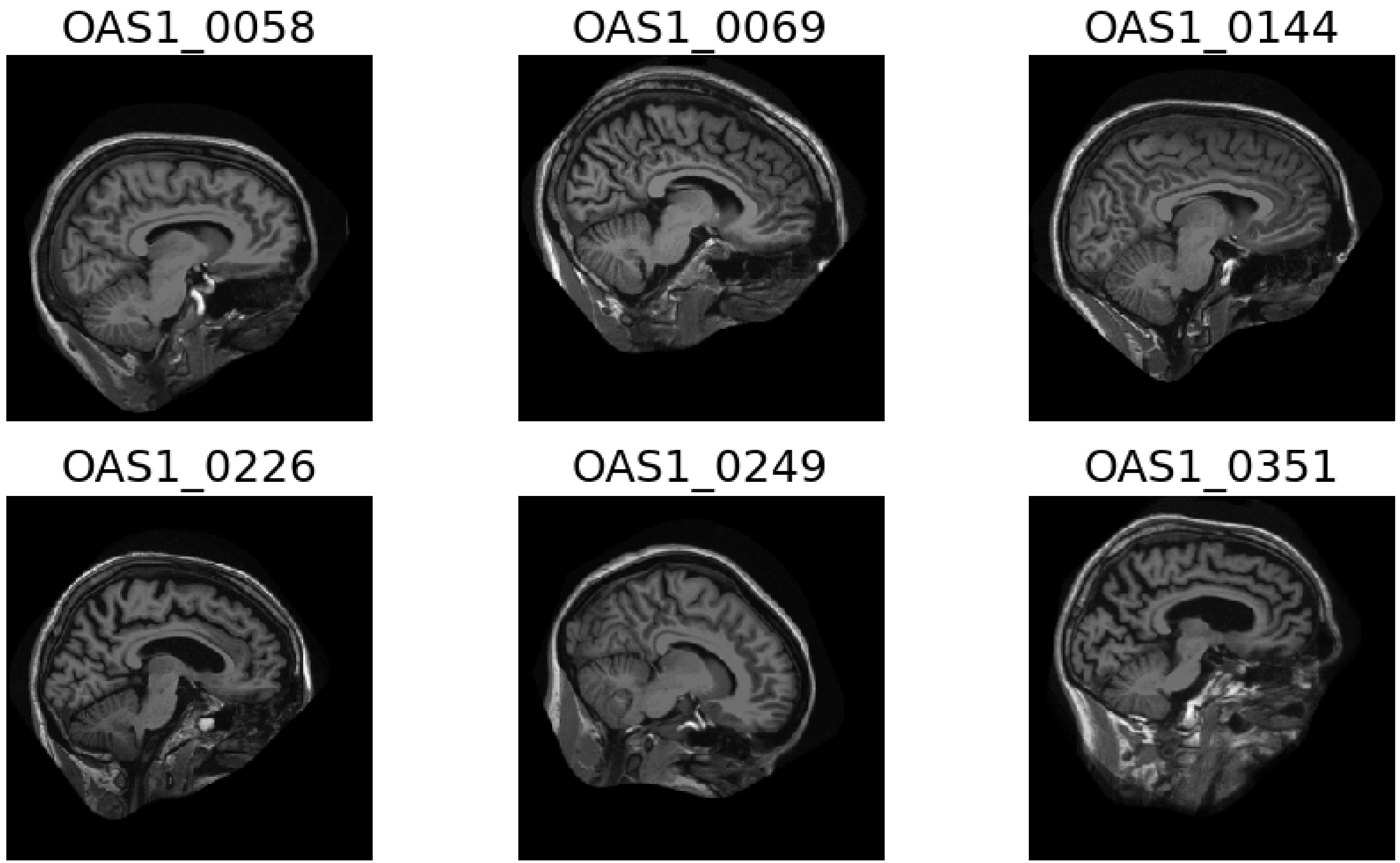
**Stephen Bailey**
Instructor

datacamp

# OASIS Database

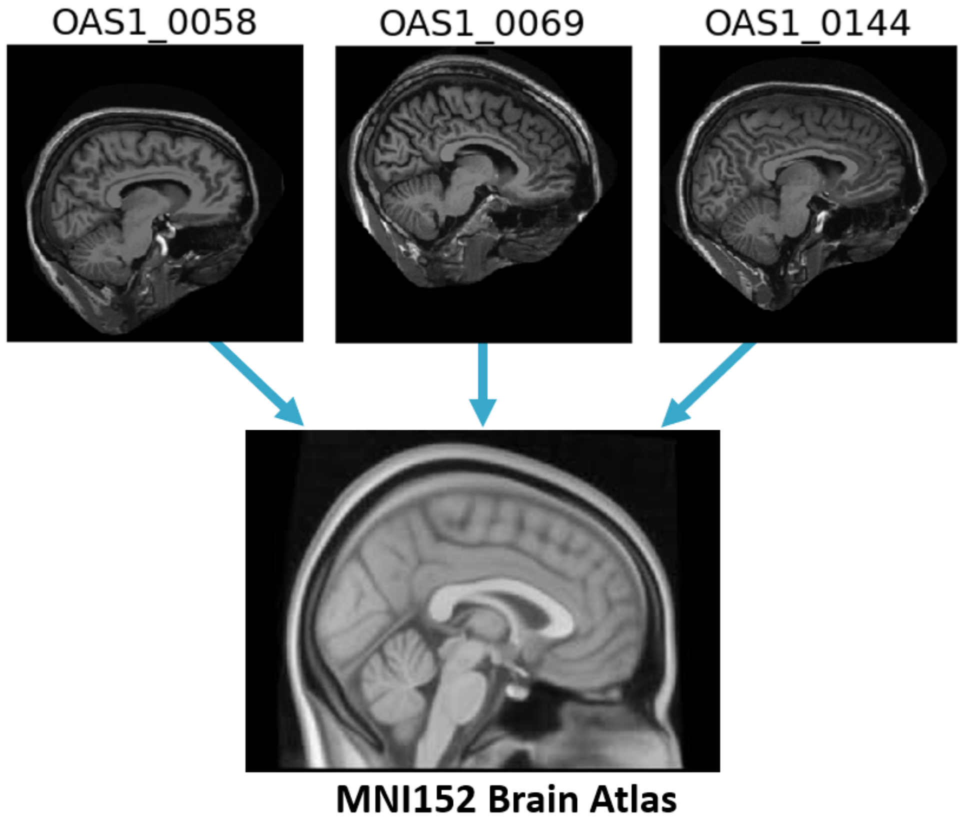


# Significant variability

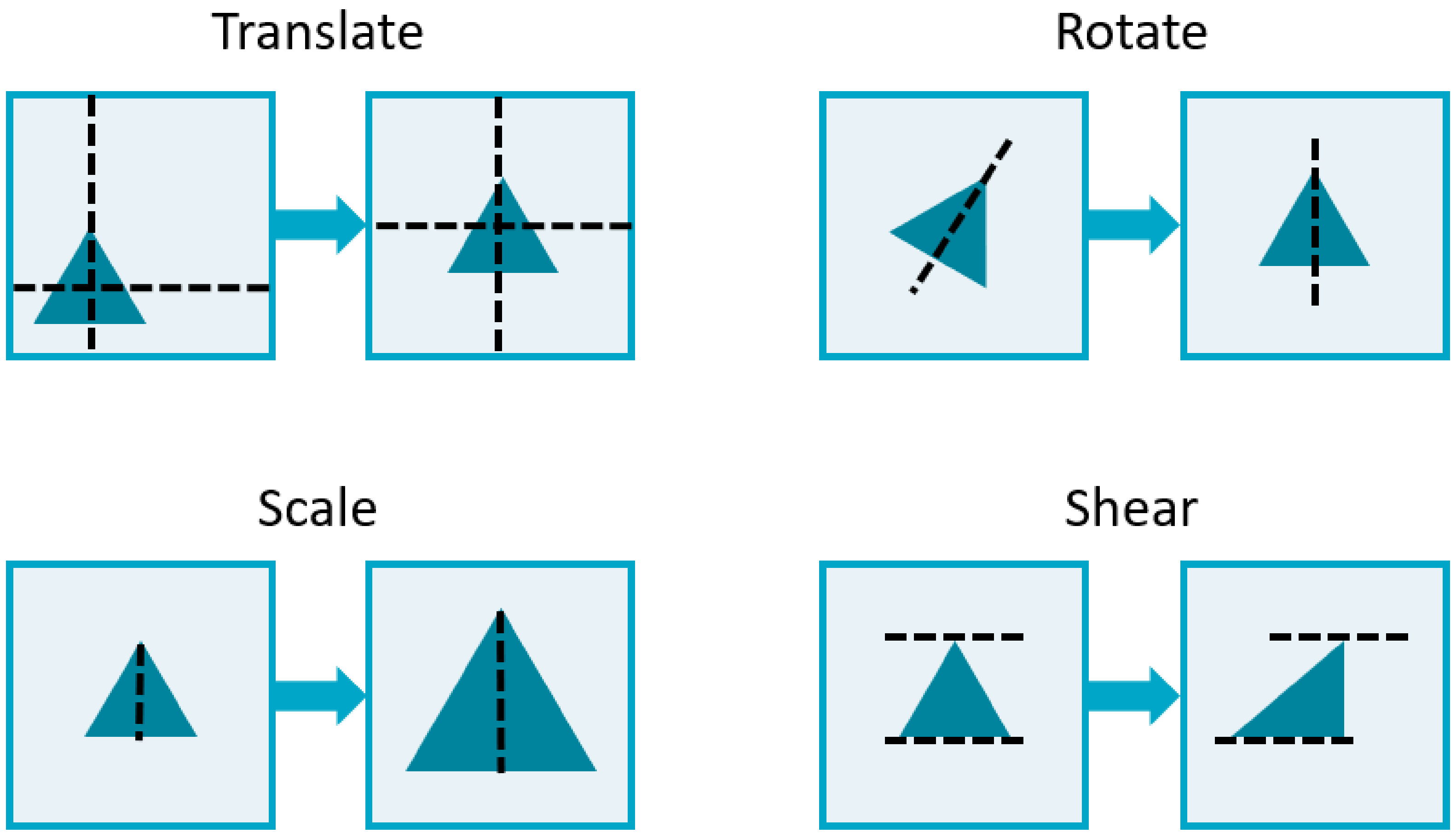


# Registration

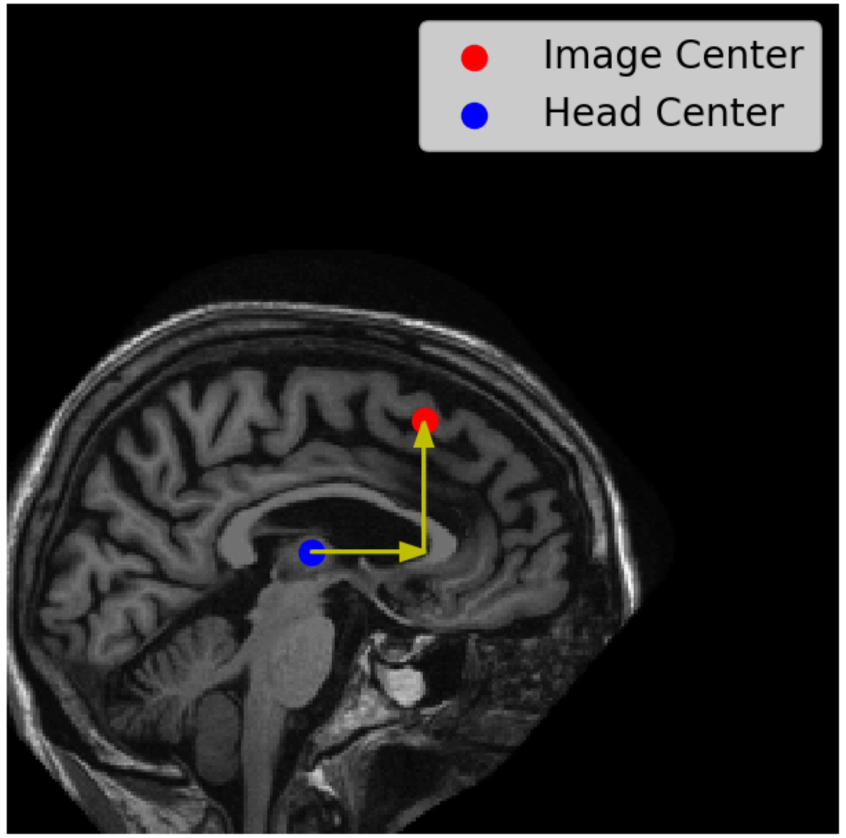
- Align images to template
- Minimize spatial variability
- Templates:
  - may represent multiple subjects
  - may be an "average" image
- Entails many spatial transformations



# Affine transformations preserve points, lines, and planes



# Translation



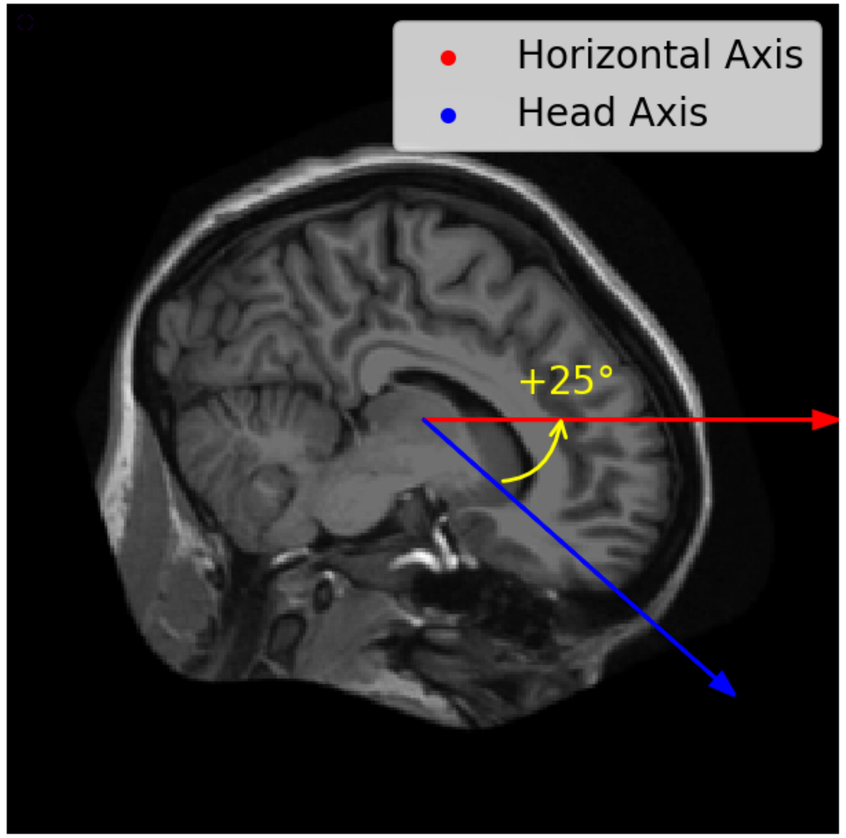

```
import imageio
import scipy.ndimage as ndi

im=imageio.imread('OAS1036-2d.dcm')
im.shape
```

```
(256, 256)
```

```
com = ndi.center_of_mass(im)
d0 = 128 - com[0]
d1 = 128 - com[1]
xfm = ndi.shift(im,
          shift=[d0, d1])
```

# Rotation

```
ndi.rotate(im,
           angle=25,
           axes=(0,1))
```



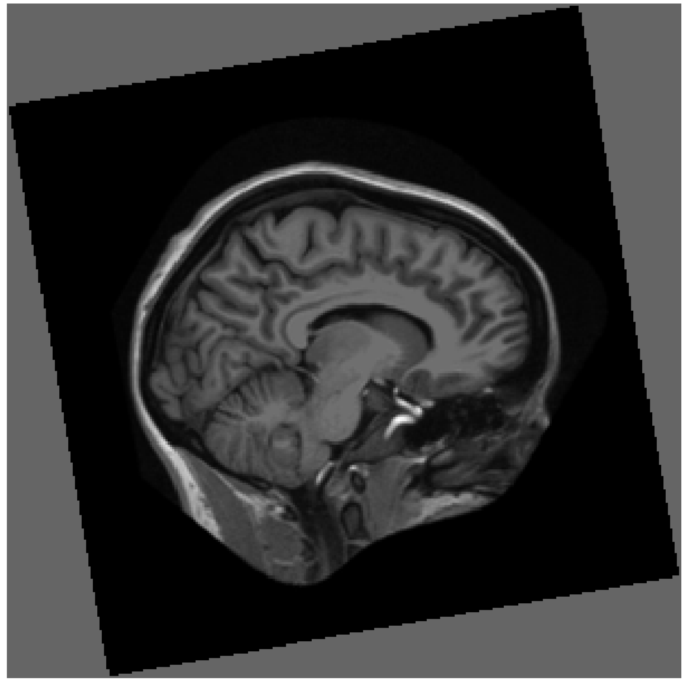
# Image rotation

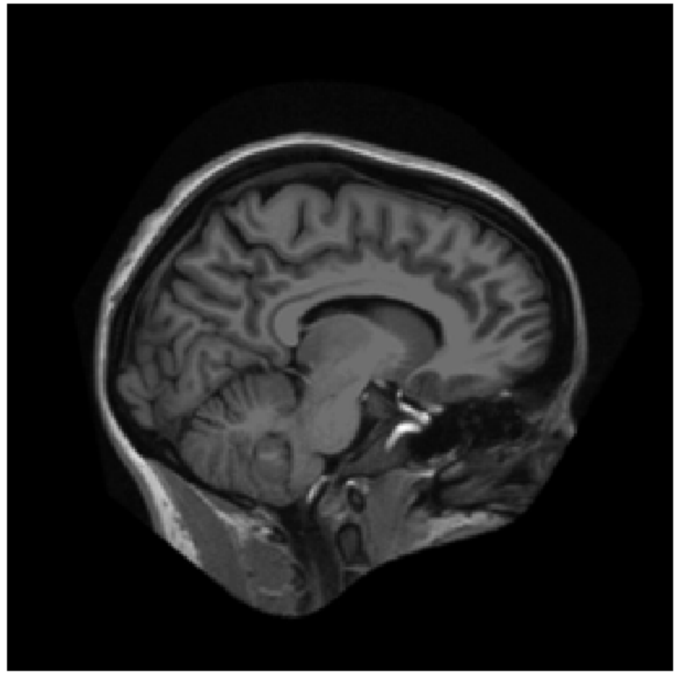
```
xfm = ndi.rotate(im, angle=25)

xfm.shape
```

```
(297, 297)
```

```
xfm = ndi.rotate(im, angle=25,
                 reshape=False)
xfm.shape
```

```
(256, 256)
```





# Transformation matrix

**Transformation matrix:** applied to one image for registration.

Elements of the matrix encode "instructions" for different affine transformations.

Translation

$$\begin{bmatrix} 1 & 0 & T_x \\ 0 & 1 & T_y \\ 0 & 0 & 1 \end{bmatrix}$$

Rotation

$$\begin{bmatrix} \cos(\Theta) & -\sin(\Theta) & 0 \\ \sin(\Theta) & \cos(\Theta) & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

Scale

$$\begin{bmatrix} S_x & 0 & 0 \\ 0 & S_y & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

Shear

$$\begin{bmatrix} 1 & Sh_x & 0 \\ Sh_y & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

# Applying a transformation matrix

```
# Identity matrix
mat = [[1, 0, 0],
       [0, 1, 0],
       [0, 0, 1]]
xfm = ndi.affine_transform(im,
                           mat)
```

```
# Translate and rescale
mat = [[0.8, 0,   -20],
       [0,   0.8, -10],
       [0,   0,    1]]
xfm = ndi.affine_transform(im,
                           mat)
```

Translate

$$\begin{bmatrix} 1 & 0 & T_x \\ 0 & 1 & T_y \\ 0 & 0 & 1 \end{bmatrix}$$

Rotate

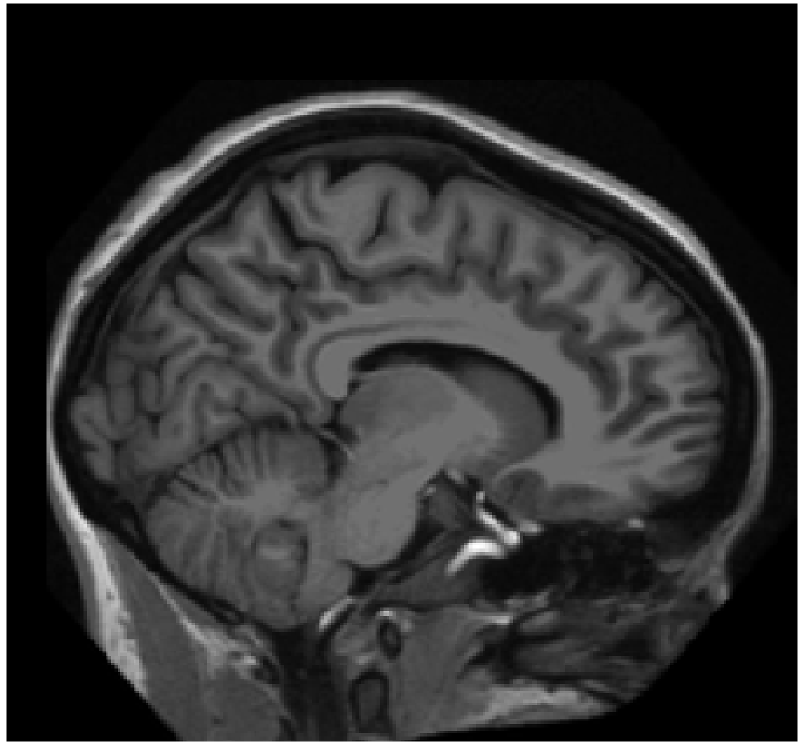
$$\begin{bmatrix} \cos(\Theta) & -\sin(\Theta) & 0 \\ \sin(\Theta) & \cos(\Theta) & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

Scale

$$\begin{bmatrix} S_x & 0 & 0 \\ 0 & S_y & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

Shear

$$\begin{bmatrix} 1 & Sh_x & 0 \\ Sh_y & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}$$



# Let's practice!

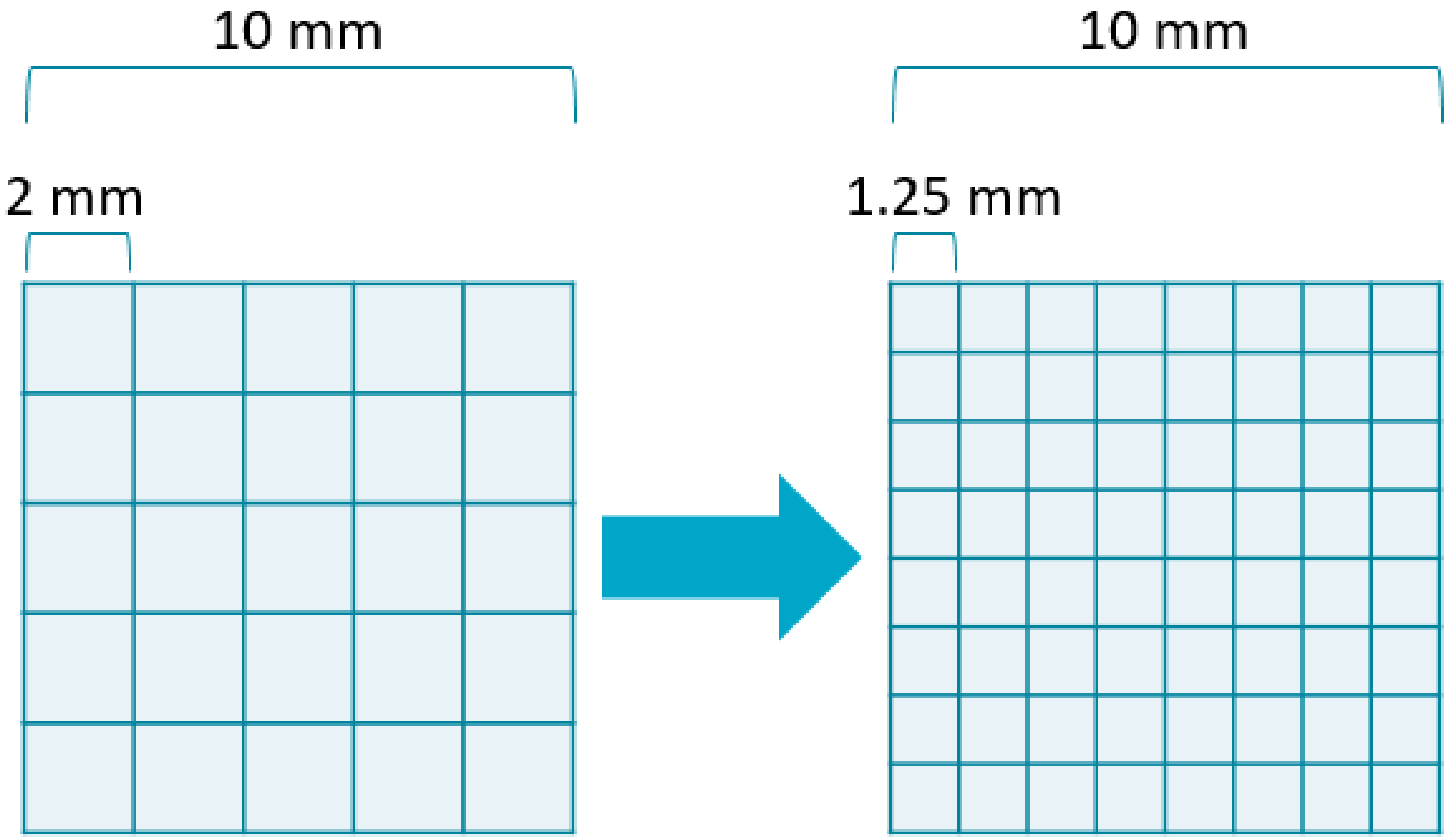
# Resampling and interpolation

BIOMEDICAL IMAGE ANALYSIS IN PYTHON

**Stephen Bailey**
Instructor

datacamp

# Resampling changes the array shape



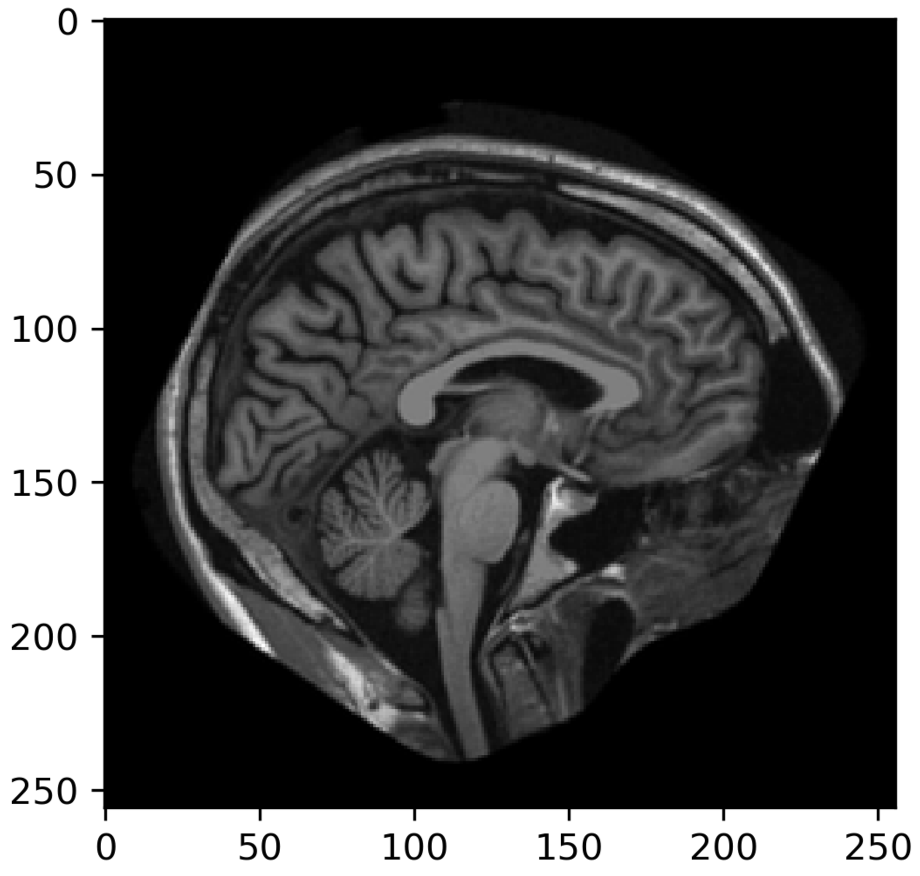
# Downsampling

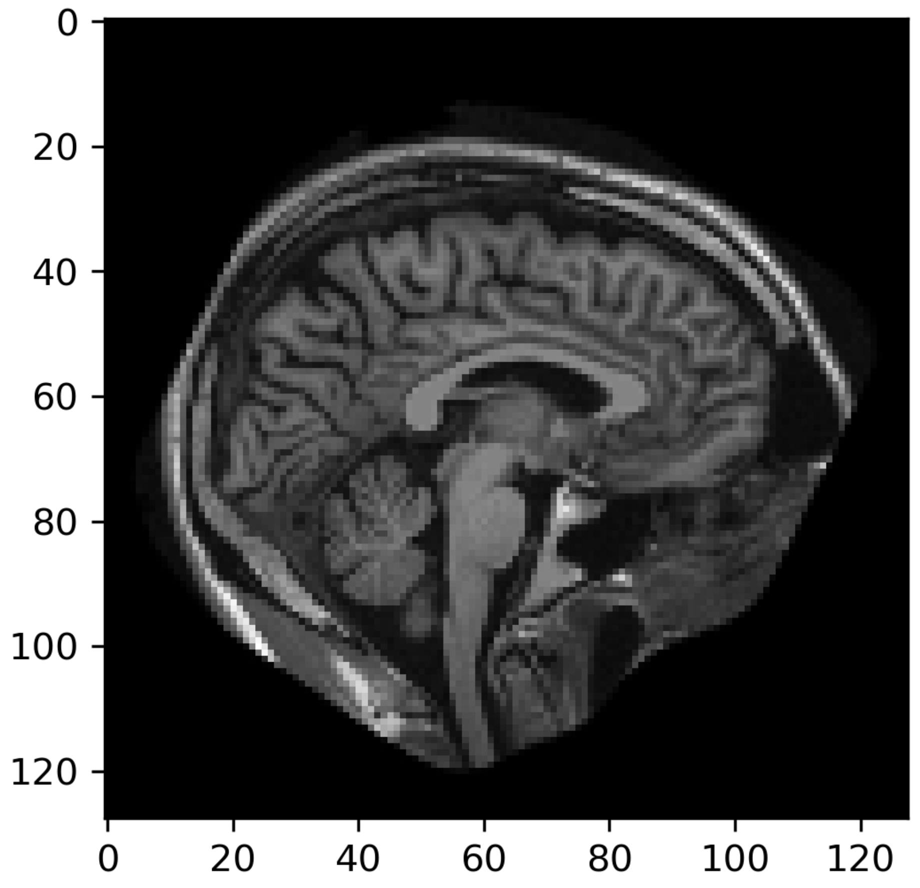
```
vol = imageio.volread('OAS1_0255')
vol.shape
```

```
(256, 256, 256)
```

```
vol_dn = ndi.zoom(vol, zoom=0.5)
vol_dn.shape
```

```
(128, 128, 128)
```



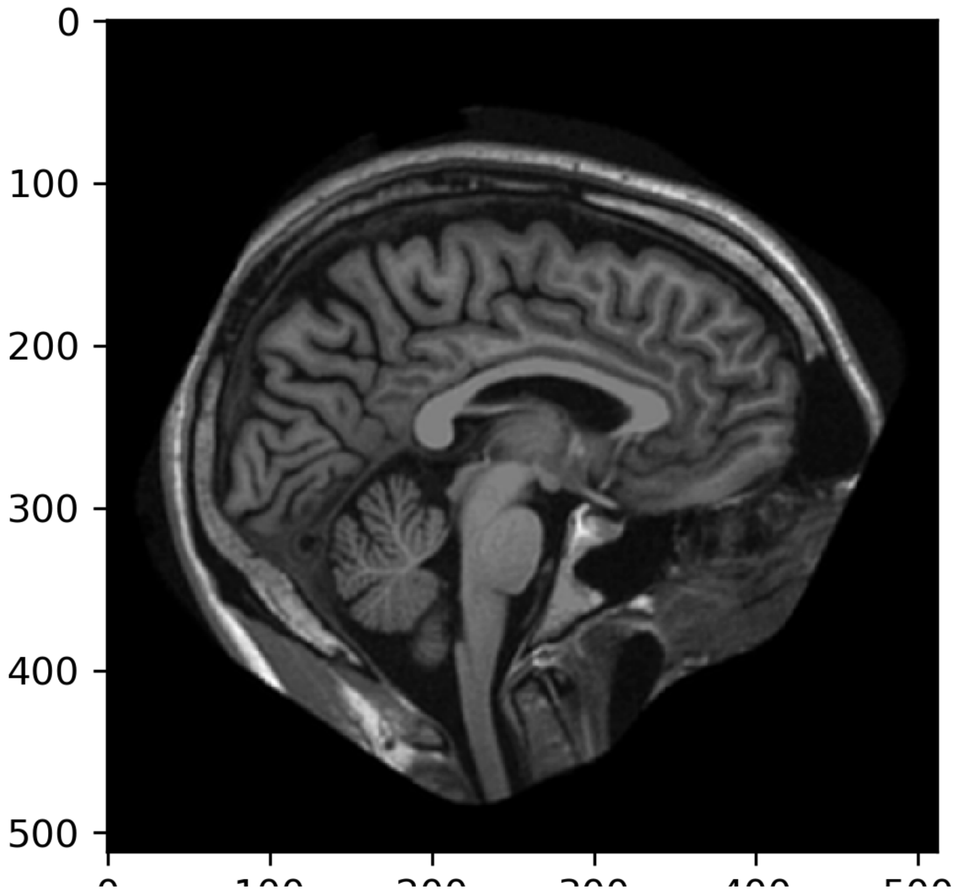



# Upsampling

- Resampling to a larger grid
- Not the same as collecting higher-resolution data
- Useful for standardizing sampling rates that are unequal

```
vol_up = ndi.zoom(vol, zoom=2)
vol_up.shape
```

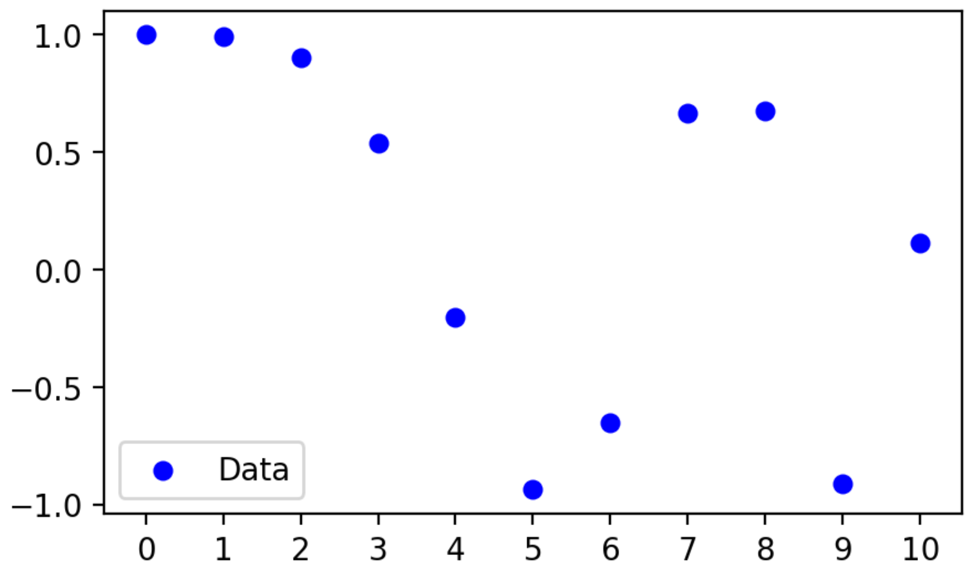
```
(512, 512, 512)
```



# Interpolation

- "Stitches together" grid points to model the space between points.

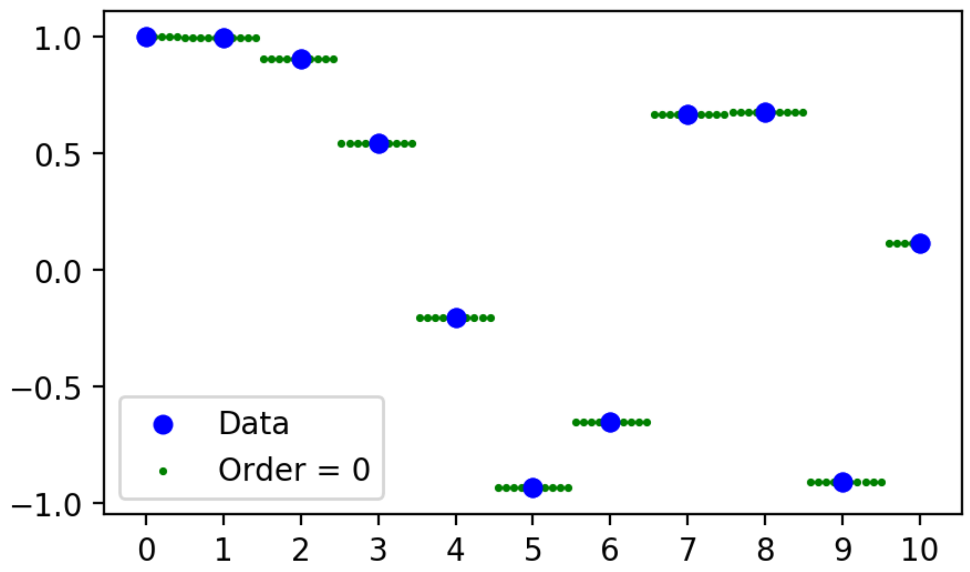
$$Interpolation\ in\ 1\ D$$



# Interpolation

- "Stitches together" grid points to model the space between points.
- **Nearest-neighbor:** uses the closest measured value.

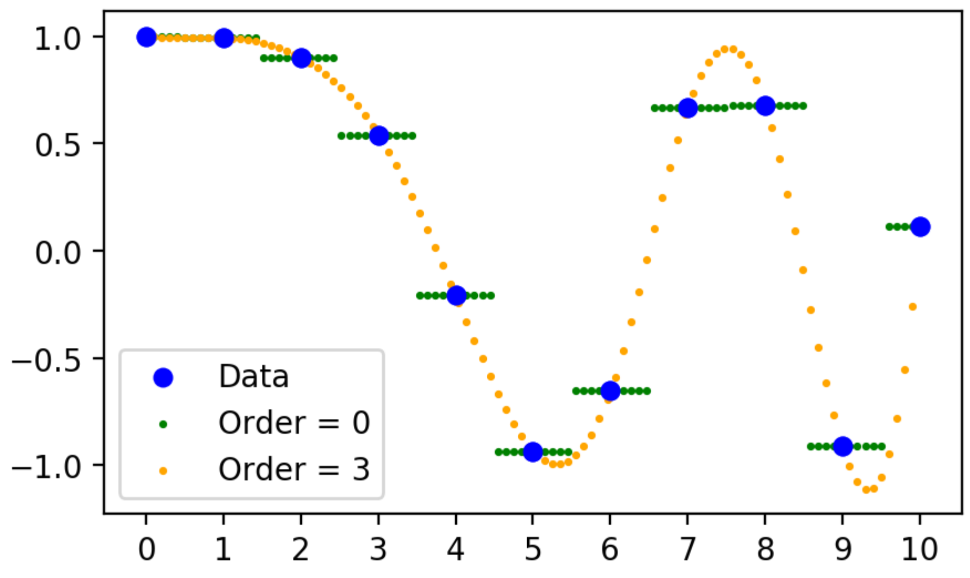
*Interpolation in 1 D*



# Interpolation

- "Stitches together" grid points to model the space between points.
- **Nearest-neighbor:** uses the closest measured value.
  - `order = 0`
- **B-spline interpolation:** models space between points with spline functions of a specified order.
  - `order` is between `1` and `5`

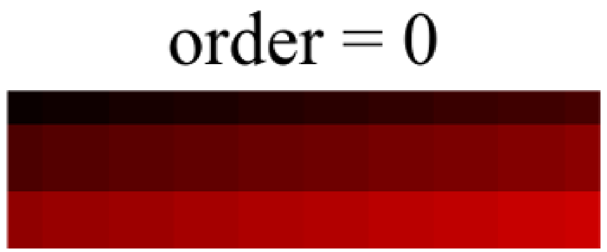
*Interpolation in 1 D*

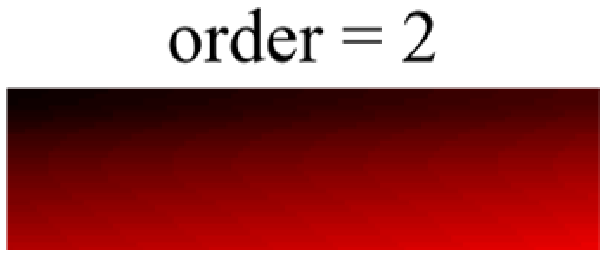


# Interpolation in 2D

```
im=np.arange(100).reshape([10,10])
```

```
zm1=ndi.zoom(im, zoom=10, order=0)
zm2=ndi.zoom(im, zoom=10, order=2)
zm3=ndi.zoom(im, zoom=10, order=4)
```



# Let's practice!

BIOMEDICAL IMAGE ANALYSIS IN PYTHON

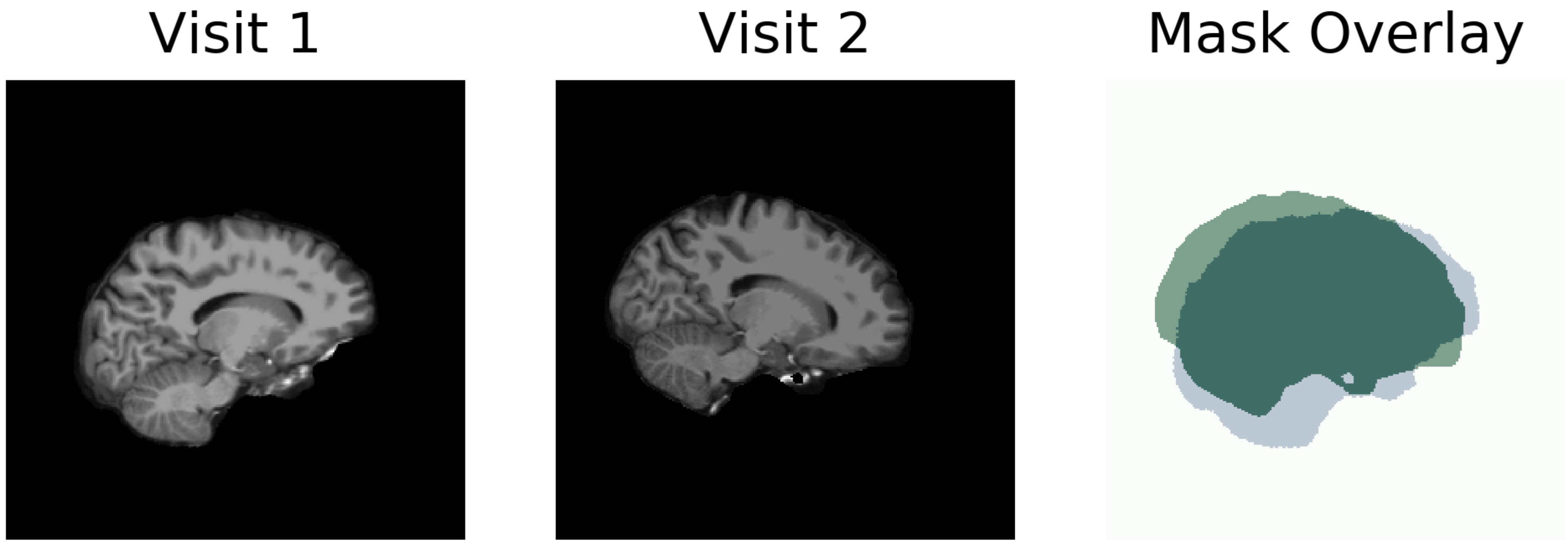
# Comparing images

BIOMEDICAL IMAGE ANALYSIS IN PYTHON

**Stephen Bailey**
Instructor

datacamp

# Comparing images



# Summary metrics

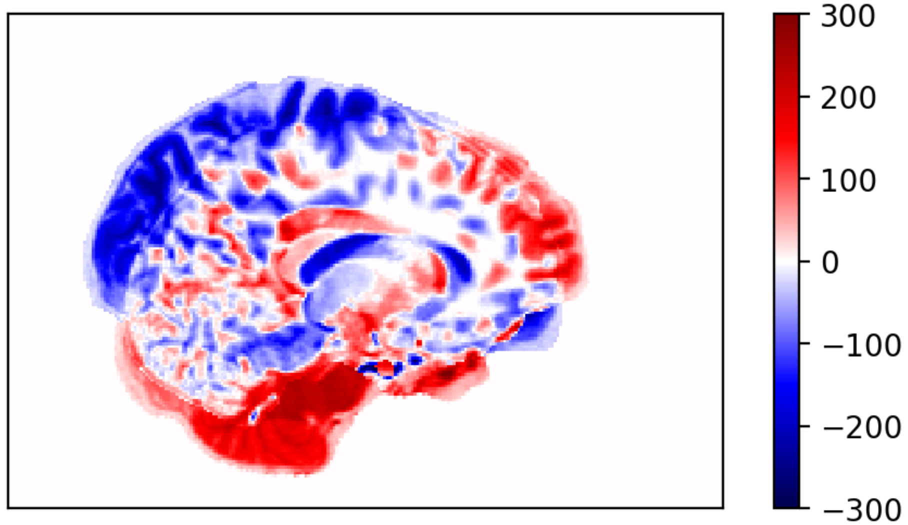
Goal: *define* a metric of similarity between two images.

**Cost functions** produce metrics to be minimized.

**Objective functions** produce metrics to be maximized.

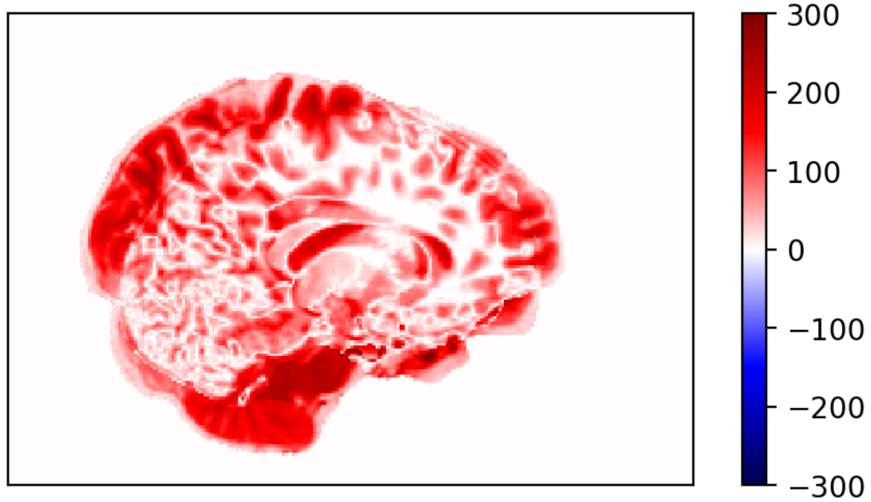
# Mean absolute error

```python
import imageio
import numpy as np
i1=imageio.imread('OAS1035-v1.dcm')
i2=imageio.imread('OAS1035-v2.dcm')
err = i1 - i2
plt.imshow(err)
```




```python
abs_err = np.abs(err)
plt.imshow(abs_err)
mae = np.mean(abs_err)
mae
```

```
29.8570
```

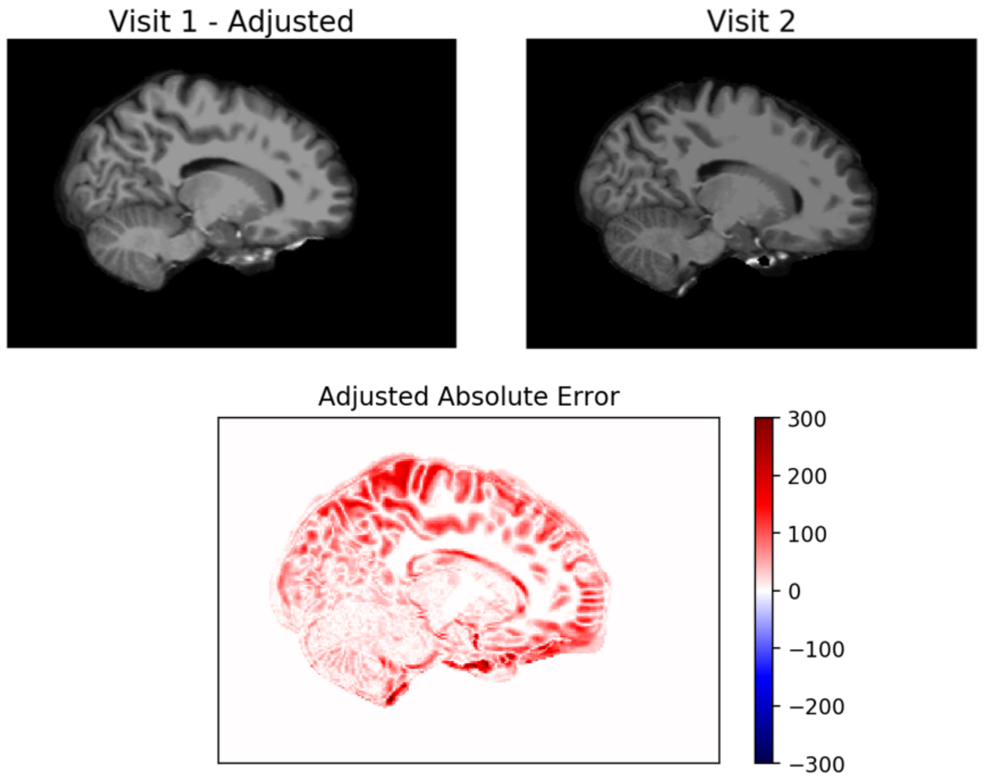


# Mean absolute error

Goal: *minimize* the cost function

```python
# Improve im1 alignment to im2
xfm=ndi.shift(im1, shift=(-8, -8))
xfm=ndi.rotate(xfm, -18,
                  reshape=False)
# Calculate cost
abs_err = np.abs(im1 - im2)
mean_abs_err = np.mean(abs_err)
mean_abs_err
```

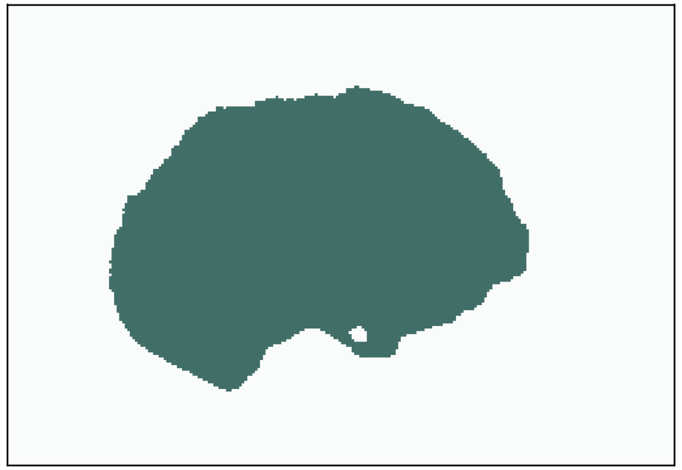
```
13.0376
```



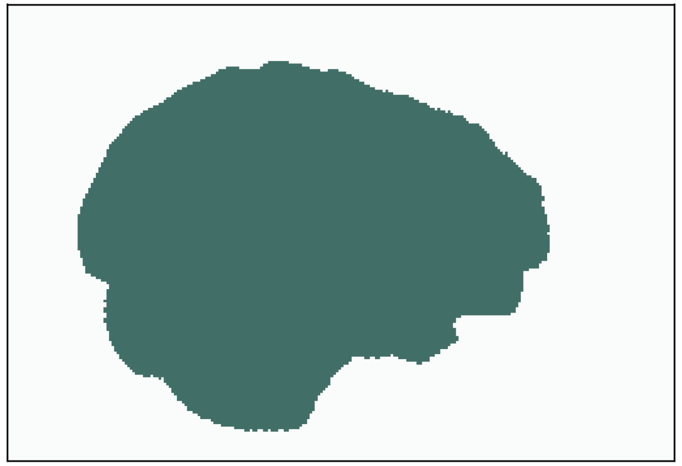
# Intersection of the union

$$IOU = \frac{I_1 \cap I_2}{I_1 \cup I_2}$$

```
mask1 = im1 > 0
mask2 = im2 > 0
intsxn = mask1 & mask2
plt.imshow(intsxn)
```



```
union = mask1 | mask2
plt.imshow(union)
iou = intsxn.sum() / union.sum()
iou
```



```
0.68392
```

# Let's practice!

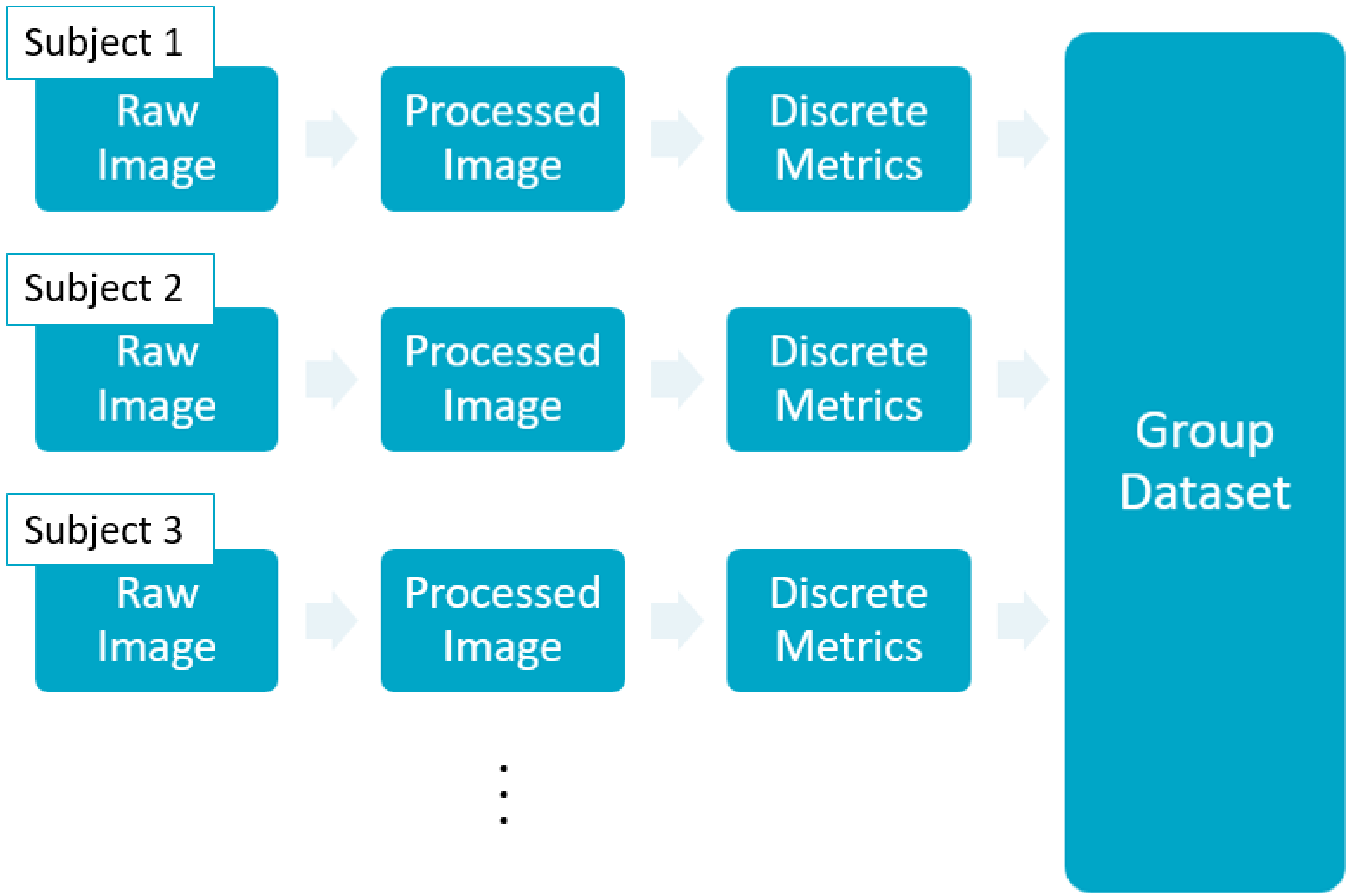
BIOMEDICAL IMAGE ANALYSIS IN PYTHON

# Normalizing measurements

BIOMEDICAL IMAGE ANALYSIS IN PYTHON

**Stephen Bailey**
Instructor

# Analysis workflow



# OASIS population

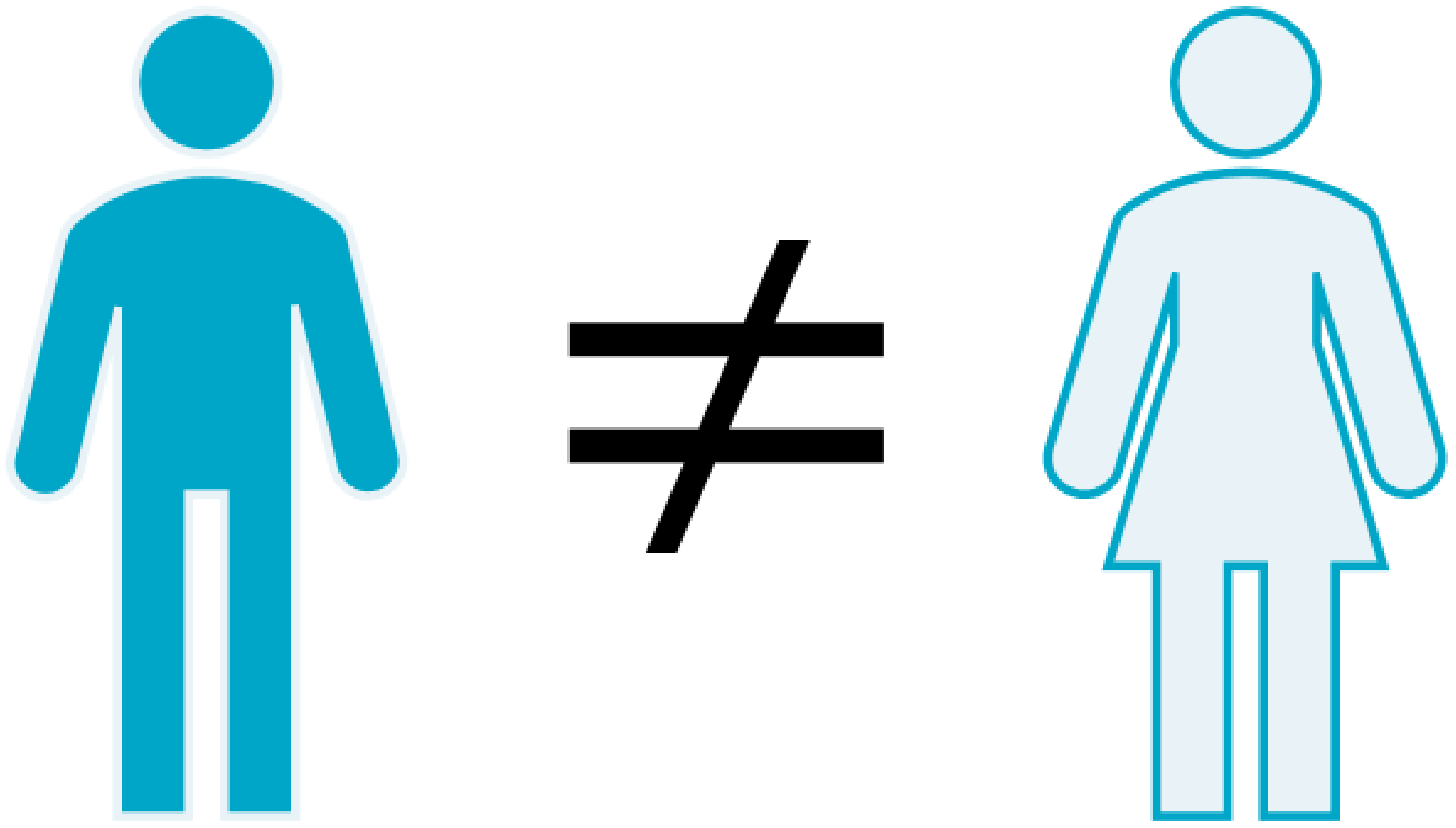
```
df.shape
```

```
(400, 5)
```

```
df.sample(5)
```

```
               age  sex  alzheimers  brain_vol    skull_vol
    ID
    OAS1_0272   75    F        True    851.451  1411.125695
    OAS1_0112   69    F       False    894.801  1434.146892
    OAS1_0213   48    F       False    925.859  1412.781004
    OAS1_0311   22    F       False    980.163  1363.413762
    OAS1_0201   85    F       False    904.104  1420.631447
```

# Hypothesis testing



# Hypothesis testing

**Null hypothesis:** two populations' mean brain volumes ($\mu_m, \mu_w$) are equal.

$$H_{null} : \mu_w = \mu_m$$

$$H_{alt} \ : \mu_w \neq \mu_m$$

$$t = \frac{\bar{X} - \mu}{s/\sqrt{n}}$$

Implemented in `scipy.stats.ttest_ind()`

# Hypothesis testing

```
brain_m = df.loc[df.sex == 'M', 'brain_vol']
brain_f = df.loc[df.sex == 'F', 'brain_vol']
from scipy.stats import ttest_ind
results = ttest_ind(brain_m, brain_f)
```

```
results.statistic
results.pvalue
```

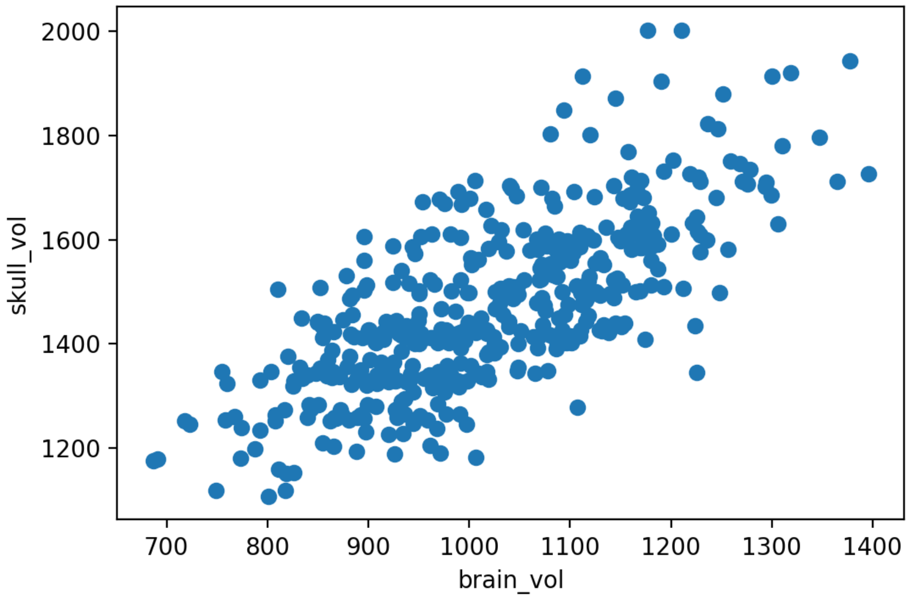
```
10.20986
5.03913e-22
```

A large $t$-statistic and low $p$-value suggests that there is a significant difference!

# Correlated measurements

```
df[['brain_vol', 'skull_vol']].corr()
```

```
               'brain_vol'  'skull_vol'
 'brain_vol'       1.000        0.736
 'skull_vol'       0.736        1.000
```



# Normalization

```
df['brain_norm'] = df.brain_vol / df.skull_vol
brain_norm_m = df.loc[df.sex == 'M', 'brain_norm']
brain_norm_f = df.loc[df.sex == 'F', 'brain_norm']
results = ttest_ind(brain_norm_m, brain_norm_f)
```

```
results.statistic
results.pvalue
```

```
-0.94011
0.34769
```

Size, not gender likely drove original results.

# Many potential confounds in imaging

**Image acquisition**

- Contrast
- Resolution
- Field of view

**Subject / object**

- Age
- Gender
- Pathology

**Context**

- Hospital
- Radiologist
- Equipment

**Data Quality**

- Format
- Artifacts

# Congratulations!

| Exploration | Masks and Filters |
| --- | --- |
| • Loading images<br>• N-D data<br>• Subplots | • Intensity distributions<br>• Convolutions<br>• Edge detection |

| Measurement | Image Comparison |
| --- | --- |
| • Labelling<br>• Multi-object measurement<br>• Morphology | • Transformations<br>• Resampling<br>• Cost functions<br>• Normalization |

# Good luck!

BIOMEDICAL IMAGE ANALYSIS IN PYTHON