



BIOMEDICAL IMAGE ANALYSIS IN PYTHON

Image Data

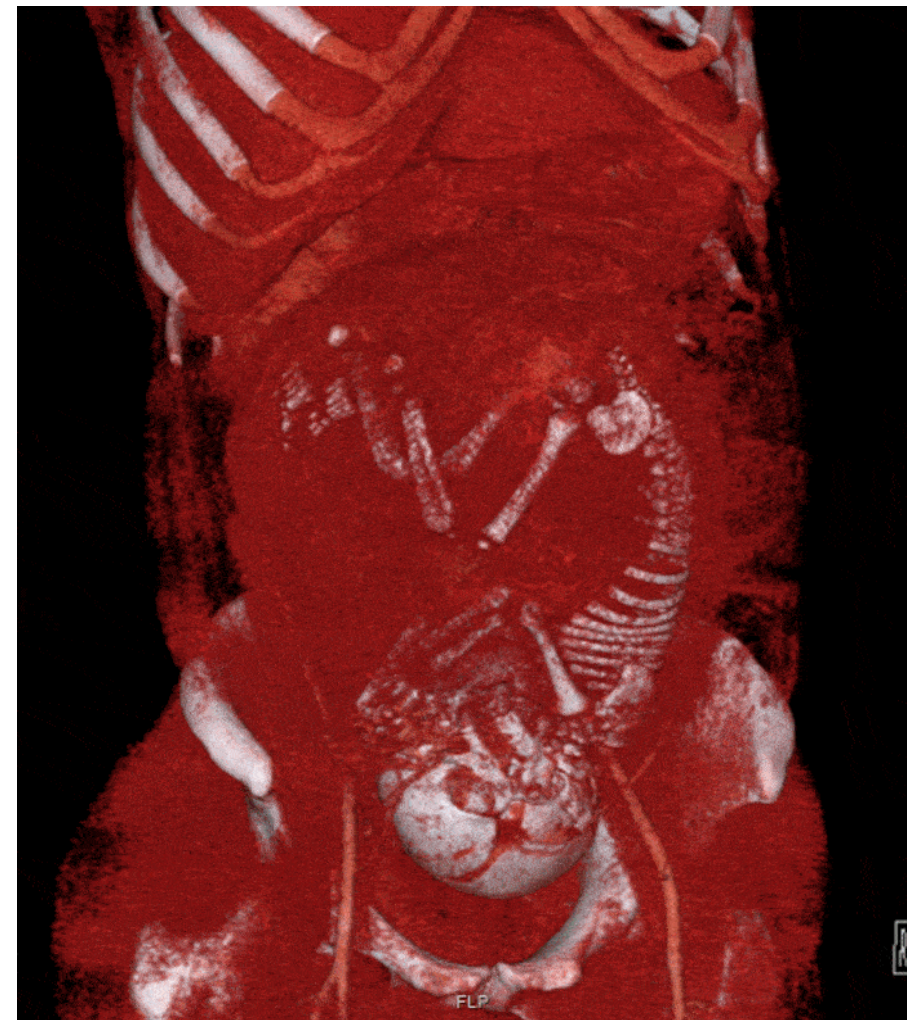
Stephen Bailey
Instructor

Biomedical imaging: more than a century of discovery

1895



2017





Course objectives

Exploration

- Loading images
- N-D data
- Subplots

Masks and Filters

- Intensity distributions
- Convolutions
- Edge detection

Measurement

- Labelling
- Multi-object measurement
- Morphology

Image Comparison

- Transformations
- Resampling
- Cost functions
- Normalization

Toolbox

- ImageIO
- NumPy
- SciPy
- matplotlib

Loading images

- `imageio`: read and save images
- Image objects are NumPy arrays.
- Slice the array by specifying values along each available dimension.

```
import imageio

im = imageio.imread('body-001.dcm')
```

```
type(im)
imageio.core.Image

im
Image([[125, 135, ..., 110],
       [100, 130, ..., 100],
       ...,
       [100, 150, ..., 100]],
      dtype=uint8)
```

```
im[0, 0]
125

im[0:2, 0:2]
Image([[125, 135],
       [100, 130]],
      dtype=uint8)
```

Metadata

- **Metadata:** the who, what, when, where and how of image acquisition
- Accessible in Image objects through the meta dictionary attribute

```
im.meta
```

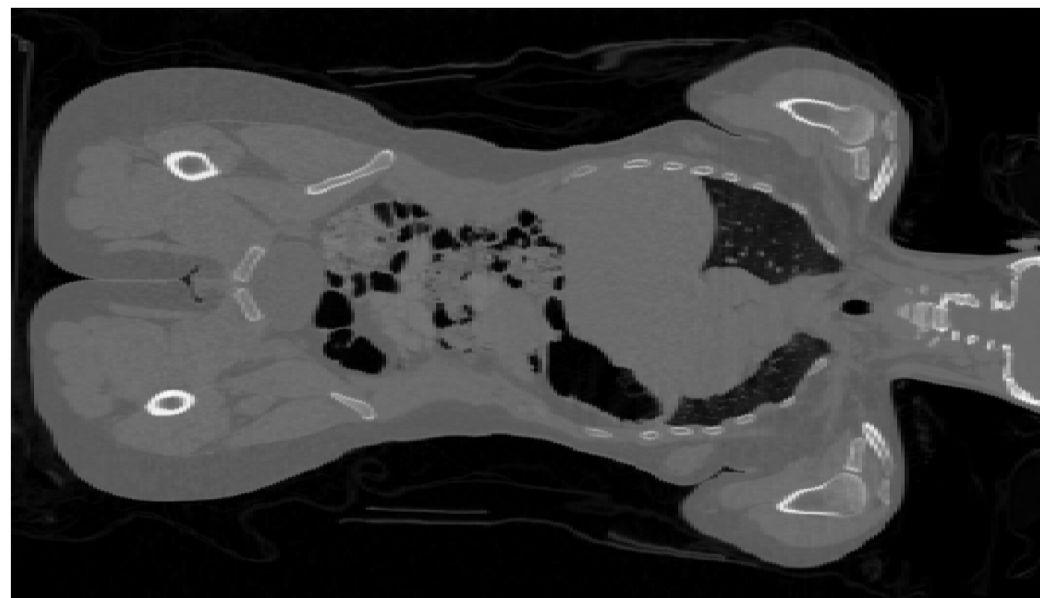
```
Dict([('StudyDate', '2017-01-01'),  
      ('Modality', 'MR'),  
      ('PatientSex', F),  
      ...  
      ('shape', (256, 256))])
```

```
im.meta['Modality']  
      'CT'  
  
im.meta.keys()  
      odict_keys(['StudyDate',  
                  'SeriesDate',  
                  'PatientSex',  
                  ...  
                  'shape'])
```

Plotting images

- Matplotlib's `imshow()` function displays 2D image data
- Many colormaps available but often shown in grayscale (`cmap='gray'`)
- Axis ticks and labels are often **not** useful for images

```
import matplotlib.pyplot as plt
plt.imshow(im, cmap='gray')
plt.axis('off')
plt.show()
```





BIOMEDICAL IMAGE ANALYSIS IN PYTHON

Let's practice!



BIOMEDICAL IMAGE ANALYSIS IN PYTHON

N-dimensional images

Stephen Bailey
Instructor

Images of all shapes and sizes

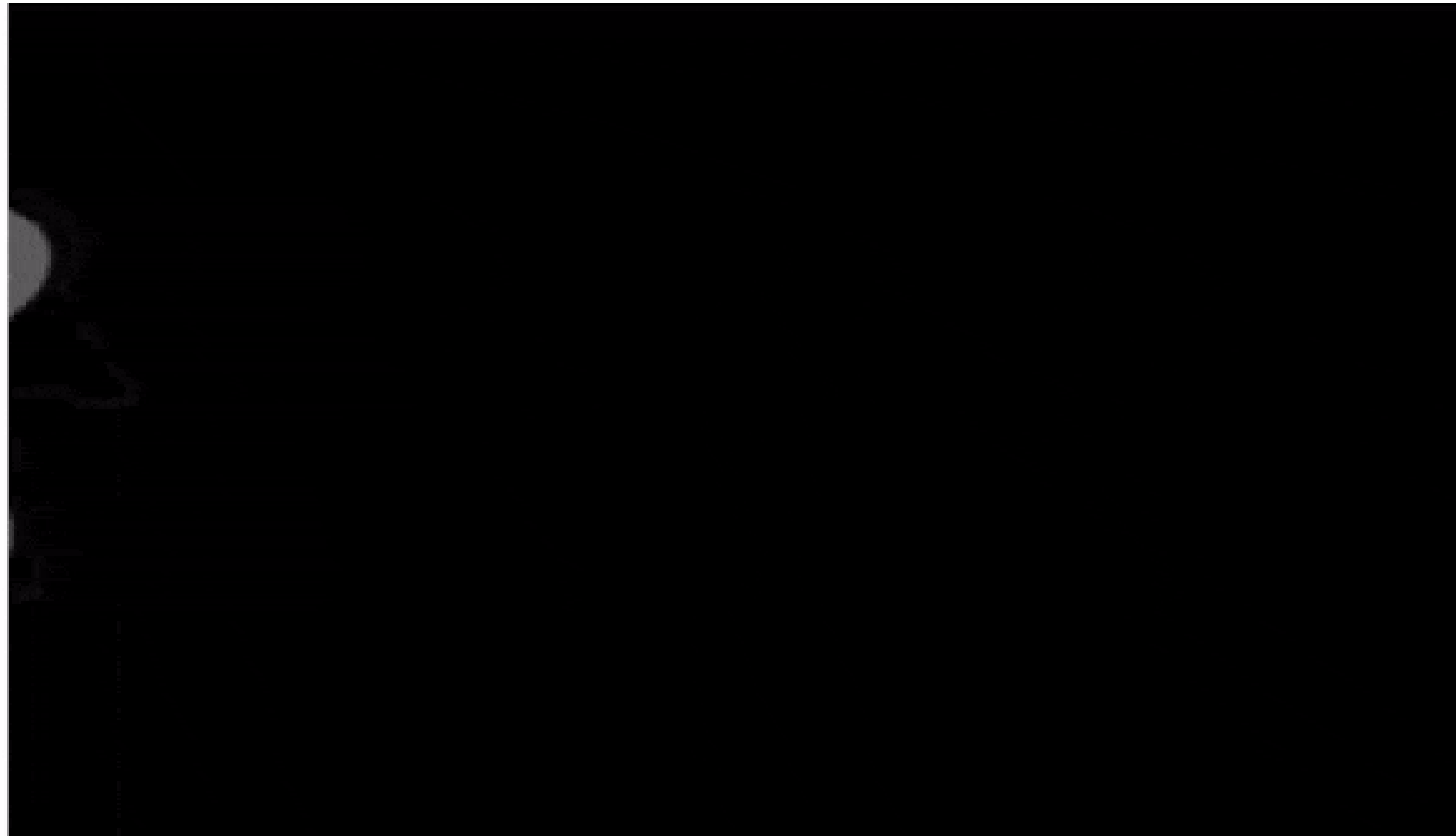
```
im[row, col]
```





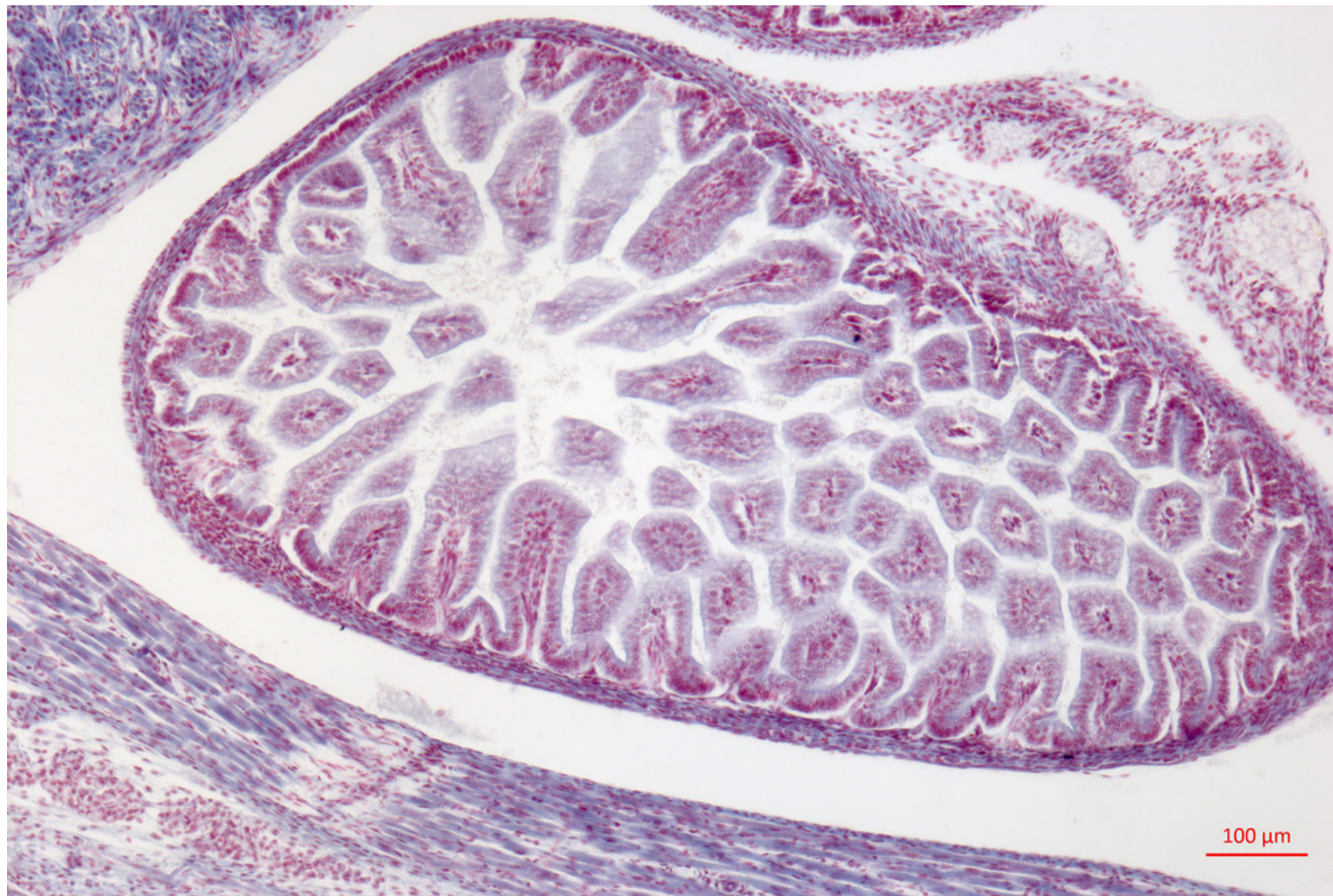
Images of all shapes and sizes

```
vol[pln, row, col]
```



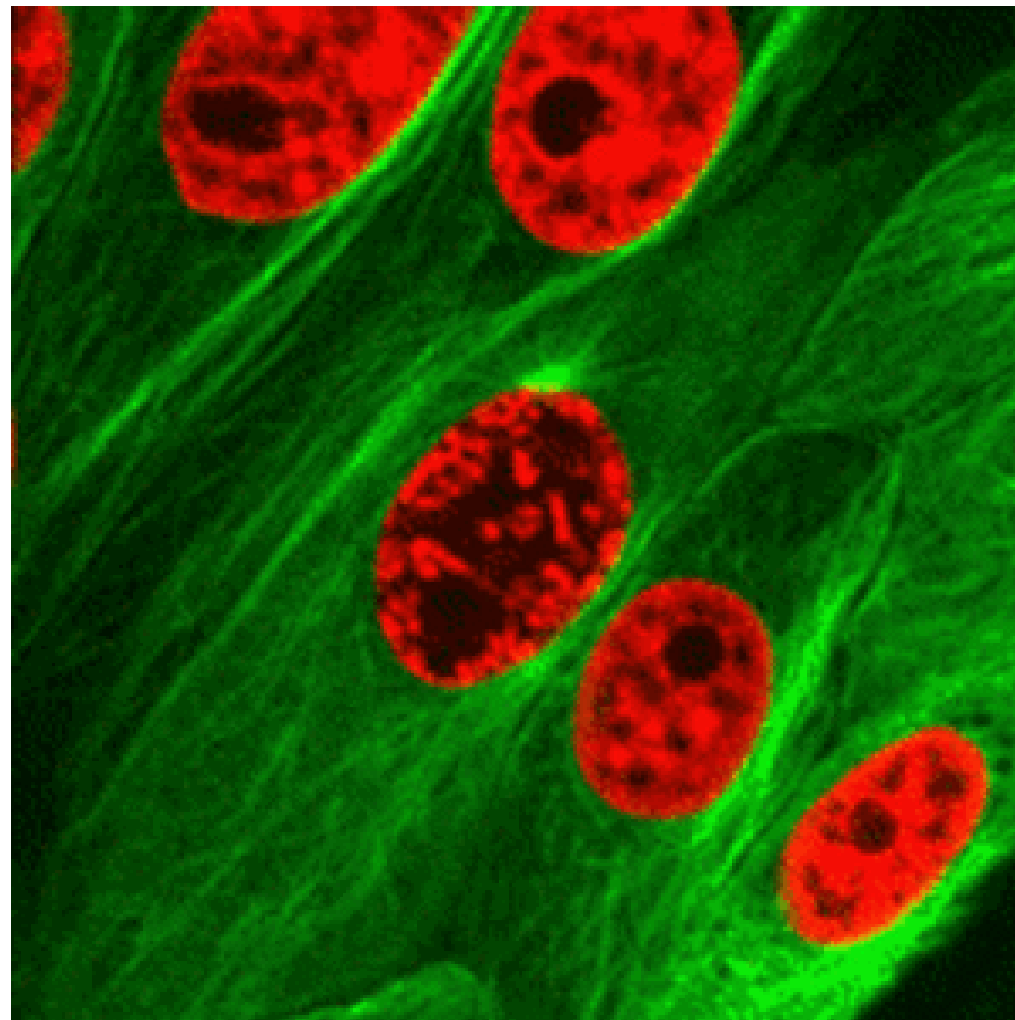
Images of all shapes and sizes

```
im[row, col, ch]
```



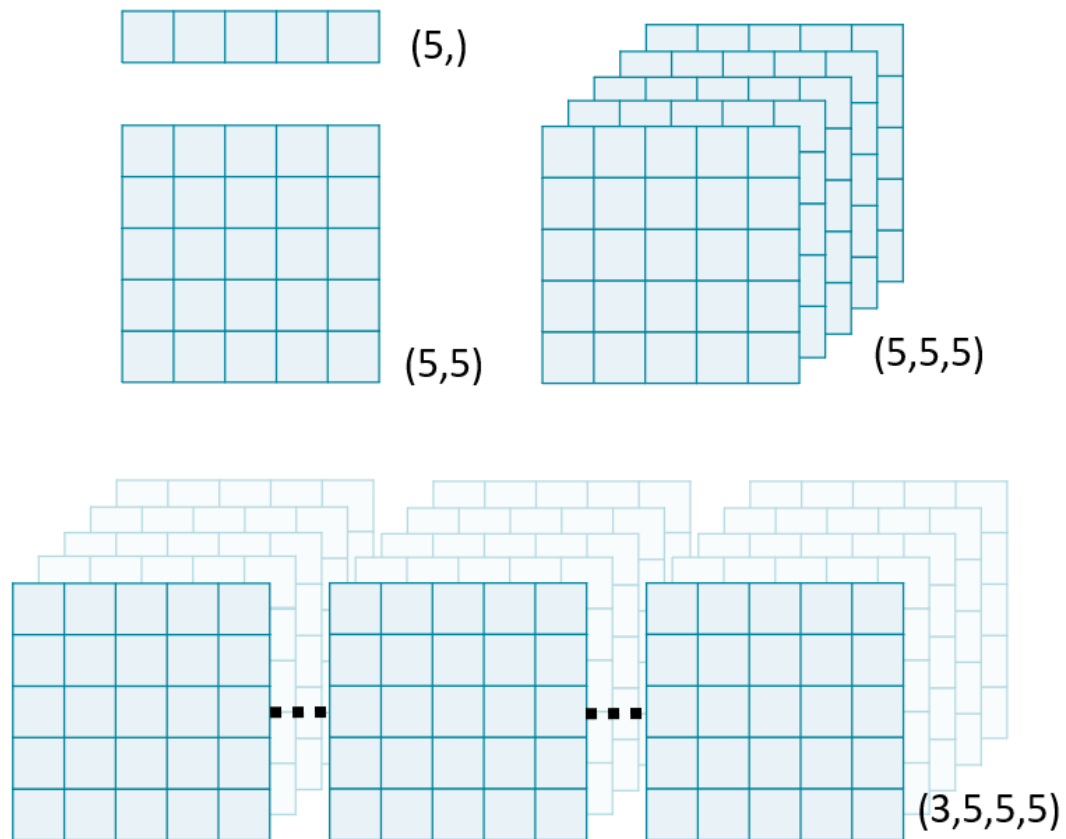
Images of all shapes and sizes

```
im_ts[time, row, col, ch]
```





N-dimensional images are stacks of arrays



```
import imageio
import numpy as np

im1=imageio.imread('chest-000.dcm')
im2=imageio.imread('chest-001.dcm')
im3=imageio.imread('chest-002.dcm')

im1.shape
(512, 512)

vol = np.stack([im1, im2, im3])

vol.shape
(3, 512, 512)
```



Loading volumes directly

`imageio.volread()`:

- read multi-dimensional data directly
- assemble a volume from multiple images

```
import os

os.listdir('chest-data')
['chest-000.dcm',
 'chest-001.dcm',
 'chest-002.dcm',
 ...,
 'chest-049.dcm']
```

```
import imageio

vol = imageio.volread('chest-data')

vol.shape
(50, 512, 512)
```


Shape, sampling, and field of view

- **Image shape:** number of elements along each axis
- **Sampling rate:** physical space covered by each element
- **Field of view:** physical space covered along each axis

```
import imageio

vol = imageio.volread('chest-data')

# Image shape (in voxels)
n0, n1, n2 = vol.shape
n0, n1, n2
(50, 512, 512)
```

```
# Sampling rate (in mm)
d0, d1, d2 = vol.meta['sampling']
d0, d1, d2
(2, 0.5, 0.5)
```

```
# Field of view (in mm)
n0 * d0, n1 * d1, n2 * d2
(100, 256, 256)
```



BIOMEDICAL IMAGE ANALYSIS IN PYTHON

Let's practice!



BIOMEDICAL IMAGE ANALYSIS IN PYTHON

Advanced plotting

Stephen Bailey
Instructor

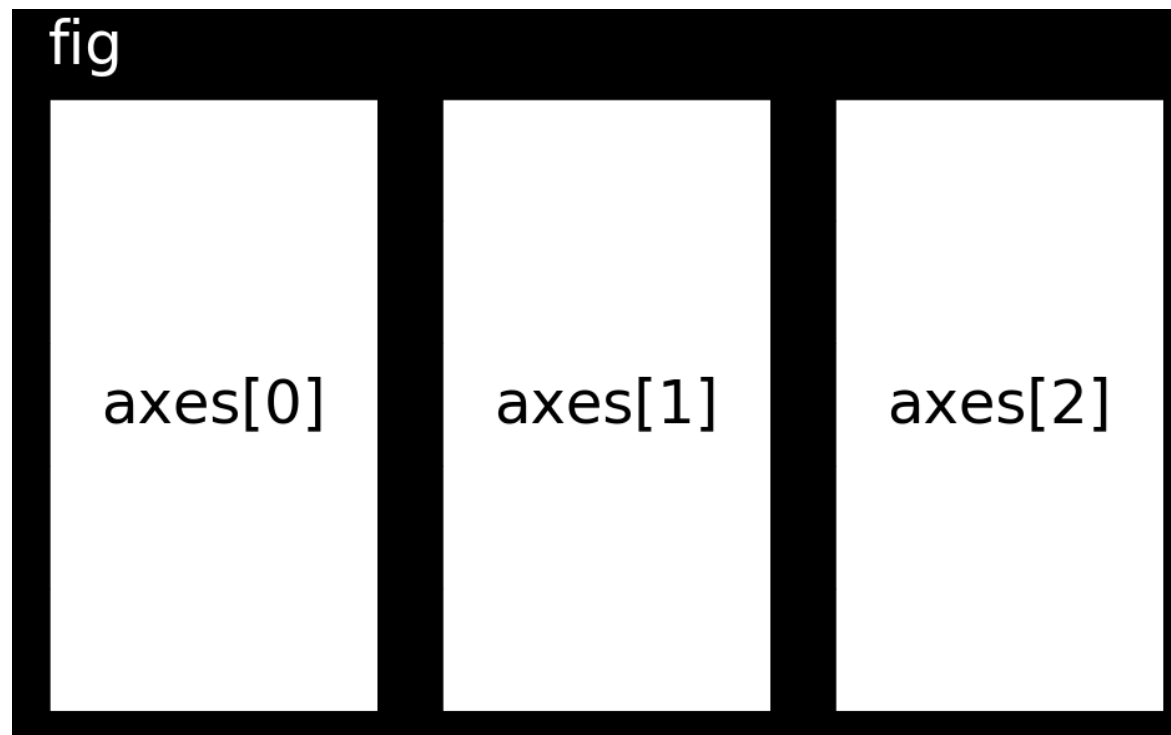
To plot N-dimensional data slice it!





Plotting multiple images at once

`plt.subplots()`: creates a figure canvas with multiple `AxesSubplots` objects.



```
import imageio

vol = imageio.volread('chest-data')

fig, axes = plt.subplots(nrows=1,
                        ncols=3)

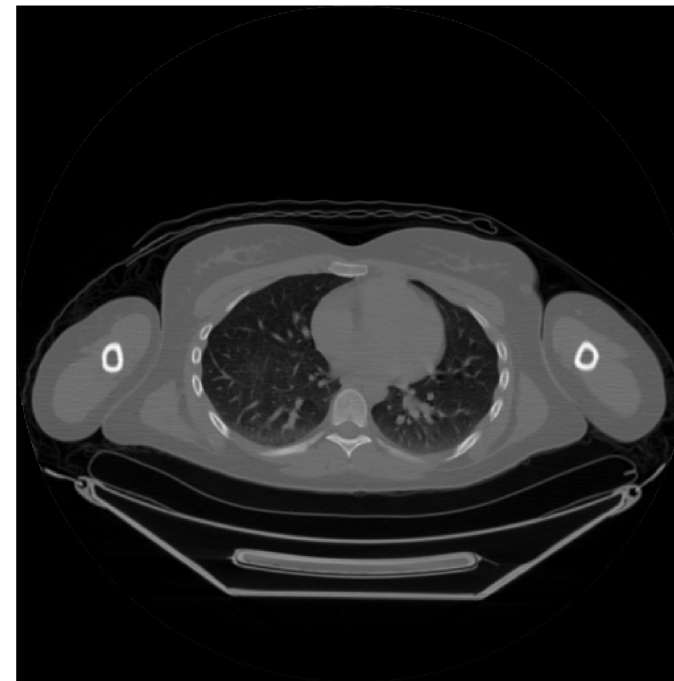
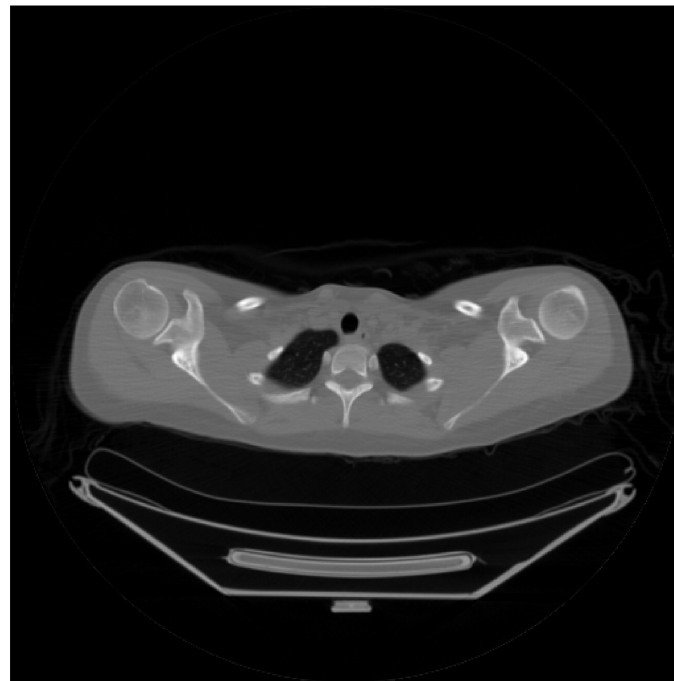
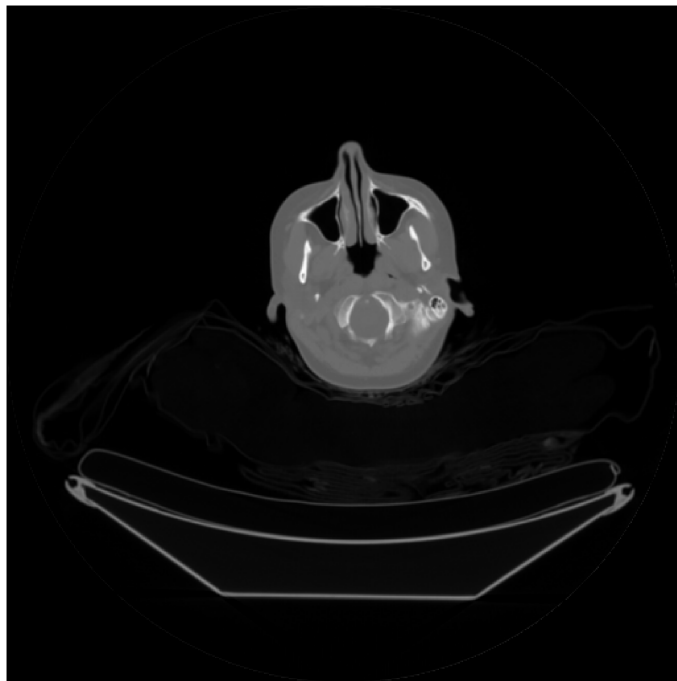
axes[0].imshow(vol[0], cmap='gray')

axes[1].imshow(vol[10], cmap='gray')
axes[2].imshow(vol[20], cmap='gray')

for ax in axes:
    ax.axis('off')

plt.show()
```

Plotting multiple images at once



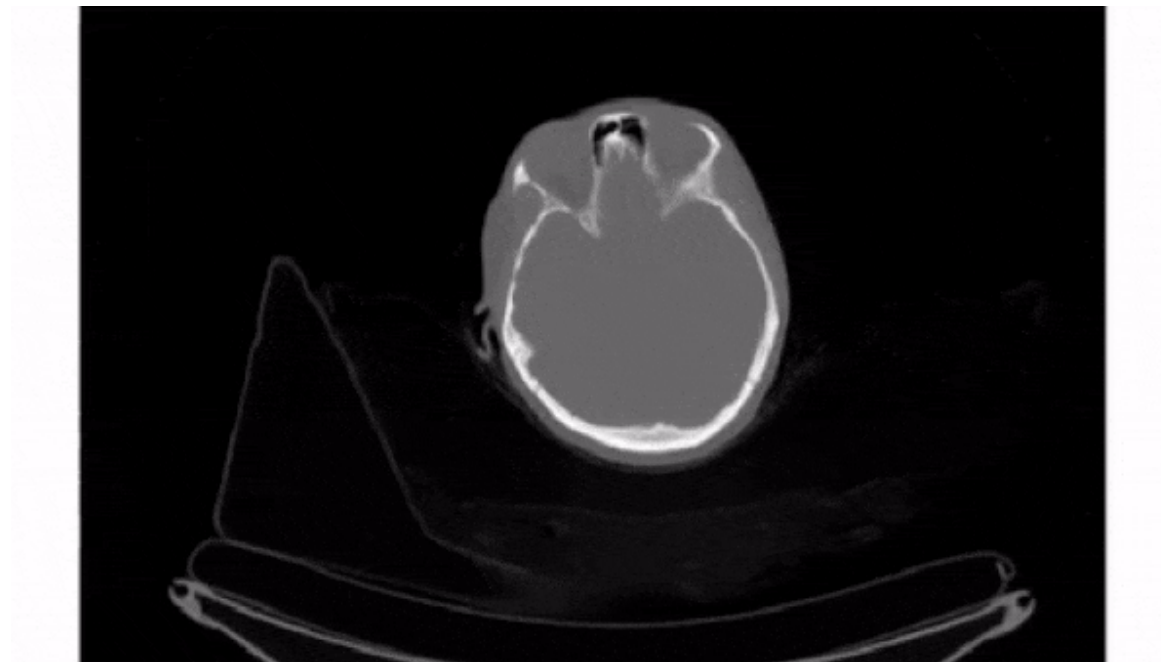
Non-standard views

```
import imageio

vol = imageio.volread('chest-data')

view_1v2 = vol[pln, :, :]
view_1v2 = vol[pln]
```

Axial





Non-standard views

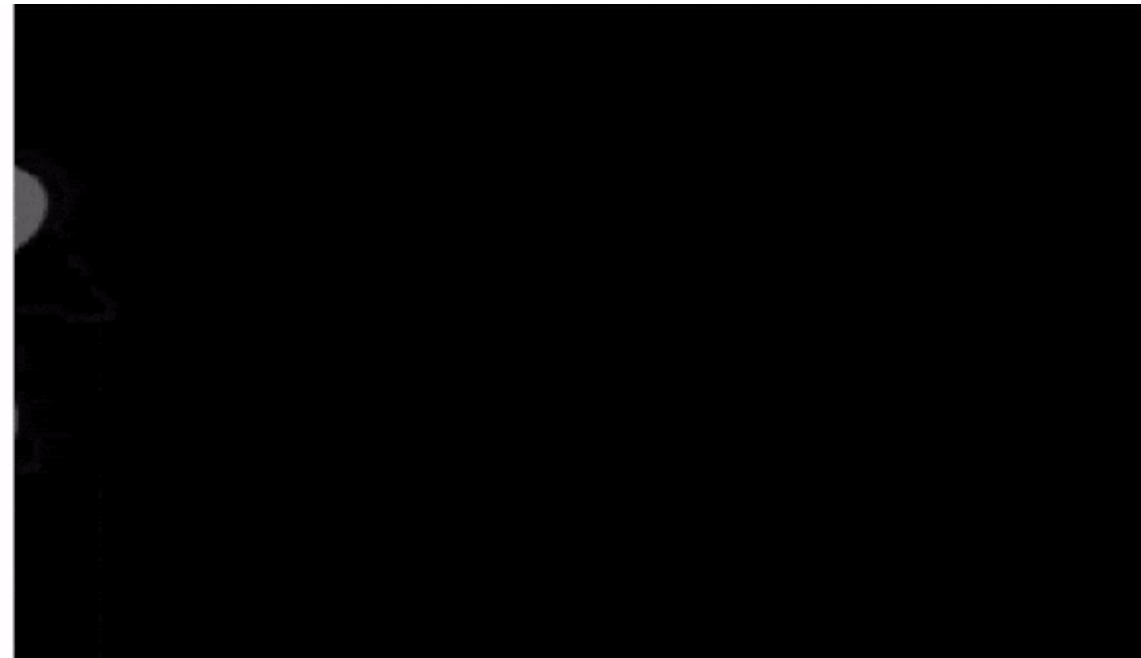
```
import imageio

vol = imageio.volread('chest-data')

view_1v2 = vol[pln, :, :]
view_1v2 = view_1v2[pln]

view_0v2 = vol[:, row, :]
```

Coronal





Non-standard views

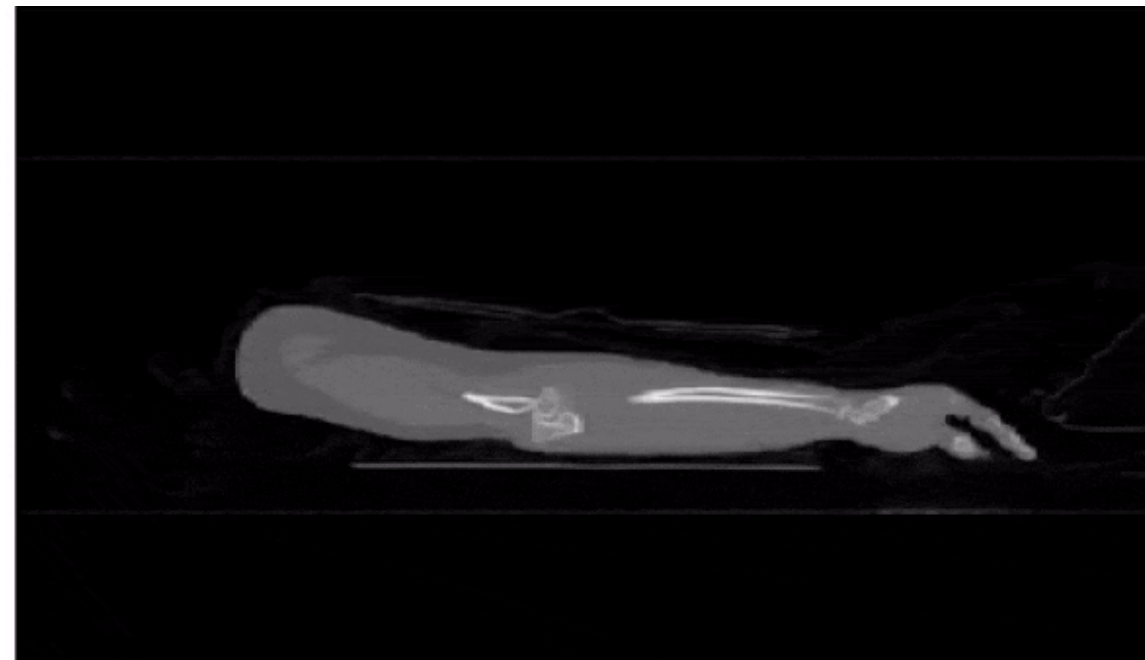
```
import imageio

vol = imageio.volread('chest-data')

view_1v2 = vol[pln, :, :]
view_1v2 = view_1v2[pln]

view_0v2 = vol[:, row, :]
view_0v1 = vol[:, :, col]
```

Sagittal





Modifying the aspect ratio

Pixels may adopt any aspect ratio:

4:1

16:9

1:1

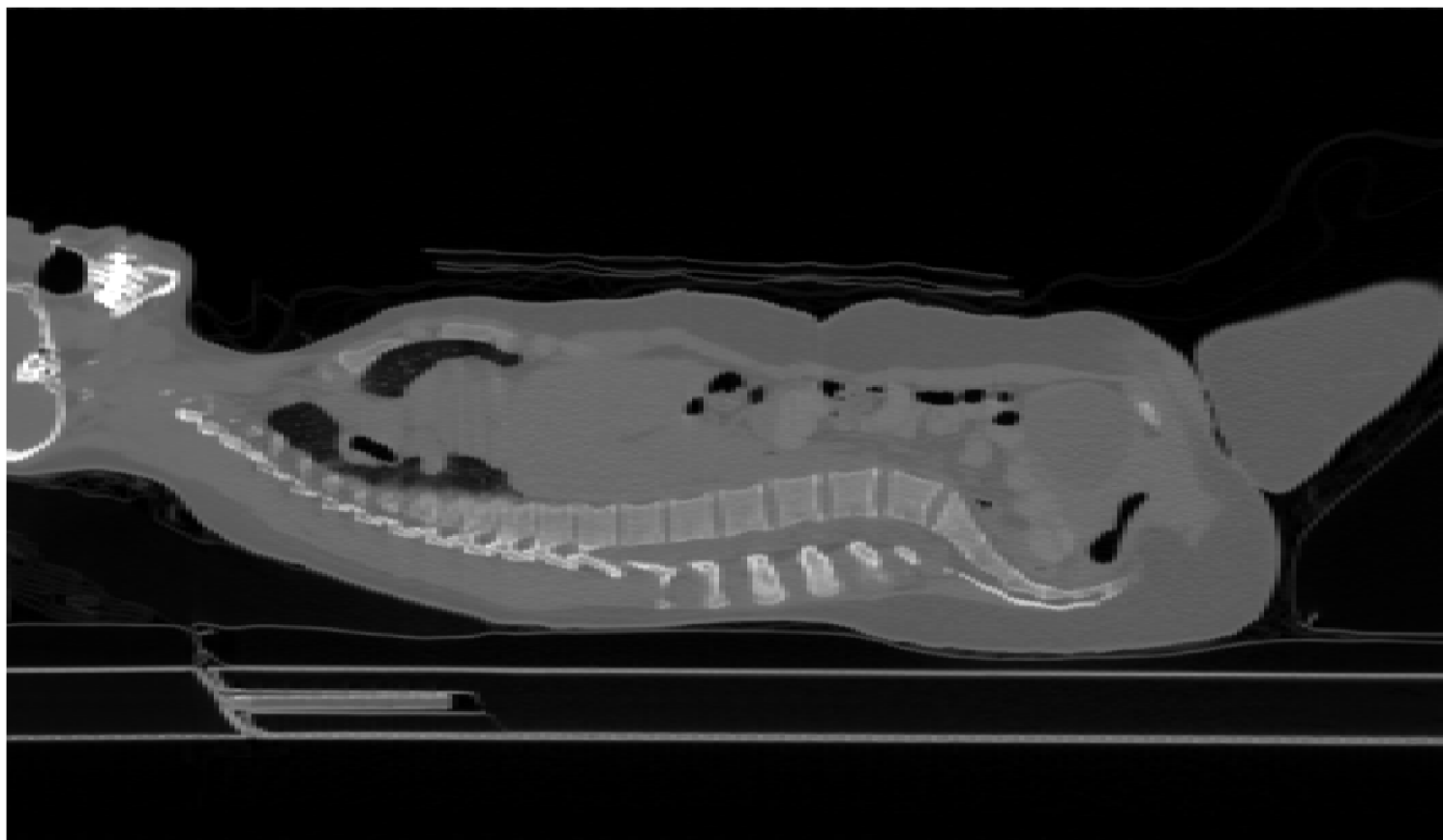
```
im = vol[:, :, 100]

d0, d1, d2 = vol.meta['sampling']
d0, d1, d2
(2, 0.5, 0.5)

asp = d0 / d1
asp
3

plt.imshow(im, cmap='gray',
            aspect=asp)
plt.show()
```


Modifying the aspect ratio





BIOMEDICAL IMAGE ANALYSIS IN PYTHON

Let's practice!