0. Relevant External Libraries

Library/ Module	Description	Import Syntax
tifffile	Reads & writes tiff files	`import tifffile`
ndv	View images interactively	`import ndv`
matplotlib.pyplot	Module within matplotlib for plotting	`import matplotlib.pyplot`
numpy	N-dim array computing	`import numpy`
bffile	Reads proprietary file types	`import bffile`

```
# importing a library w/ alias
import tifffile as tiff

# importing portion of a library
import matplotlib.pyplot
from matplotlib import pyplot

# combine portion + alias
import matplotlib.pyplot as plt
```

1. Reading .tif Image w/ `tifffile`

Use **`tifffile`** to read .tif images and store as **`numpy`** array

```
import tifffile as tiff

image_path = r"/path/to/image.tif"

image = tiff.imread(image_path)
```

2. Image Viewing w/ `ndv`

View images in an interactive viewer

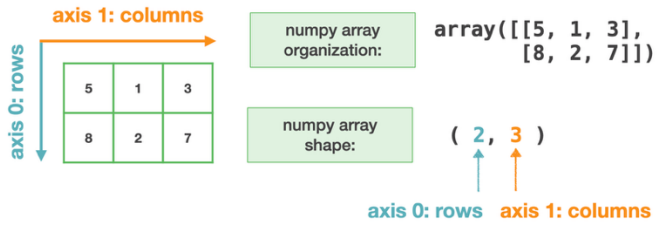
```
import ndv
ndv.imshow(image)
```

3. Image Viewing w/ `matplotlib`

View images statically as a plot

```
import matplotlib.pyplot as plt

plt.imshow(image)
```

4. numpy Arrays: 2D Images



Working with 2D numpy arrays

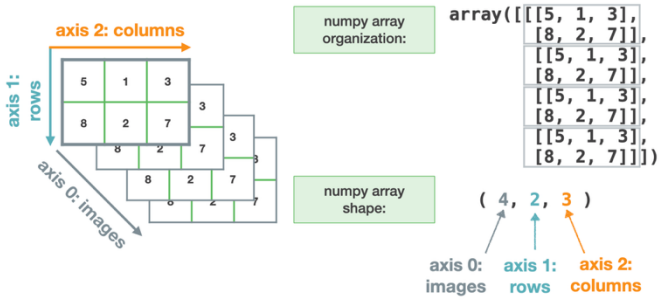
Task	Syntax
Find shape	`image.shape`
Find bit depth (dtype)	`image.dtype`
Confirm type	`type(image)`
Access individual pixel values with indices	`image[0,0]`
Access an entire row of pixel values w/indices	`image[0,:]` or `image[0]`
Access an entire column of pixel values w/indices	`image[:,0]`

Accessing pixel statistics across the array

Statistic	Syntax
Minimum	`image.min()`
Maximum	`image.max()`
Mean	`image.mean()`

5. numpy Arrays: Multi-D Images

Examples of multi-dimensional images:
z-stacks, time lapses, multiple fluorescence channels, etc.



Working with multi-d numpy arrays

Task	Syntax
Find shape	`image.shape`
Find bit depth	`image.dtype`
Confirm type	`type(image)`

Access individual images from 3D array w/indices	`image[0,:,:]` or `image[0]`

Image projections

Projection Type	Syntax
Max Intensity Projection	`numpy.max(image, axis=0)`
Min Intensity Projection	`numpy.min(image, axis=0)`
Sum Projection	`numpy.sum(image, axis=0)`
Avg Intensity Projection	`numpy.mean(image, axis=0)`
Stdev Projection	`numpy.std(image, axis=0)`
Variance Projection	`numpy.var(image, axis=0)`

6. Image Saving w/ `tifffile`

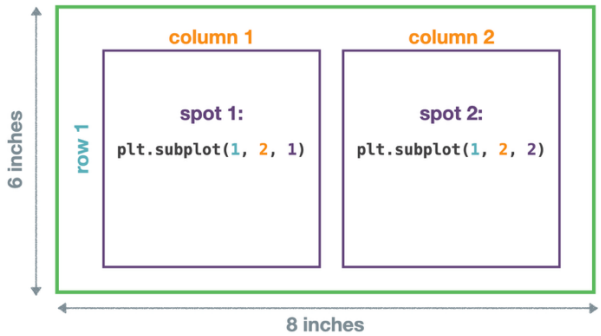
Use **`tifffile`** to save .tif images

```
import tifffile as tiff

output_path =
     "/path/to/image_name.tif"

tiff.imwrite(output_path,
          data=image)
```

7. Visualizing Multiple Images

matplotlib.pyplot figures & subplots

`plt.figure(figsize=(8,6))`




```
import matplotlib.pyplot as plt
plt.figure(figsize=(8,6))
plt.subplot(1, 2, 1)
plt.imshow(img_1)
plt.subplot(1, 2, 2)
plt.imshow(img_2)
plt.show()
```

Miscellaneous plot settings

Setting	Syntax
Remove axis labels	`plt.axis("off")`
Changing colormap	`plt.imshow(img_1, cmap="gray")`
Adjusting B&C	`plt.imshow(img_1, vmin=img_1.min(), vmin=img_1.max())`
Fitting subplots in figure	`plt.figure(figsize=(8,6), layout="constrained")`

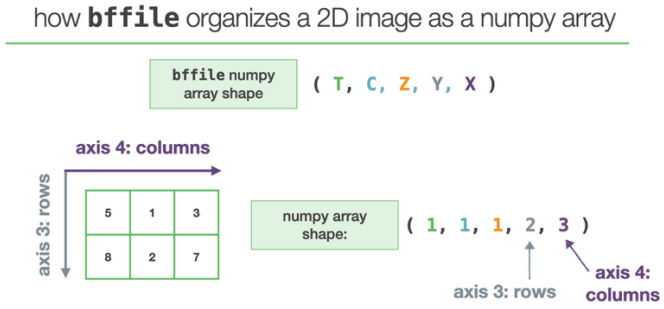
ndv display options

Setting	Syntax
View composite	`ndv.imshow(img, channel_mode="composite")`
Specify LUTs	`ndv.imshow(img, luts={0:{"cmap":"green", 1:{"cmap":"magenta"}})`

8. Reading Proprietary File Types w/ `bffile`

Use **`bffile`** to read proprietary image file types (e.g., .nd2, .czi, .lif, etc)

```
import bffile

image_path = "/path/to/image.czi"

image = bffile.imread(image_path)
```



Use **`numpy.squeeze()`** to drop axes with length 1 from a **`numpy`** array

Strategy	Syntax
Drop single axis w/ len 1	`np.squeeze(image, axis = 0)`
Drop any axes w/ len 1	`np.squeeze(image)`