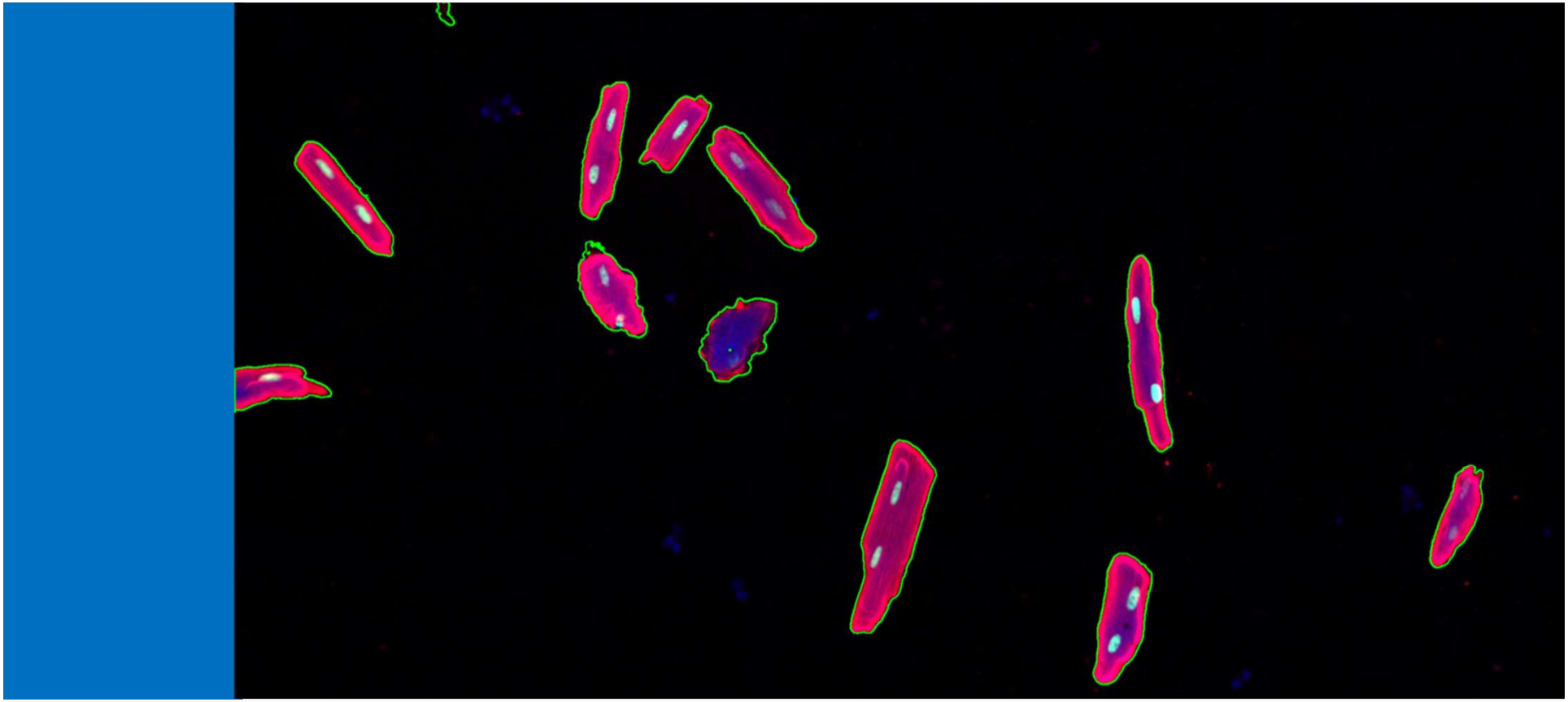


Week 4: Segmentation

MCDB-BCHM 4312-5312

Before we start…

Download the following image from Canvas:

`l4_cardio.tif`

- For MATLAB homework:

Look at the answer key for Problem Set 3 on Canvas to see an example of what to turn in

Learning goals

- Image intensity histograms

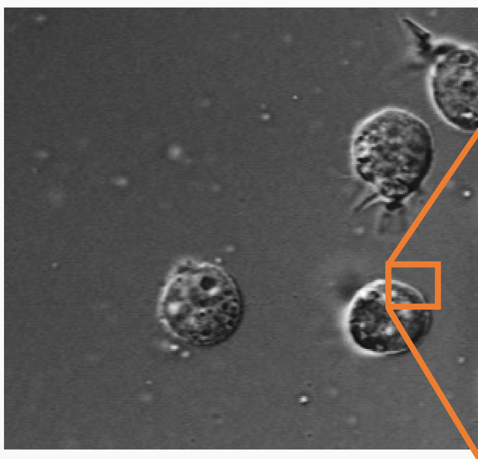
- Segmentation by thresholding
  - Otsu's Method

- Measuring data from masks using `regionprops`

Pixel values

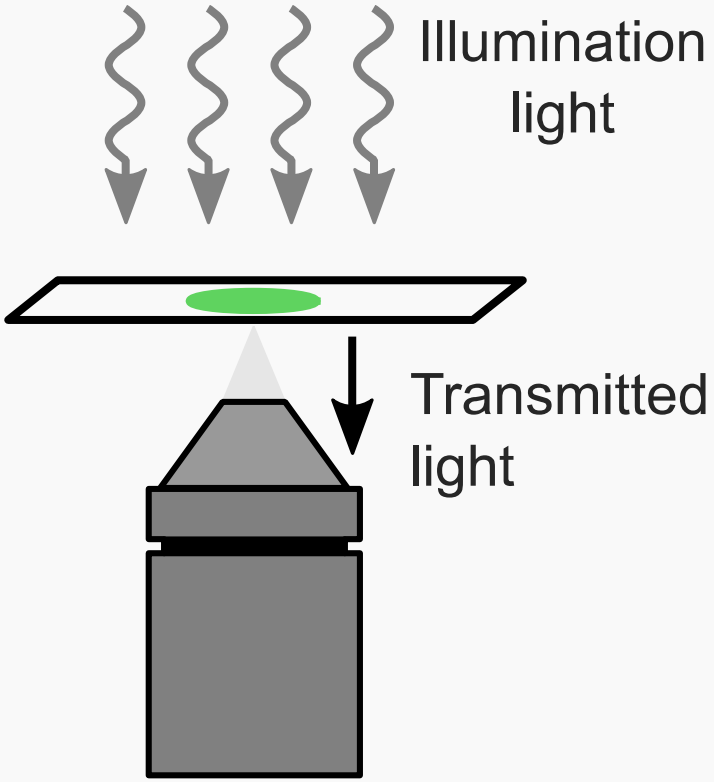
= values in the image matrix

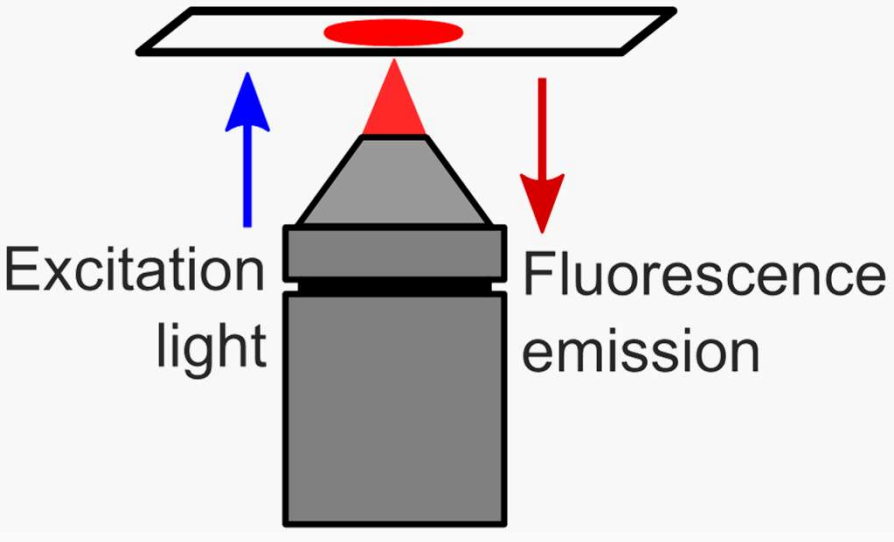
= amount of light recorded by microscope camera

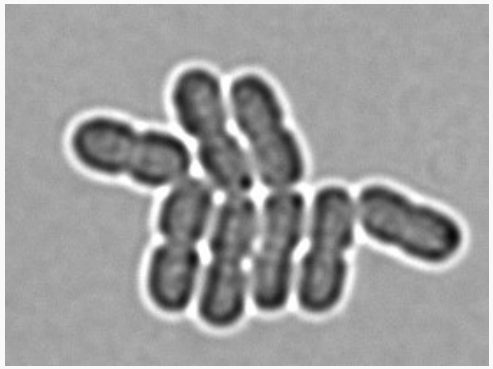


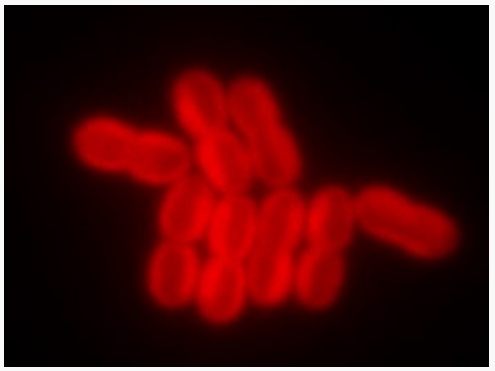
0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	256	0	0
0	0	0	0	0	0	256	0	0	0
0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0
256	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0
256	256	0	0	0	0	0	0	0	0
256	256	256	0	0	0	0	0	0	0
256	256	256	256	256	256	0	0	0	0
256	512	256	512	512	256	256	0	0	0
256	256	256	256	256	512	512	256	0	0
256	512	256	256	512	256	256	256	256	0
512	512	512	512	512	256	512	512	512	256
512	256	512	512	512	512	256	512	256	512
256	512	512	512	256	512	256	512	256	256
256	512	512	512	256	256	256	512	512	512
768	512	512	768	512	512	256	256	256	512
256	512	256	512	512	256	512	512	256	256
256	512	512	512	512	512	512	512	512	512
512	512	256	512	512	256	512	512	256	512
512	256	512	512	256	256	512	256	256	256
768	512	512	512	512	512	256	512	512	256
512	256	512	256	512	256	256	512	512	512

Brightfield vs Fluorescence images



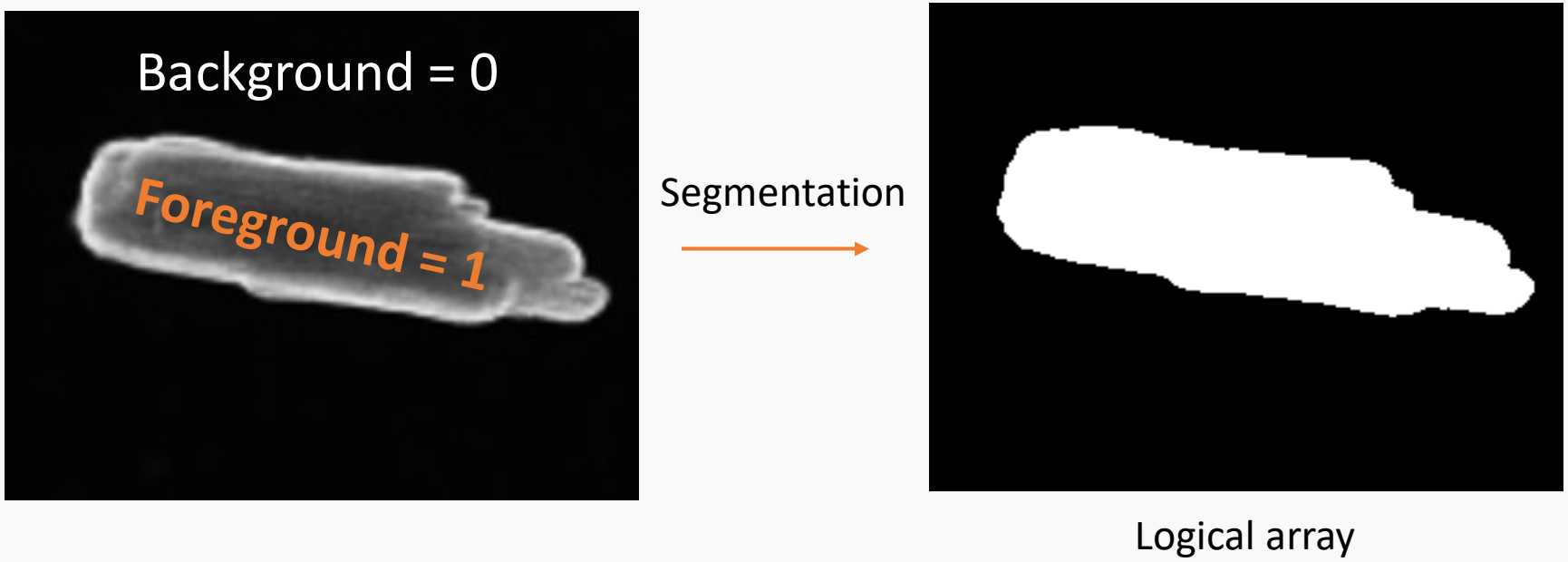






What is segmentation

Segmentation is the labeling of pixels between objects of interest (foreground) and background



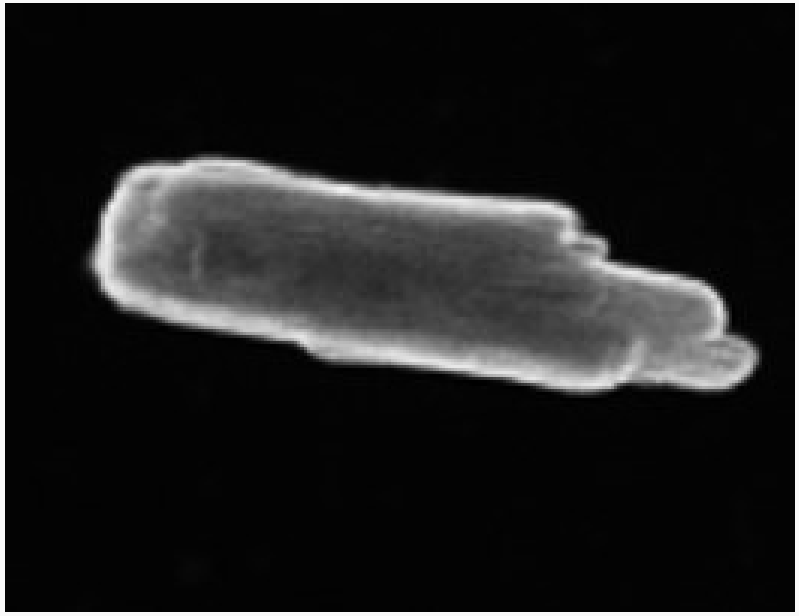
Segmentation is not a "solved problem"

- Cell segmentation is a difficult problem because it relies on many imaging factors:
  - How bright are the cells
  - How dark is the background
  - How many cells are in the image
  - …etc.
- There is no "universal segmentation algorithm"
- Try different techniques to see which works best

Segmentation by intensity threshold

- Assumption: The cell is brighter than the background

- We can choose a **threshold intensity** – every pixel that is brighter than the threshold must belong to the cell



Question

1. Read and display the image `'l4_cardio.tif'`

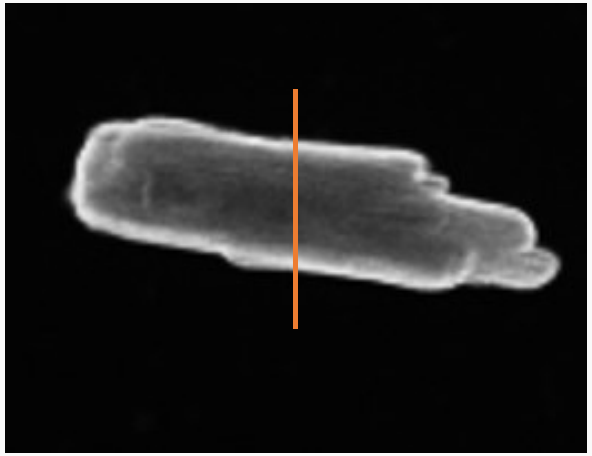
2. What pixel value should we choose as the intensity?

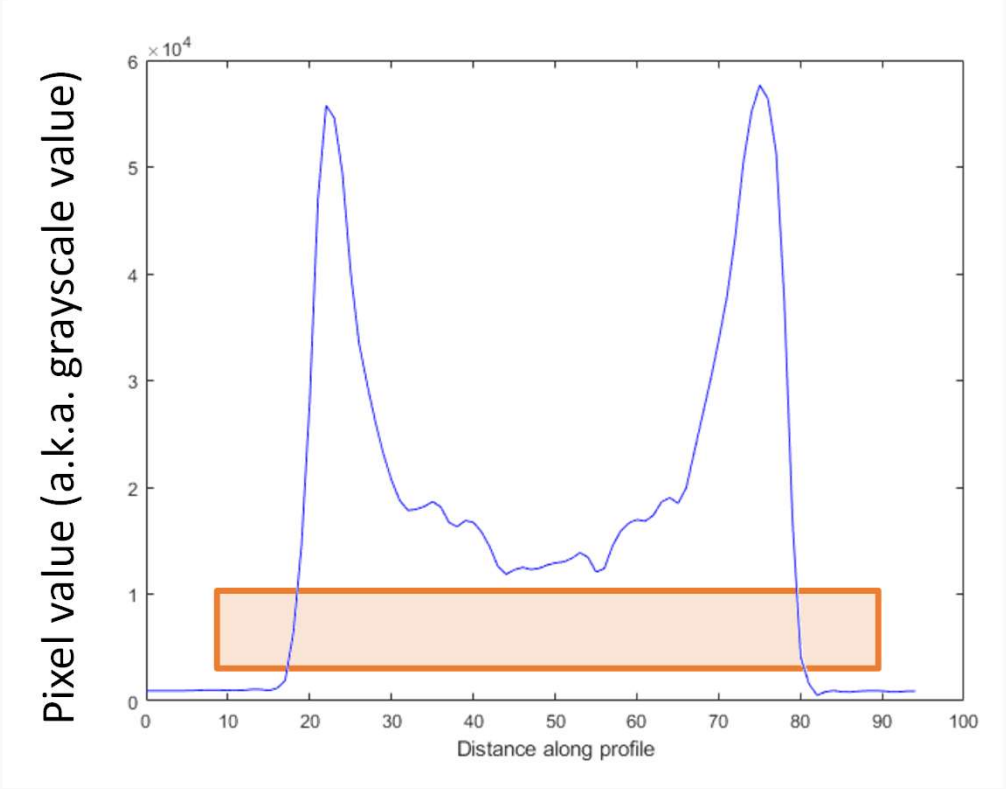
Intensity profiles (line profiles)

`improfile`

- Click to select end points
- Press enter when done
- A new figure window will appear to show profile along selected line

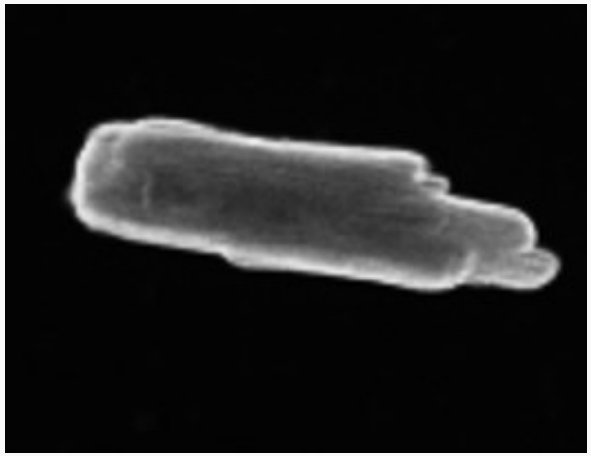
Intensity profiles (line profiles)





What value would you select as the threshold intensity?

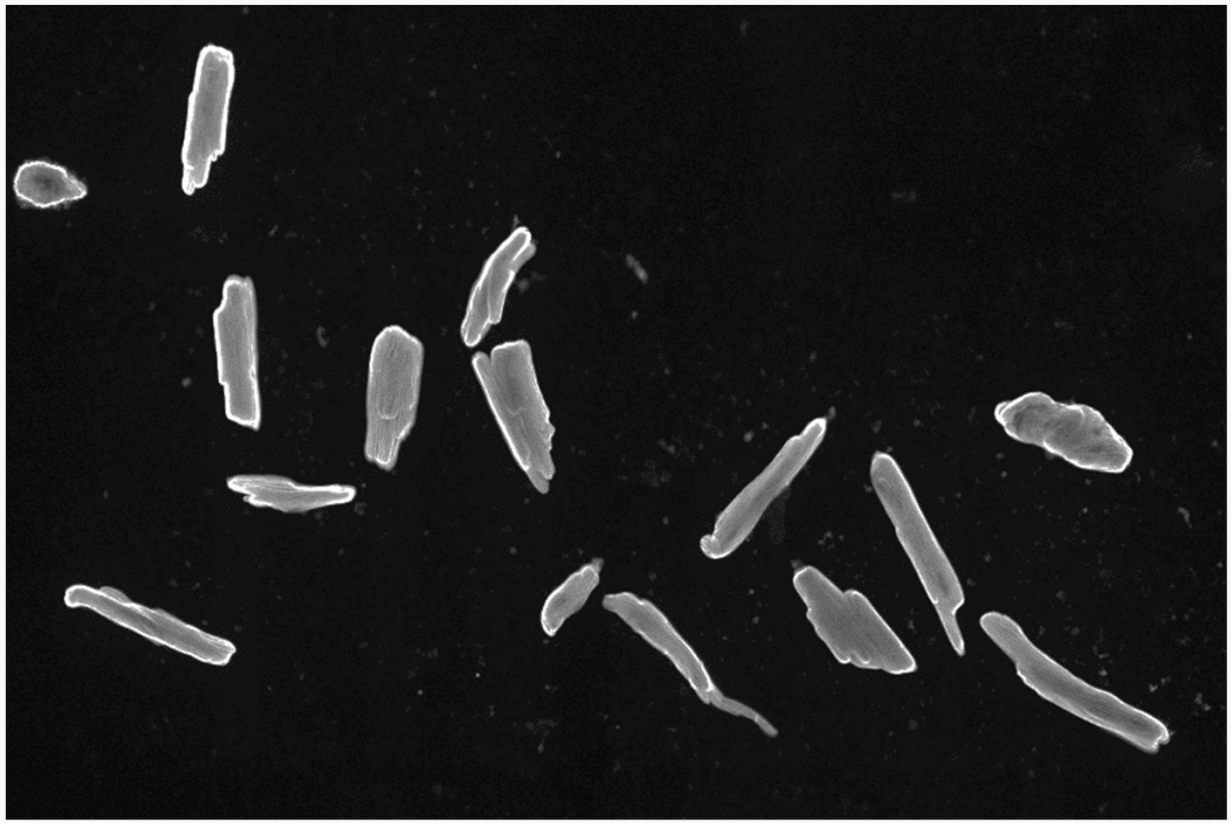
Generating the mask



```
>> mask = I > 1e4;

>> figure;
>> imshow(mask)
```

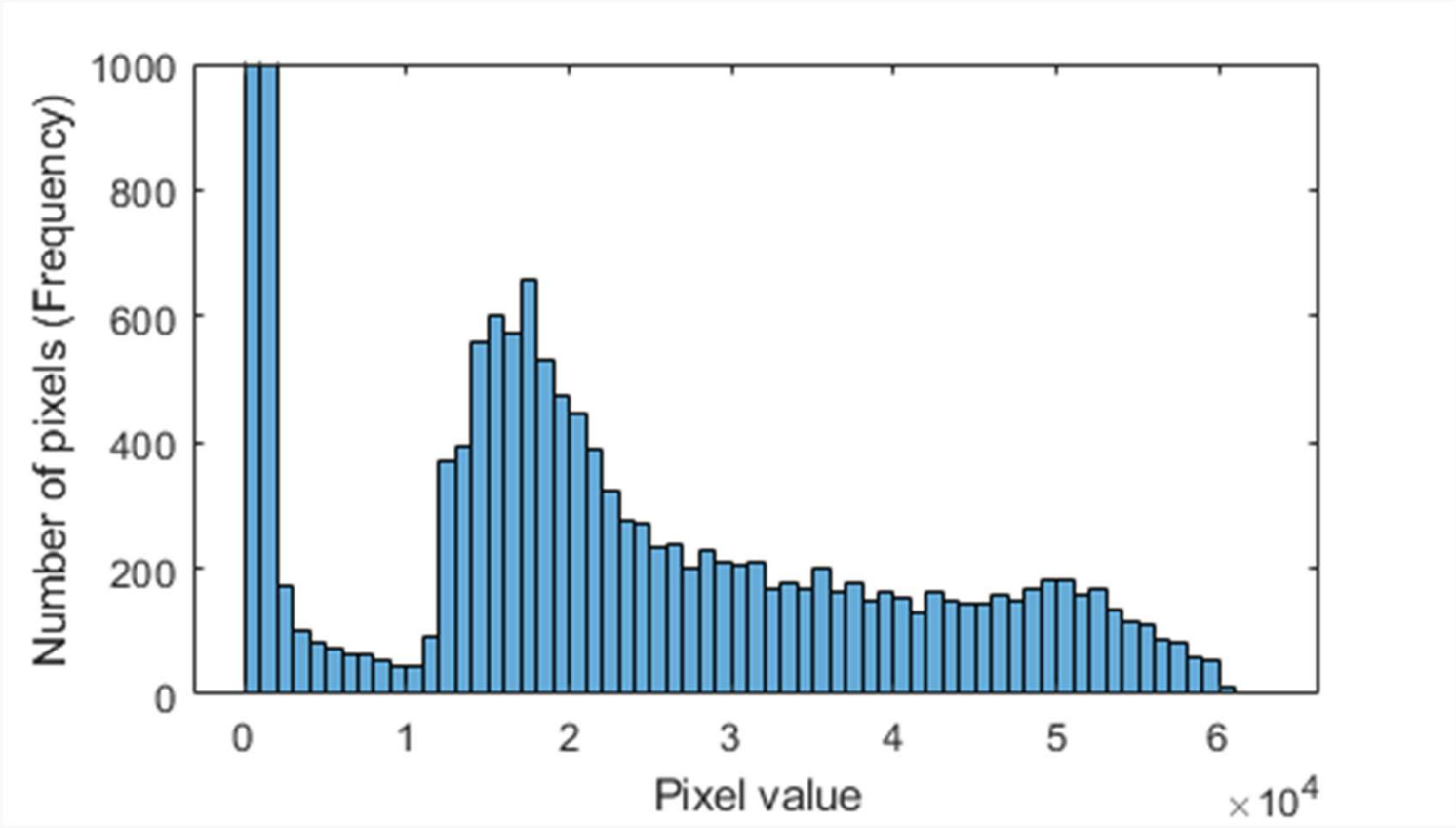
 Creates a new figure window

What if image had many cells?



Intensity histograms

```
histogram(I)

ylim([0 1000])
```

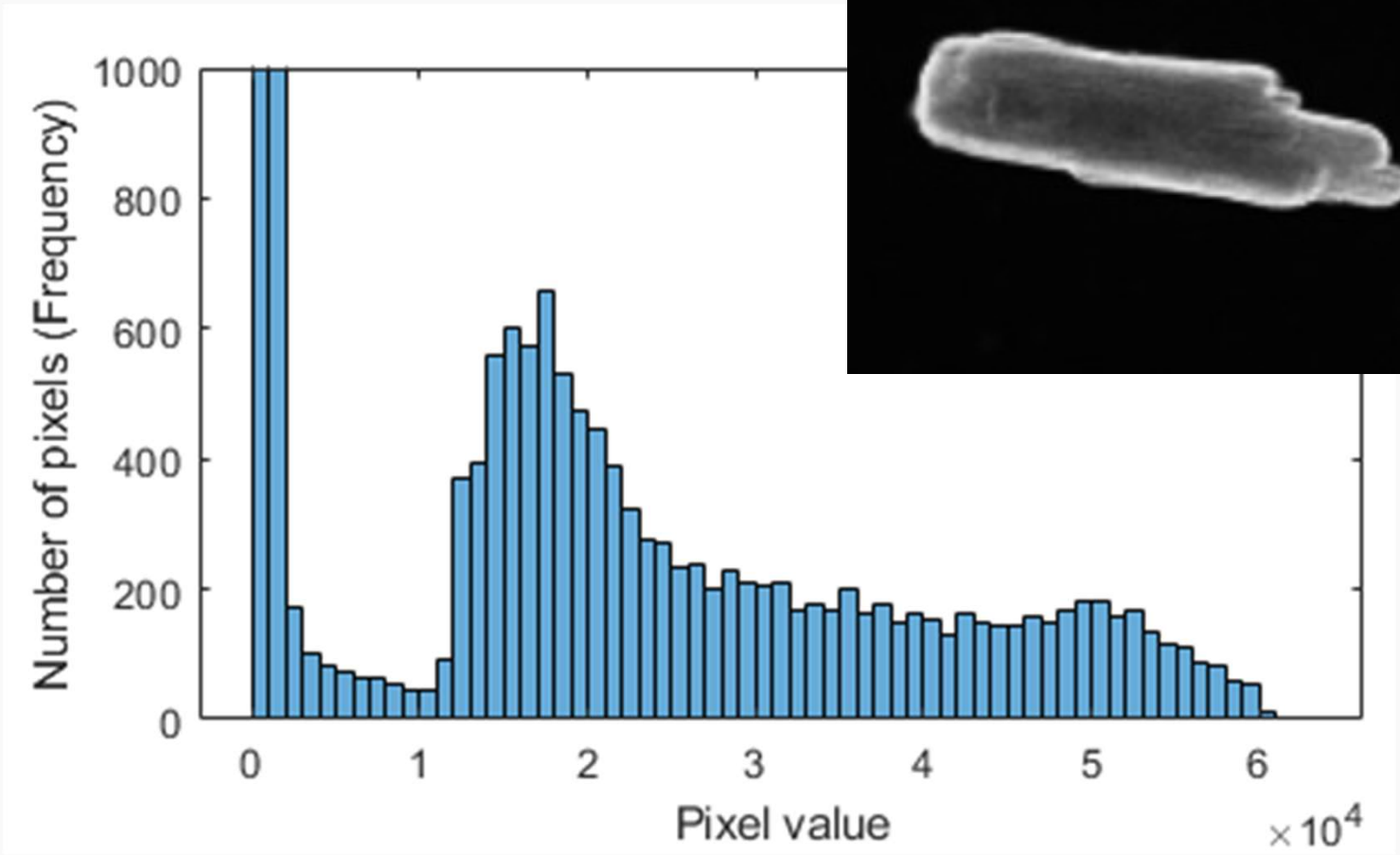


Intensity histograms

```
histogram(I)

%Set limit of y-axis
ylim([0 1000])

%Label the axes
ylabel('Number of pixels (Frequency)')
xlabel('Pixel value')
```

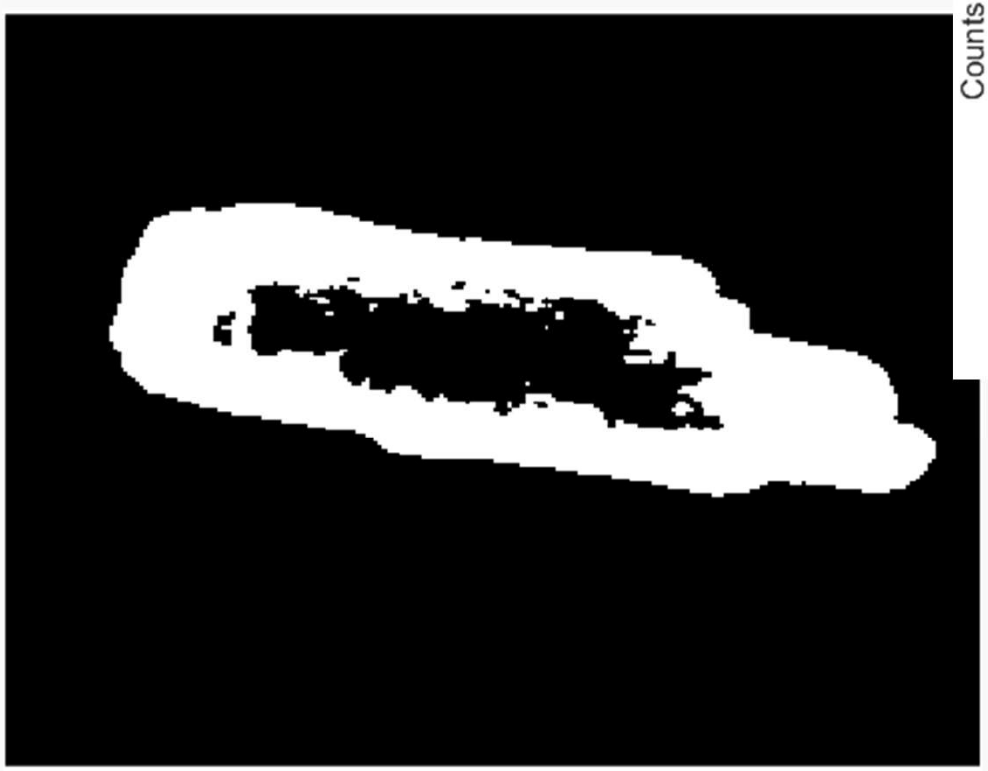
Intensity histograms

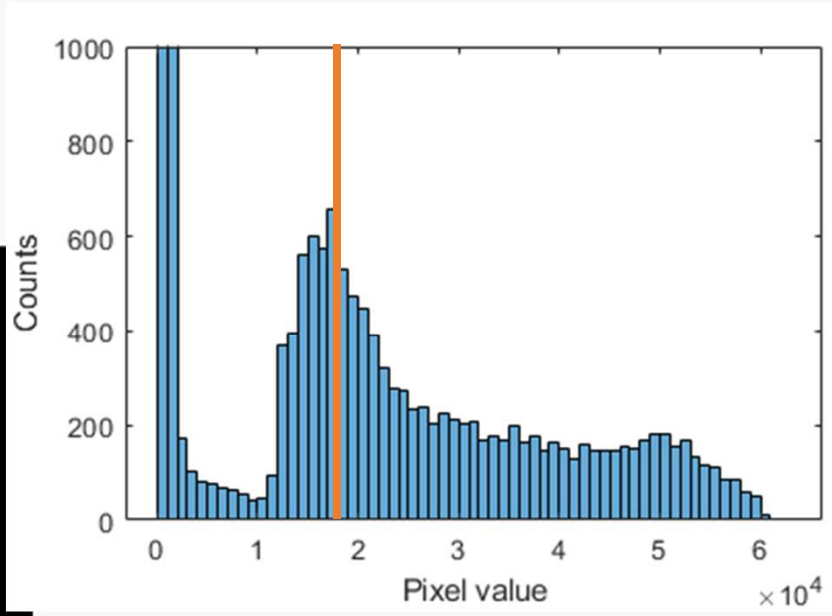
```
histogram(I)
```



Automatic binarization algorithm

```
>> mask = imbinarize(I);
```





Otsu's method: An automatic binarization algorithm

- Assumption: object(s) and background are well-separated in intensity
  - Background is dark
  - Cells are bright

Fluorescent images

- Looks for a threshold that minimizes the sum of the variance of the two groups
  - Groups clustered as tightly as possible around their individual means
  - Variance is measure of spread of numbers around mean

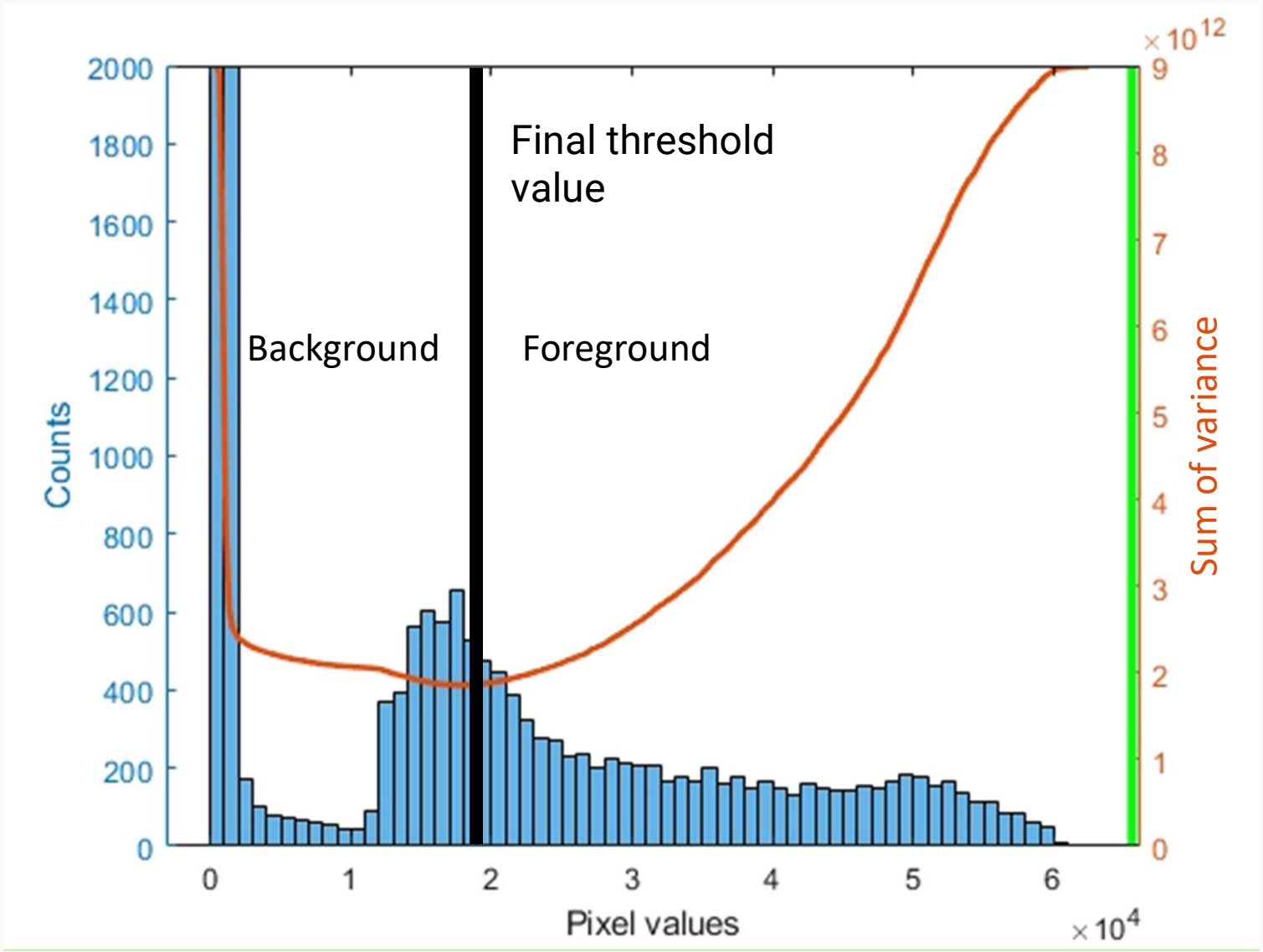
Implementing Otsu's method

- You don't need to know how to program Otsu's method, but you should know conditions that affect the outcome of the algorithm

- For those interested:

  - Otsu's paper: http://dx.doi.org/10.1109/TSMC.1979.4310076

  - Steve Eddins blog: https://blogs.mathworks.com/steve/2016/06/14/image-binarization-otsus-method/


Final threshold value
Background
Foreground
Counts
Pixel values
Sum of variance
×10^12
×10^4
2000
1800
1600
1400
1200
1000
800
600
400
200
0
0
1
2
3
4
5
6
9
8
7
6
5
4
3
2
1
0

Pros and cons with Otsu's algorithm

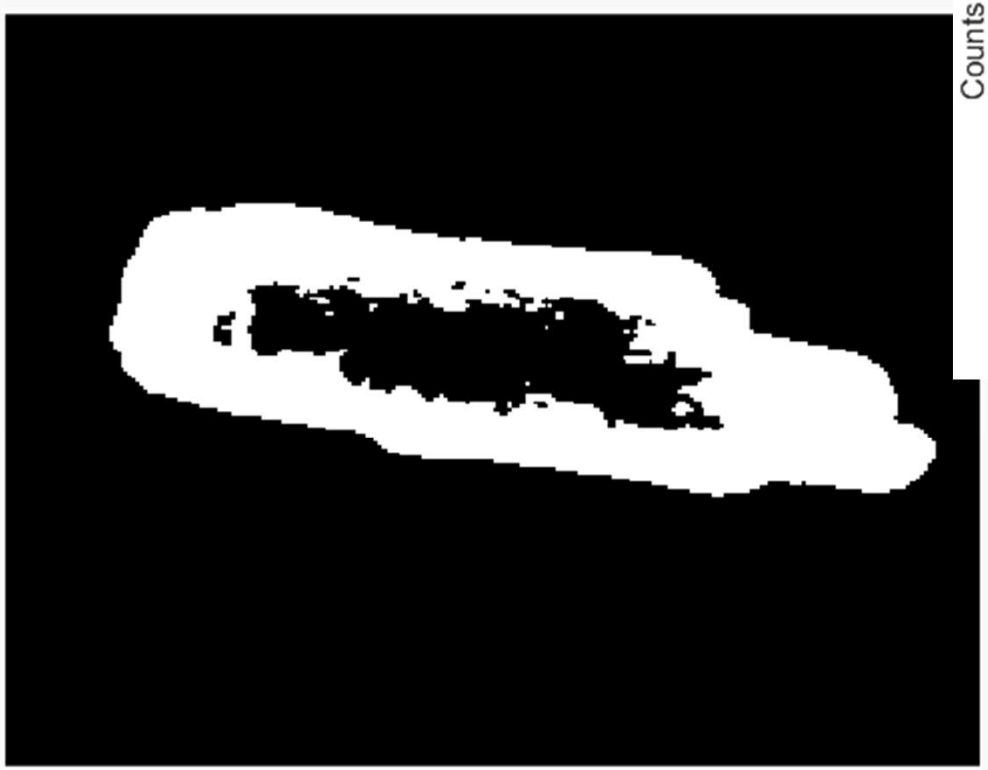
- Otsu's method uses **global statistics** (i.e. statistics of the whole image) to compute the threshold

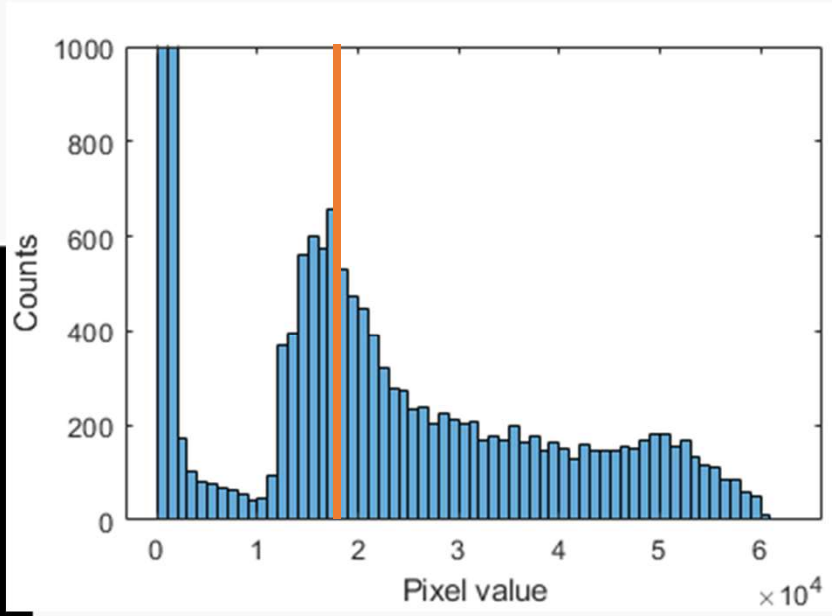
- Works well if object(s) have intensity distributions that are distinctly separated from the background – i.e. deep "valley" between them

- But does not work as well if:
  - The image intensity is uneven
  - There is not much contrast between the object and background
  - There is a lot of noise in the image
  - … examples in homework!

Automatic binarization algorithm

```
>> mask = imbinarize(I);
```





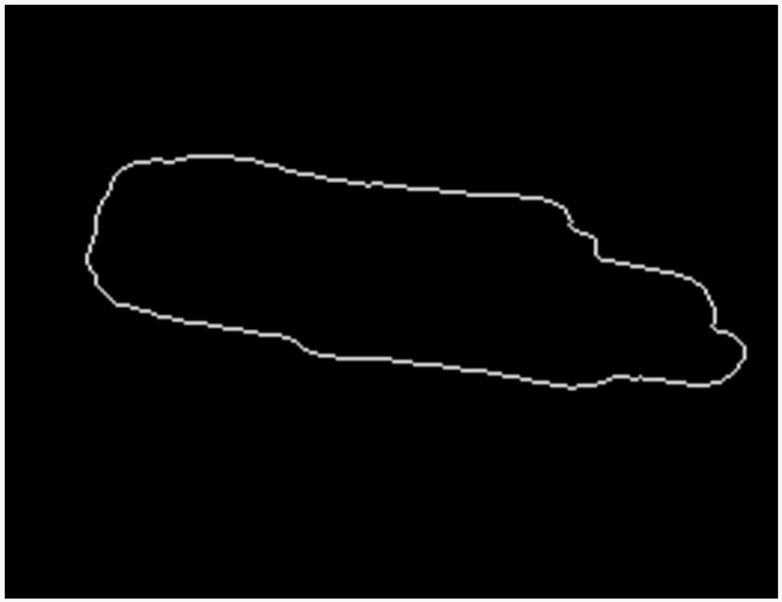
Fill

```
>> mask = imfill(mask, 'holes');
```

Fills in holes in the image
(zeros surrounded by ones)

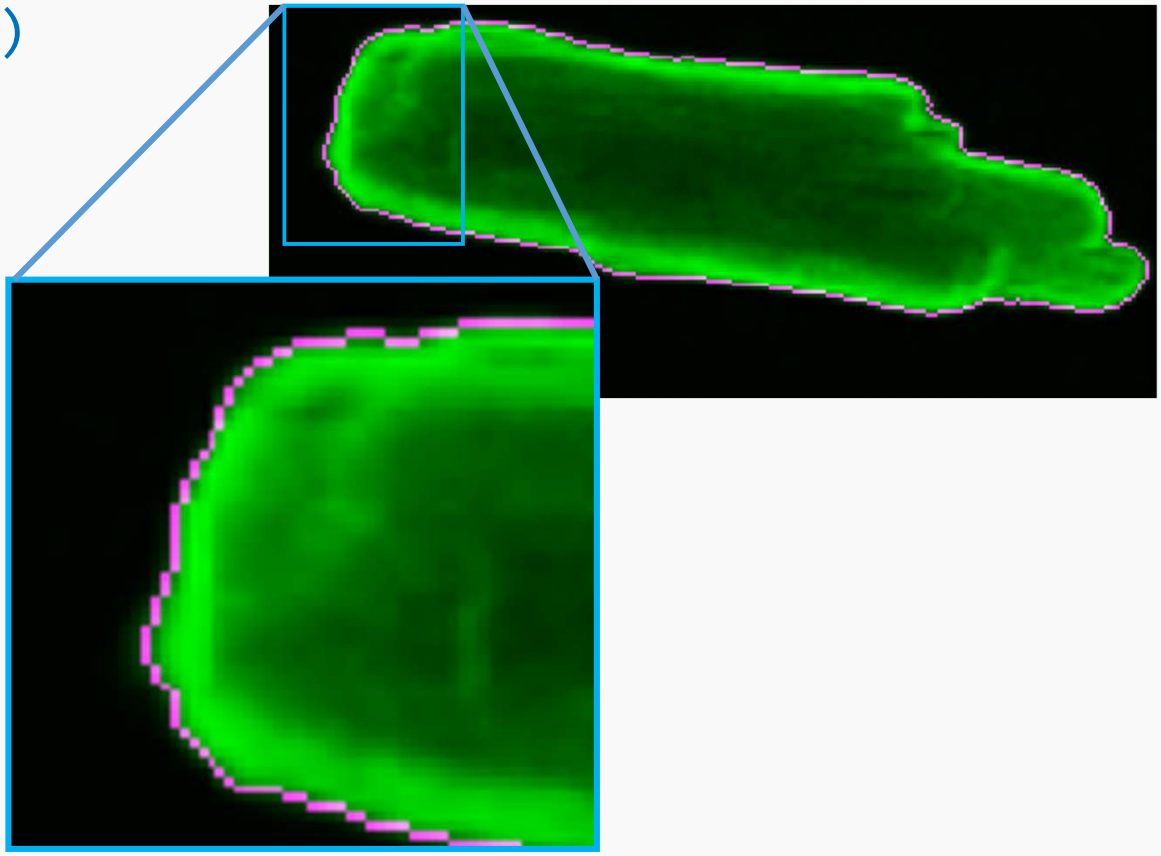
Validating the mask

```
%Get a mask of the perimeter
perimeter = bwperim(mask)
imshow(perimeter)
```



Validating the mask

```
%Get a mask of the perimeter
perimeter = bwperim(mask)
imshowpair(I, mask)
```



First image `I` is shown in green

Second image `mask` is shown in magenta

Measuring cell properties

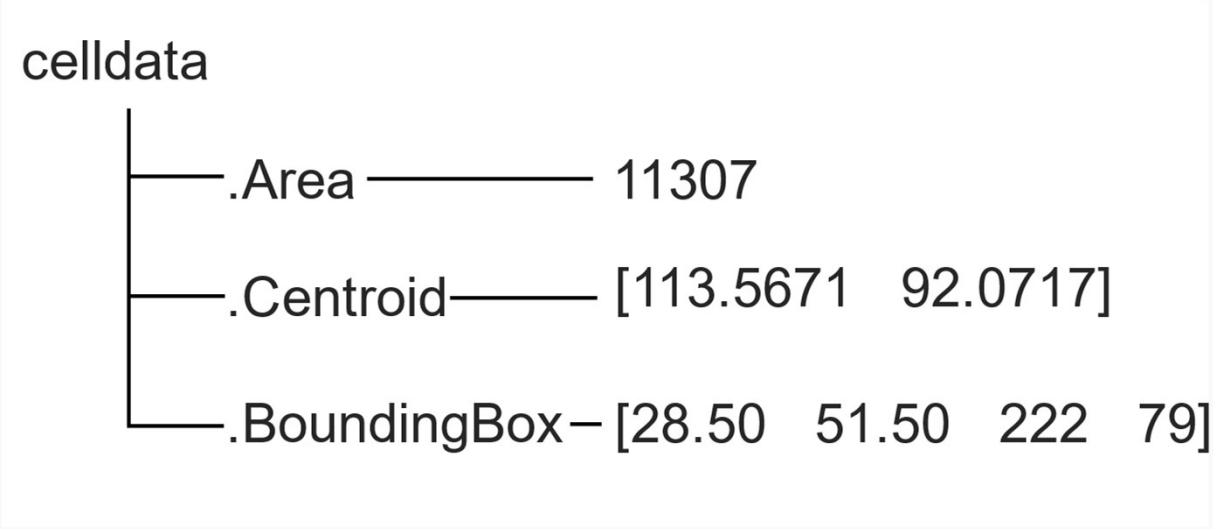
- Once you have the mask, you can measure properties of the cell using the function `regionprops`

```
celldata = regionprops(mask)
```

`celldata` is a `struct` or a structured array

Structured Arrays (`struct`)

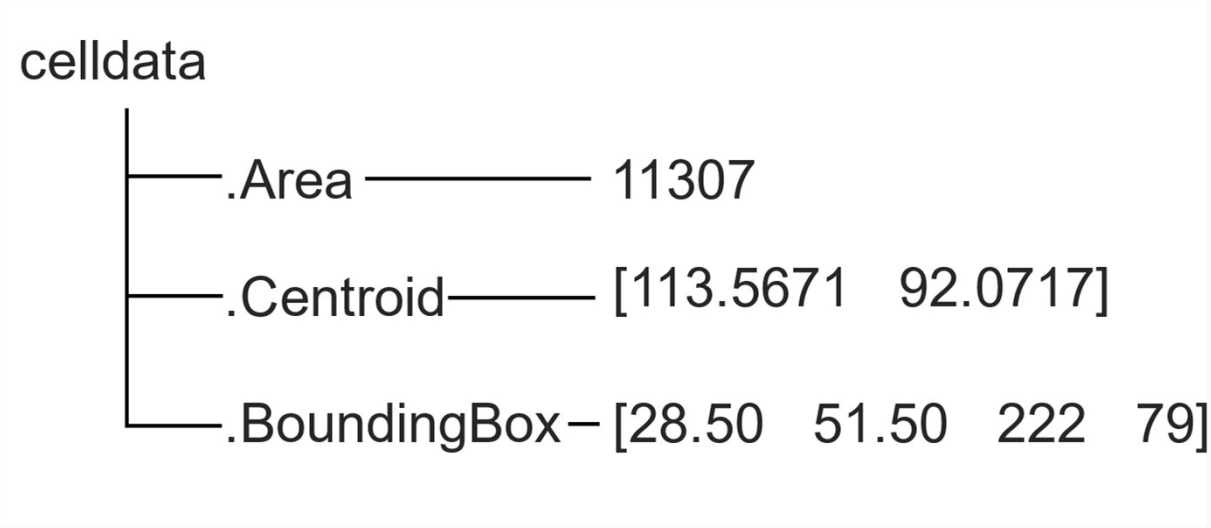
- `struct` is a basic MATLAB data type



- Data is stored in named **fields**

- Data in different fields can have different types and sizes

Accessing data from a struct




```
>> celldata.Area
```

- Fieldnames are case-sensitive
- For `regionprops`, fields always have first letter uppercase

Creating a struct

- If you want to create a struct:

```
>> S.name = 'Jian';
>> S.Area = 20;
```

- To add to the array:

```
>> S(2).name = 'Joe';
>> S(2).Area = 40;
```

Specifying properties

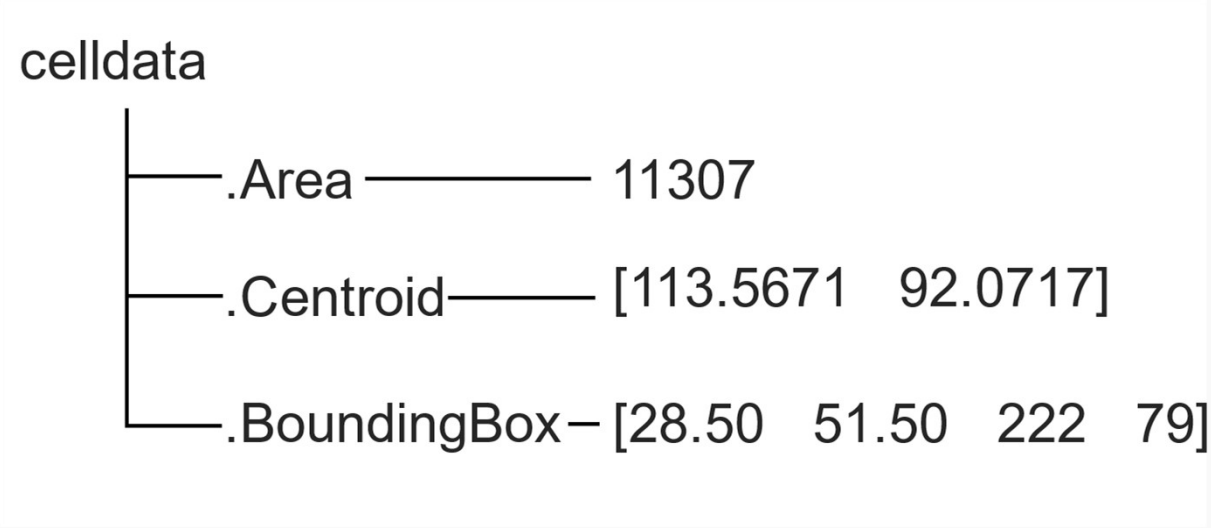
```
celldata = regionprops(mask, ...
     'Area', 'Centroid', 'MajorAxisLength')
```

Find a complete list of properties
In documentation

- Area = number of pixels in mask

- Centroid = (x, y) coordinate of the center of mass

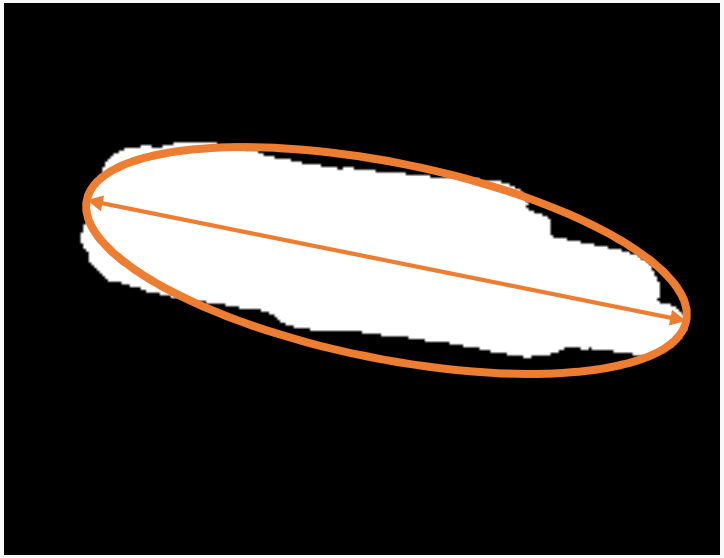
Accessing data from a struct



- How to retrieve the y-coordinate of the centroid position?

Measuring length

- MajorAxisLength



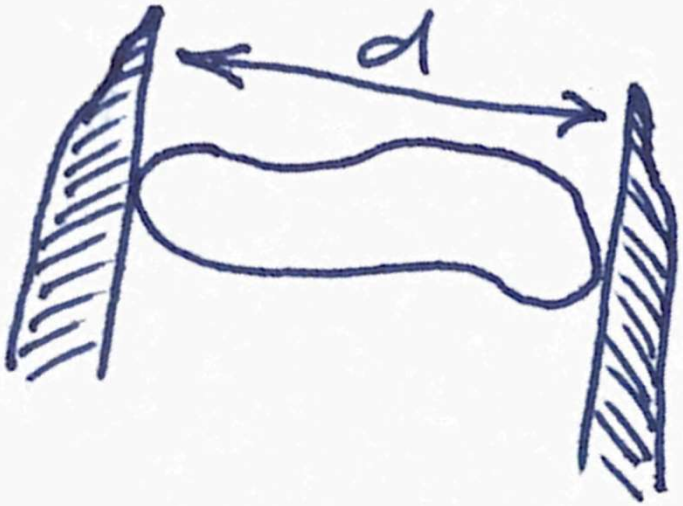
To figure out length, MATLAB
fits an ellipse to the mask

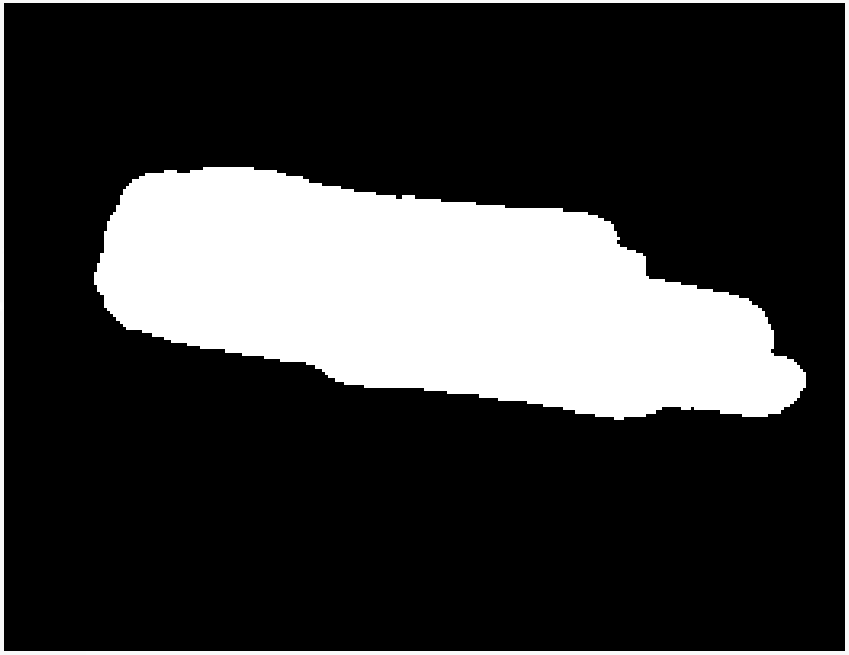
Same method for MinorAxisLength and Orientation (angle of the object)

https://blogs.mathworks.com/steve/2010/07/30/visualizing-regionprops-ellipse-measurements/

Measuring length

- MaxFeretProperties and MinFeretProperties might be more accurate





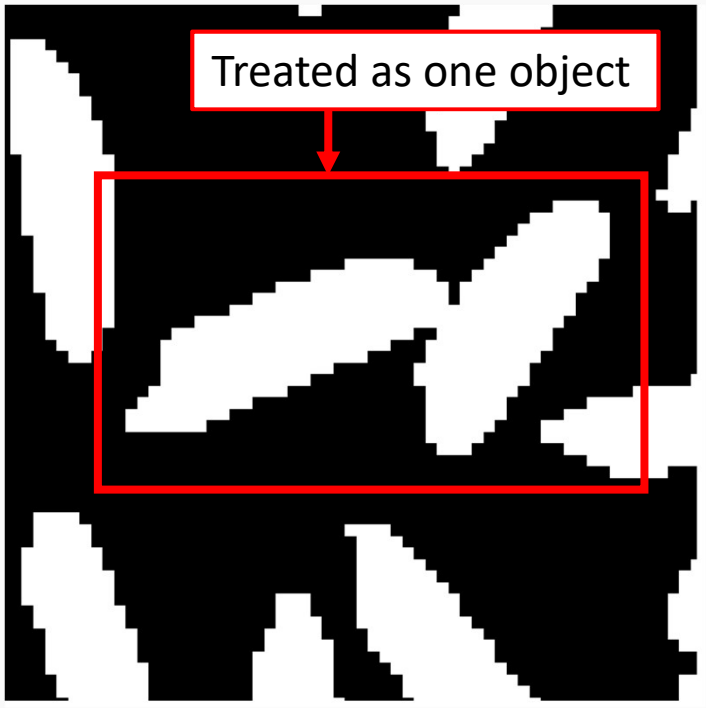
MajorAxisLength: 231.3899
MaxFeretDiameter: 224.9000

https://blogs.mathworks.com/steve/2018/02/20/minimum-feret-diameter/

Questions?

Regionprops with multiple objects

- `regionprops` treats each **unconnected region** as a separate object

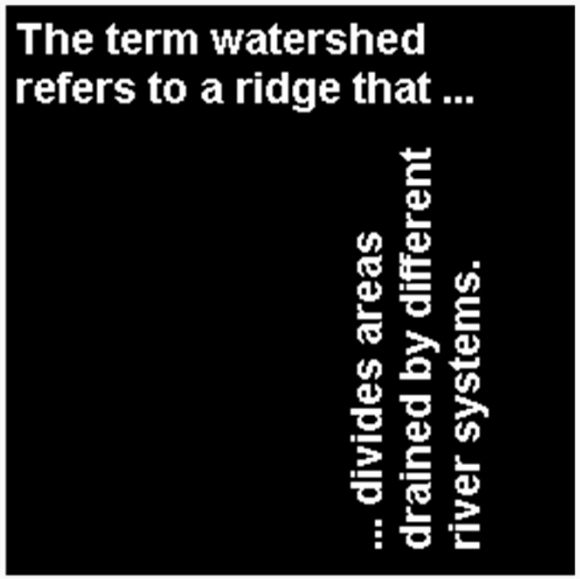


We'll look at techniques to separate objects next week

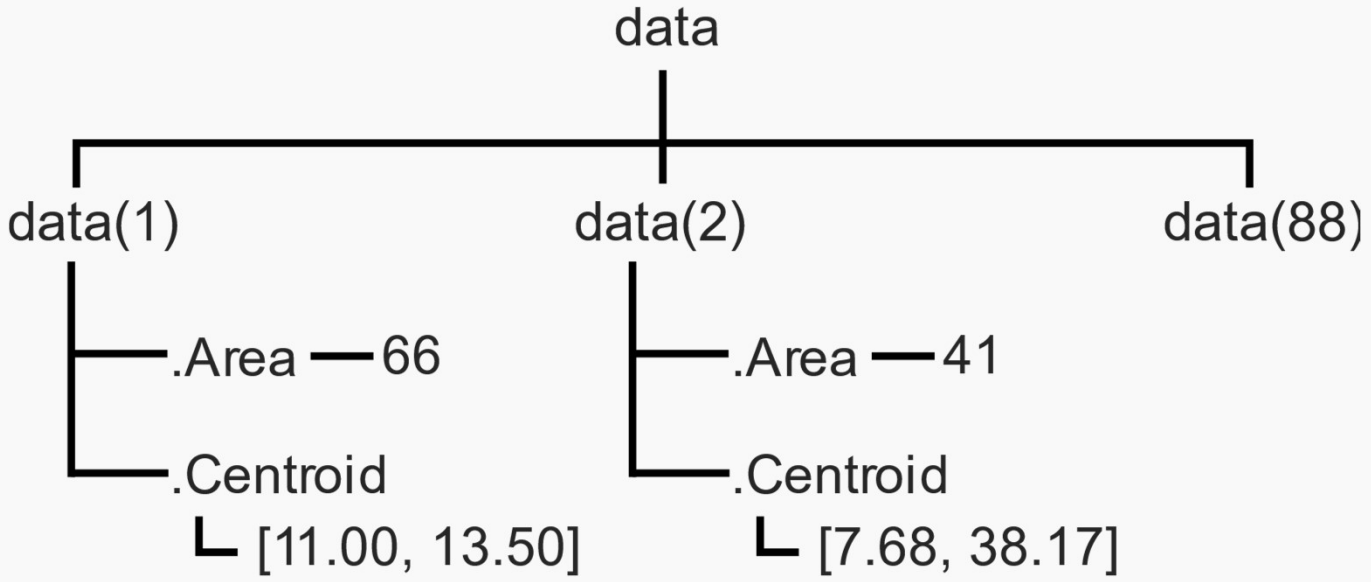
Regionprops with multiple objects

- Pretend that this is a mask that we have segmented

```
>>  BW = imread('text.png');
>>  data  = regionprops(BW, 'centroid', 'area');
```

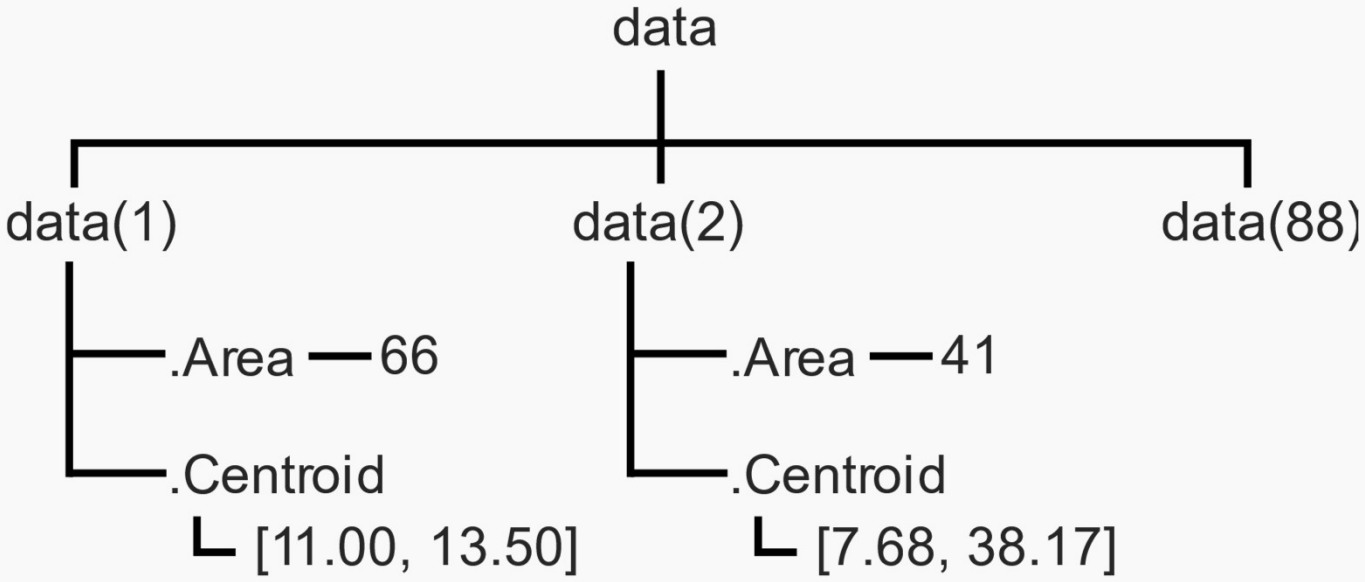


Multi-element structure arrays



- Every struct element has the SAME fields (although not the same values)

Indexing a struct element




```
>> data(1).Area

>> data(2).Area
```

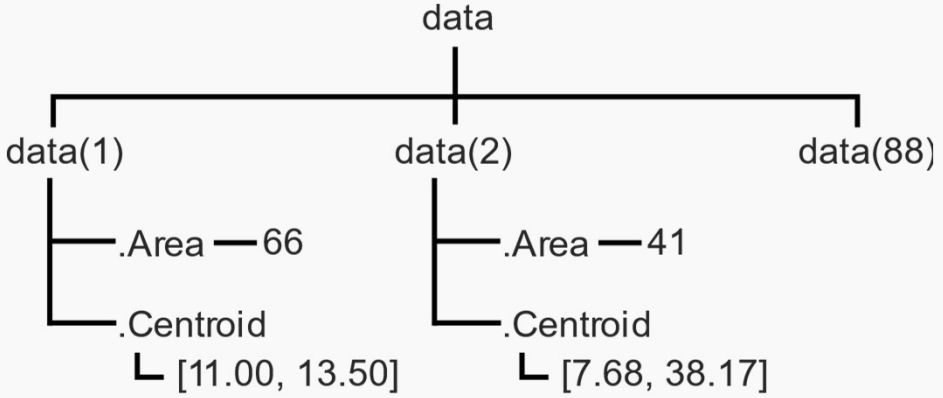
Number of detected objects

Which of the following gives you the total number of detected objects

a) `numel(data)`

b) `size(data)`

c) `numberelements(data)`



Analyzing data from struct arrays

Example: Compute the mean area of the letters

1. Concatenate (join) the values together using the function `cat`

```
areas = cat(1, data.Area);
```

Index of dimension to concatenate

dim = 1 is rows

Analyzing data

Example: Compute the mean area of the letters

1. Concatenate (join) the values together using the function `cat`

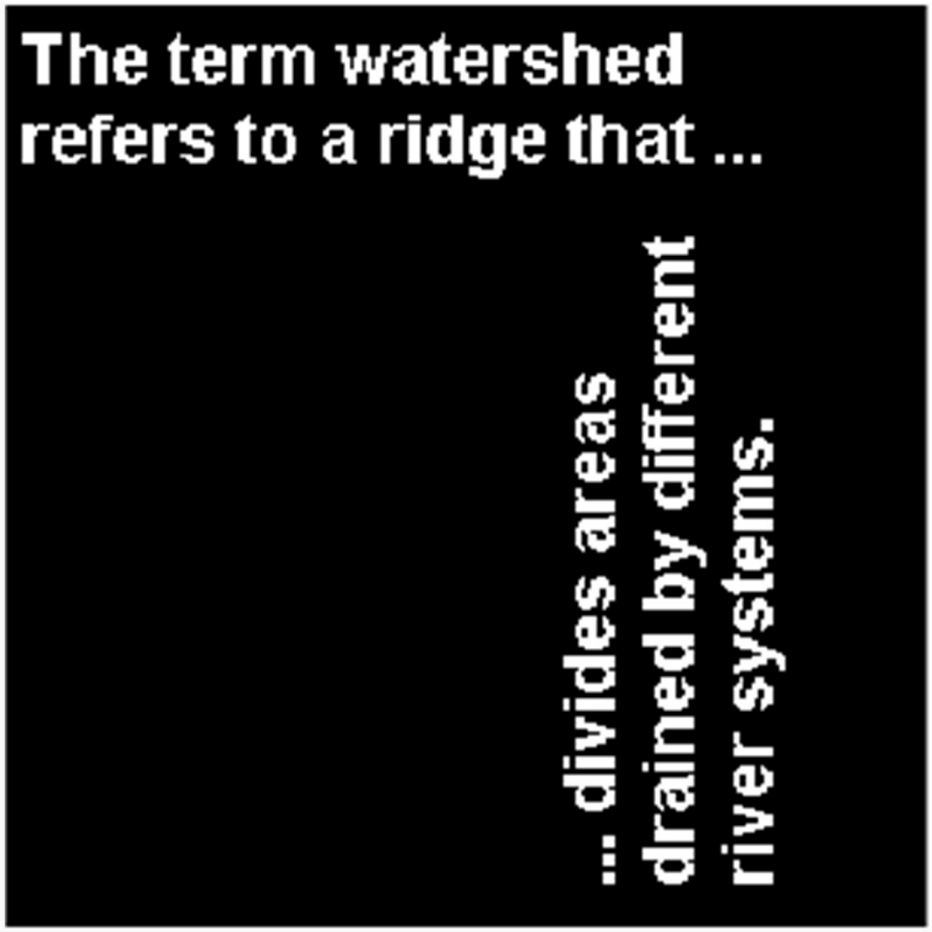
```
>> areas = cat(1, data.Area);
```

2. Compute the mean

```
>> mean(areas)
```

Removing small areas

- Say we want to get rid of the dots in the image

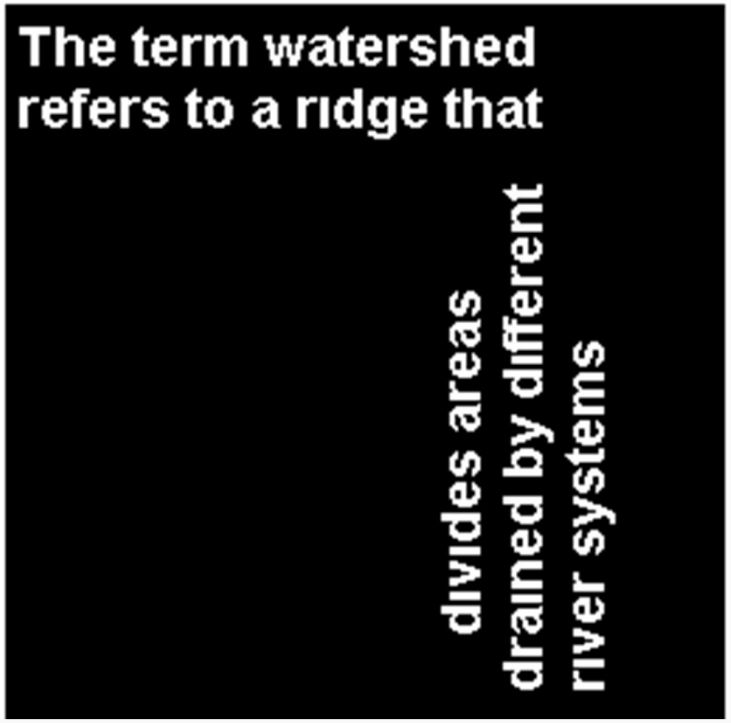


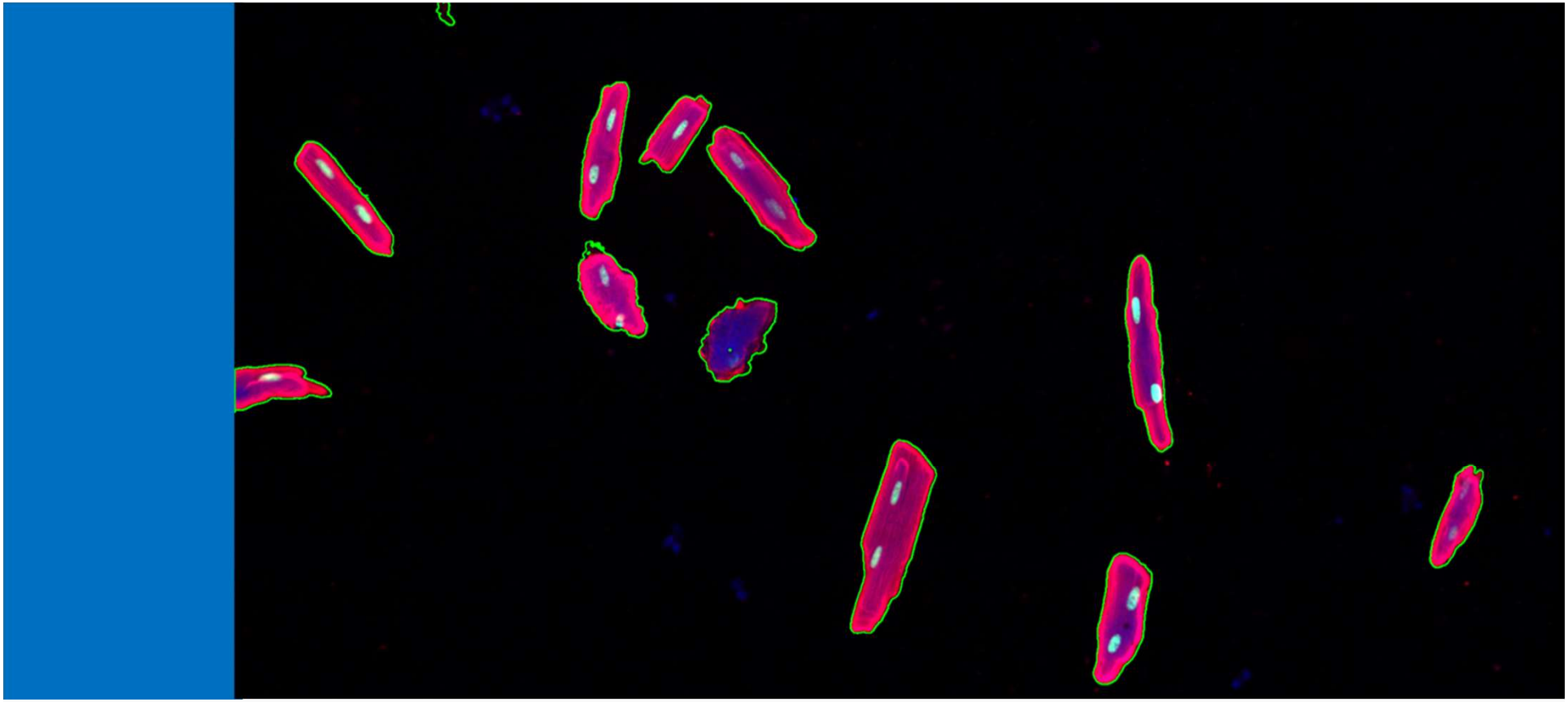
Removing small areas

```
mask = bwareaopen(BW, 20);
```

Minimum area in pixels

```
imshow(mask)
```

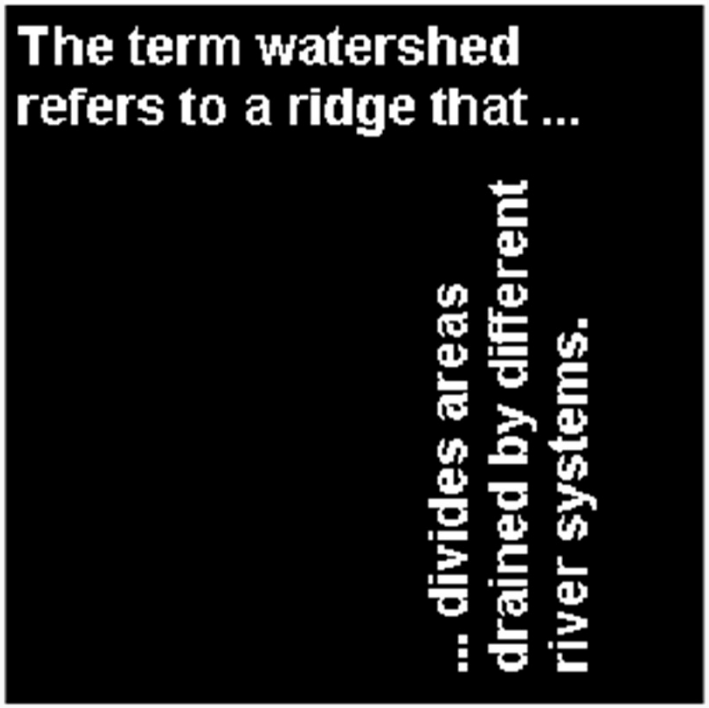




Week 4: Segmentation

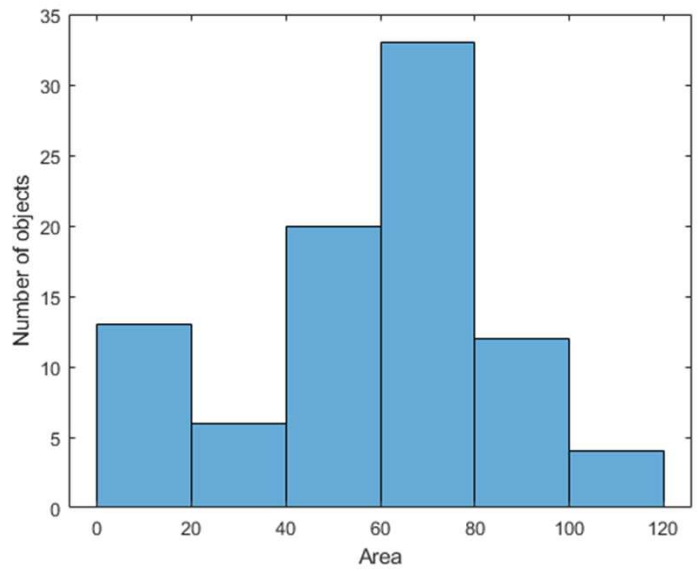
MCDB-BCHM 4312-5312

Filtering data



- Count the number of dots in the text (e.g. periods and the dots above i)
- One way is to distinguish them by area since the dots are small
- Plot a histogram to see distribution

```
>> histogram(areas)
```



Using logical indexing to filter data

- Select only matrix elements that are smaller than 20

```
dotAreas = areas(areas < 20)
```

How many dots are there in the image? 13

Validating the data

```
>>  BW = imread('text.png');
>>  s  = regionprops(BW, 'centroid', 'area',  ...
                'PixelIdxList');
```

`PixelIdxList` = List of pixel indices

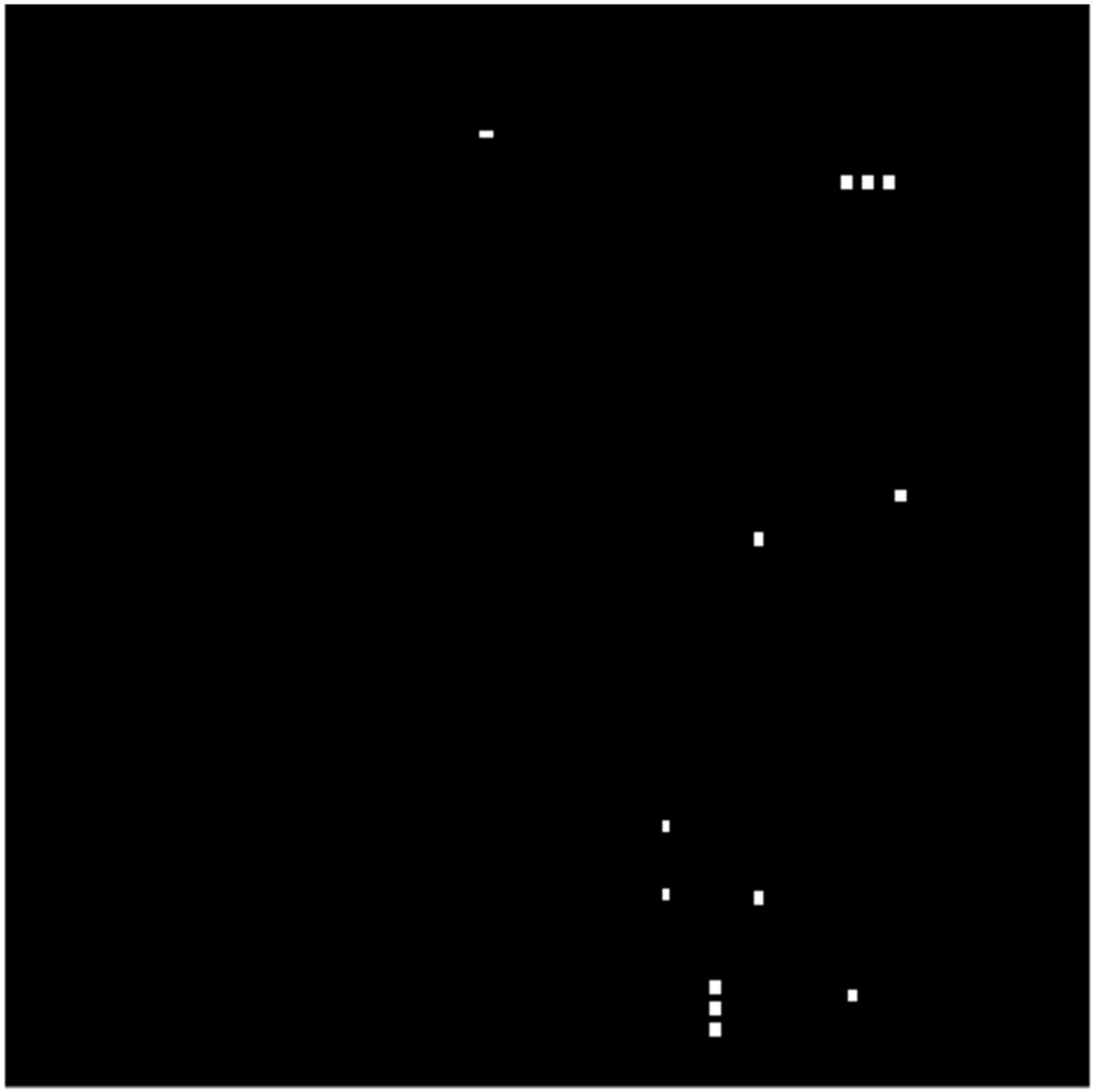
How to select only elements which have area < 20?

```
>> dots = s(areas < 20);
```

Create a mask

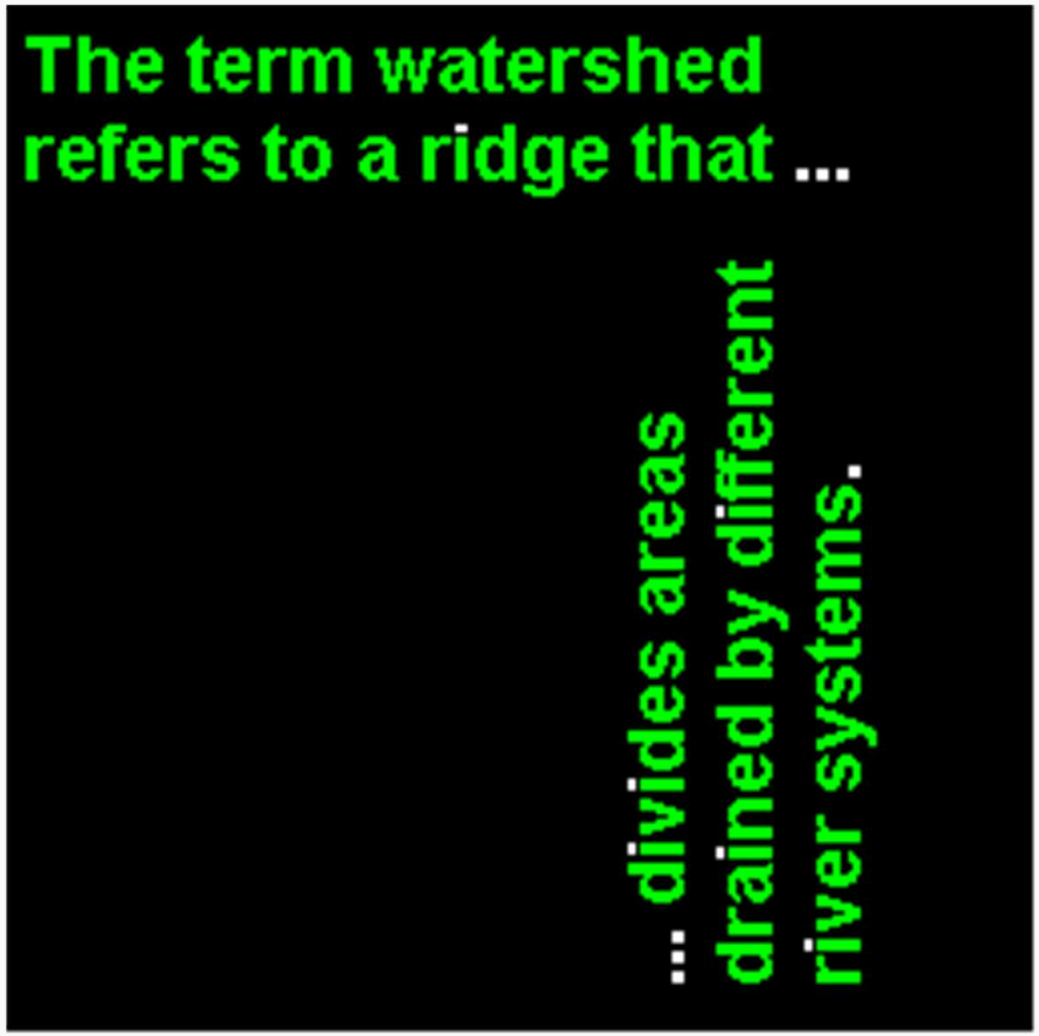
You can use `false(size)` to create logical array

```
%Initialize a logical array with value of false
dotMask = false(size(BW));

%Concatenate (combine) the pixel indices into a vector
allPixels = cat(1, s.PixelIdxList);

%Now set the pixels to true
dotMask(allPixels) = true;

imshow(dotMask)
```

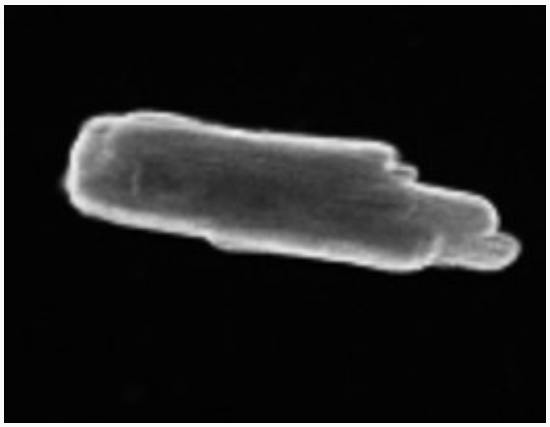


Overlay the mask over the original image

```
>> imshowpair(BW, mask)
```



How do we evaluate quantitatively that segmentation is accurate?



- Questions:
  - How do we know that every pixel in the original image was identified correctly?
  - imshowpair gives a **qualitative** result (i.e. we can tell if the code is mostly correct or mostly wrong)