MCDB/BCHM 4312 & 5312 – Quantitative Optical Imaging

Lecture 23:

The watershed algorithm

Lecturer: Jian Wei Tay

Date: 15 October 2021

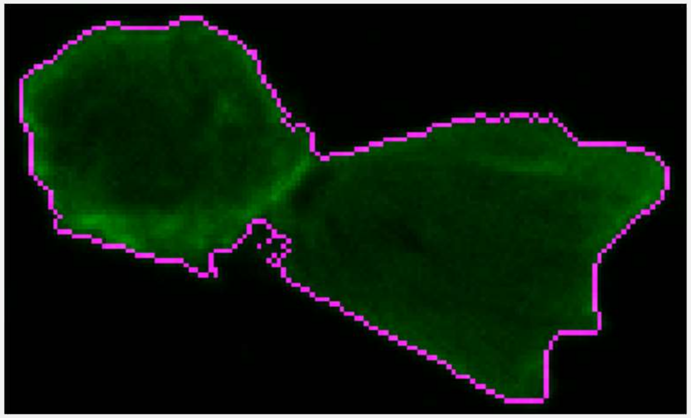
University of Colorado **Boulder**

Learning objectives

- Understand how the watershed algorithm works
- Understand the distance transform algorithm
- Explain what oversegmentation and undersegmentation are
- Use `imhmin` to refine the watershedding results
- Understand the limitations of watershedding

Segmentation

- Segmentation refers to the process of labeling individual objects within an image
- However, when using intensity thresholding, we often get connected objects

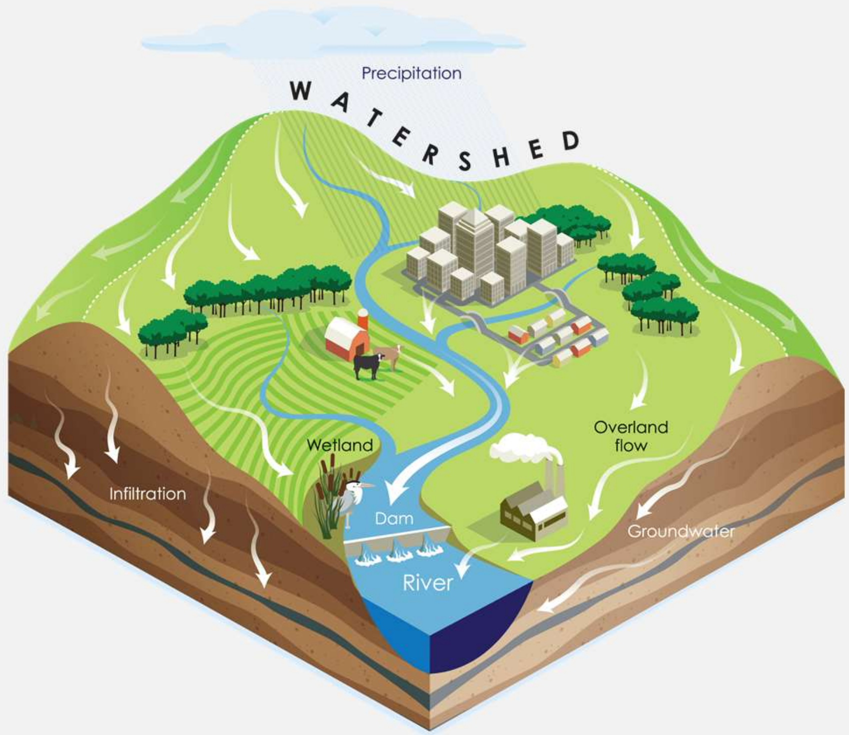


Note: Remember that connected true regions in a mask are treated as one object

The watershed algorithm

- To separate clusters of objects, we can try using the watershed algorithm

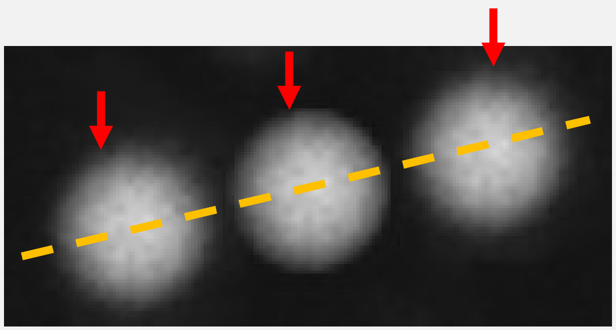
A (geological) watershed is an area of land that captures rainfall and funnels it to a lake/stream

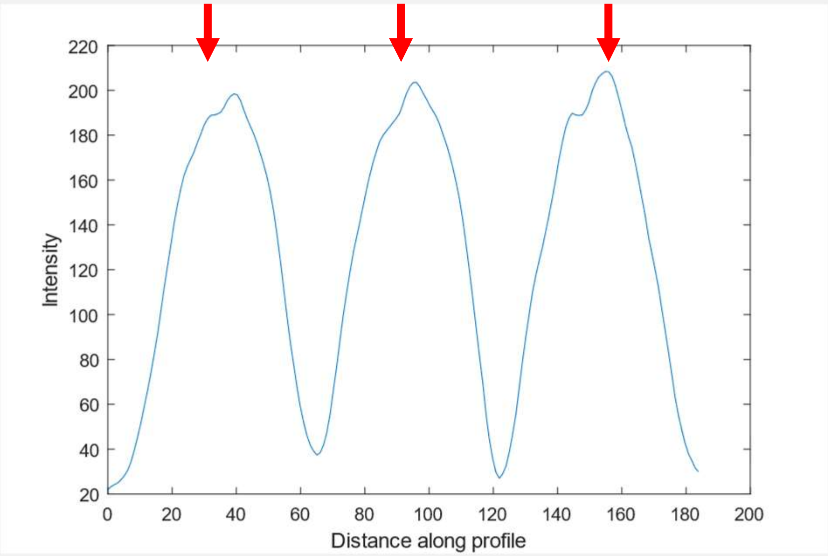


Note: The watershed algorithm uses the same idea

Watershed (image analysis)

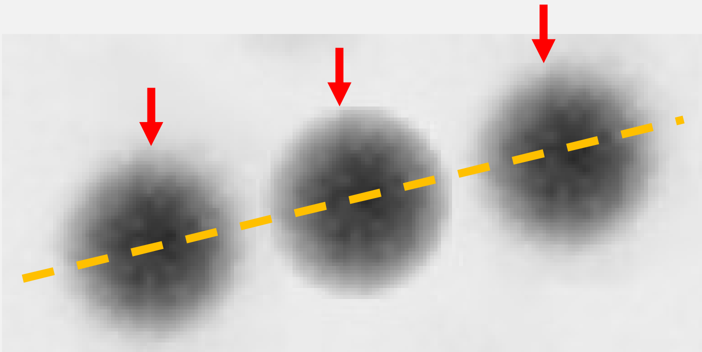
- The algorithm treats the input image as a height map, where intensity = height



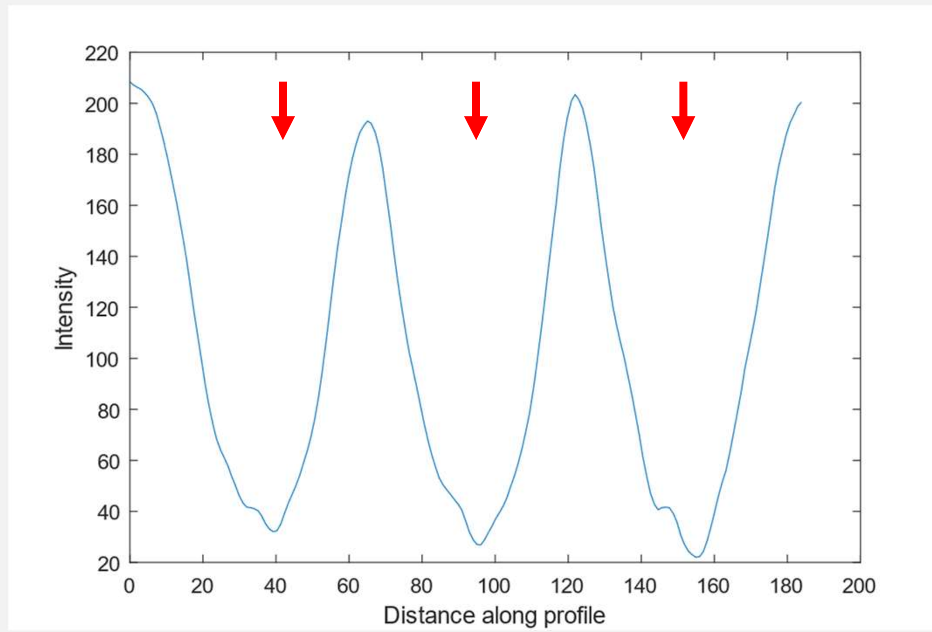


Watershed (image analysis)

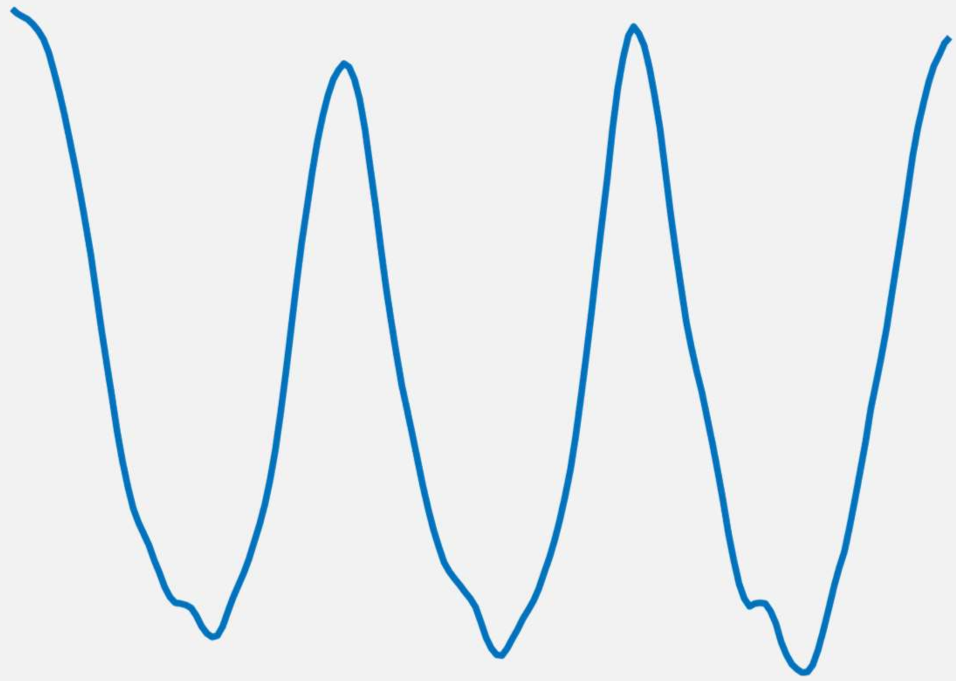
- The algorithm treats the input image as a height map, where intensity = height



Note: We need a profile that look like "basins"

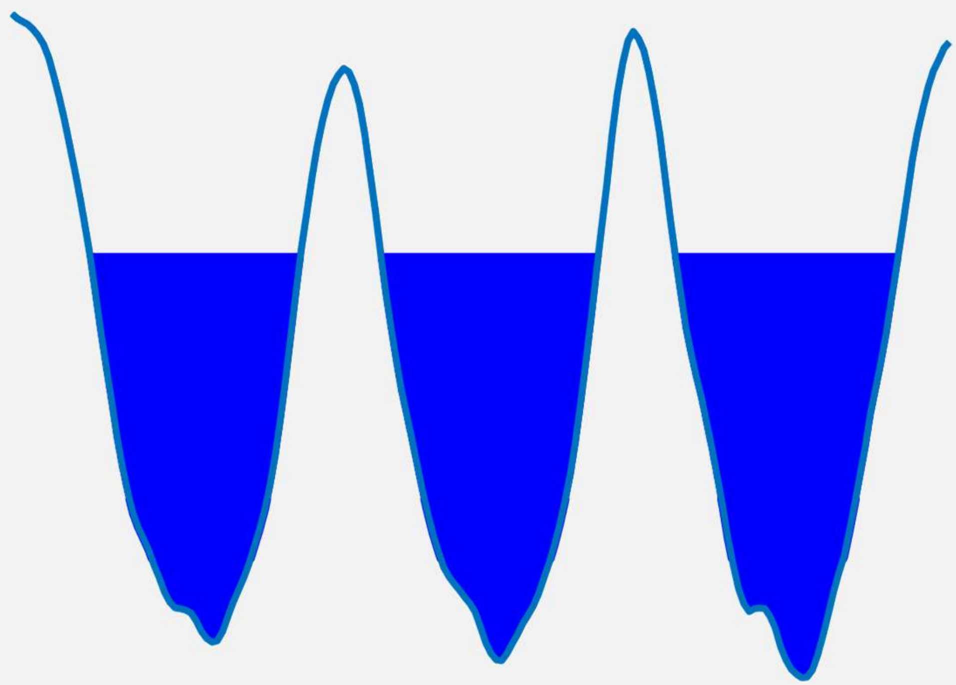


Watershed (image analysis)



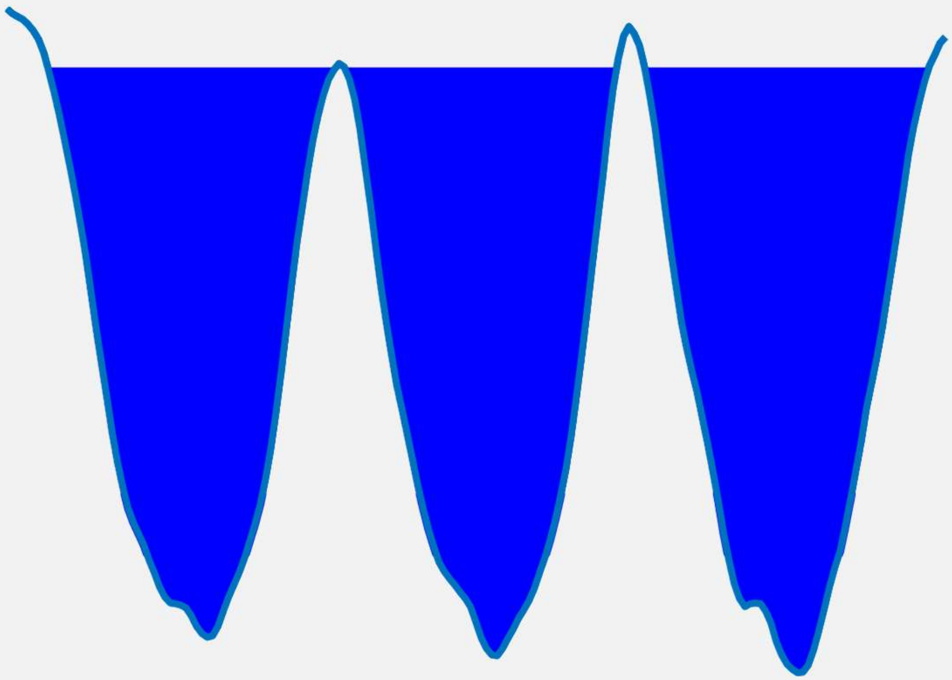
Watershed (image analysis)

Note: The algorithm works by fills the basins with water



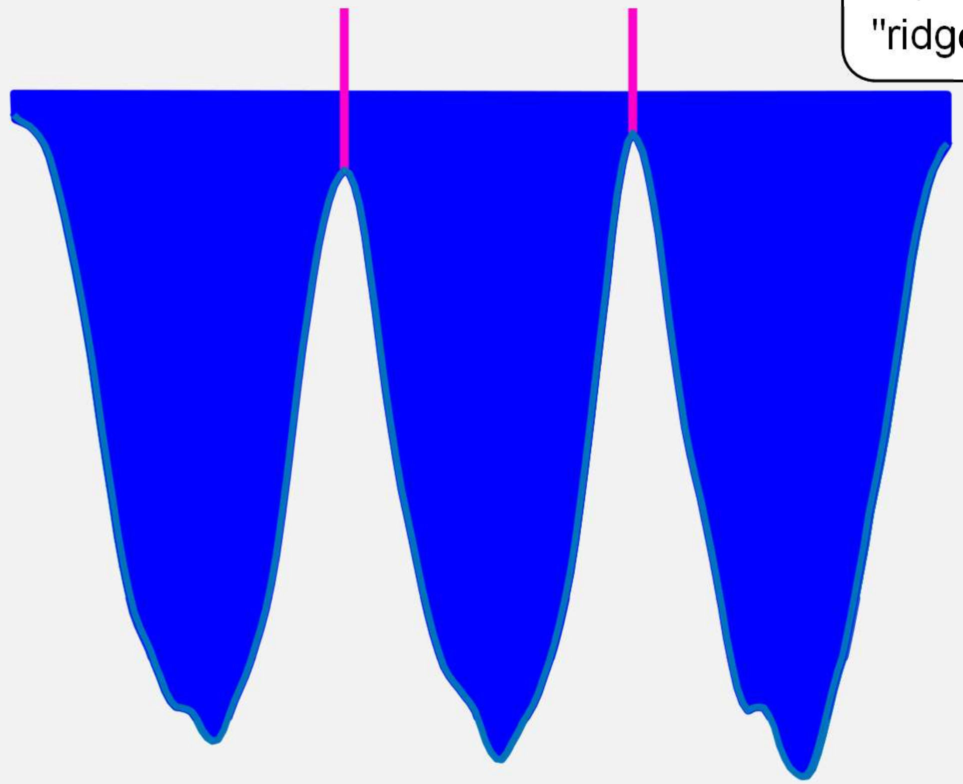
Watershed (image analysis)

Note: The algorithm works by fills the basins with water



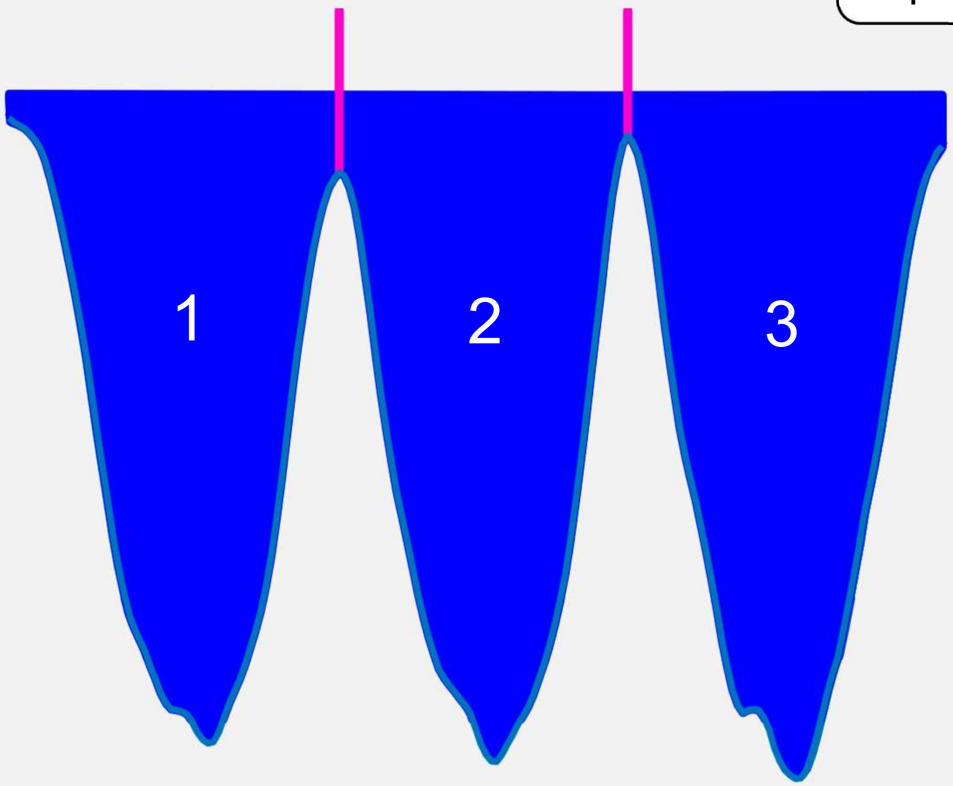
Watershed (image analysis)

Note: The locations where the water touches are "ridge lines" (in magenta)



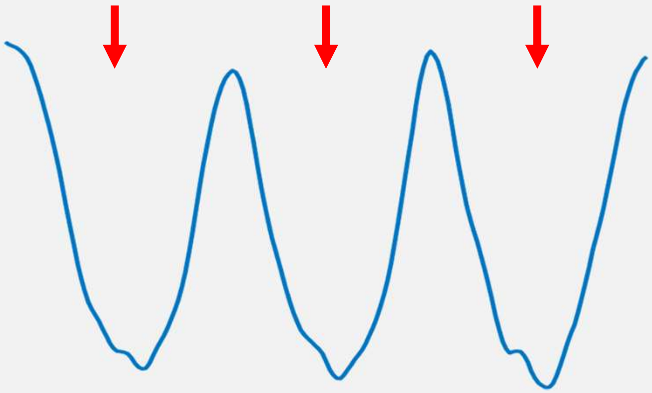
Watershed (image analysis)

Note: Each object is now separated by a ridge line



Basic requirements for the watershed algorithm

- Each object in the input image to the function must be a "basin"
- The center of each object should be near the deepest part of the basin



Questions?

Steps for performing the watershed transform

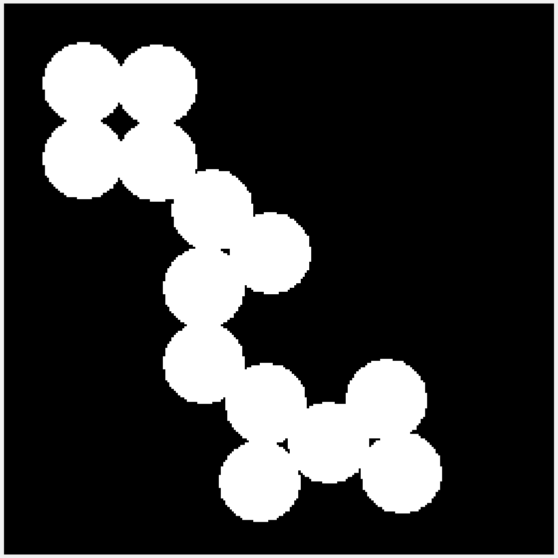
1. Make an initial mask of the objects of interest (e.g., by using manual intensity thresholding or Otsu's/Bradley's method)
2. Convert the mask into an intensity profile using the distance transform
3. Refine the distance transform
4. Run the watershed algorithm
5. Update the original mask

Steps for performing the watershed transform

1. Make an initial mask of the objects of interest (e.g., by using manual intensity thresholding or Otsu's/Bradley's method)
2. Convert the mask into an intensity profile using the distance transform
3. Refine the distance transform
4. Run the watershed algorithm
5. Update the original mask

Practice

- Create a new script
- Read the mask 'circles.png' into a variable

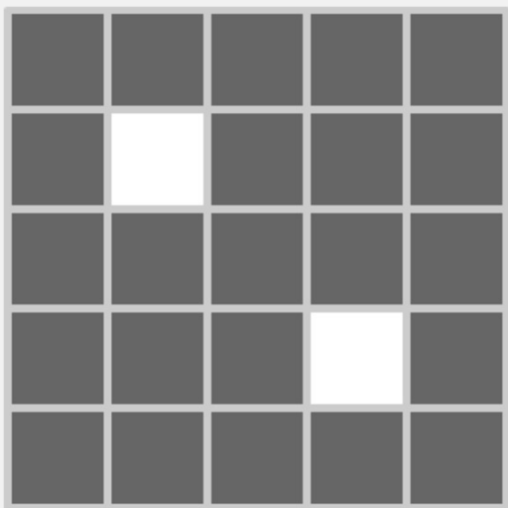


Steps for performing the watershed transform

1. Make an initial mask of the objects of interest (e.g., by using manual intensity thresholding or Otsu's/Bradley's method)
2. Convert the mask into an intensity profile using the distance transform
3. Refine the distance transform
4. Run the watershed algorithm
5. Update the original mask

The distance transform

- For each pixel in the image, the distance transform calculates the distance to the nearest nonzero pixel



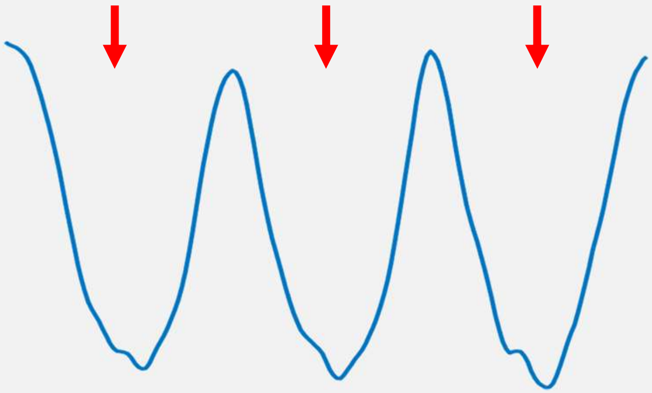
Input

1.41	1	1.41	2.24	3.16
1	0	1	2	2.24
1.41	1	1.41	1	1.41
2.24	2	1	0	1
3.16	2.24	1.41	1	1.41

Distance transform

Basic requirements for the watershed algorithm

- Each object in the input image to the function must be a "basin"
- The center of each object should be near the deepest part of the basin



Note: The distance transform is used to convert the mask into an intensity gradient

The distance transform

```
dd = bwdist(M)
```

M = logical array (mask)

dd = distance transform (double)

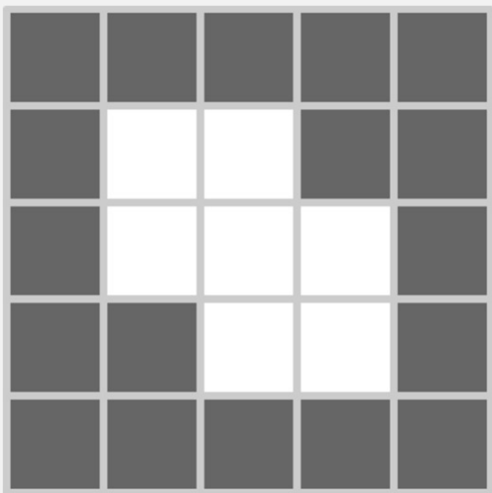
Practice

- Write a line of code in your script to calculate the distance transform of the circles mask
- Display the distance transform using imshow

Will the distance-transformed image work for the watershed algorithm?

The problem

- Objects in the distance-transformed image do look like basins, but they are not separated by peaks



Input

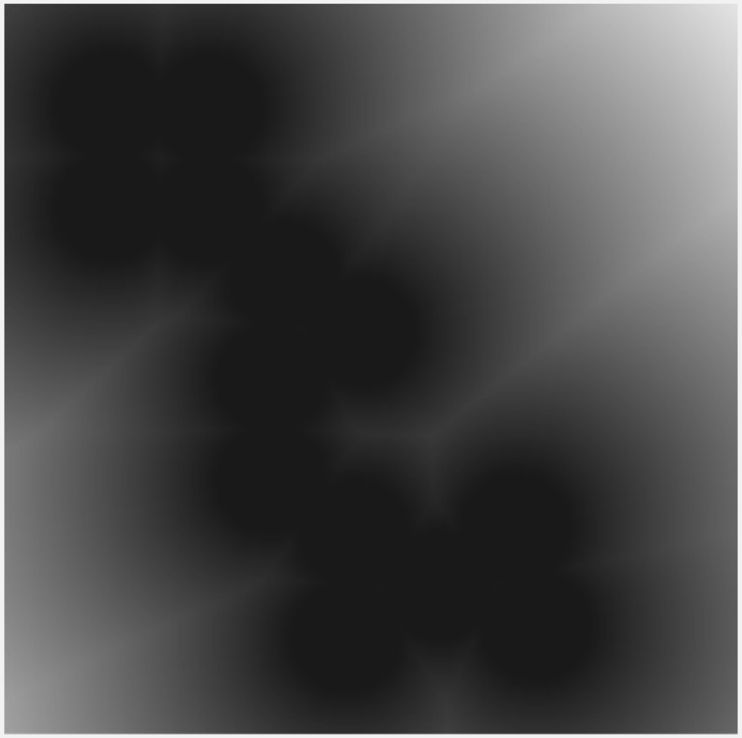
1.41	1	1	1.41	2.24
1	0	0	1	1.41
1	0	0	0	1
1.41	1	0	0	1
2.24	1.41	1	1	1.41

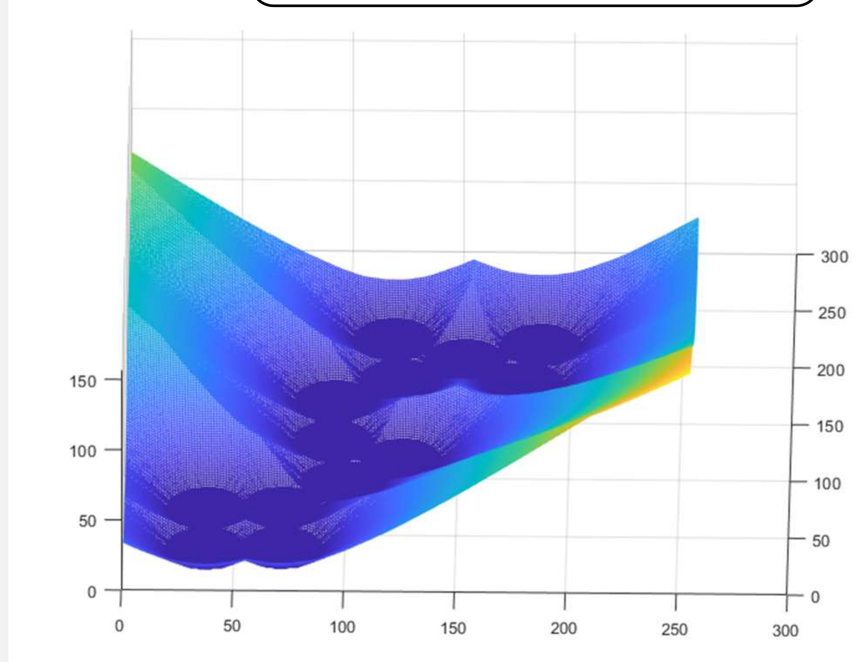
Distance transform

Note: The distance transform calculates distance of a pixel to the nearest nonzero pixel. So true pixels have a distance transform of 0.

Visualizing the mask in 3D

Note: You can use `mesh(dd)` to get a 3D plot



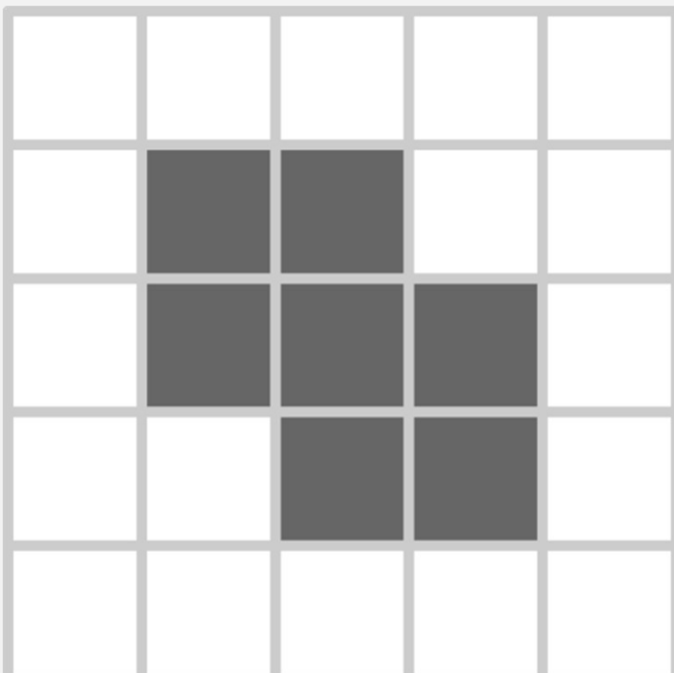


Need to manipulate the mask before using `bwdist`

1. Invert the mask using the not operator (~)
2. Compute the distance transform of the inverted mask
3. Take the negative of the transform

```
dd = -bwdist(~mask);
```

Example

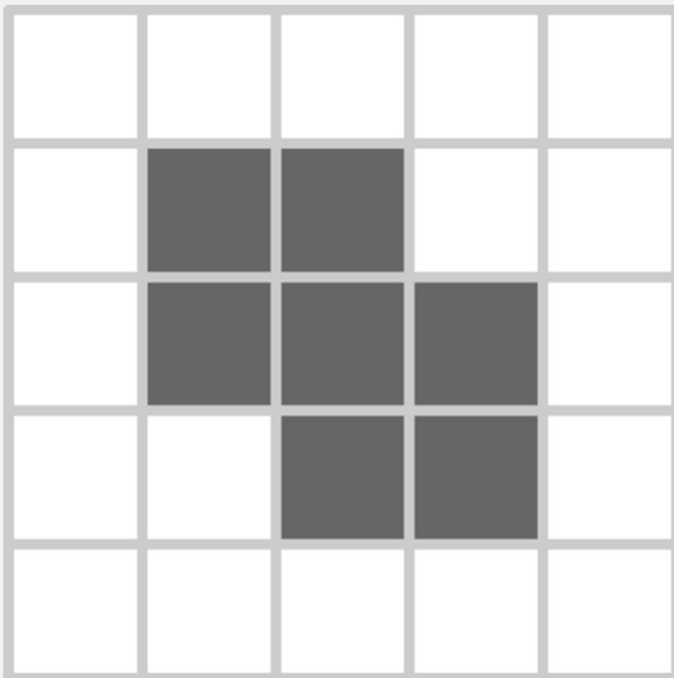


Input

0	0	0	0	0
0	1	1	0	0
0	1	1.41	1	0
0	0	1	1	0
0	0	0	0	0

Distance transform

Example



Input

0	0	0	0	0
0	-1	-1	0	0
0	-1	-1.41	-1	0
0	0	-1	-1	0
0	0	0	0	0

Distance transform

Practice

Update your script to include the following command (updating variable names as necessary)

```
dd = -bwdist(~mask);
```

Exclude regions outside the mask

- To exclude regions outside the mask, set the background regions of the mask in the resulting distance transform to `-Inf` (negative infinity)

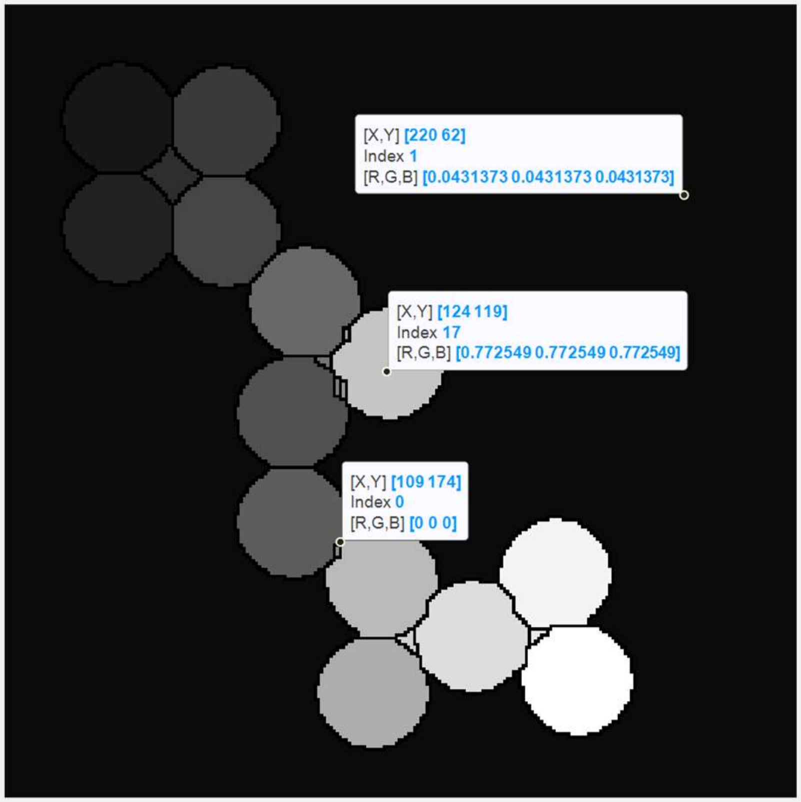
```
dd(~mask) = -Inf
```

The watershed function

```
L = watershed(dd)
```

The watershed function returns `L` = label matrix

Label matrix

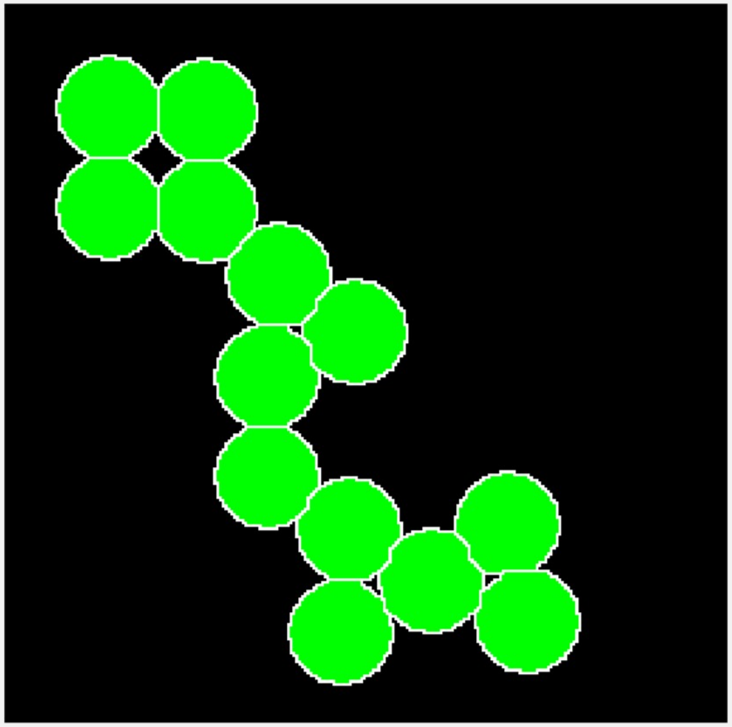


- A label matrix identifies individual objects like a mask
- Connected regions in a label matrix have the same value (not just true/false like a mask)
- **The ridge lines (regions between objects) are labeled 0**

Visualizing the segmentation results

- The ridge lines indicate the object boundaries
- To visualize these, you can plot just the regions where the label matrix is 0, e.g., using imshowpair

```
imshowpair(I, L == 0);
```



Using a label matrix with regionprops

- You can use the label matrix from the watershed algorithm in regionprops instead of a mask

- Example:

```
data = regionprops(L, 'MajorAxisLength'…)
```

Using a label matrix with `regionprops`

- The watershed algorithm will label the background (usually as region 1). This will show up as an impossibly large object in the `regionprops` data.

- Make sure to remove this from the final data by excluding the element when concatenating the data, e.g.,

```
areas = cat(1, celldata(2:end).Area);
```

Optional practice

- Go through the steps of the distance transform:
  - Why do we need to invert the mask when calculating the distance transform?
  - Why do we need to make the results of the distance transform negative?
  - Why did we set regions outside the mask to –Inf?
- For each step, run the watershed algorithm and look at the results

Questions?

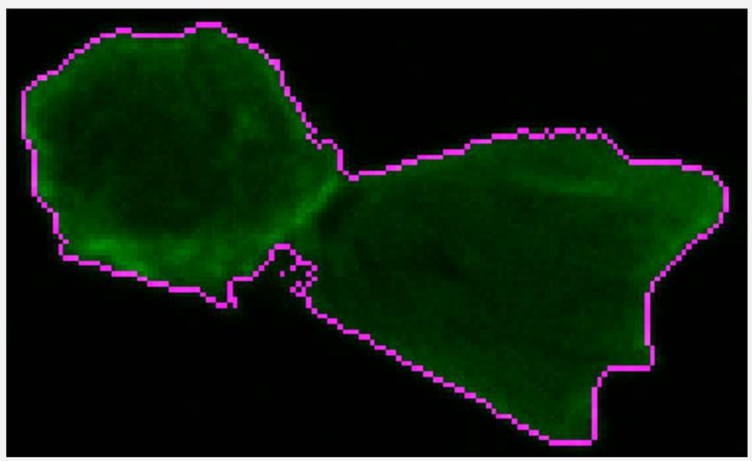
Oversegmentation and Undersegmentation

- Undersegmentation occurs when multiple objects are not divided (i.e., have the same label)

- Oversegmentation occurs when a single object is divided into multiple parts/labels

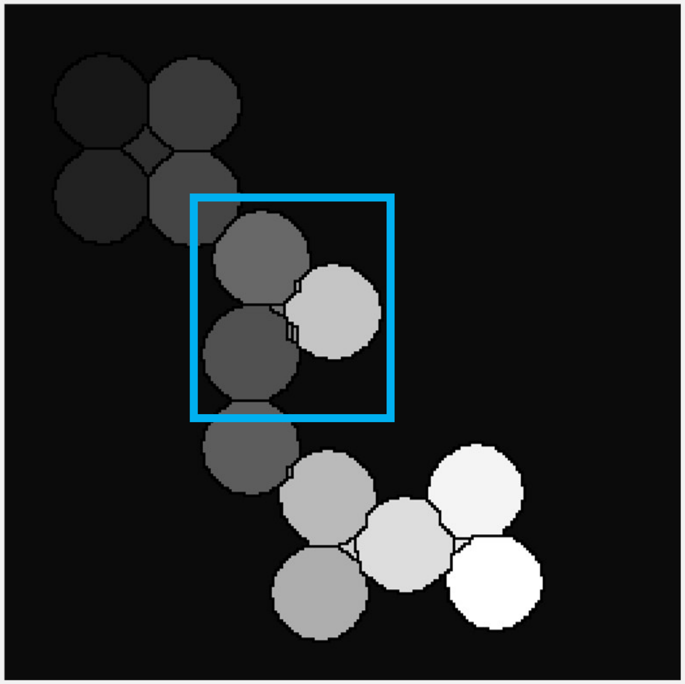
Undersegmentation

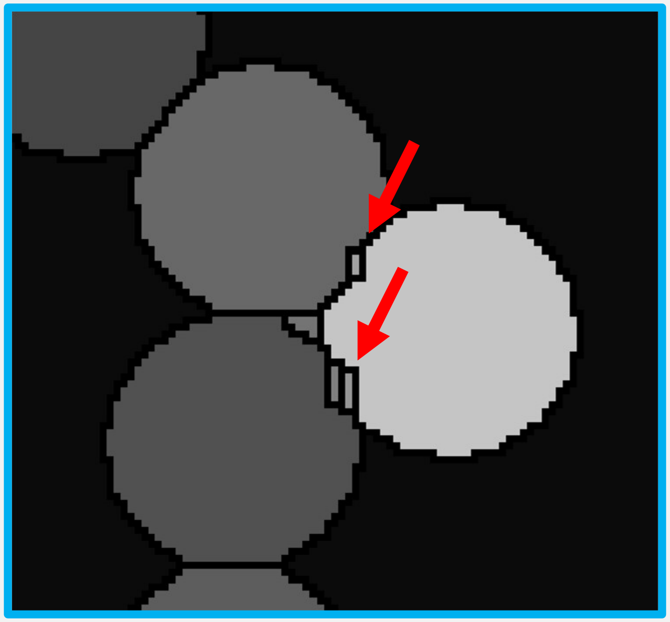
- Undersegmentation tends to occur in intensity thresholding
- Possible fixes: Watershedding



Oversegmentation

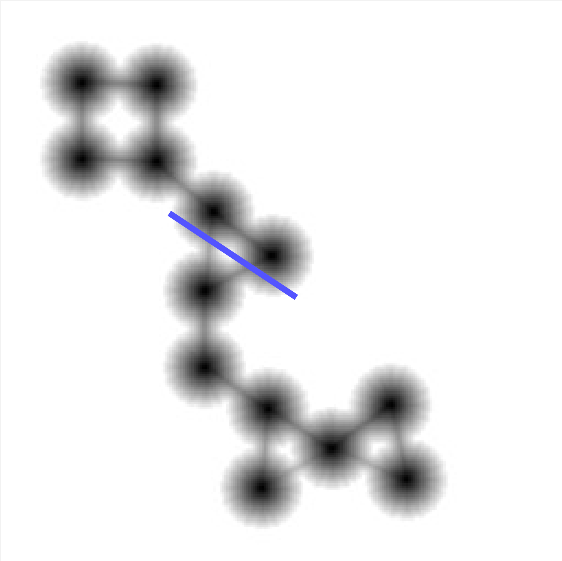
- Oversegmentation tends to occur in watershedding

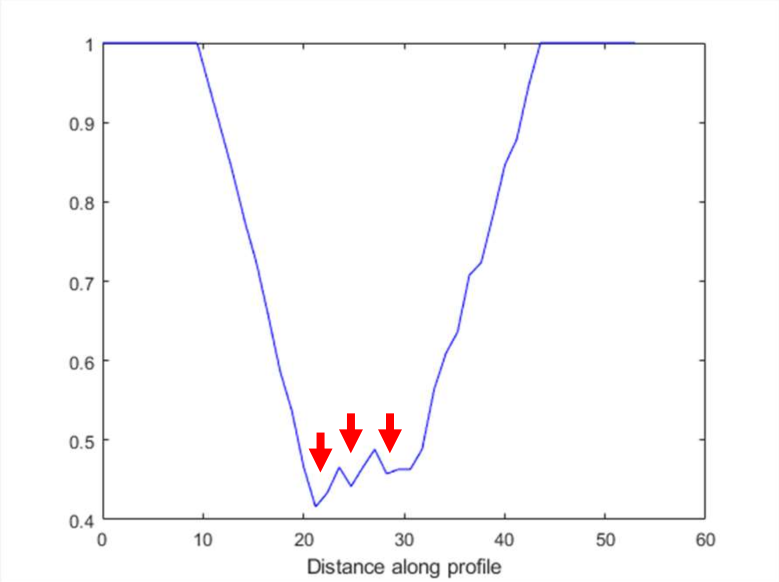




Oversegmentation

- Oversegmentation occurs due to local minima in the distance-transform image

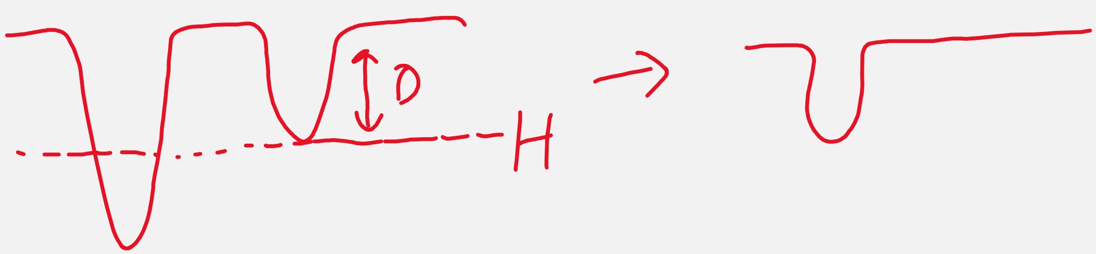




Use `imhmin` to reduce oversegmentation

```
dd2 = imhmin(dd, H)
```

Suppresses minima in the distance-transformed image `dd` with depths less than `H`



Practice

- Update your script to include `imhmin` to remove basins with depths of less than 2
- `imhmin` should go after the distance transform

```
dd2 = imhmin(dd, H)
```

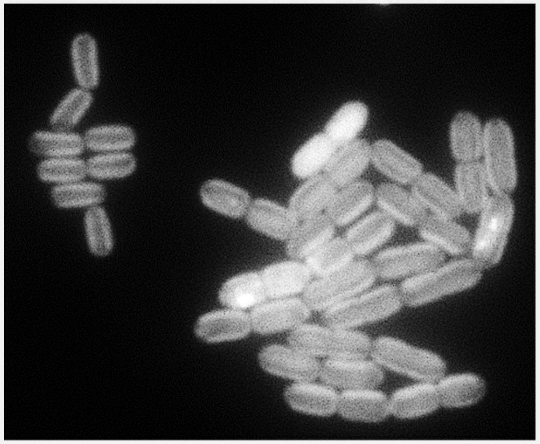
Example watershed code

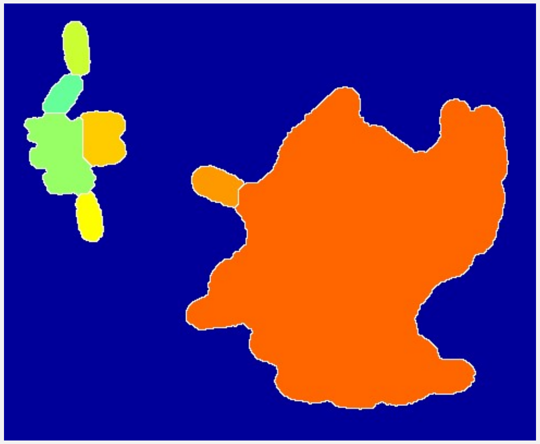
```
M = imread('circles.png');
dd = -bwdist(~M);
dd(~M) = -Inf;
dd2 = imhmin(dd, 2);
L = watershed(dd2);

%Plot segmentation results
imshowpair(M, L == 0)
```

Questions?

What are the limitations of the watershed algorithm?

- The shape of the objects in the initial mask affects the distance transform
- Thus, this algorithm works best for <u>circular objects</u> or objects that are only partly connected





Improving watershed results

- The "art" to using the watershed algorithm well is to find the correct threshold intensities and use morphological operations to refine masks that will segment properly

- Other manipulations include using markers to label objects (marker-controlled watershedding) – this is beyond the scope of this course (optional reading)
  - Koyuncu et al. PLOS One 7:e48664 (2012)
  - MATLAB Blog