+/- Message

```
## Loading required package: knitr
```

+/- R Code

Quantitative Big Imaging

author: Kevin Mader date: 6 March 2014 width: 1440 height: 900 transition: rotate

Basic Segmentation and Discrete Binary Structures

Course Outline

- 20th February - Introductory Lecture
- 27th February - Filtering and Image Enhancement (A. Kaestner)
- 6th March - **Basic Segmentation, Discrete Binary Structures**
- 13th March - Advanced Segmentation
- 20th March - Analyzing Single Objects
- 27th March - Analyzing Complex Objects
- 3rd April - Spatial Distribution
- 10th April - Statistics and Reproducibility
- 17th April - Dynamic Experiments
- 8th May - Big Data
- 15th May - Guest Lecture - Applications in Material Science
- 22th May - Project Presentations

Lesson Outline

- Motivation
- The old ways
- Thresholding
    - Other types of images
    - Selecting a good threshold
- Implementation
- Morphology

- Applications

Literature / Useful References

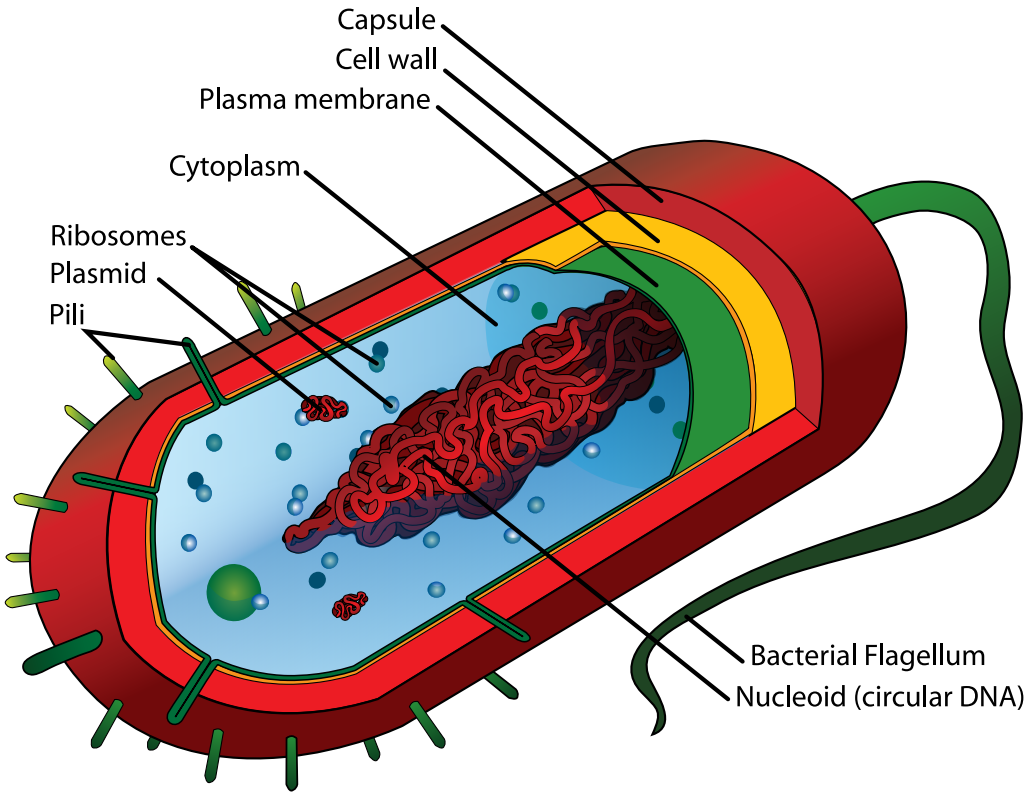
- Jean Claude, Morphometry with R
  - Online (http://link.springer.com/book/10.1007%2F978-0-387-77789-4) through ETHZ
  - Buy it (http://www.amazon.com/Morphometrics-R-Use-Julien-Claude/dp/038777789X)
- John C. Russ, “The Image Processing Handbook”,(Boca Raton, CRC Press)
  - Available online (http://dx.doi.org/10.1201/9780203881095) within domain ethz.ch (or proxy.ethz.ch / public VPN)

Motivation: Why do we do imaging experiments?

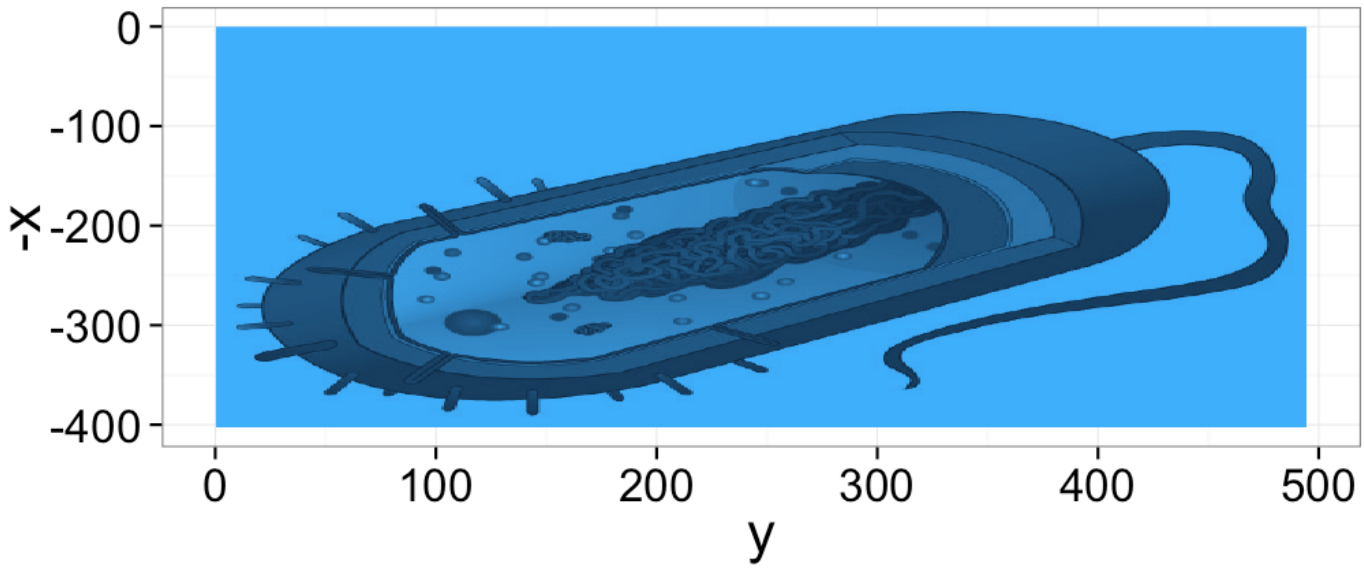
#incremental: true

- To get an idea of what is going on
- To test a hypothesis
  - Does temperature affect bubble size?
  - Is this gene important for cell shape and thus mechanosensation in bone?
  - Does higher canal volume make bones weaker?
  - Does the granule shape affect battery life expectancy?

- What we are looking at



- What we get from the imaging modality

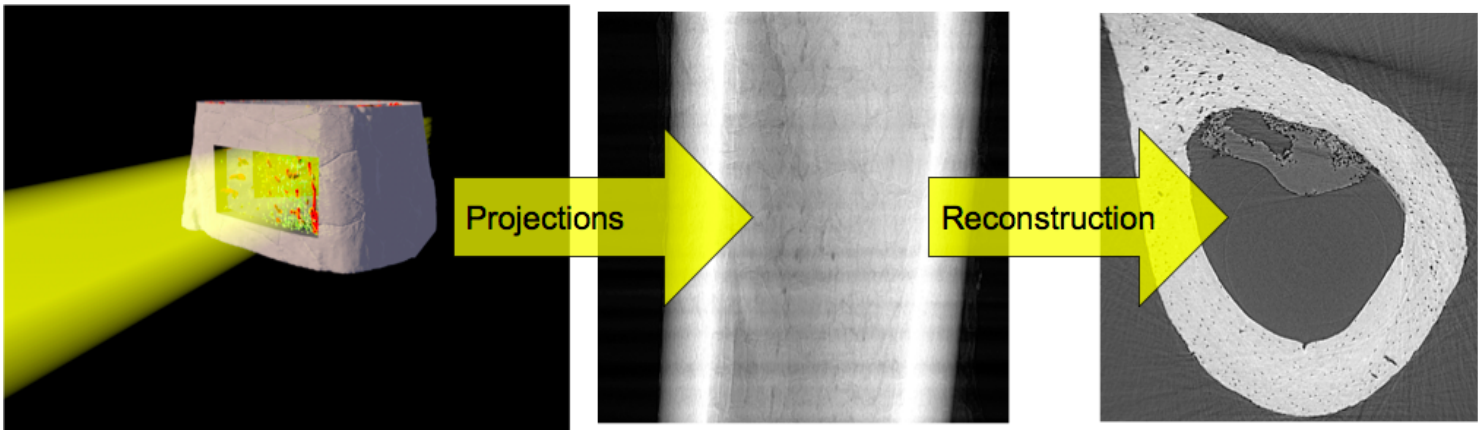


+/- R Code

To test a hypothesis

- We perform an experiment bone to see how big the cells are inside the tissue

↓



2560 x 2560 x 2160 x 32 bit = 56GB / sample

- Filtering and Preprocessing!

↓

- 20h of computer time later ...
- 56GB of less noisy data
- Way too much data, we need to reduce

What did we want in the first place

Single number:

- volume fraction,
- cell count,
- average cell stretch,
- cell volume variability

Why do we perform segmentation?

- In model-based analysis every step we peform, simple or complicated is related to an underlying model of the system we are dealing with
- *Occam's Razor* (http://en.wikipedia.org/wiki/Occams_Razor) is very important here : The simplest solution is usually the right one
    - Bayesian, neural networks optimized using genetic algorithms with Fuzzy logic has a much larger parameter space to explore, establish sensitivity in, and must perform much better and be tested

much more thoroughly than thresholding to be justified

Review: Filtering and Image Enhancement

incremental: true

- This was a noise process which was added to otherwise clean imaging data
- $$I_{measured}(x,y) = I_{sample}(x,y) + \text{Noise}(x,y)$$
- What would the perfect filter be
  - $$Filter * I_{sample}(x,y) = I_{sample}(x,y)$$
  - $$Filter * \text{Noise}(x,y) = 0$$
  - $$Filter * I_{measured}(x,y) = Filter * I_{real}(x,y) + Filter * \text{Noise}(x,y) \rightarrow \mathbf{I_{sample}(x,y)}$$
- What most filters end up doing

$$Filter * I_{measured}(x,y) = 90\% I_{real}(x,y) + 10\% \text{Noise}(x,y)$$

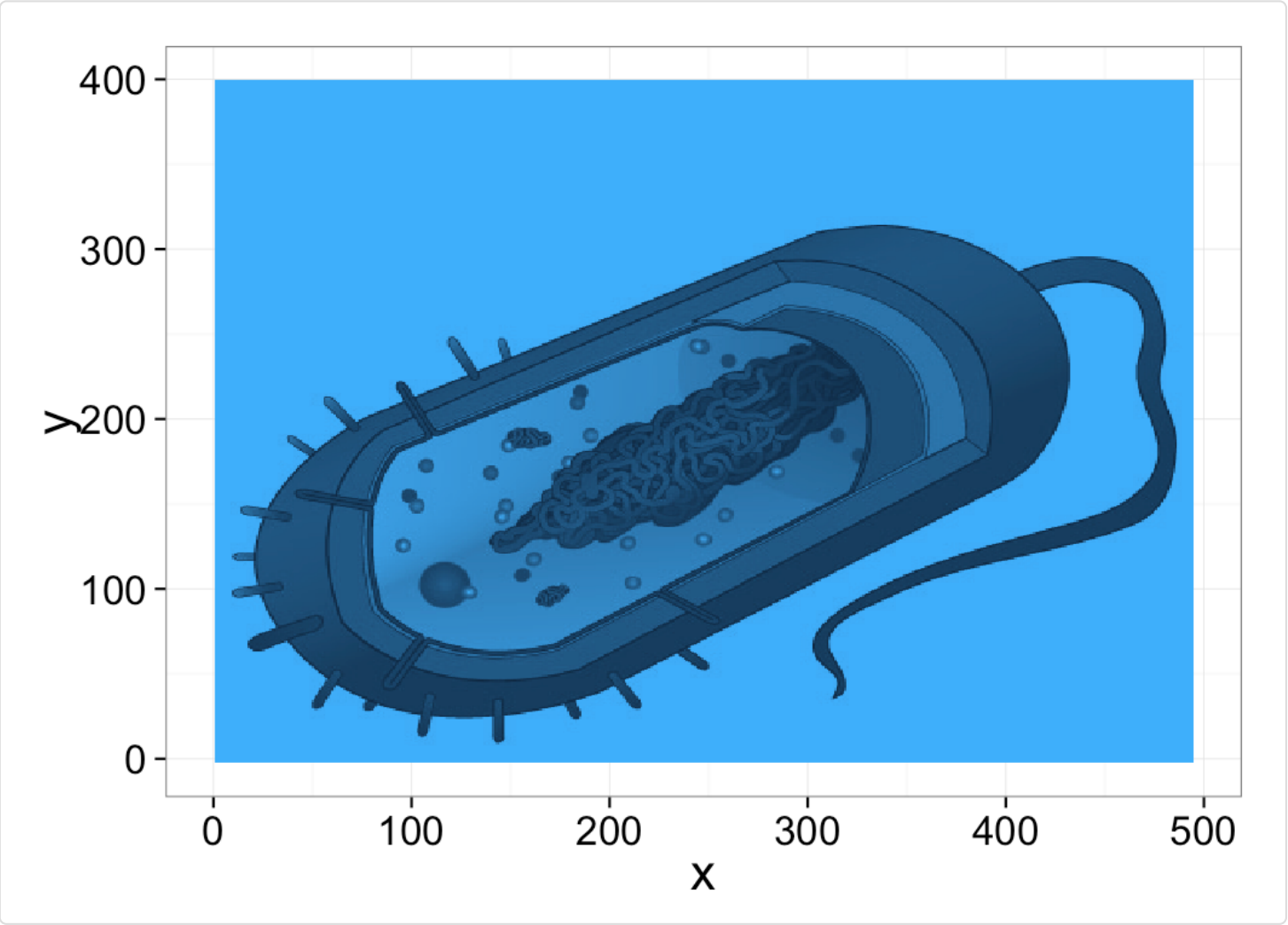
- What bad filters do

$$Filter * I_{measured}(x,y) = 10\% I_{real}(x,y) + 90\% \text{Noise}(x,y)$$

What did people used to do?

- What comes out of our detector / enhancement process

+/- R Code



- Identify objects by eye
  - Count, describe qualitatively: "many little cilia on surface", "long curly flaggelum", "elongated nuclear structure"
- Morphometrics
  - Trace the outline of the object (or sub-structures)
  - Can calculate the area by using equal-weight-paper and employing the "cut-and-weigh (http://ion.chem.usu.edu/%7Esbialkow/Classes/361/GC/GC.html)" method

Quantitative Analysis

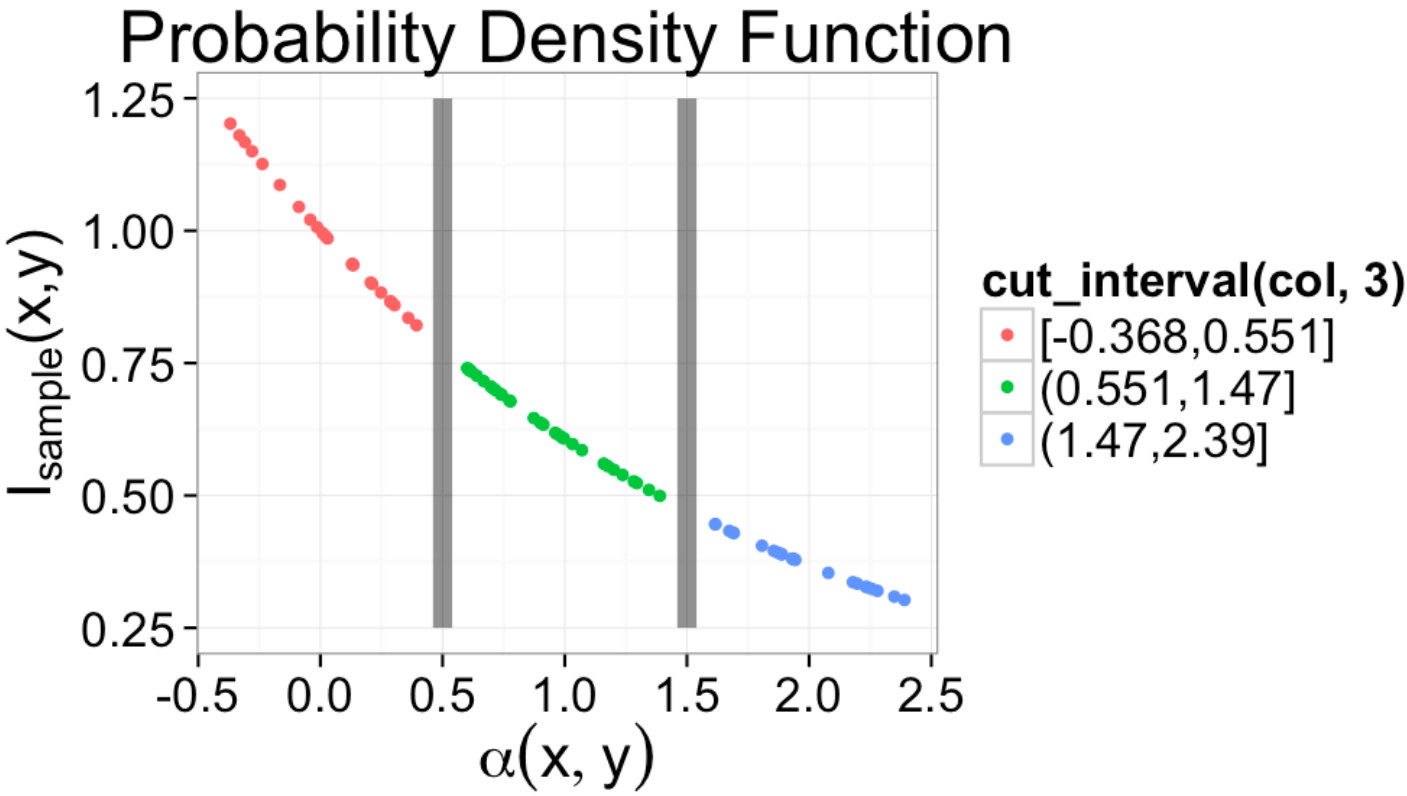
- For segmentation this model is:
  - there are 2 (or more) distinct components that make up the image
  - these components are distinguishable by their values (or vectors, colors, tensors, ...)
  - For absorption/attenuation microscopy, Beer-Lambert Law (http://en.wikipedia.org/wiki/Attenuation_coefficient)

$$I_{detector} = I_{source} \exp(-\alpha d)$$

- Different components have a different α based on the strength of the interaction between the light and the chemical / nuclear structure of the material

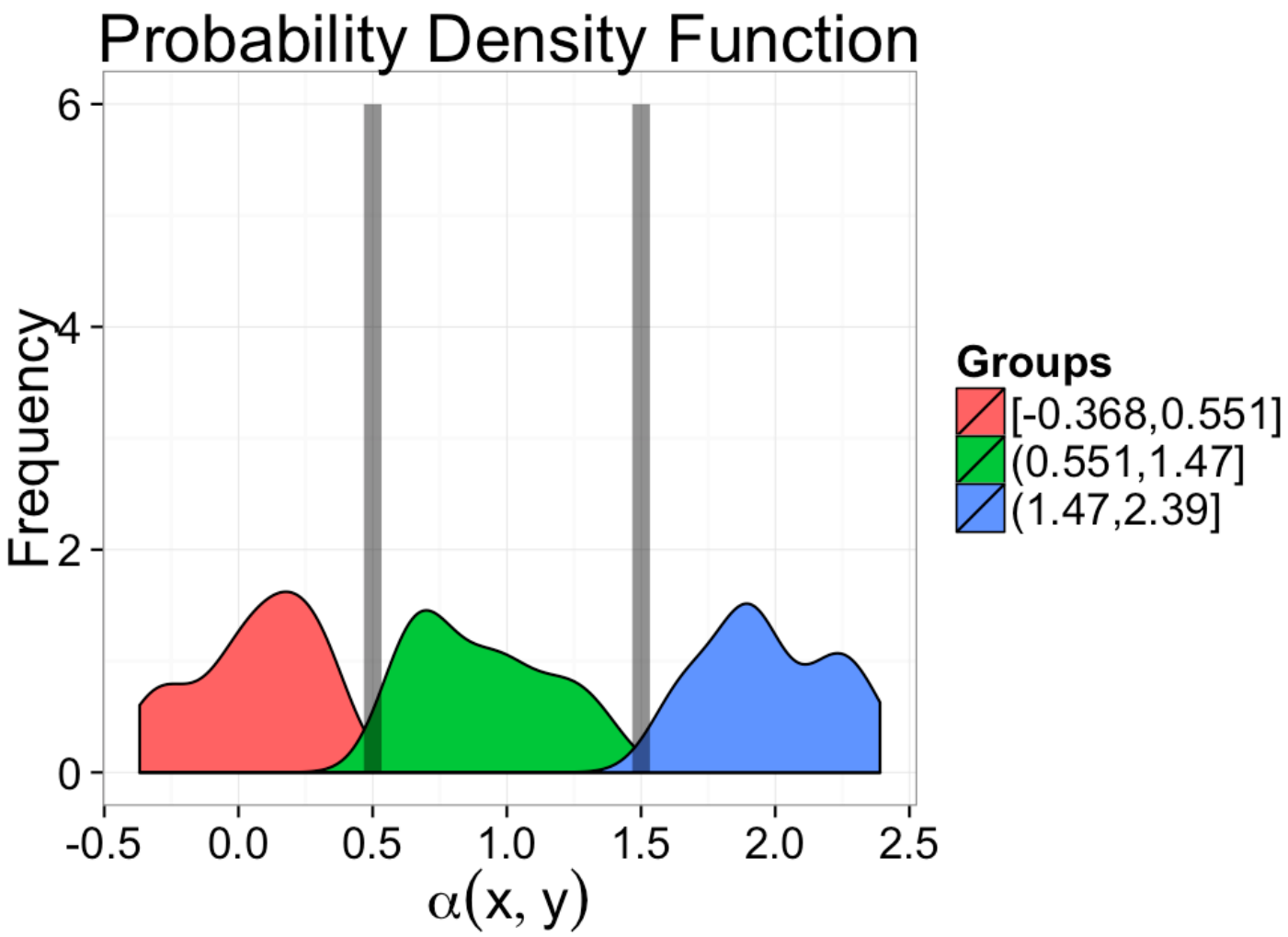
$$I_{sample}(x,y) = I_{source} \exp(-\alpha(x,y) d)$$

$$\alpha = f(N, Z, \sigma, \cdots)$$



+/- R Code

+/- R Code



Where does segmentation get us?

incremental: true

- We convert a decimal value (or something even more complicated like 3 values for RGB images, a spectrum for hyperspectral imaging, or a vector / tensor in a mechanical stress field)
- to a single, discrete value (usually true or false, but for images with phases it would be each phase, e.g. bone, air, cellular tissue)
- **2560 x 2560 x 2160 x 32 bit = 56GB / sample**

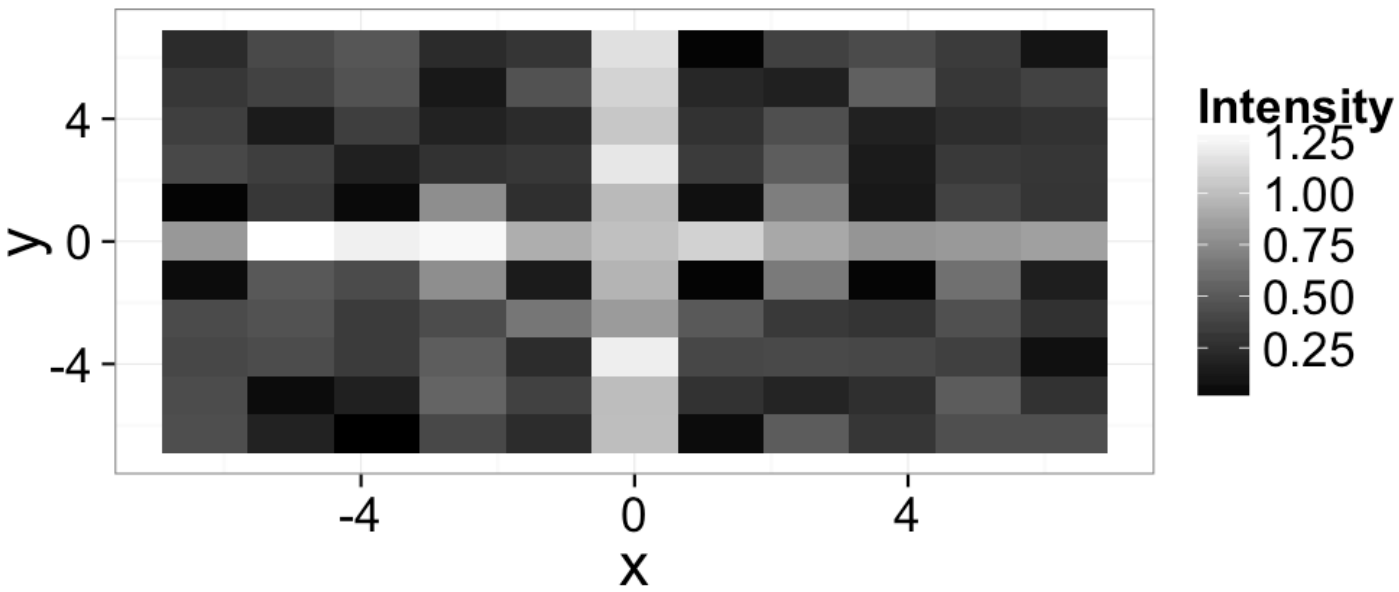
↓

- 2560 x 2560 x 2160 x **1 bit** = 1.75GB / sample

Applying a threshold to an image

Start out with a simple image of a cross with added noise

$$I(x,y) = f(x,y)$$

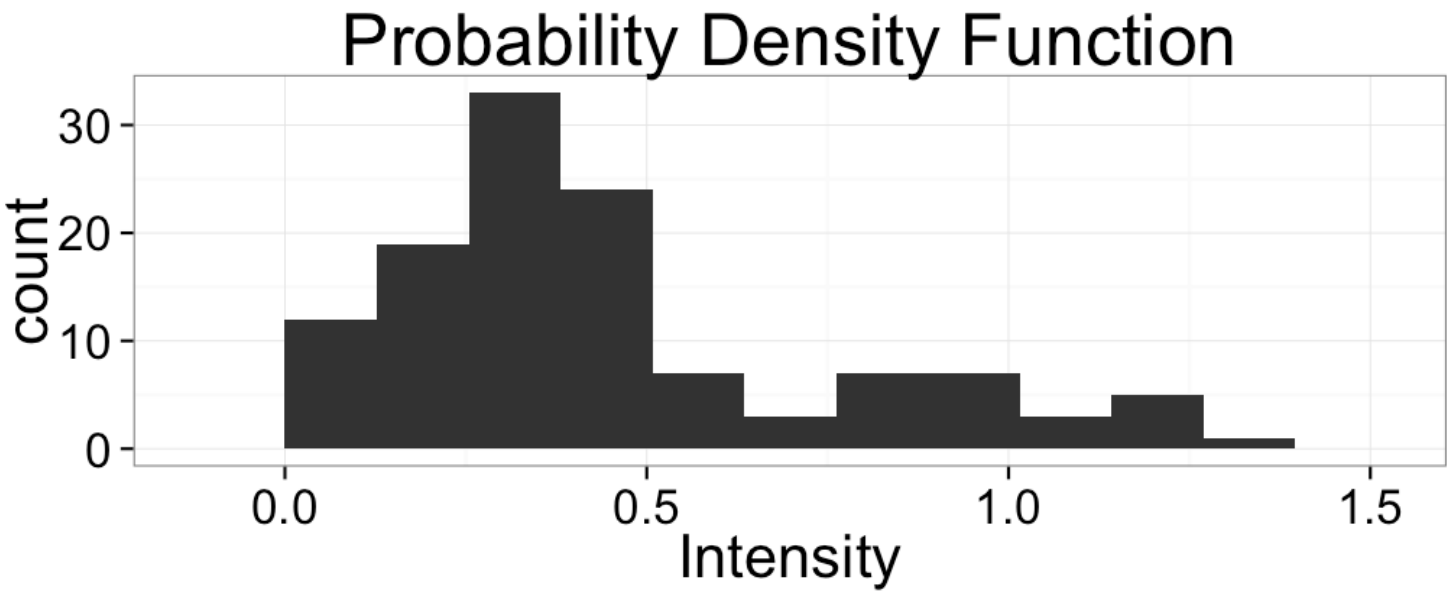


+/- R Code

The intensity can be described with a probability density function

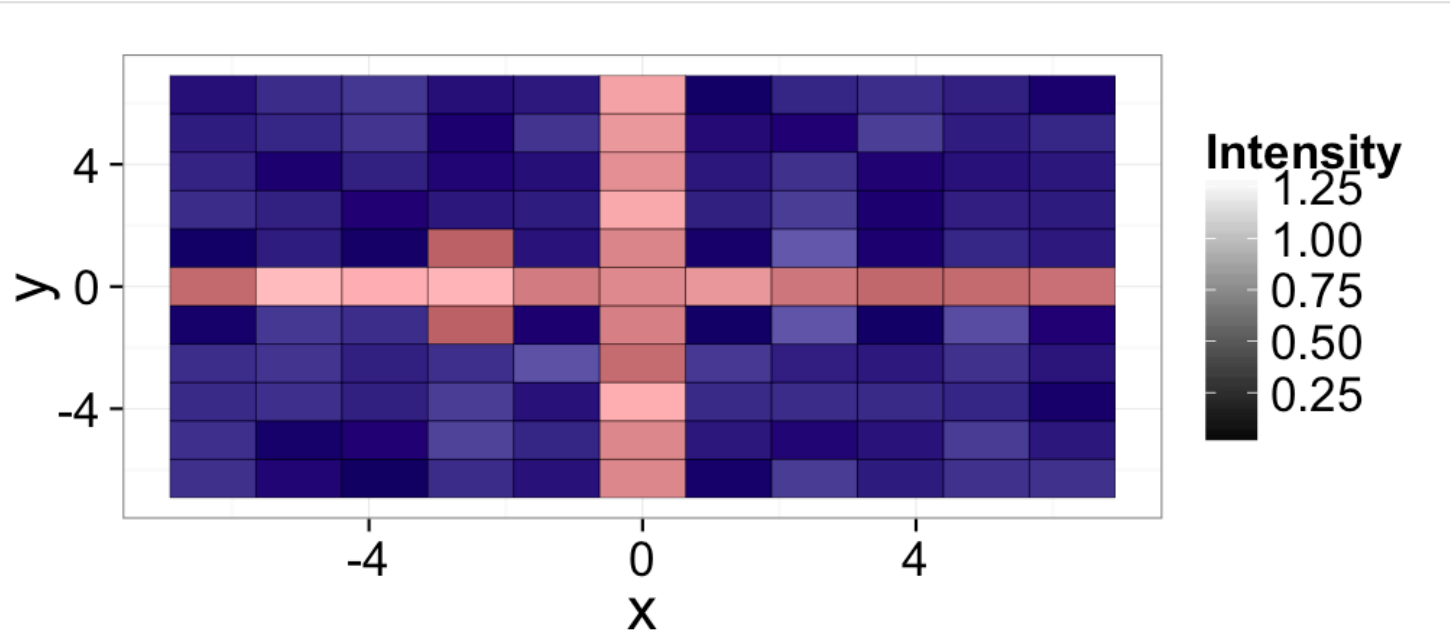
$$P_f(x,y)$$

+/- R Code



Applying a threshold to an image

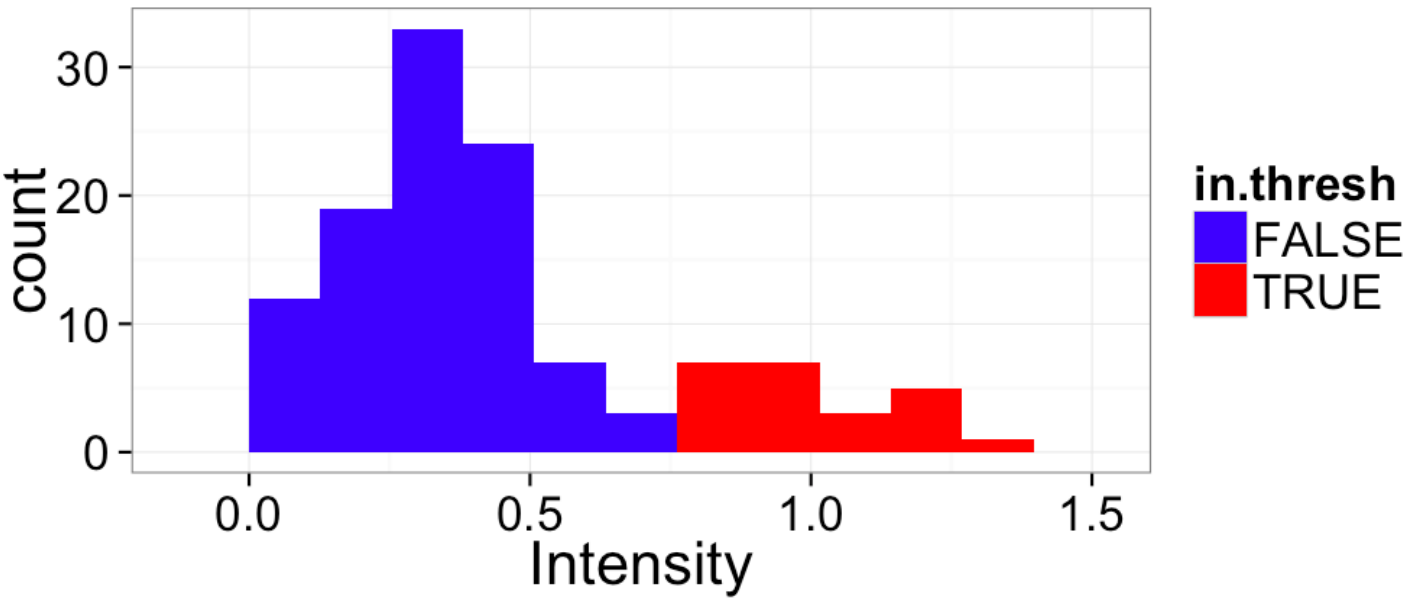
By examining the image and probability distribution function, we can *deduce* that the underyling model is a whitish phase that makes up the cross and the darkish background



+/- R Code

Applying the threshold is a deceptively simple operation

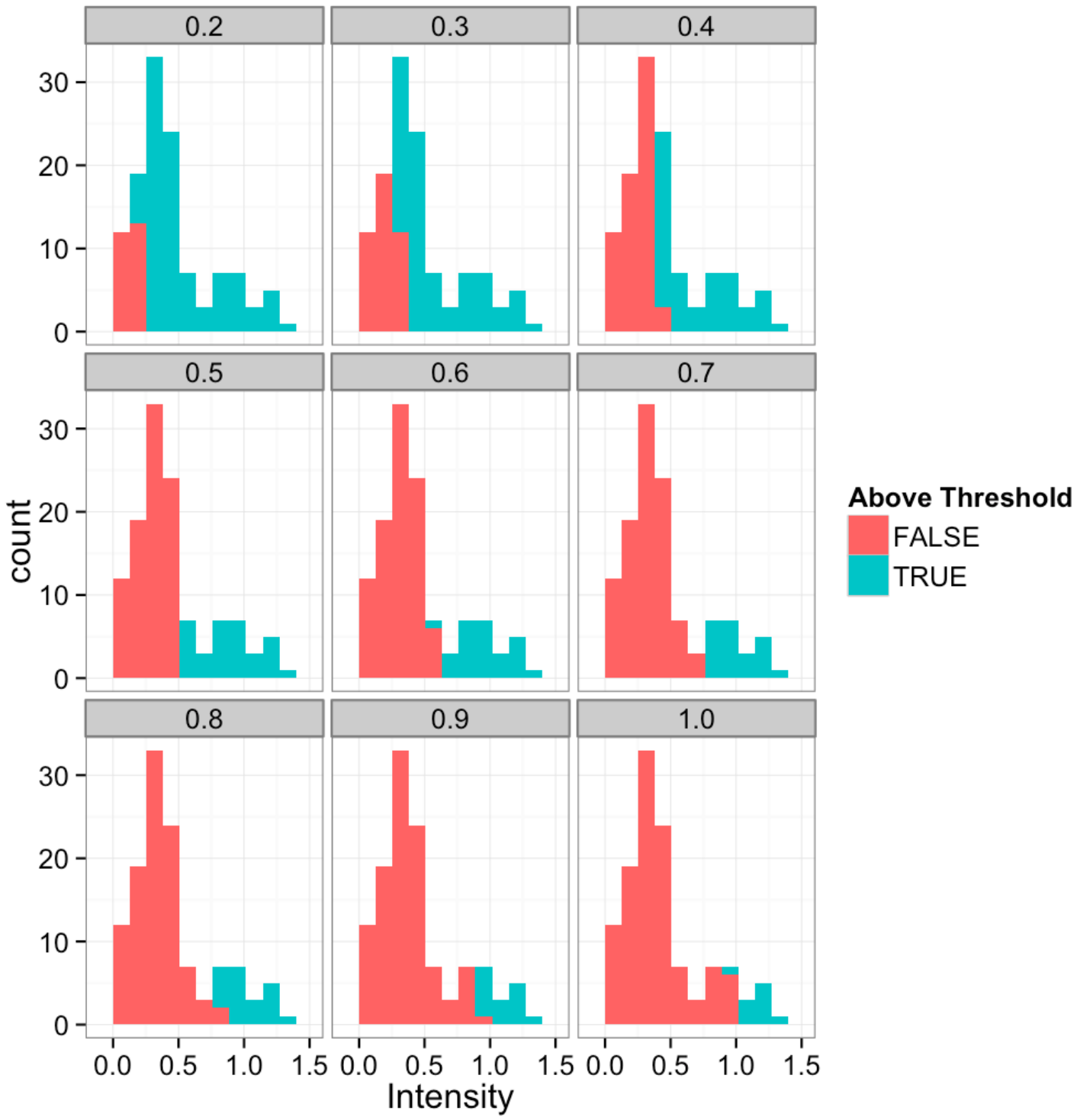
$$I(x,y) = \begin{cases} 1, & f(x,y) \geq 0.5 \\ 0, & f(x,y) < 0.5 \end{cases}$$



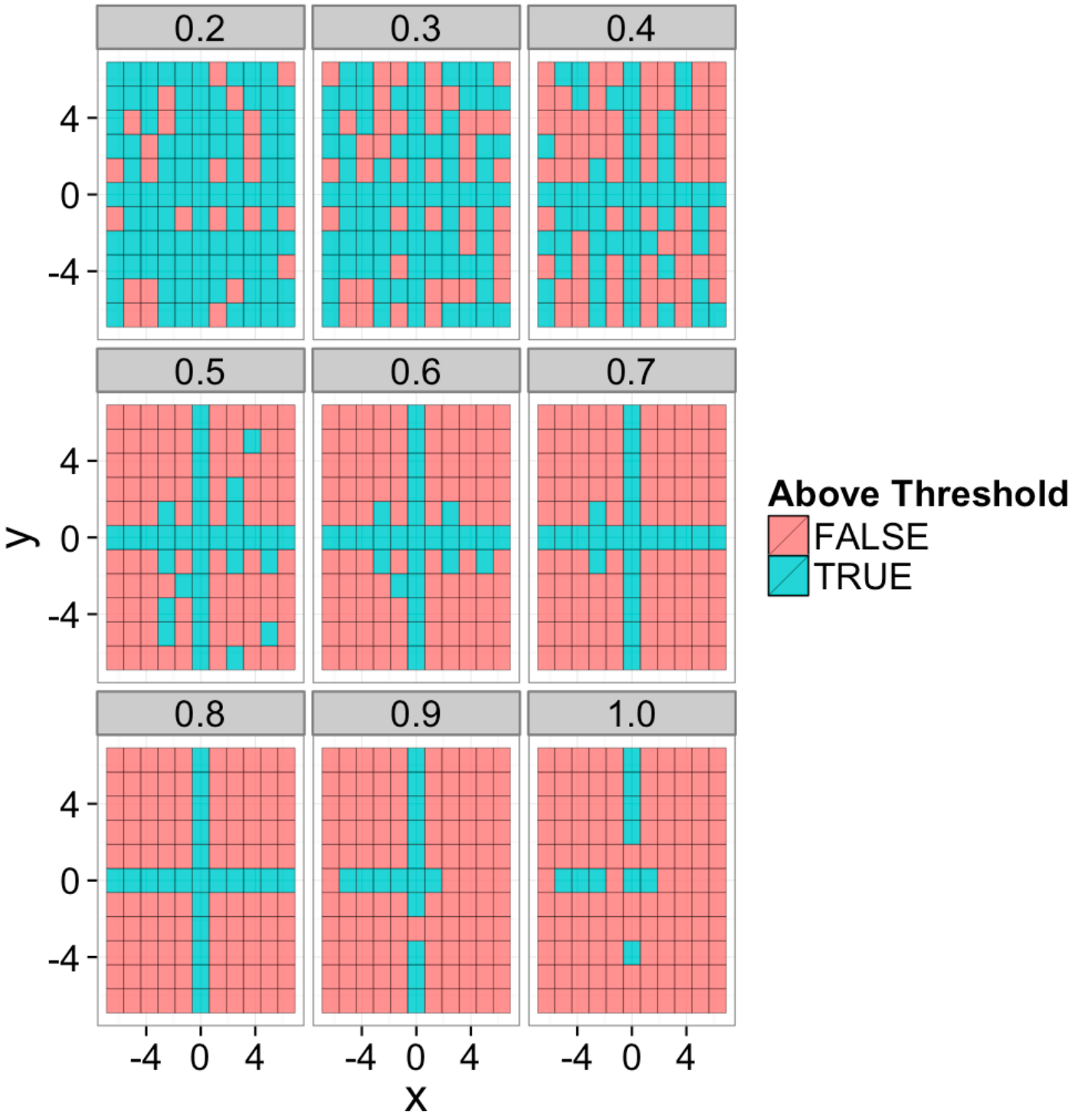
+/- R Code

Various Thresholds

+/- R Code

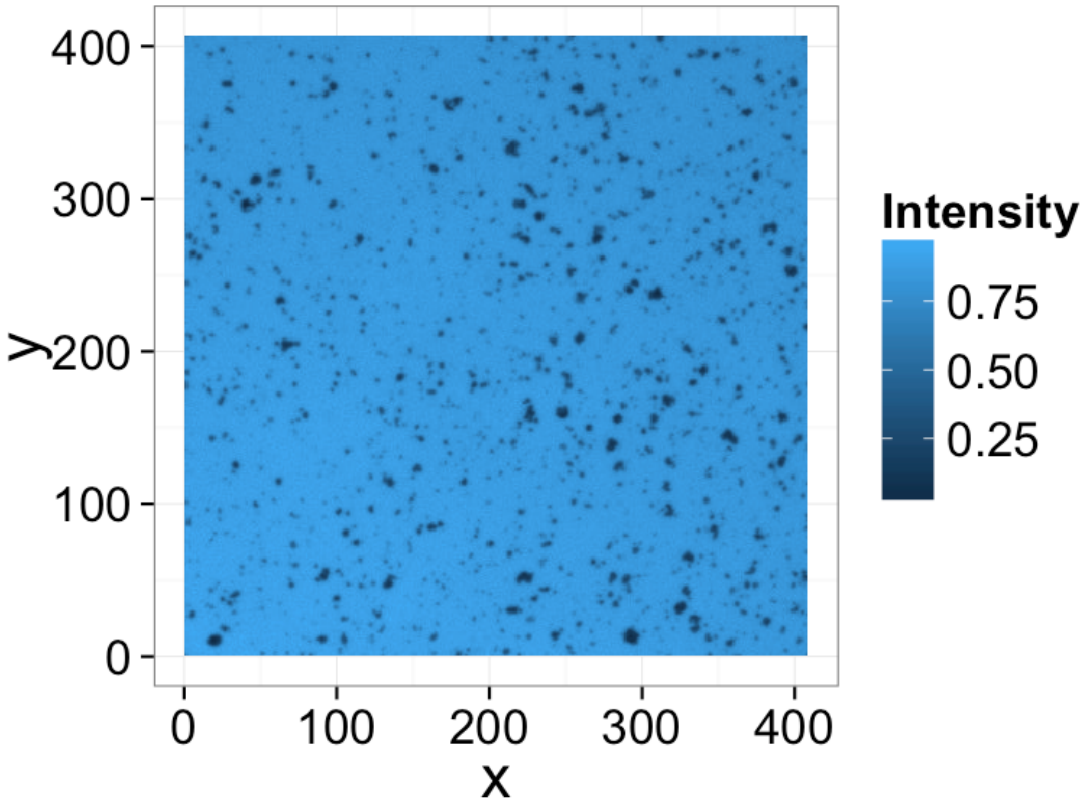

0.2
0.3
0.4
0.5
0.6
0.7
0.8
0.9
1.0
count
Intensity
0
10
20
30
0.0
0.5
1.0
1.5
Above Threshold
FALSE
TRUE

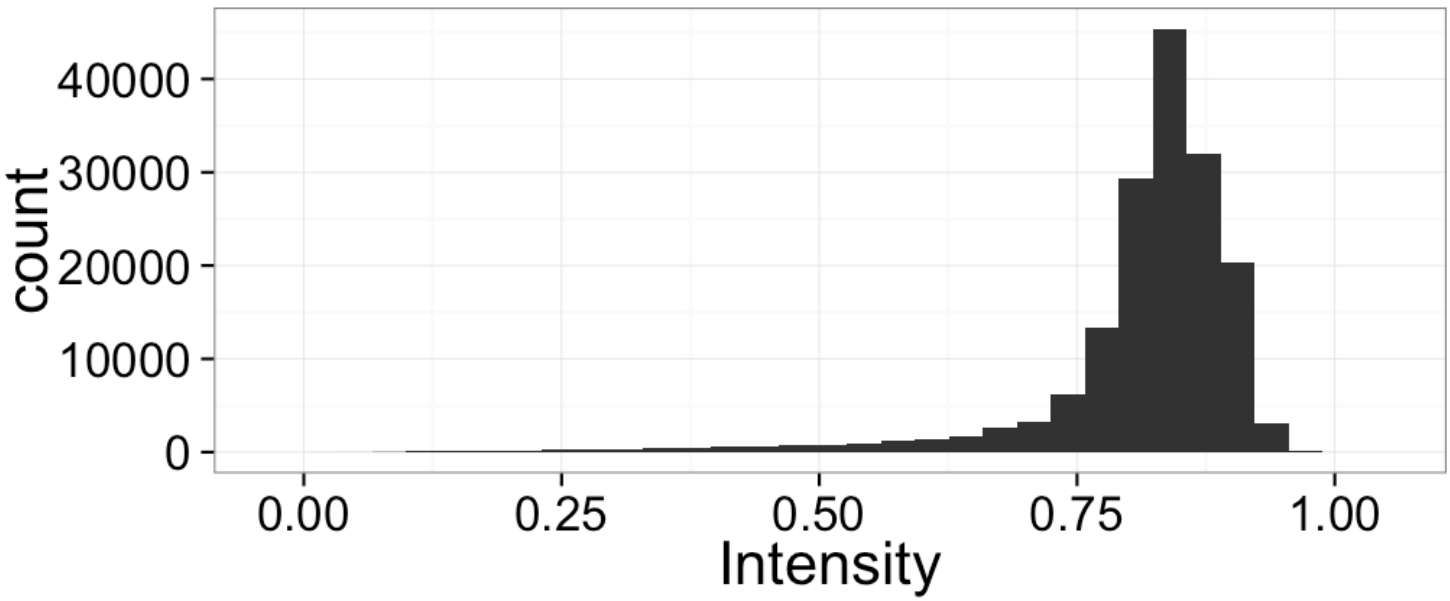
+/- R Code



Segmenting Cells

+/- R Code



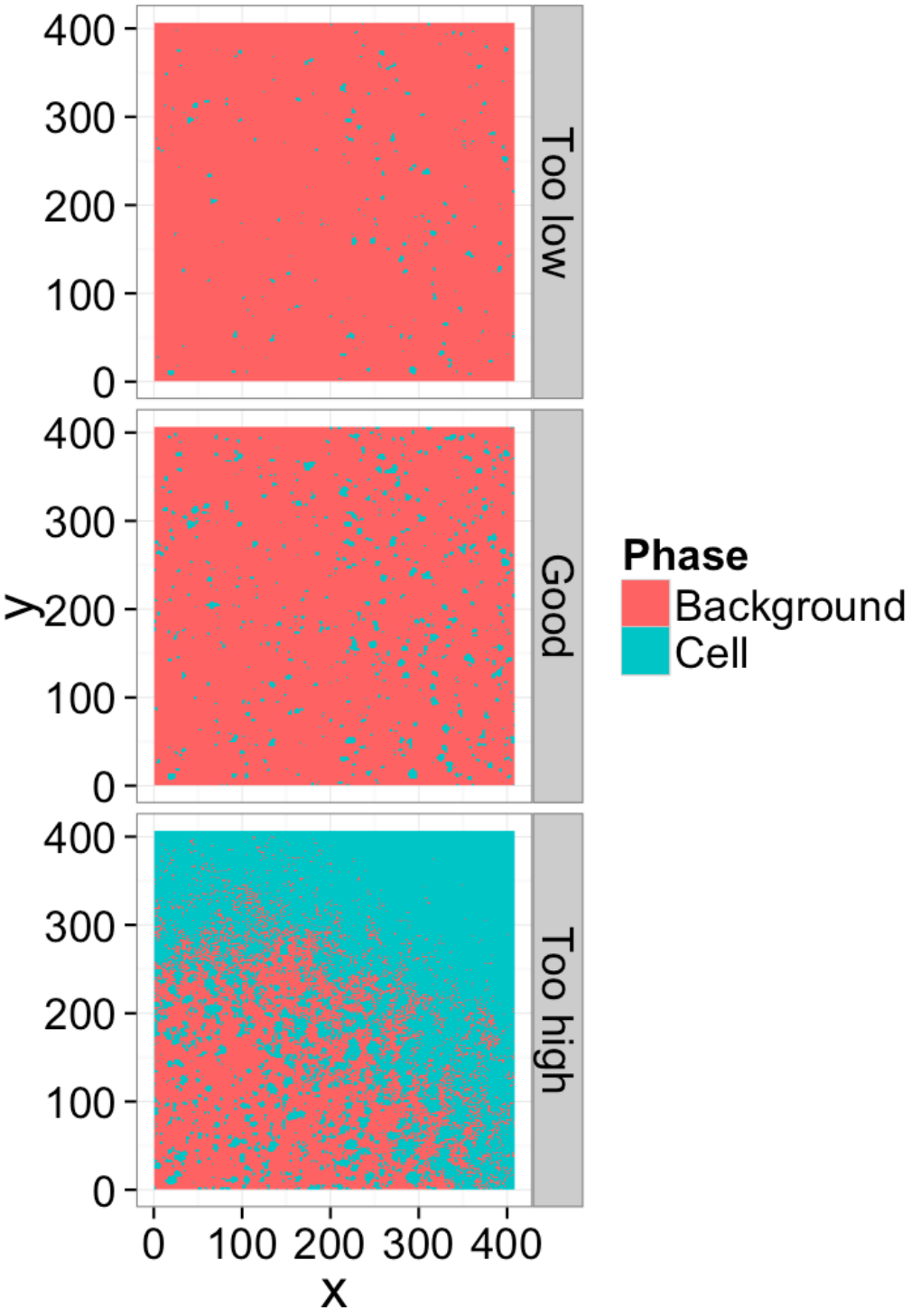


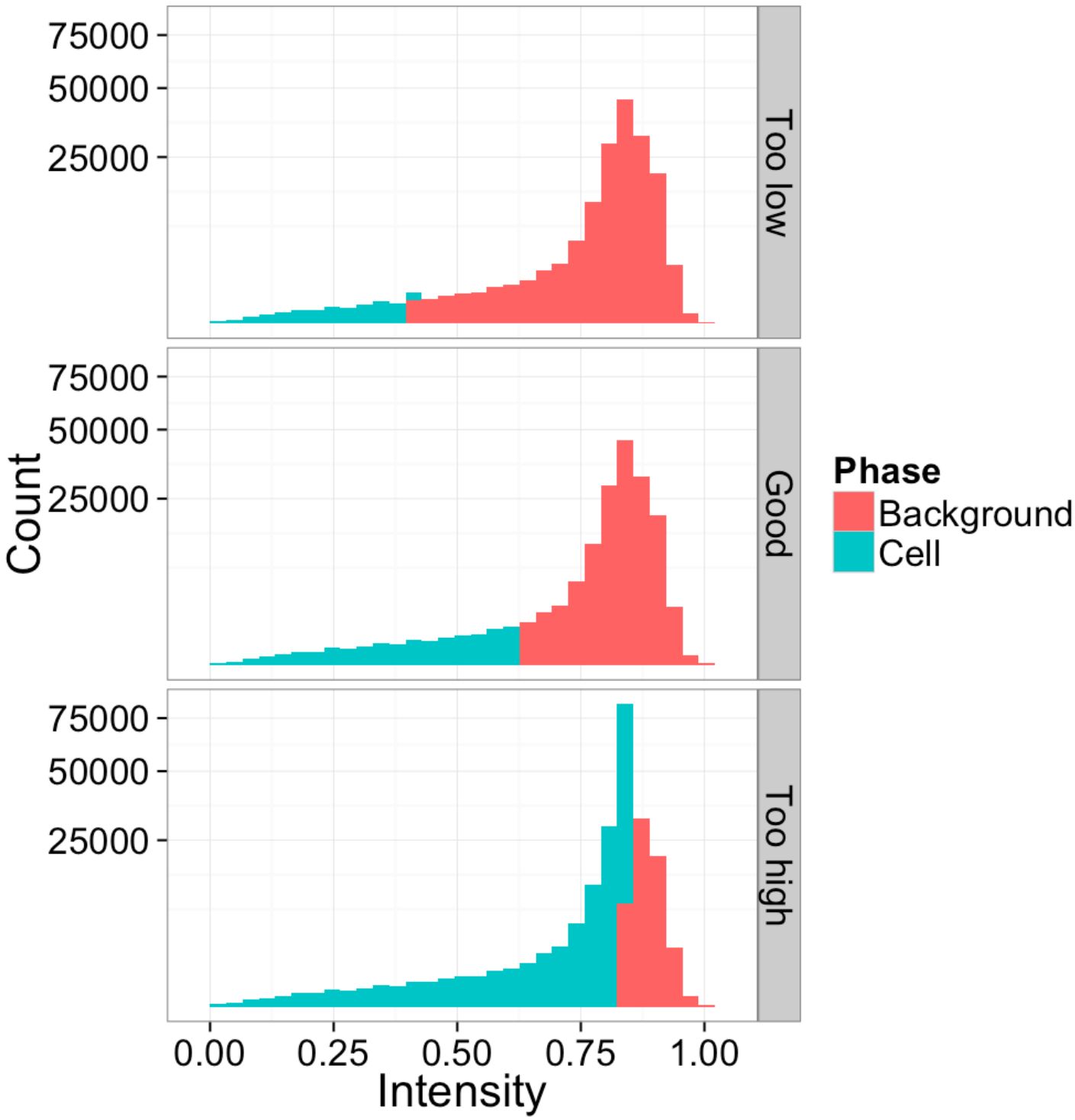
- We can peform the same sort of analysis with this image of cells
- This time we can derive the model from the basic physics of the system
  - The field is illuminated by white light of nearly uniform brightness

- Cells absorb light causing darker regions to appear in the image
- *Lighter* regions have no cells
- **Darker** regions have cells

Different Threshold Values

+/- R Code

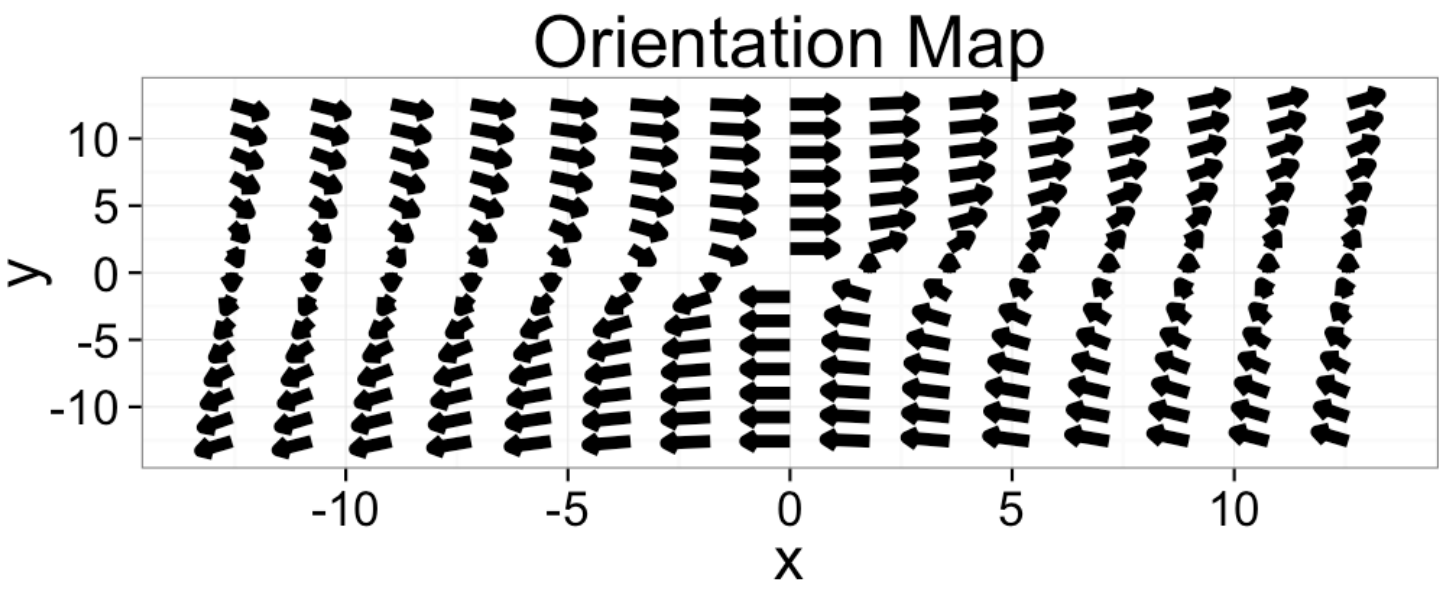

Too low
Good
Too high
y
x
0
100
200
300
400
Phase
Background
Cell



Other Image Types

While scalar images are easiest, it is possible for any type of image

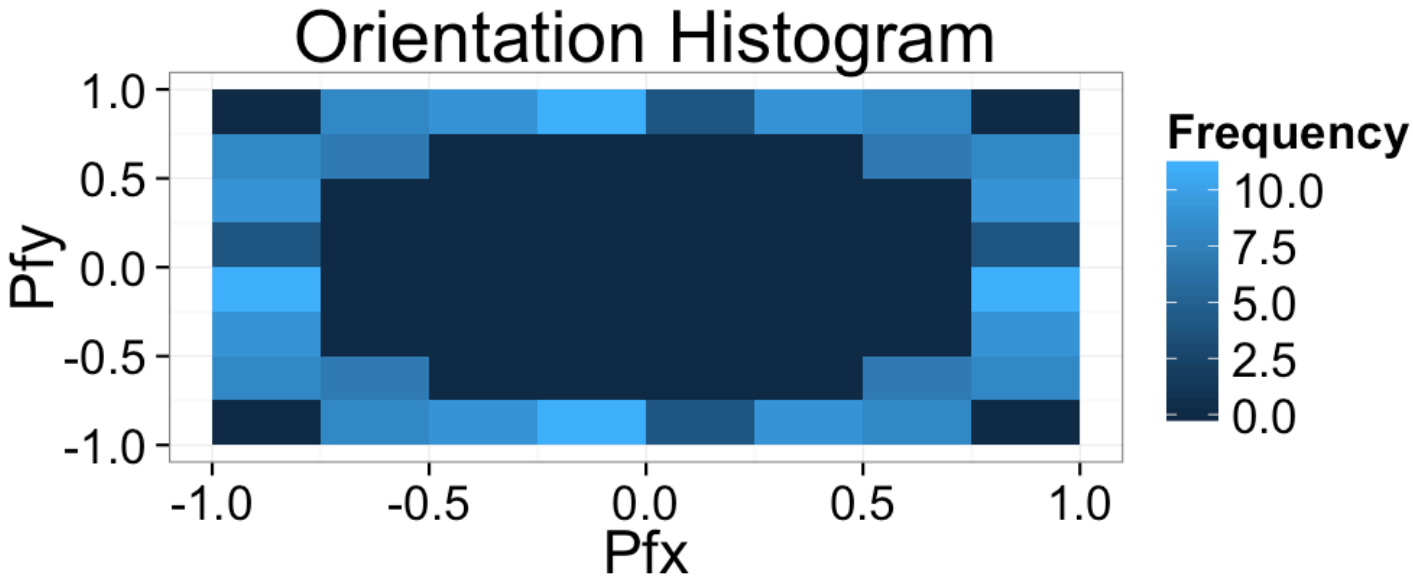
$$I(x,y) = \vec{f}(x,y)$$



+/- R Code

The intensity can be described with two seperate or a single joint probability density function

$$P_{\vec{f}\cdot\vec{i}}(x,y), P_{\vec{f}\cdot\vec{j}}(x,y)$$

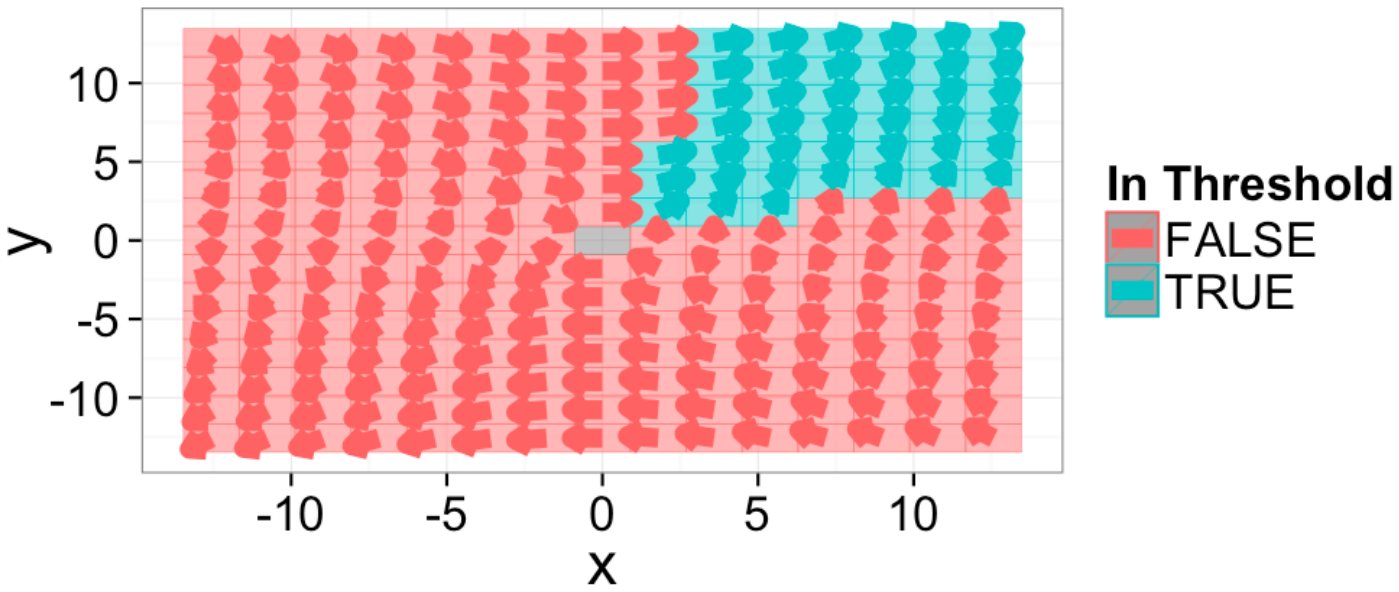


+/- R Code

Applying a threshold

A threshold is now more difficult to apply since there are now two distinct variables to deal with. The standard approach can be applied to both

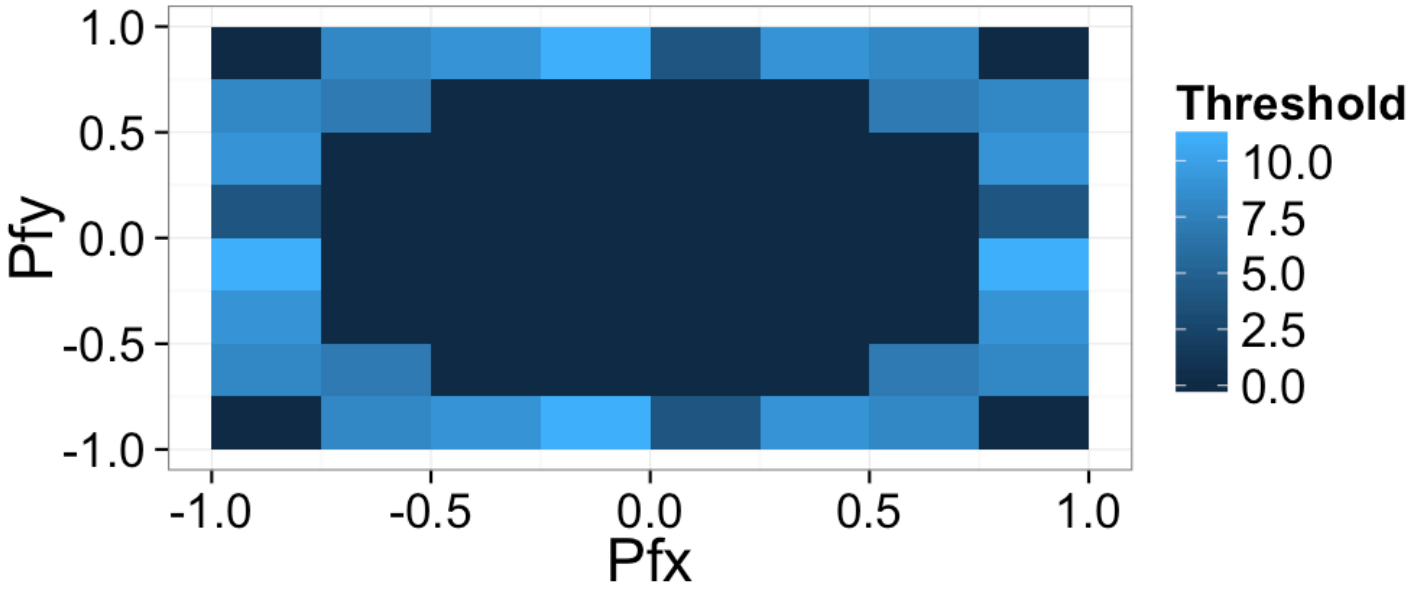
$$I(x,y) = \begin{cases} 1, & \vec{f}_x(x,y) \geq 0.25 \text{ and} \\ & \vec{f}_y(x,y) \geq 0.25 \\ 0, & \text{otherwise} \end{cases}$$



+/- R Code

This can also be shown on the joint probability distribution as

+/- R Code

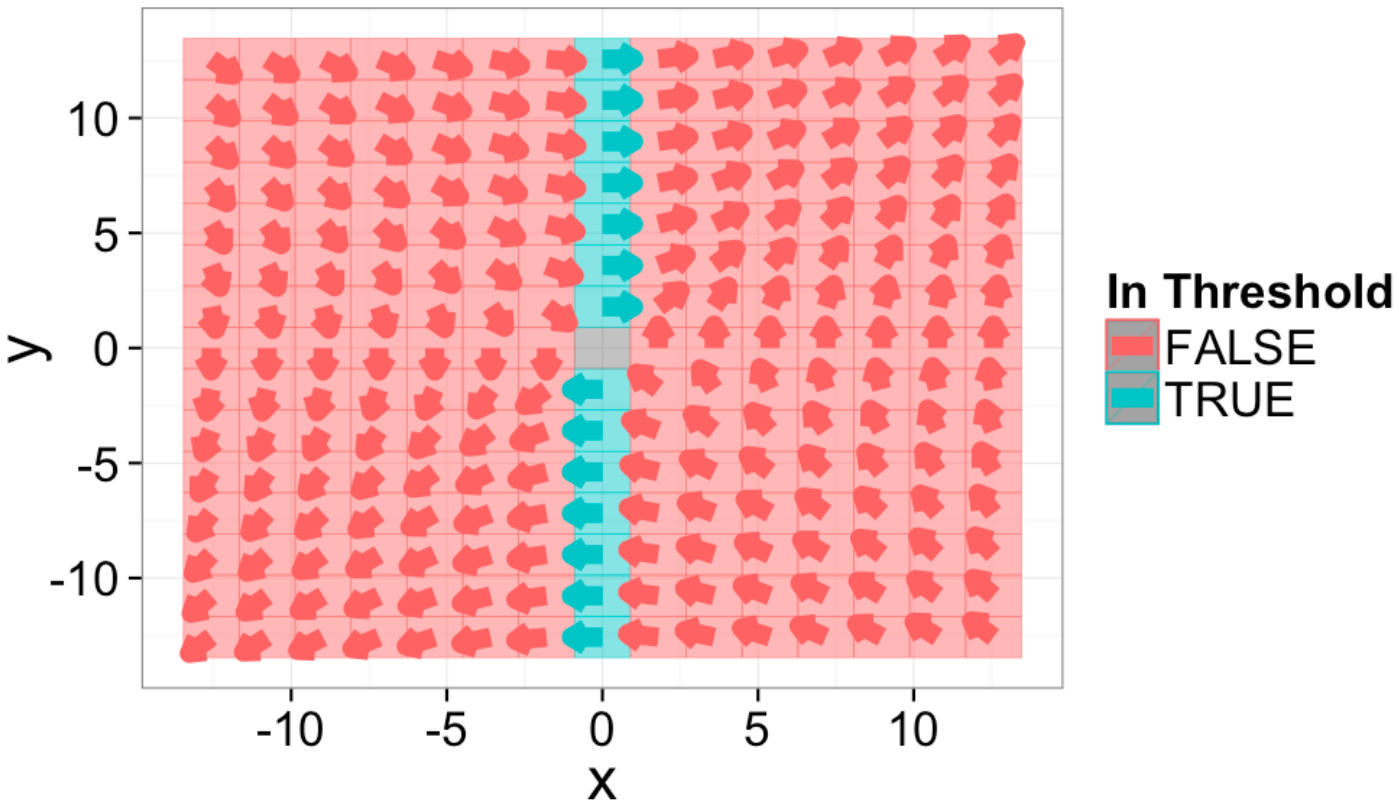


Applying a threshold

Given the presence of two variables; however, more advanced approaches can also be investigated. For example we can keep only components parallel to the x axis by using the dot product.

$$I(x,y) = \begin{cases} 1, & |\vec{f}(x,y) \cdot \vec{i}| = 1 \\ 0, & \text{otherwise} \end{cases}$$

+/- R Code



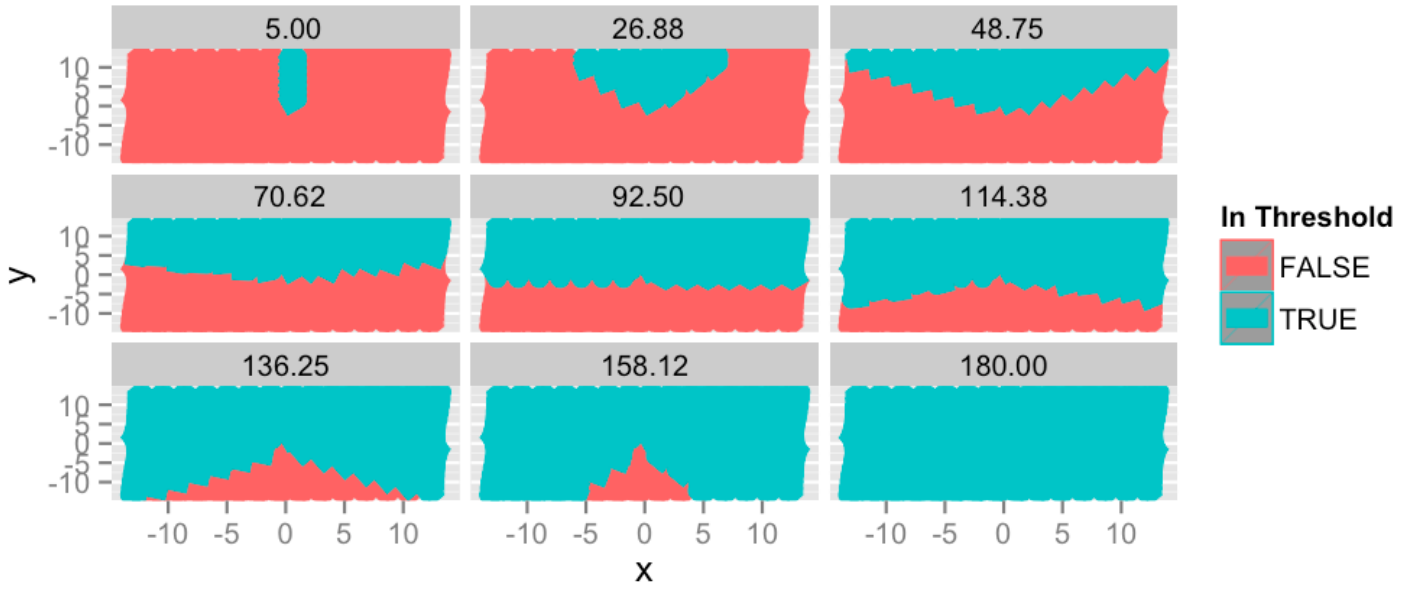
Looking at Orientations

We can tune the angular acceptance by using the fact

$$\vec{x} \cdot \vec{y} = |\vec{x}||\vec{y}| \cos(\theta_{x \to y})$$

$$I(x,y) = \begin{cases} 1, & \cos^{-1}(\vec{f}(x,y) \cdot \vec{i}) \leq \theta^\circ \\ 0, & \text{otherwise} \end{cases}$$

+/- R Code



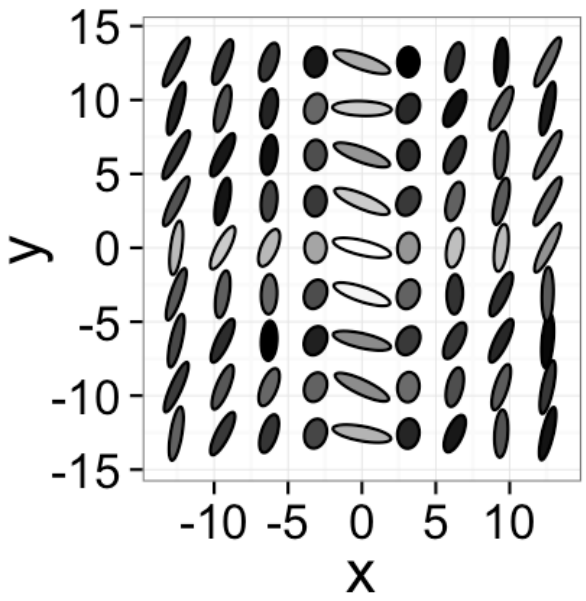
+/- R Code

Other Image Types

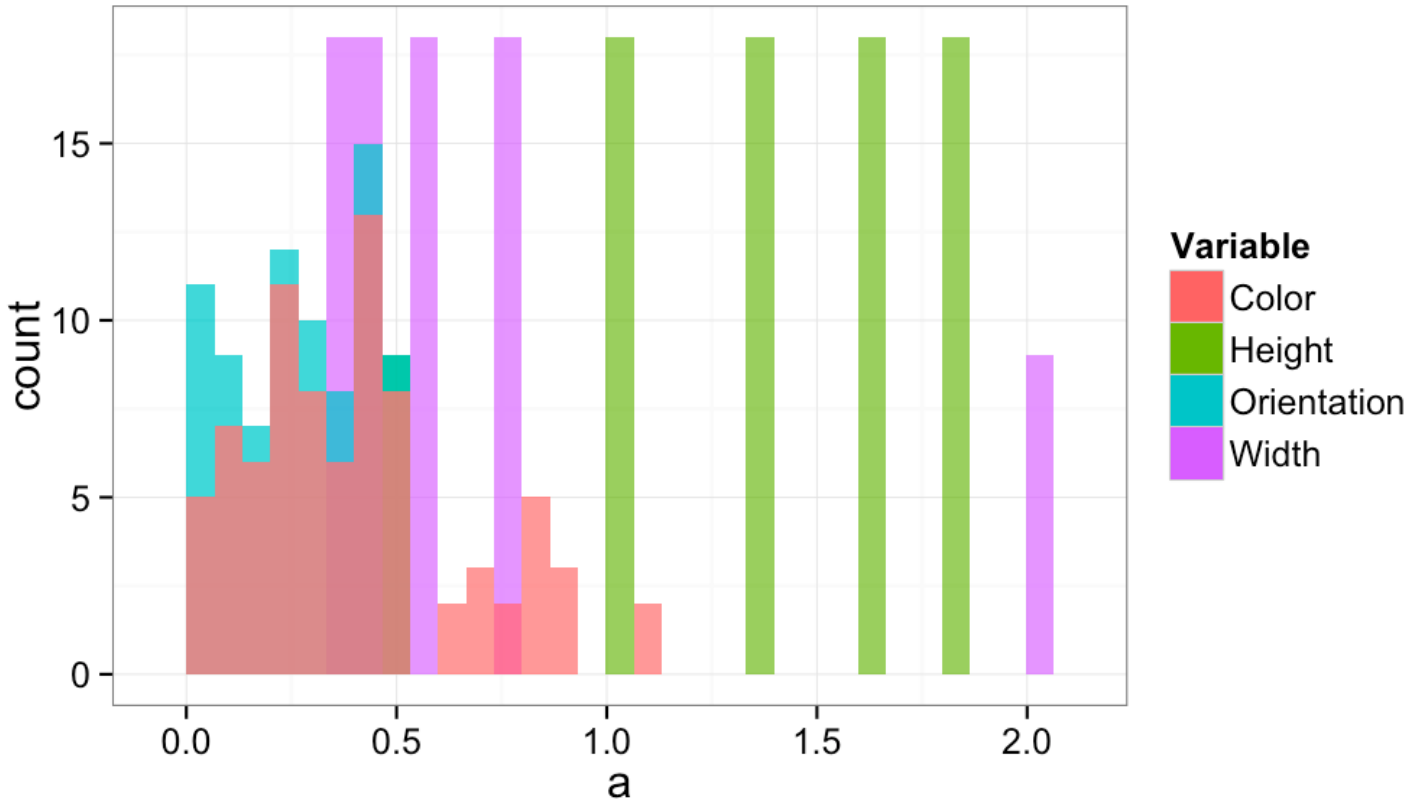
Going beyond vector the possibilities for images are limitless. The following shows a tensor plot with the tensor represented as an ellipse.

$$I(x,y) = \hat{f}(x,y)$$

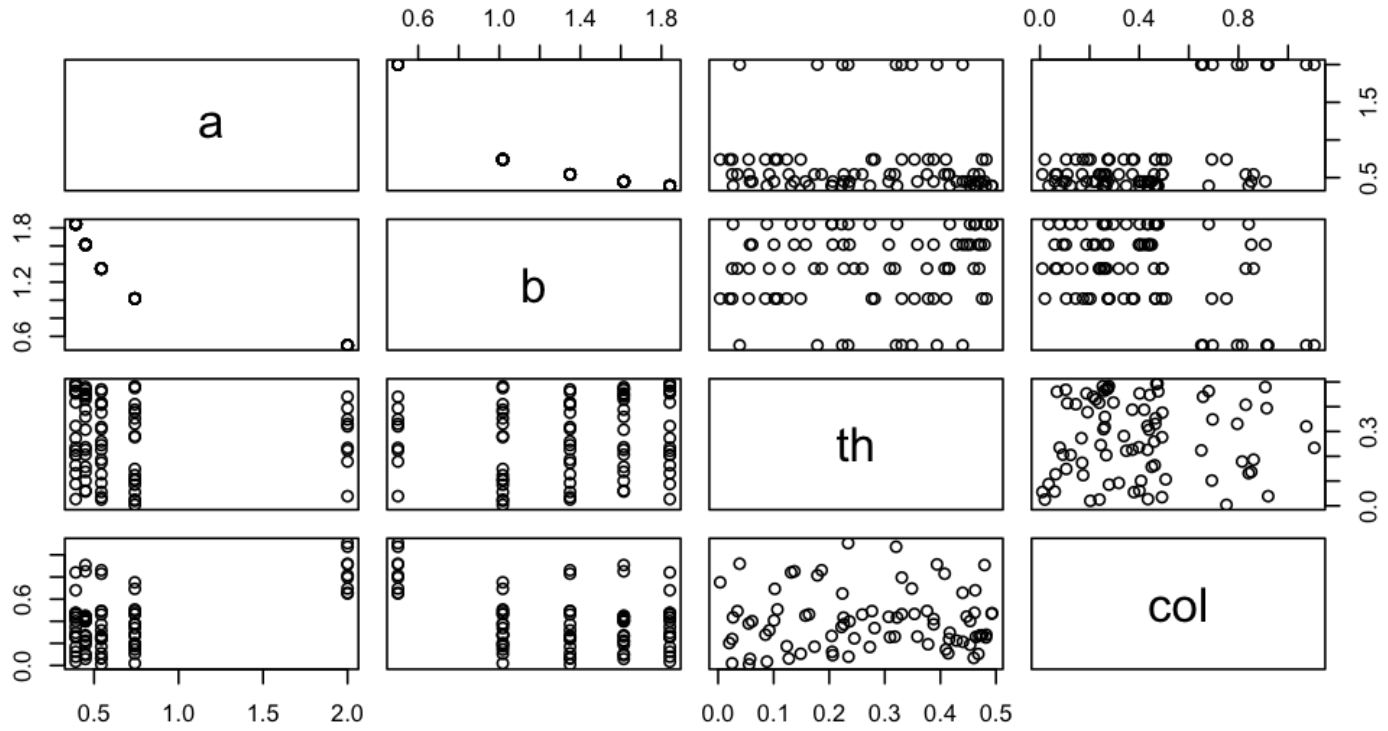
+/- R Code



Once the variable count is above 2, individual density functions and a series of cross plots are easier to interpret than some multidimensional density hypervolume.



+/- R Code



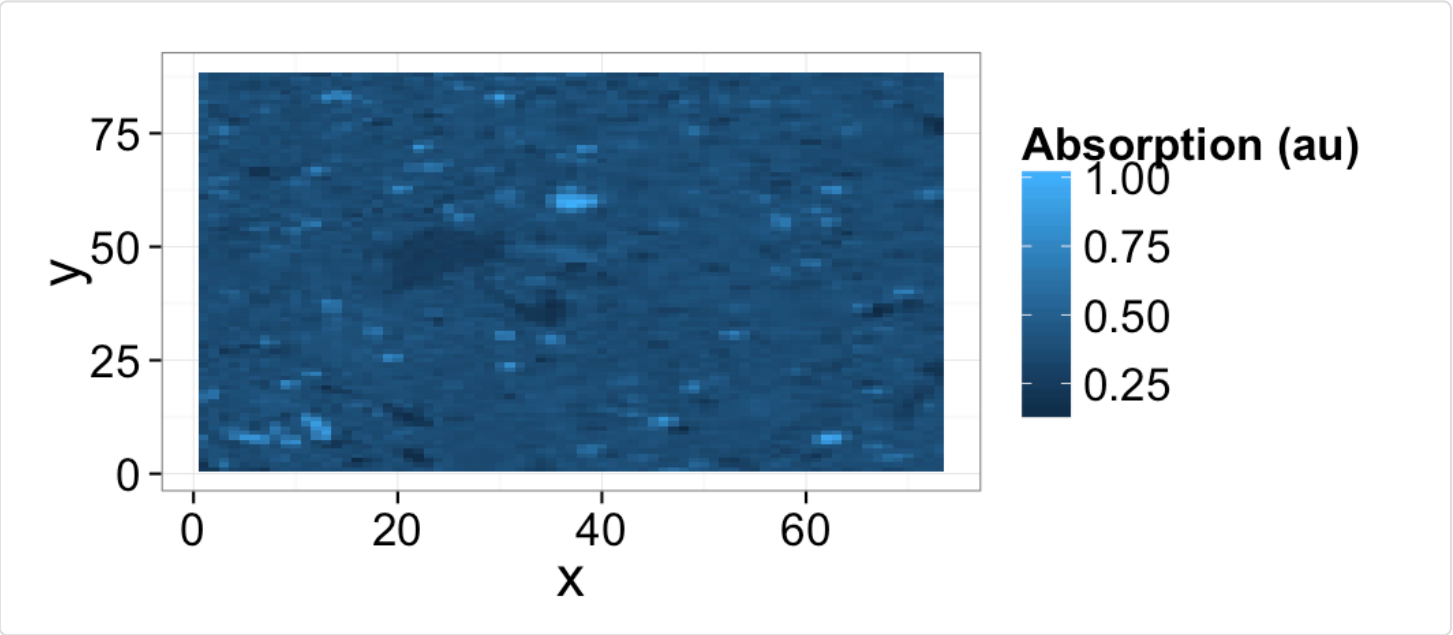
+/- R Code

Multiple Phases: Segmenting Shale

% Shale provided from Kanitpanyacharoen, W. (2012). Synchrotron X-ray Applications Toward an Understanding of Elastic Anisotropy.

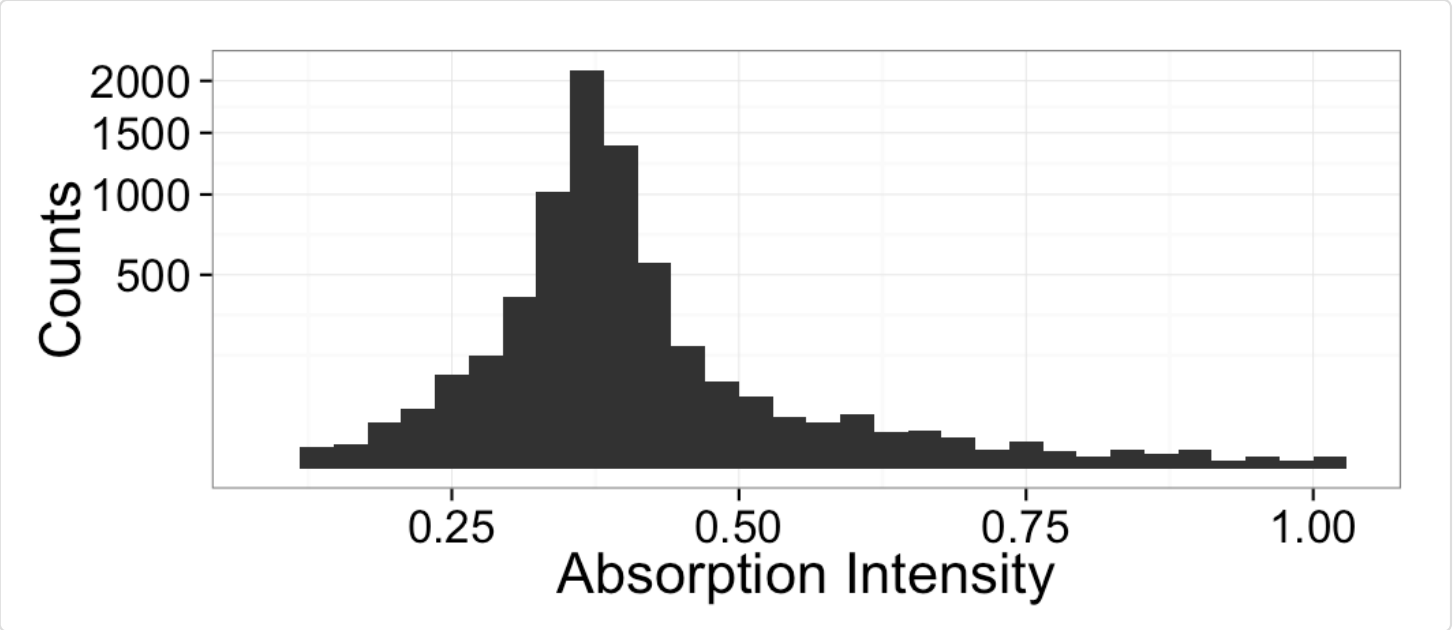
- Here we have a shale sample measured with X-ray tomography with three different phases inside (clay, rock, and air).
- The model is that because the chemical composition and density of each phase is different they will absorb different amounts of x-rays and appear as different brightnesses in the image

+/- R Code



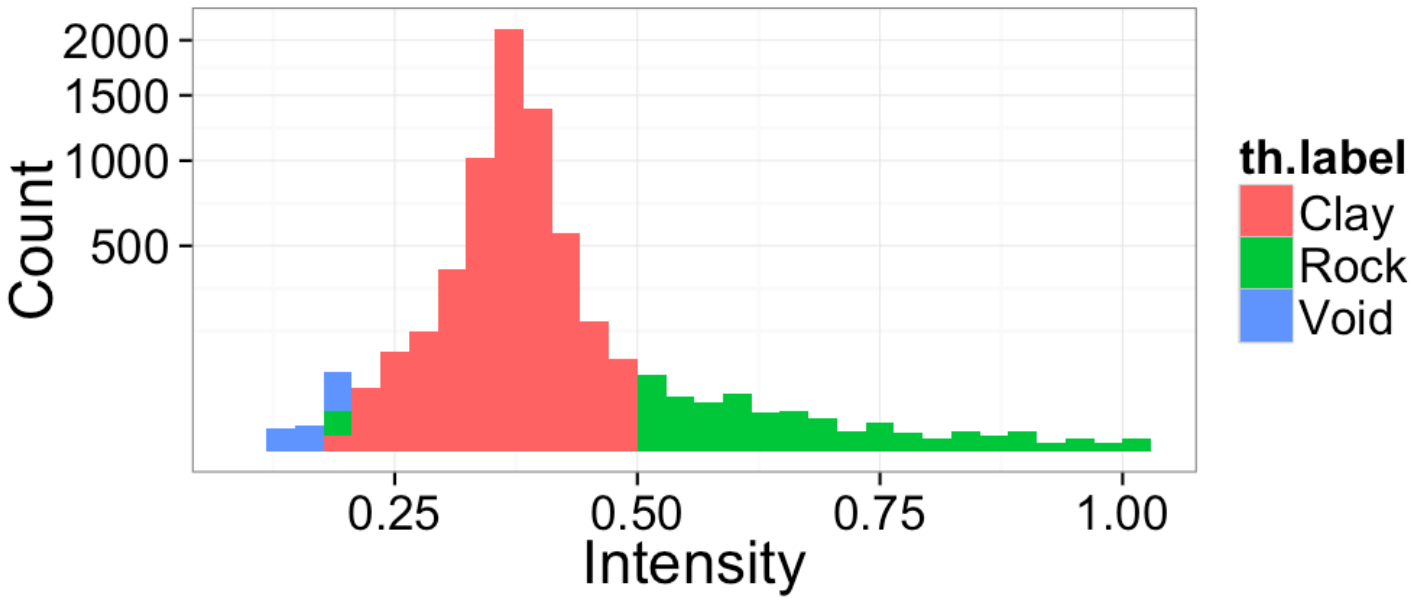
Ideally we would derive 3 values for the thresholds based on a model for the composition of each phase and how much it absorbs, but that is not always possible or practical.

- While there are 3 phases clearly visible in the image, the histogram is less telling (even after being re-scaled).



Multiple Segmentations

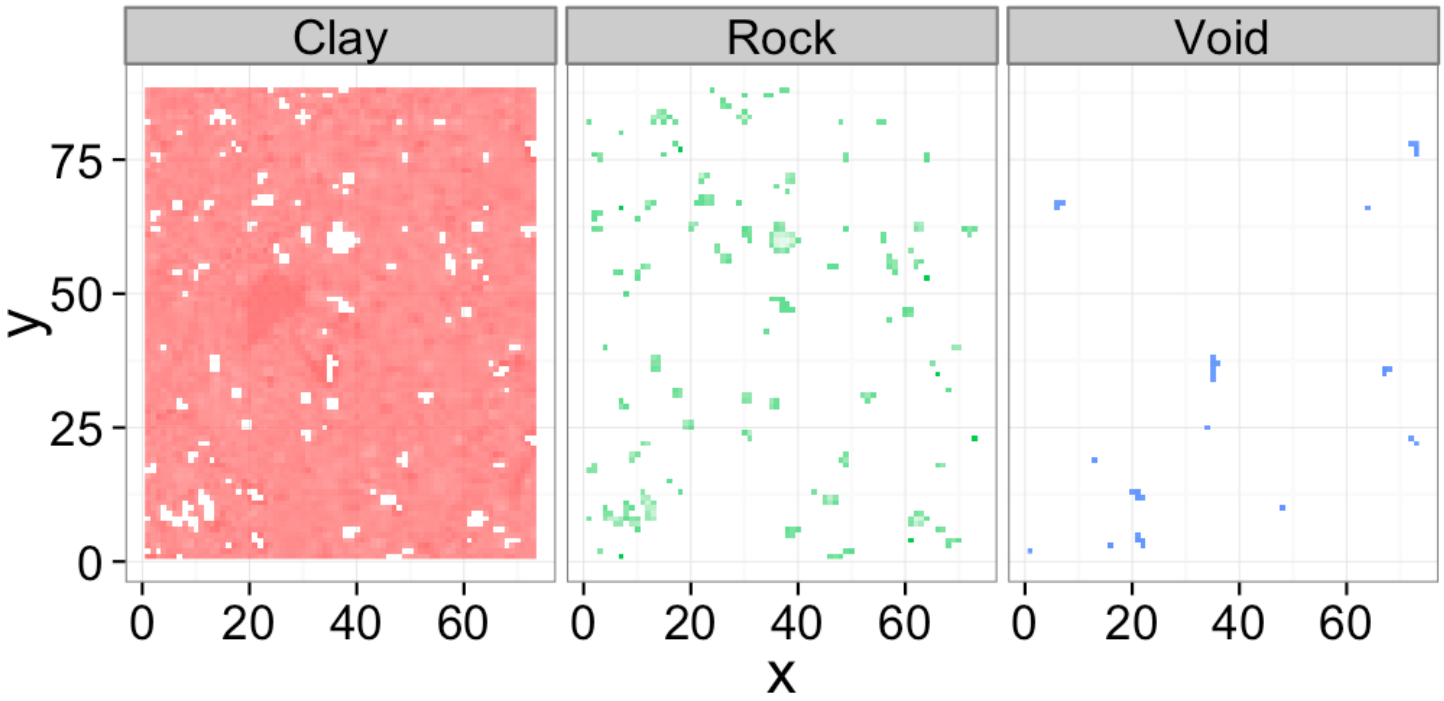
For this exercise we choose arbitrarily 3 ranges for the different phases and perform visual inspection



The relation can explicitly be written out as

$$I(x) = \begin{cases} \text{Void}, & 0 \leq x \leq 0.2 \\ \text{Clay}, & 0.2 < x \leq 0.5 \\ \text{Rock}, & 0.5 < x \end{cases}$$

+/- R Code



Implementation

The implementations of basic thresholds and segmentations is very easy since it is a unary operation of a single image

$$f(I(\vec{x}))$$

In mathematical terms this is called a map and since it does not require information from neighboring voxels or images it can be calculated for each point independently (*parallel*). Filters on the other hand almost always depend on neighboring voxels and thus the calculations are not as easy to seperate.

Implementation Code

Matlab

The simplist is a single threshold in Matlab:

```
thresh_img = gray_img > thresh
```

A more complicated threshold:

```
thresh_img = (gray_img > thresh_a) & (gray_img > thresh_b)
```

Python (numpy)

```
thresh_img = gray_img > thresh
```

Python

```
thresh_img = map(lambda gray_val: gray_val>thresh,
                 gray_img)
```

Java

```
boolean[] thresh_img = new boolean[x_size*y_size*z_size];
for(int x=x_min;x<x_max;x++)
  for(int y=y_min;y<y_max;y++)
    for(int z=z_min;z<z_max;z++) {
      int offset=(z*y_size+y)*x_size+x;
      thresh_img[offset]=gray_img[offset]>thresh;
    }
```

In C/C++

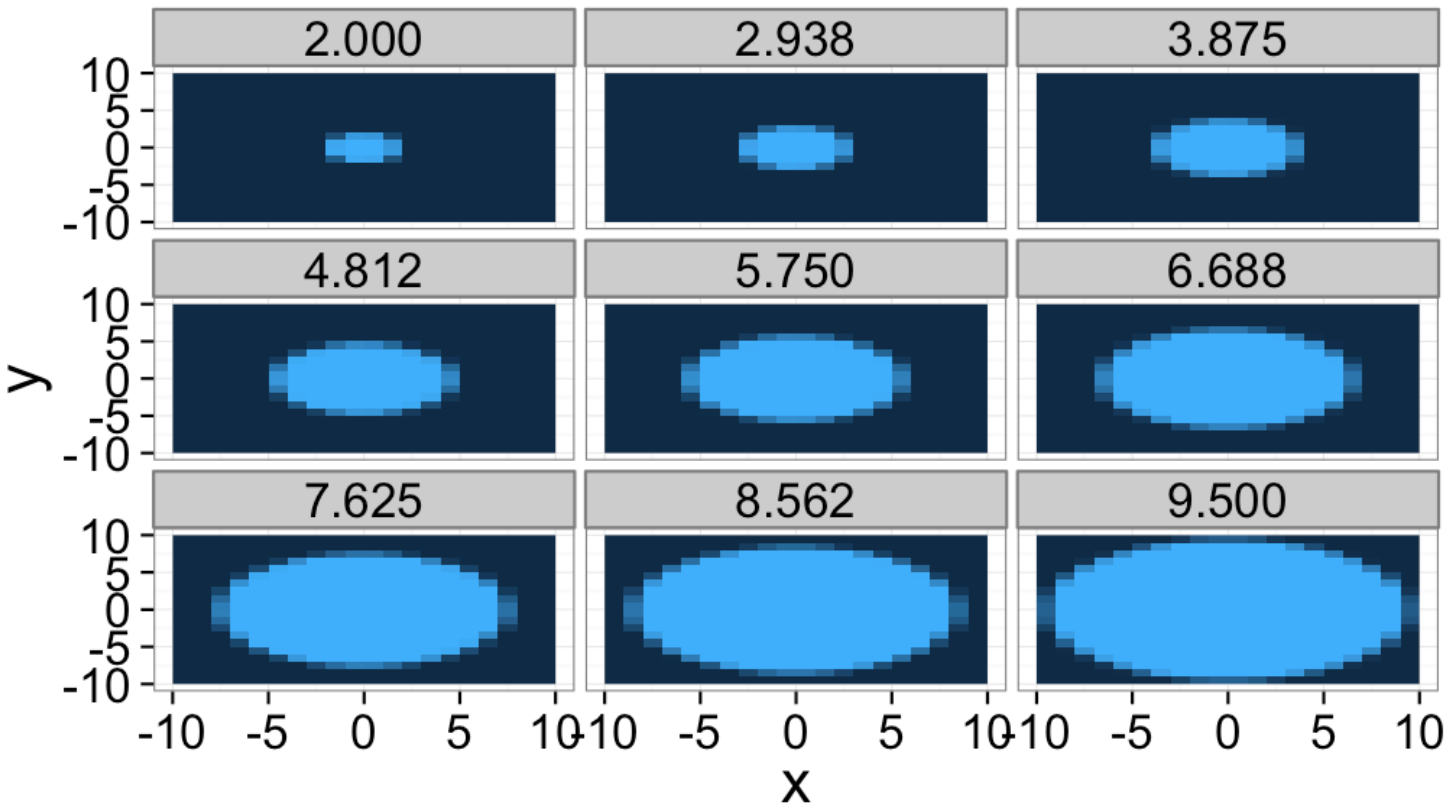
```
bool* thesh_img = malloc(x_size*y_size*z_size * sizeof (bool));

for(int x=x_min;x<x_max;x++)
  for(int y=y_min;y<y_max;y++)
    for(int z=z_min;z<z_max;z++) {
      int offset=(z*y_size+y)*x_size+x;
      thresh_img[offset]=gray_img[offset]>thresh;
    }
```

Pitfalls with Segmentation

type: alert

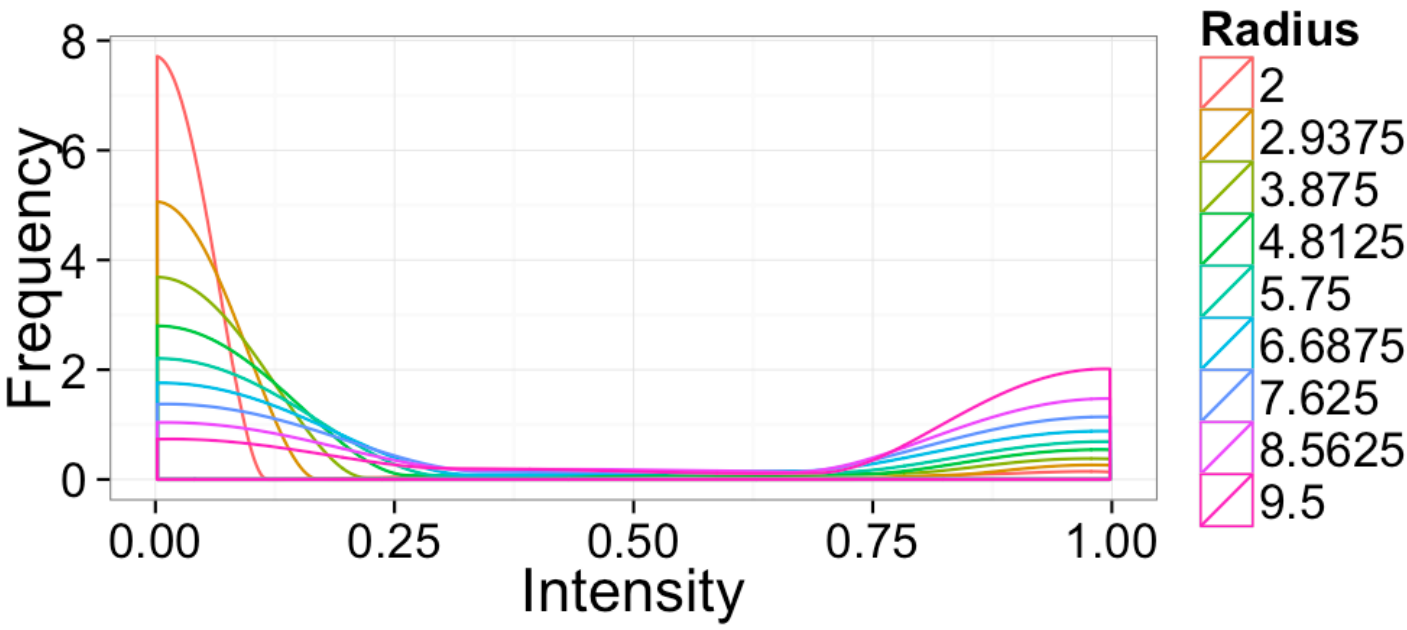
Partial Volume Effect

- The partial volume effect (http://bit.ly/1mW7kdP) is the name for the effect of discretization on the image into pixels or voxels.
- Surfaces are complicated, voxels are simple boxes which make poor representations
- Many voxels are only partially filled, but only the voxels on the surface
- Removing the first layer alleviates issue



+/- R Code

+/- R Code



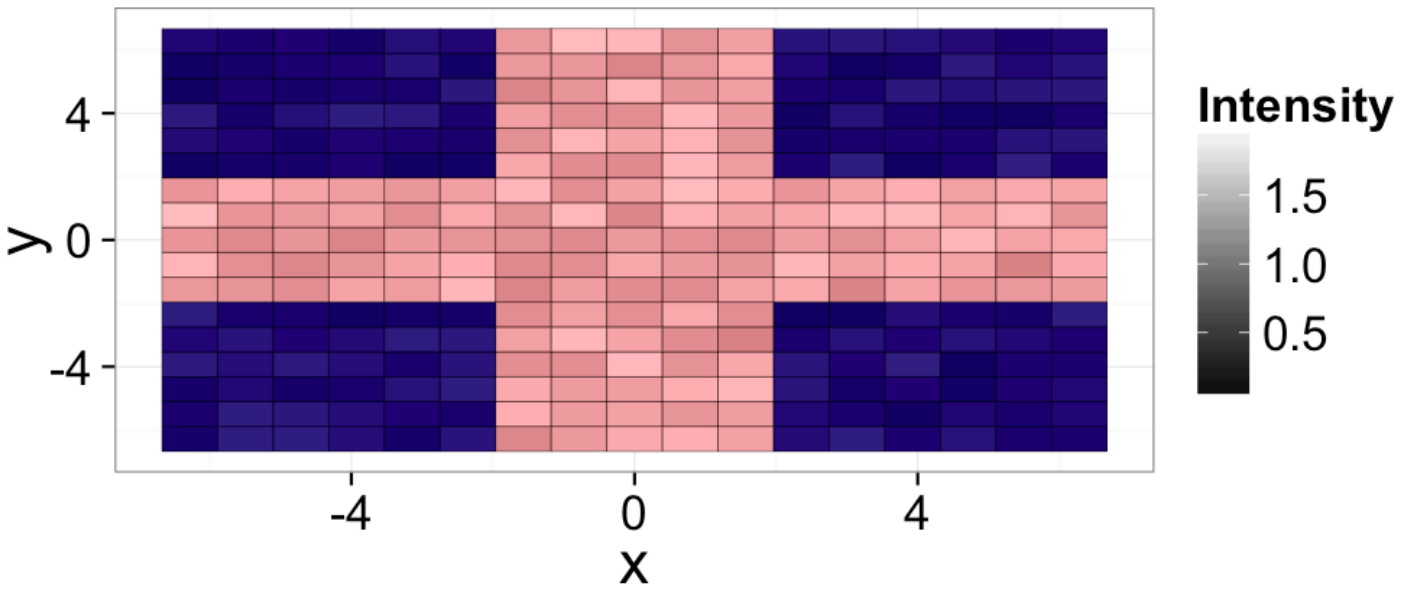
- Shown as a function of radius

+/- R Code

Radius	Mean Intensity	Sd Intensity
2.000	0.0311	0.1623
2.938	0.0677	0.2370
3.875	0.1177	0.3061
4.812	0.1822	0.3688
5.750	0.2601	0.4196
6.688	0.3511	0.4567
7.625	0.4569	0.4763
8.562	0.5761	0.4690
9.500	0.7073	0.4259

Morphology

Returning to the original image of a cross



+/- R Code

We can now utilize information from neighborhood voxels to improve the results. These steps are called morphological operations.

Like filtering the assumption behind morphological operations are

- nearby voxels in **real** images are related / strongly correlated with one another
- noise and imaging artifacts are less spatially correlated.

Therefore these imaging problems can be alleviated by adjusting the balance between local and neighborhood values.

Fundamentals: Neighborhood

A neighborhood consists of the pixels or voxels which are of sufficient proximity to a given point. There are a number of possible definitions which largely affect the result when it is invoked.

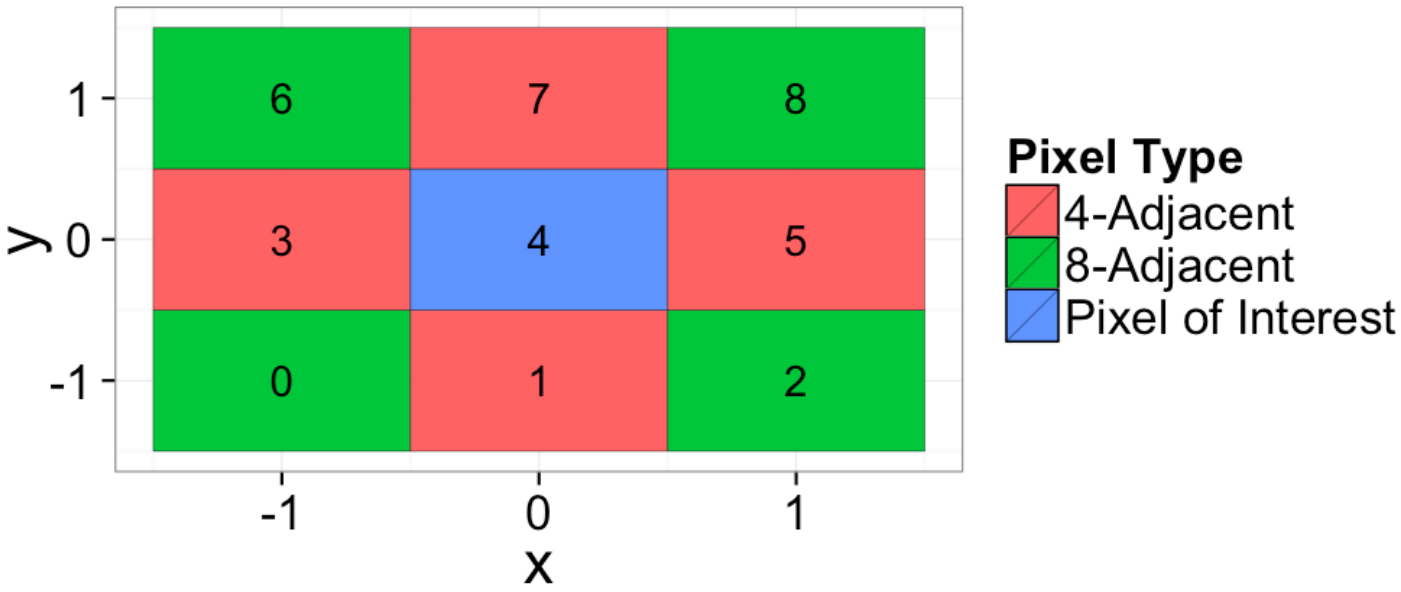
- A large neighborhood performs operations over larger areas / volumes
  - Computationally intensive
  - Can *smooth* out features
- A small neighborhood performs operations over small areas / volumes
  - Computationally cheaper
  - Struggles with large noise / filling large holes

The neighborhood is important for a large number of image and other (communication, mapping, networking) processing operations:

- filtering
- morphological operations
- component labeling
- distance maps
- image correlation based tracking methods

Fundamentals: Neighbors in 2D

For standard image operations there are two definitions of neighborhood. The 4 and 8 adjacent neighbors shown below. Given the blue pixel in the center the red are the 4-adjacent and the red and green make up the 8 adjacent.



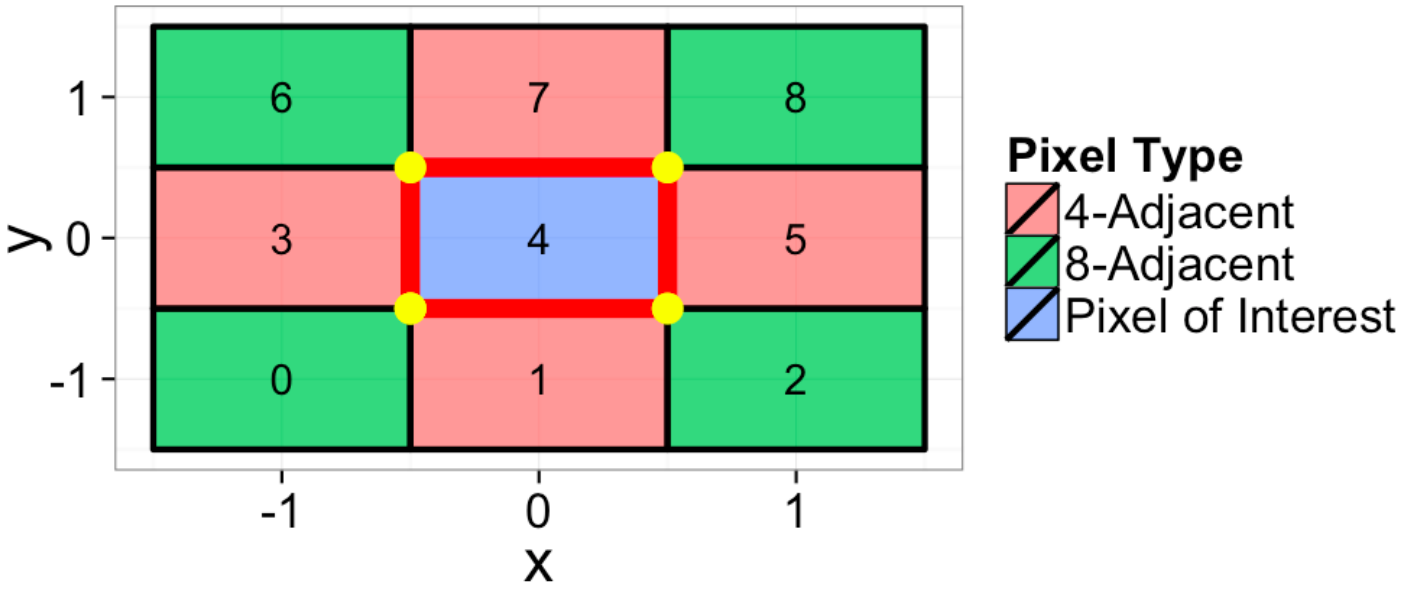
+/- R Code

Formal Neighborhood Definitions

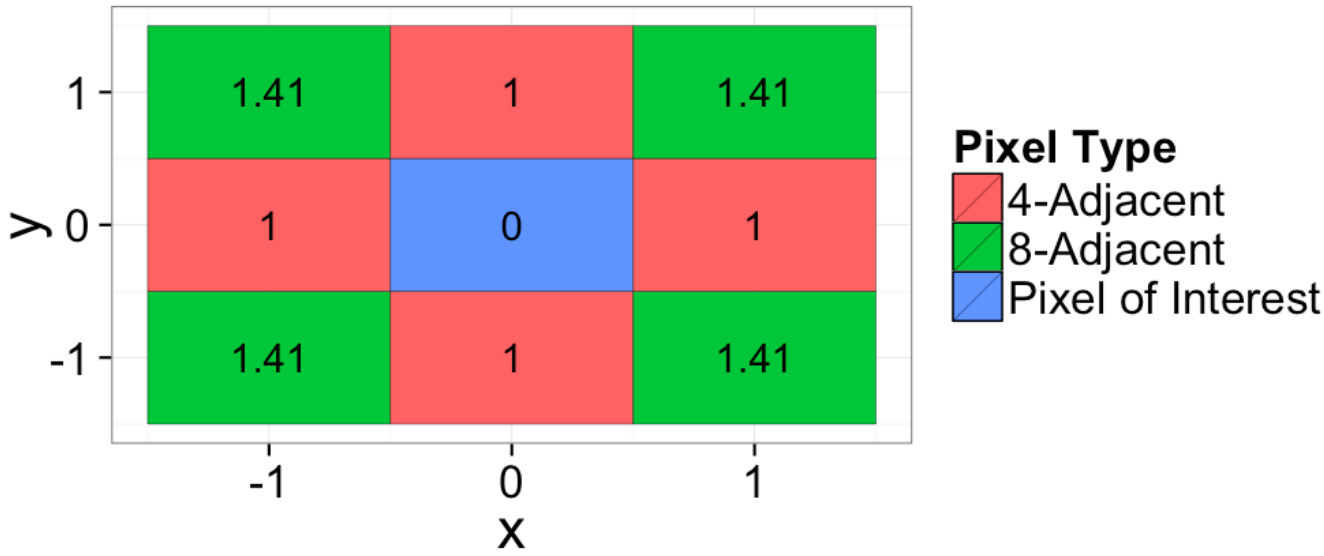
Formally these have two effectively equivalent, but different definitions.

- Pixels which share a face (red line) with the pixel
- Pixels which share a vertex (yellow dot) with the pixel

+/- R Code



- Pixels which are distance 1 from the pixel
- Pixels which are distance $\sqrt{2}$ from the pixel

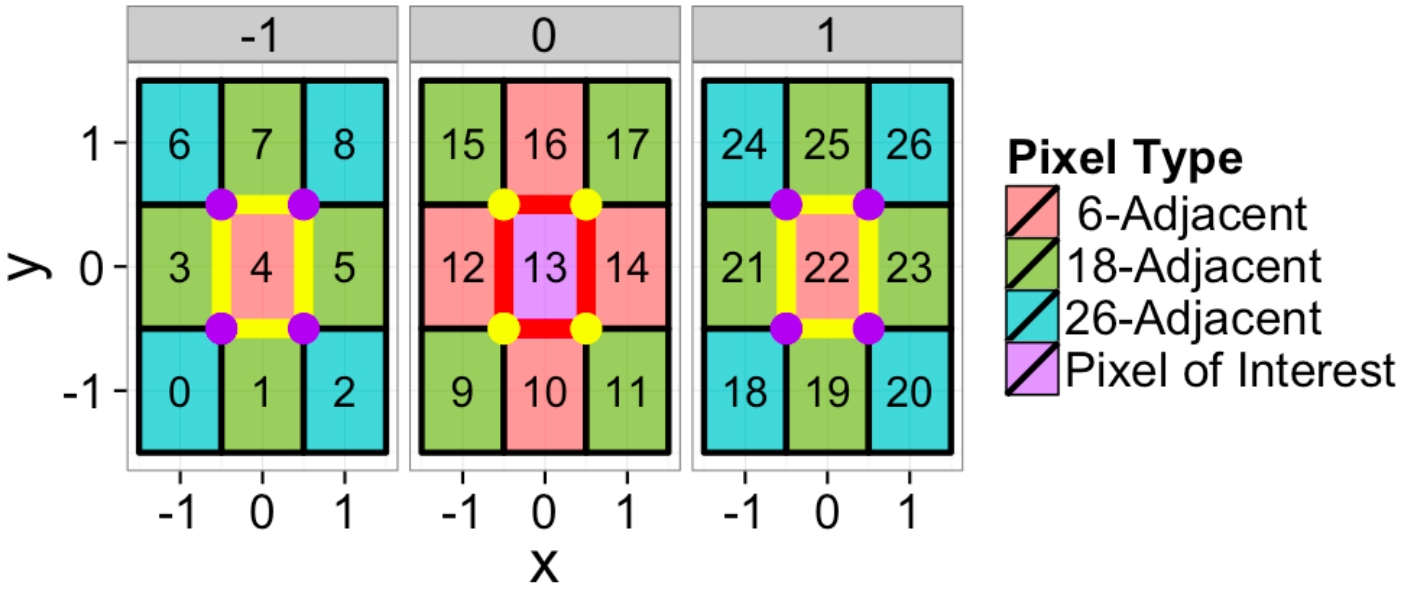


+/- R Code

Formal Neighborhood Definitions in 3D

In 3D there is now an additional group to consider because of the extra-dimension

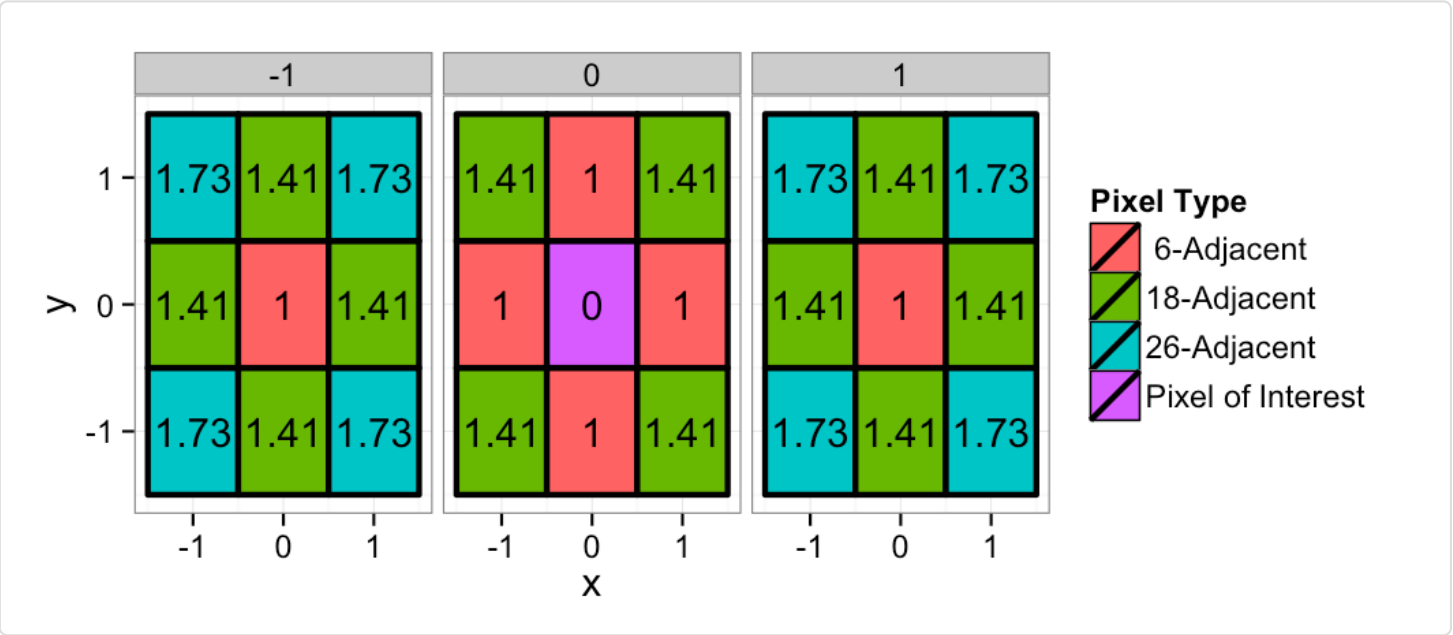
- Voxels which share a face (red line) with the voxel (6-adjacent)
- Voxels which share an edge (yellow dot) with the voxel (18-adjacent)
- Voxels which share a vertex (purple dot) with the voxel (26-adjacent)



+/- R Code

- Voxels which are distance 1 from the voxel
- Voxels which are distance $\sqrt{2}$ from the voxel
- Voxels which are distance $\sqrt{3}$ from the voxel

+/- R Code



Erosion and Dilation

Erosion

If any of the voxels in the neighborhood are 0/false than the voxel will be set to 0

- Has the effect of peeling the surface layer off of an object

---

Dilation

If any of the voxels in the neigbhorhood are 1/true then the voxel will be set to 1

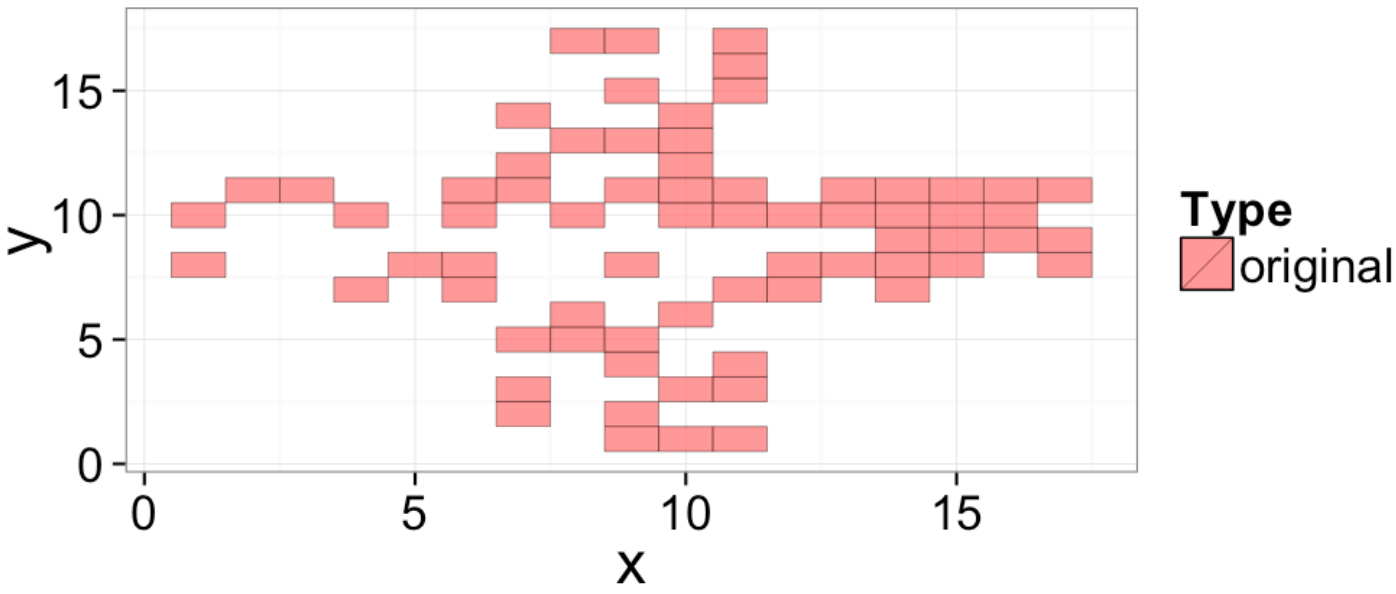
- Has the effect of adding a layer onto an object (dunking an strawberry in chocolate, adding a coat of paint to a car)

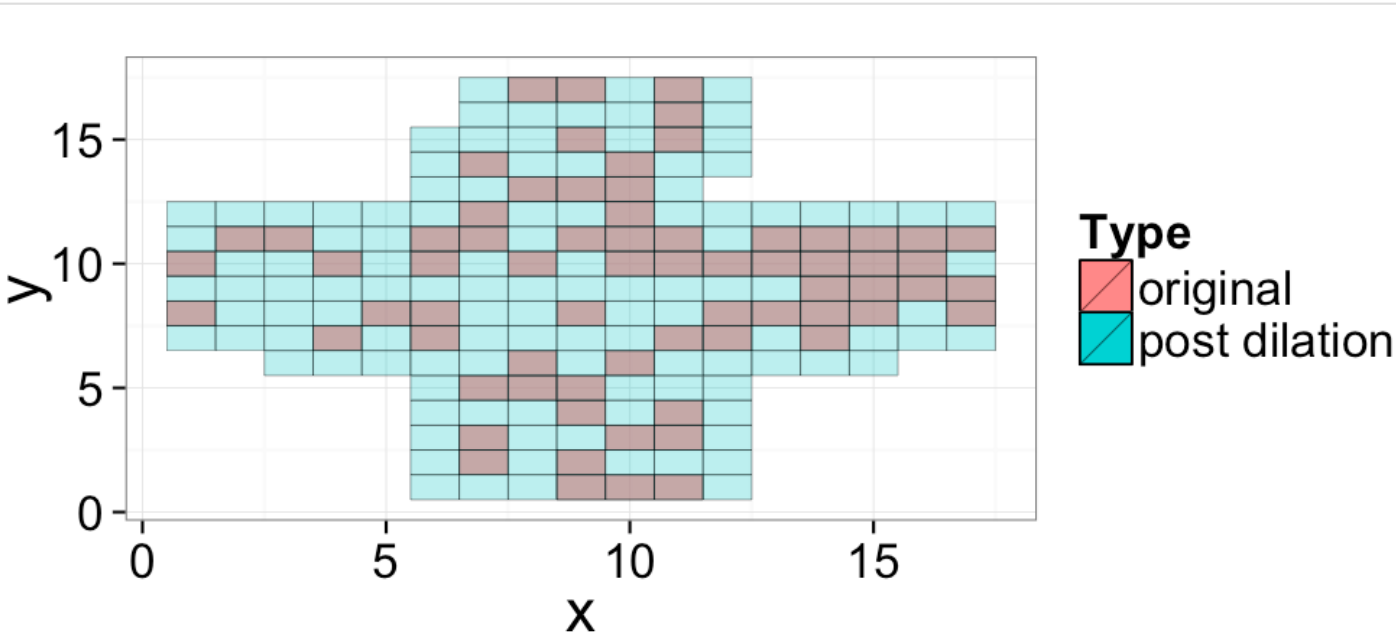
Applied Erosion and Dilation

Erosion

Taking an image of the cross at a too-high threshold, we show how dilation can be used to recover some of the missing pixels

+/- R Code

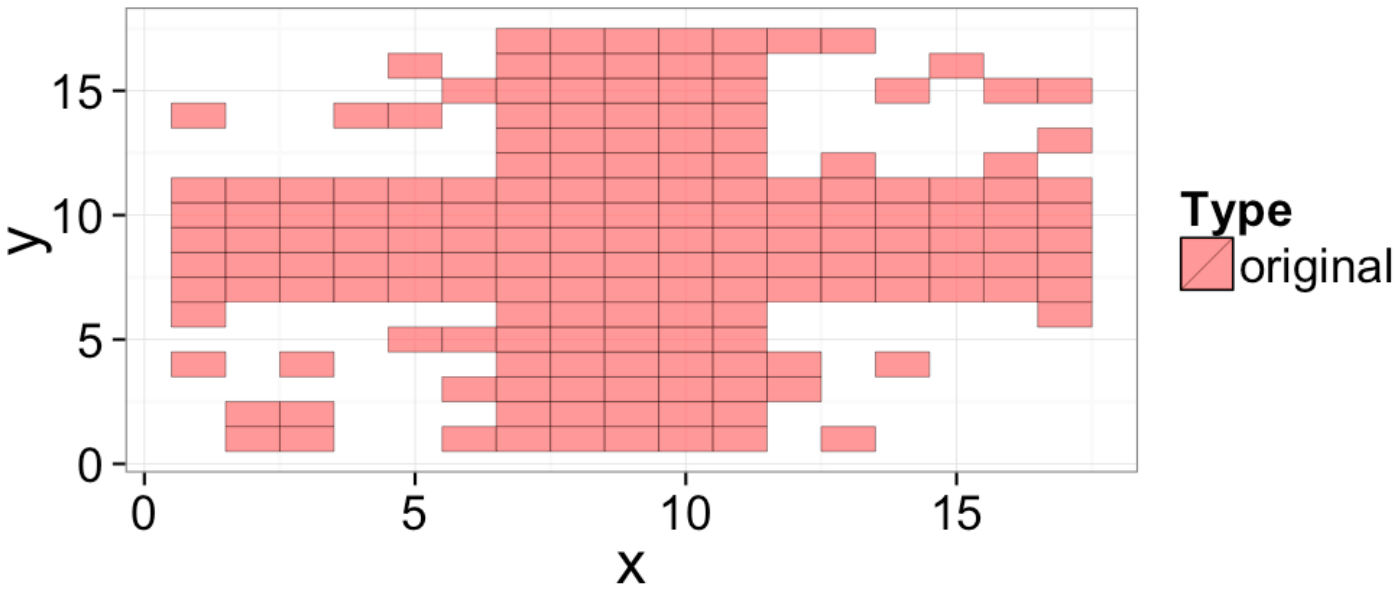




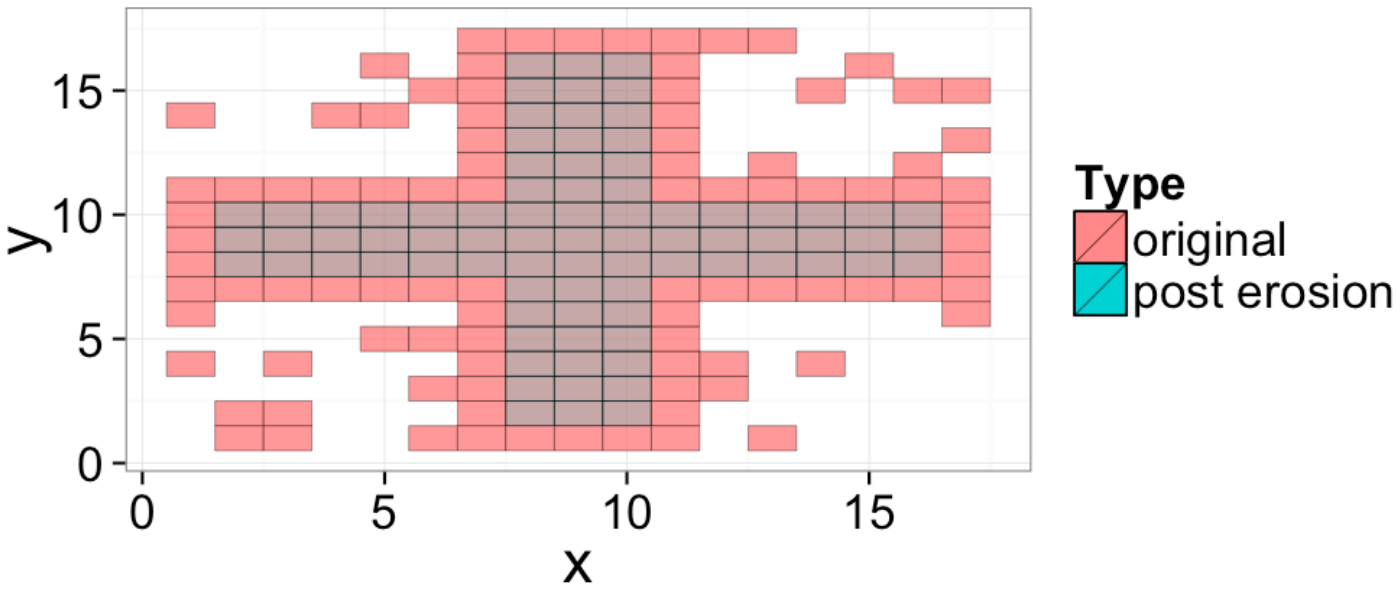
+/- R Code

Dilation

Taking an image of the cross at a too-low threshold, we show how erosion can be used to remove some of the extra pixels



+/- R Code



+/- R Code

Opening and Closing

Opening

An erosion followed by a dilation operation

- Peels a layer off and adds a layer on
- Very small objects and connections are deleted in the erosion and do not return the image is thus **open**ed
- A cube larger than several voxels will have the exact same volume after (conservative)

Closing

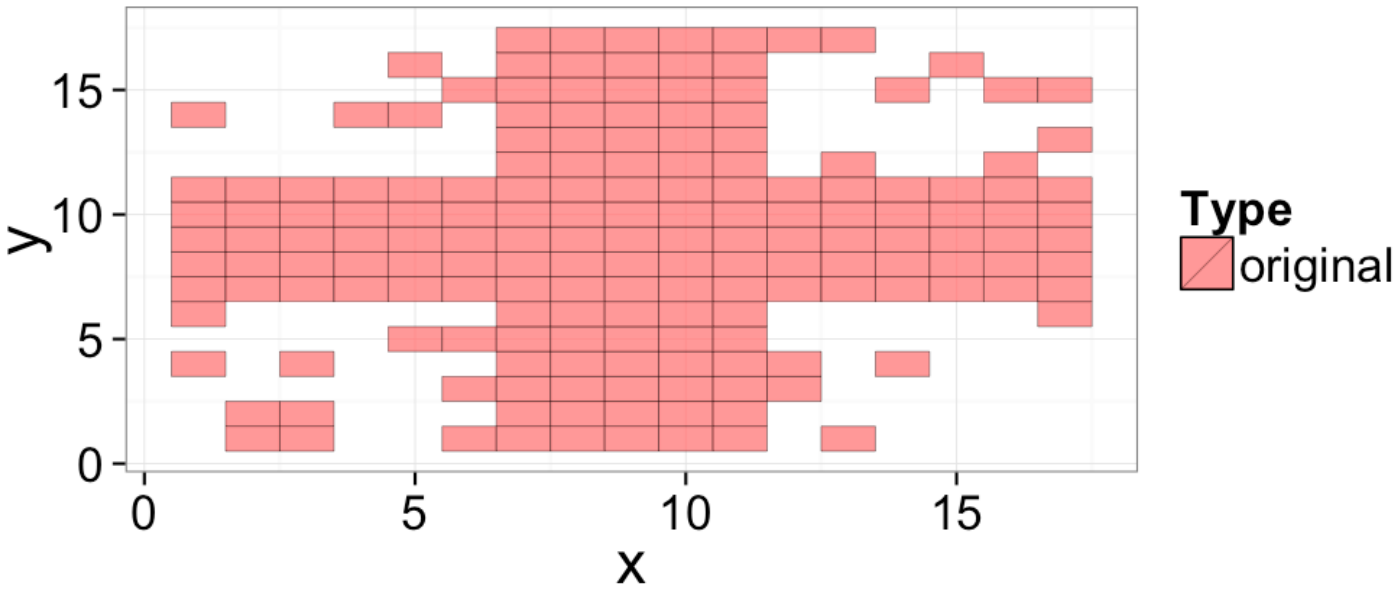
A dilation followed by an erosion operation

- Adds a layer and then peels a layer off
- Objects that are very close are connected when the layer is added and they stay connected when the layer is removed thus the image is **close**d
- A cube larger than one voxel will have the exact same volume after (conservative)

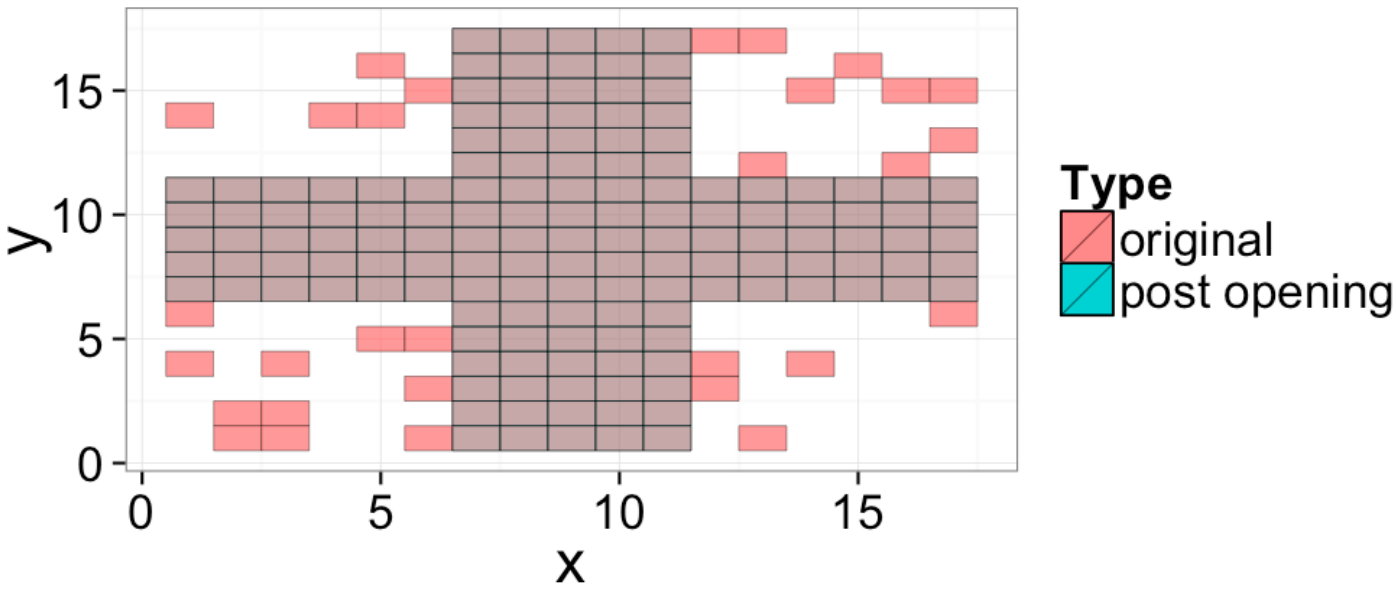
Applied Opening and Closing

Opening

Taking an image of the cross at a too-low threshold, we show how opening can be used to remove some of the extra pixels



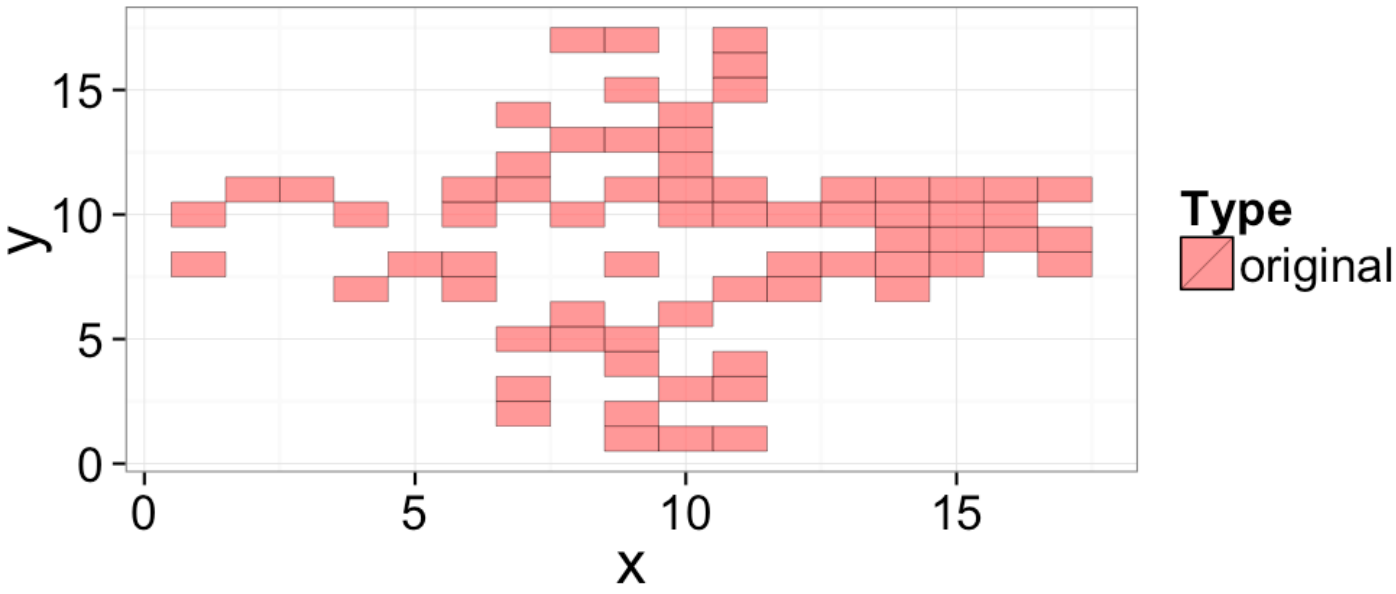
+/- R Code



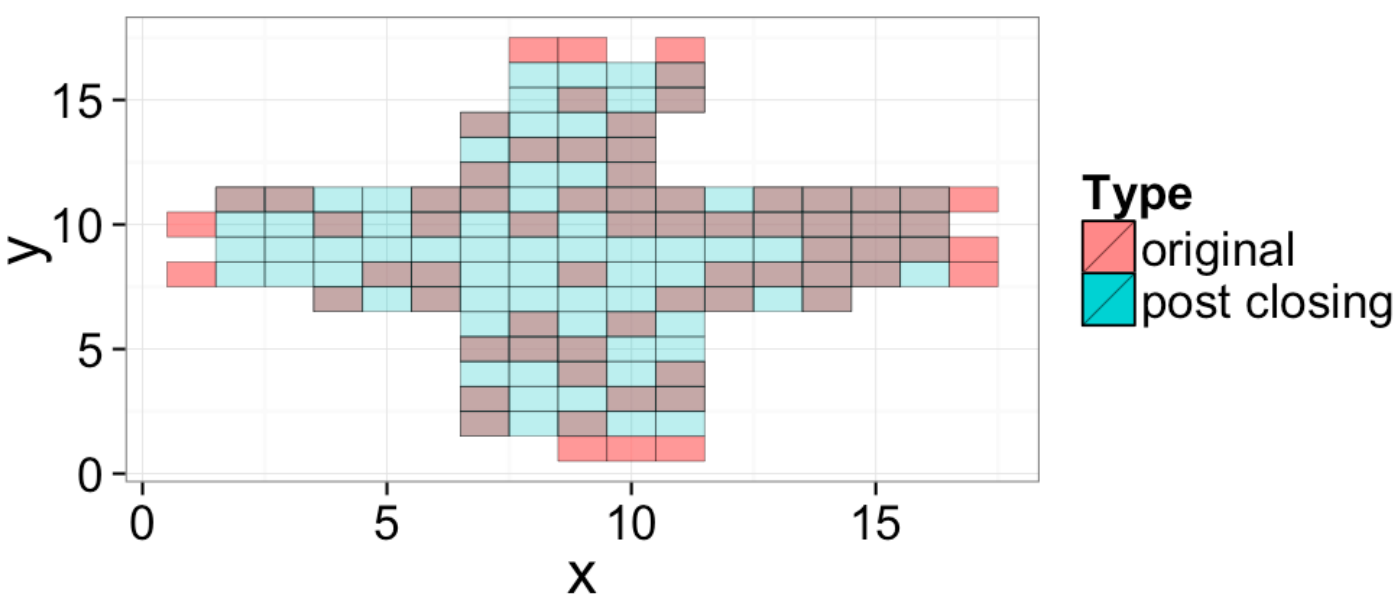
+/- R Code

Closing

Taking an image of the cross at a too-high threshold, we show how closing can be used to recover some of the missing pixels



+/- R Code



+/- R Code

Applying Morphological Tools

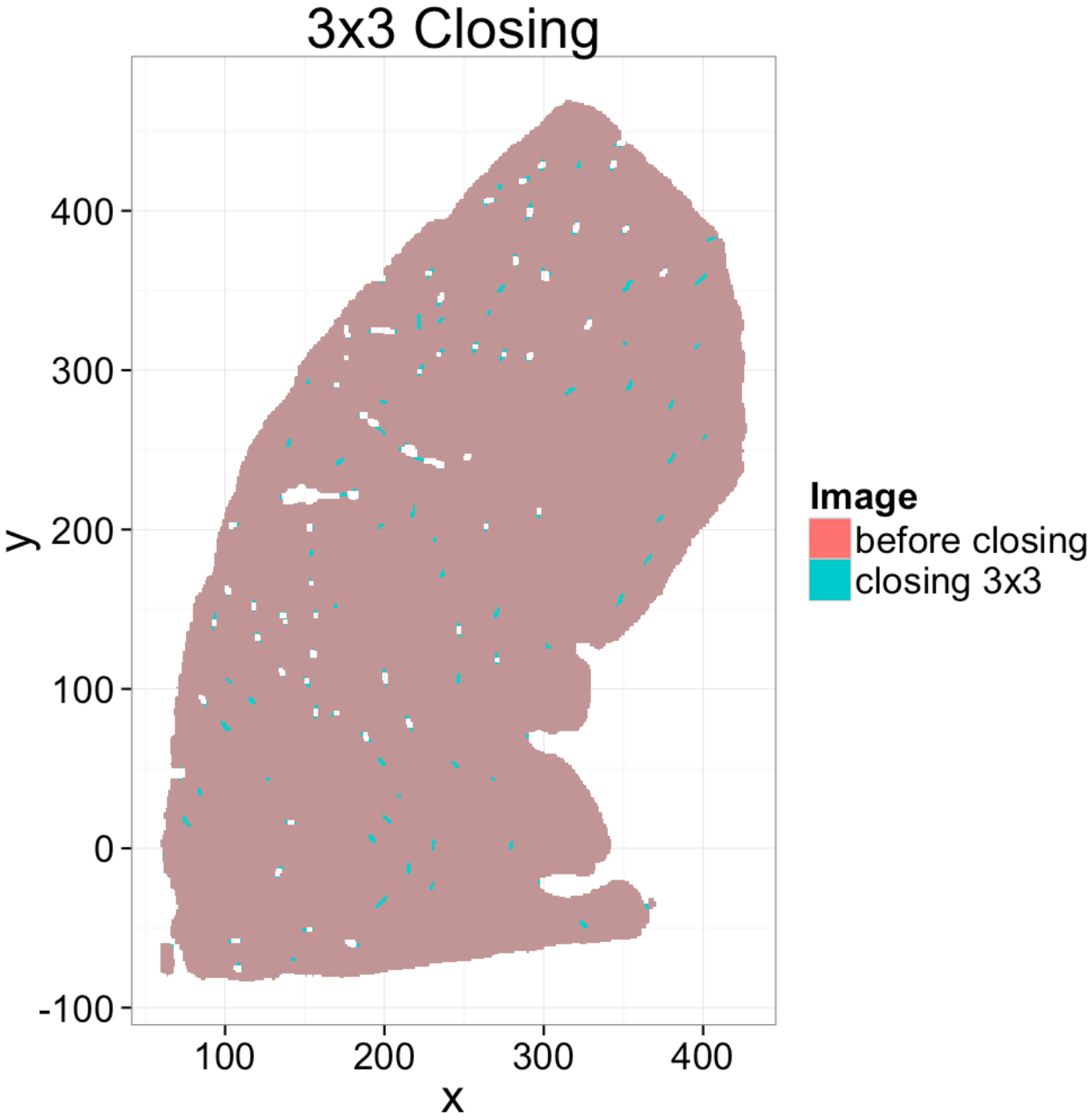
- For many applications morphology appears to be the same as filter
- The key difference is the avalanche or non-linear nature of the results
  - A single off voxel can turn its entire neighborhood off (erosion)
  - A single on voxel can turn its entire neighborhood on (dilation)
- The effects of this are most pronounced when performed iteratively
- This is very useful for filling holes, connecting objects, creating masks

---

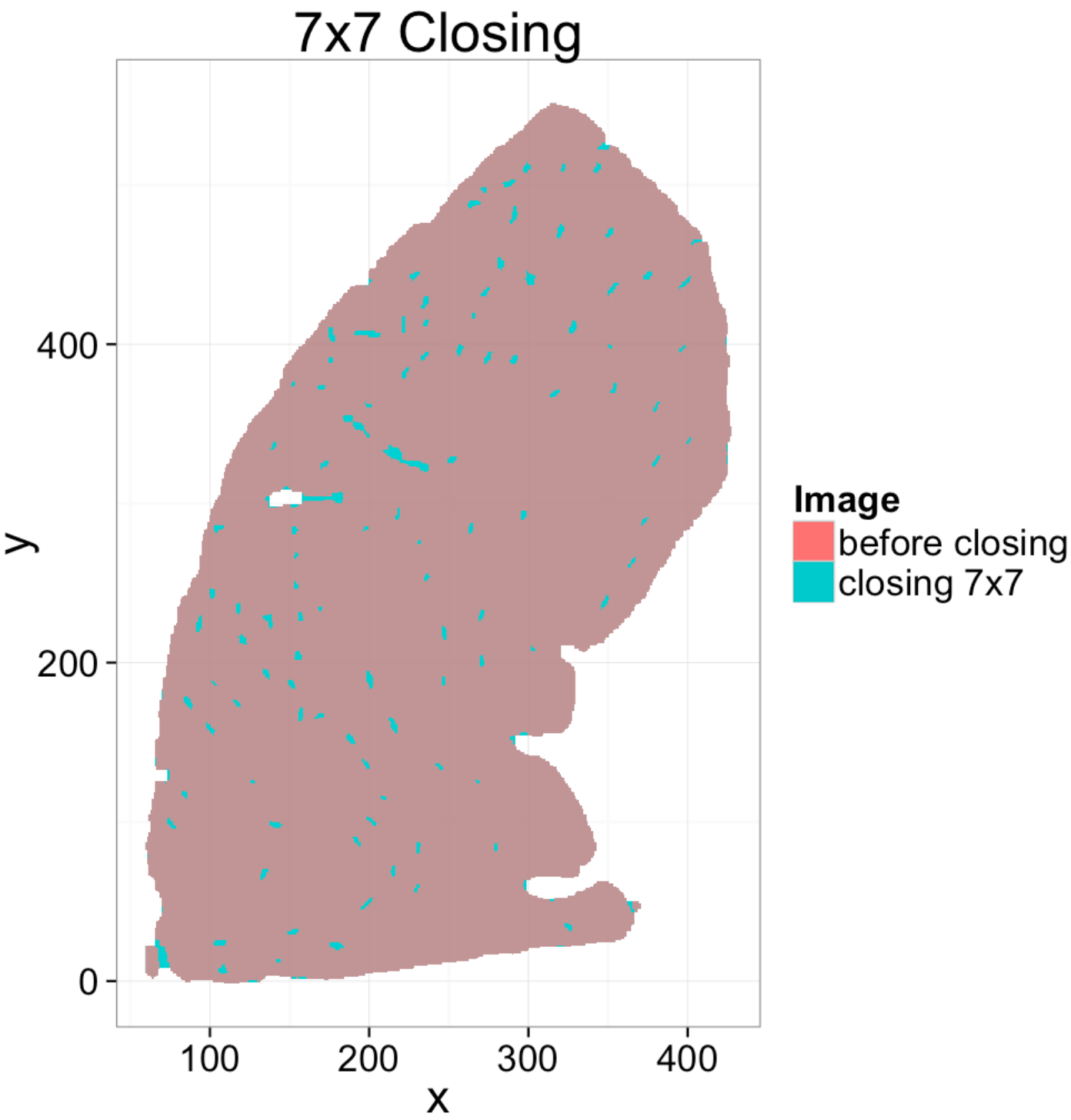
A segmented slice taken from a cortical bone sample. The dark is the calcified bone tissue and the white are holes in the image

Filling Holes / Creating Masks

+/- R Code

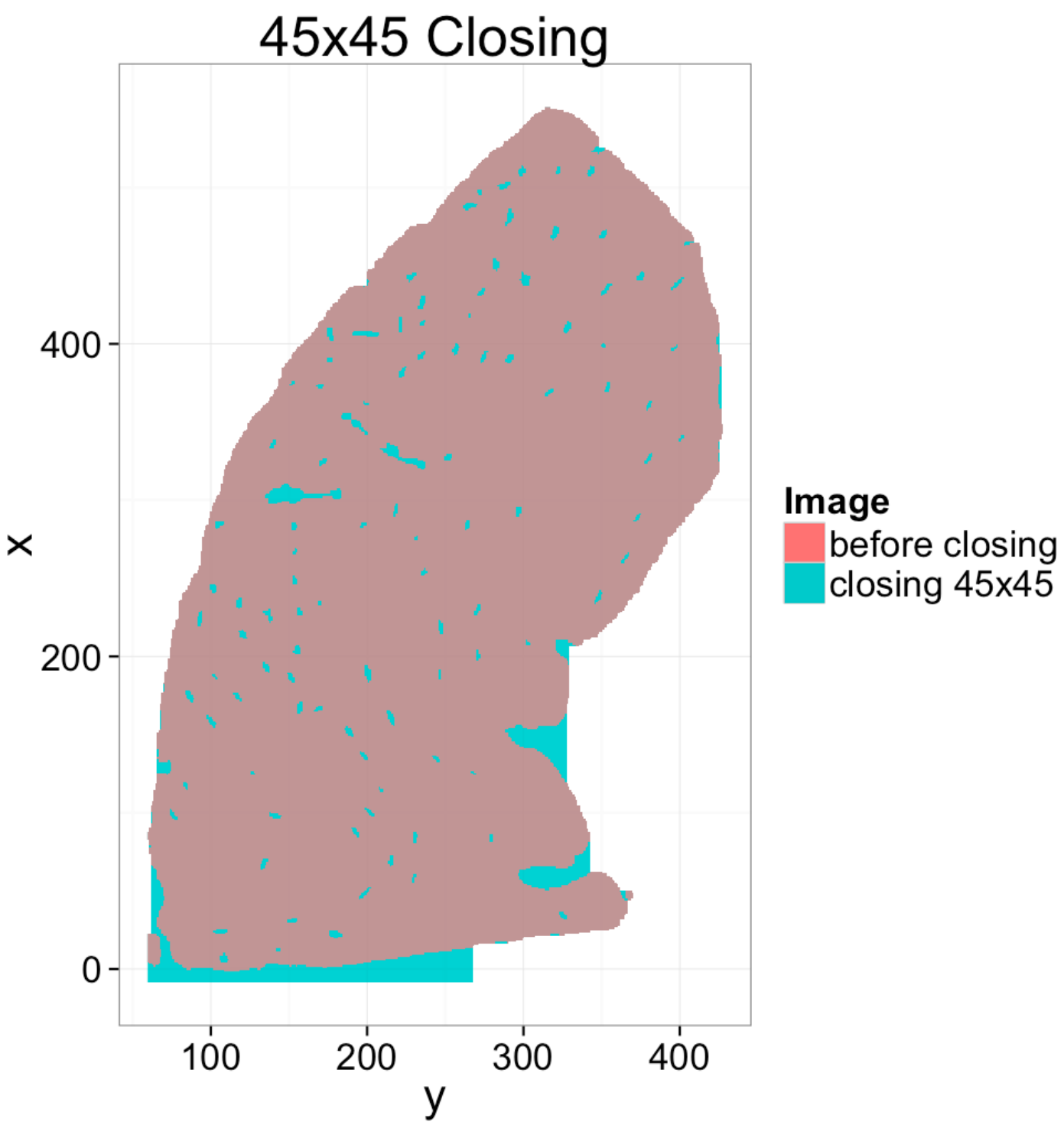

3x3 Closing
400
300
200
100
0
-100
y
100
200
300
400
x
Image
before closing
closing 3x3

+/- R Code



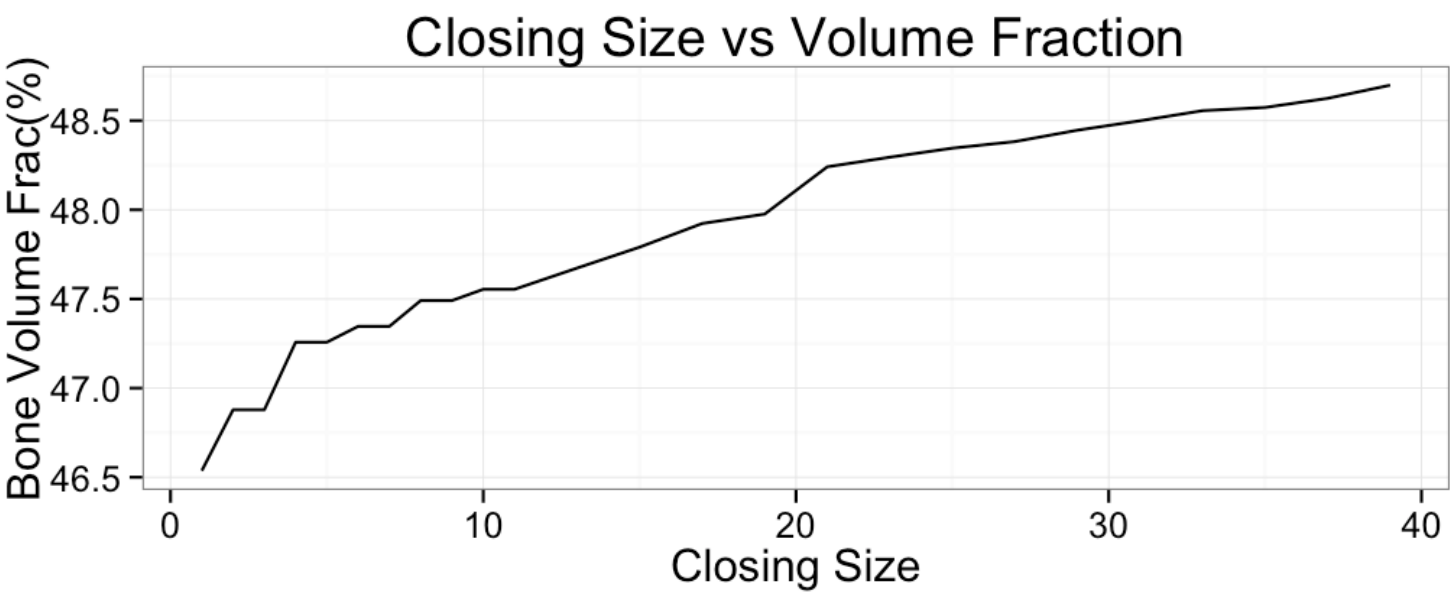
Filling Holes / Creating Masks

Applying the closing with even larger windows will close everything but being to destroy the underlying structure of the mask



+/- R Code

We can characterize this by examining the dependency of the closing size and the volume fraction (note scale)



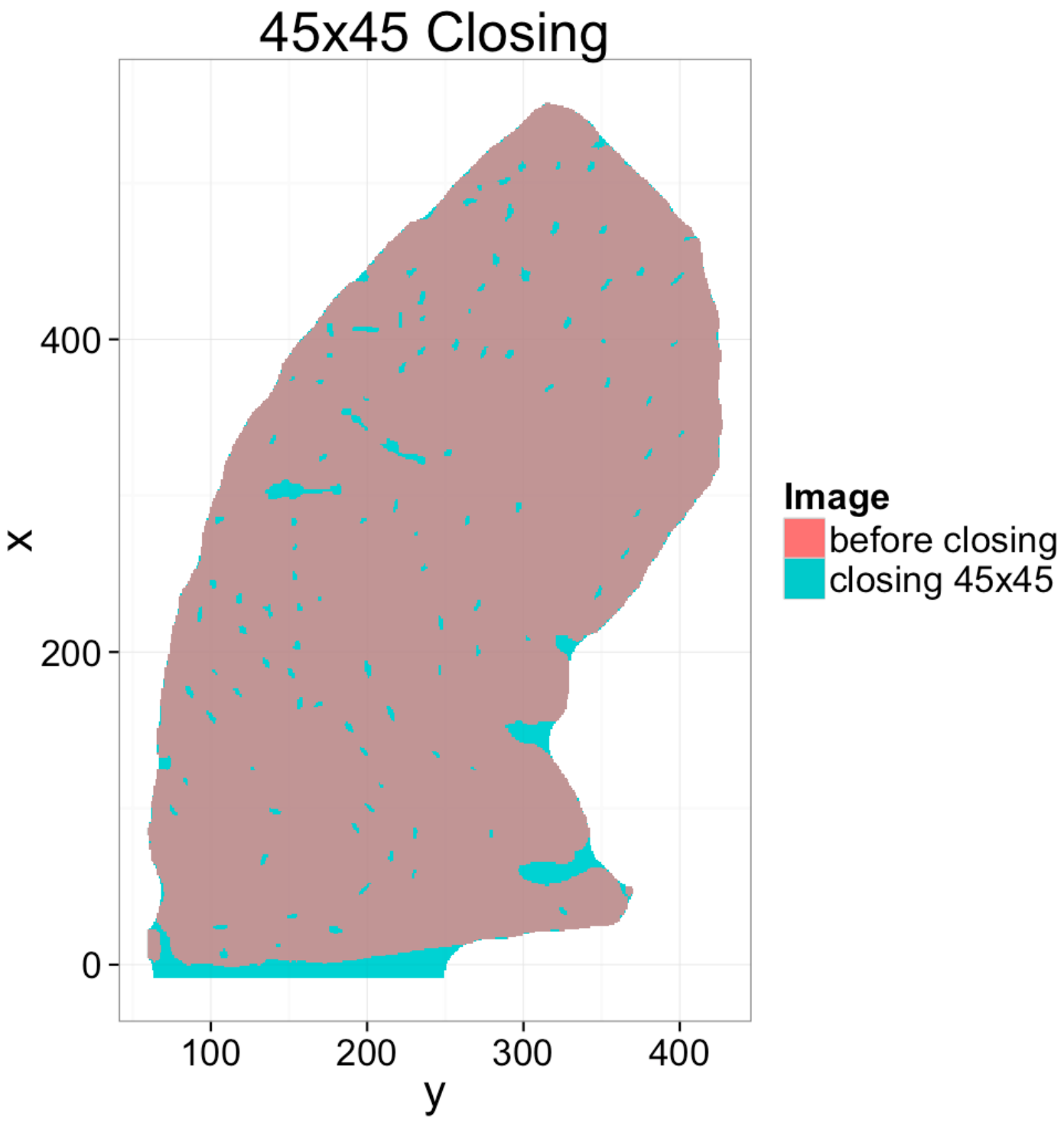
+/- R Code

- Be careful when using large area opening / closing operations (always check the images)
- Noise and defects in images can be amplified with these larger operations

Different Kernels

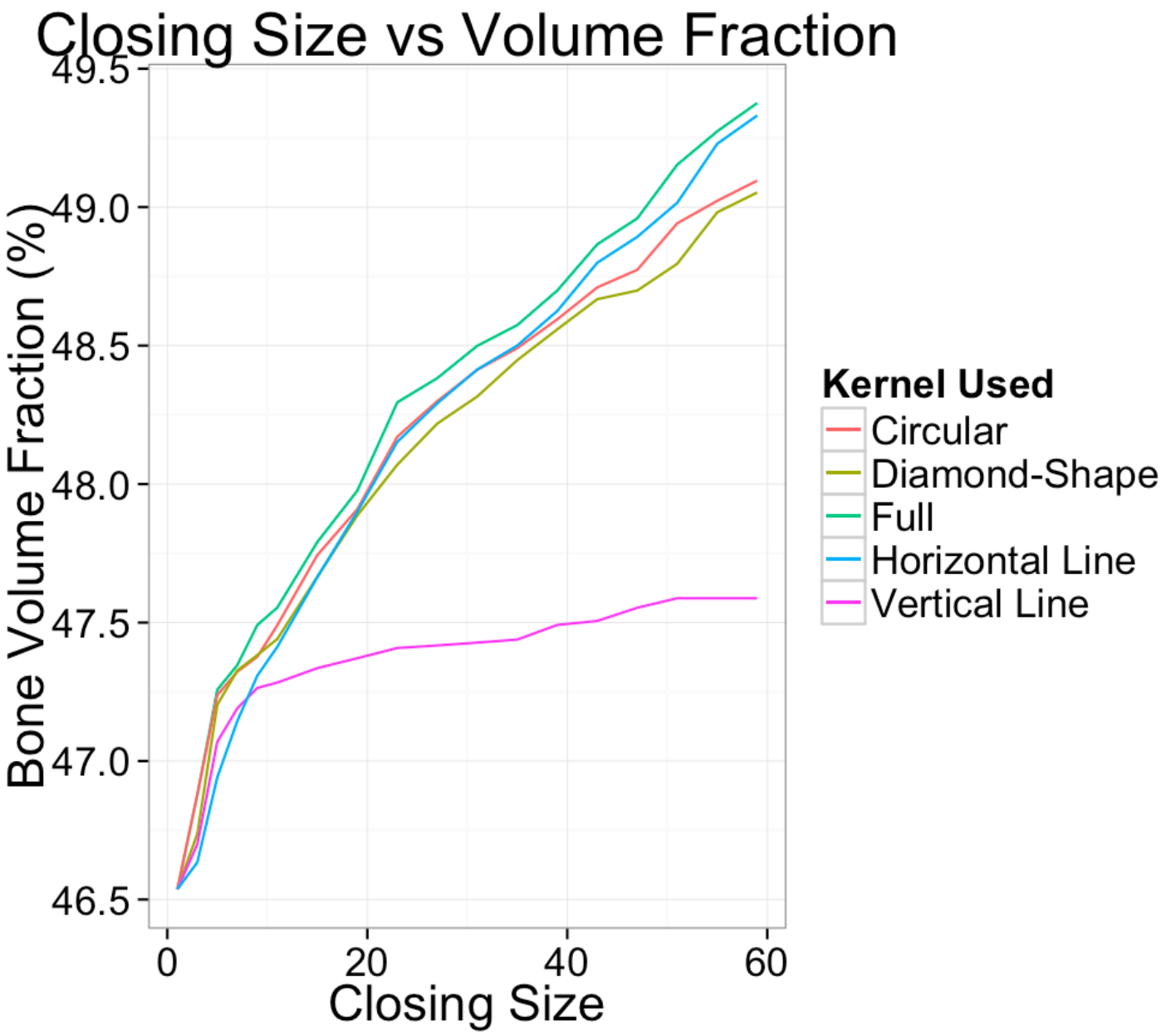
Using a better kernel shape (circular, diamond shaped, etc) the artifacts from morphological operations can be reduced

+/- R Code



We can characterize this by examining the dependency of the closing size and the volume fraction (note scale)

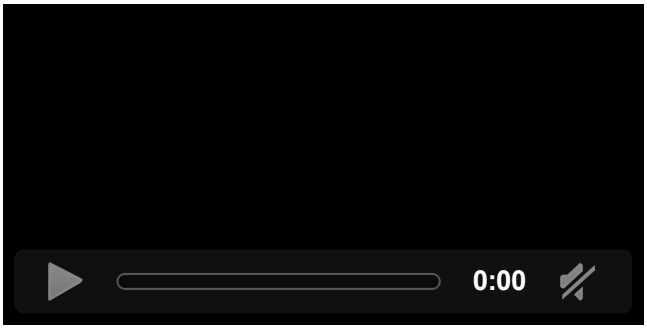
+/- R Code



- Alternative techniques
  - Convex Hull (http://en.wikipedia.org/wiki/Convex_hull) - assuming convex shapes
  - Flood Filling (http://en.wikipedia.org/wiki/Flood_fill) - fill connected pixels like in Microsoft Paint

Segmenting Fossils

Taken from the BBC Documentary First Life (http://www.bbc.co.uk/programmes/b00vw49d) by David Attenborough



- Gut Data
  - Slice 182 you can make out the intenstine track
- Teeth Data
  - Explore the sample and apply a threshold to locate the teeth using the 3D viewer

Quantifying Alzheimers: Segment the Cortex

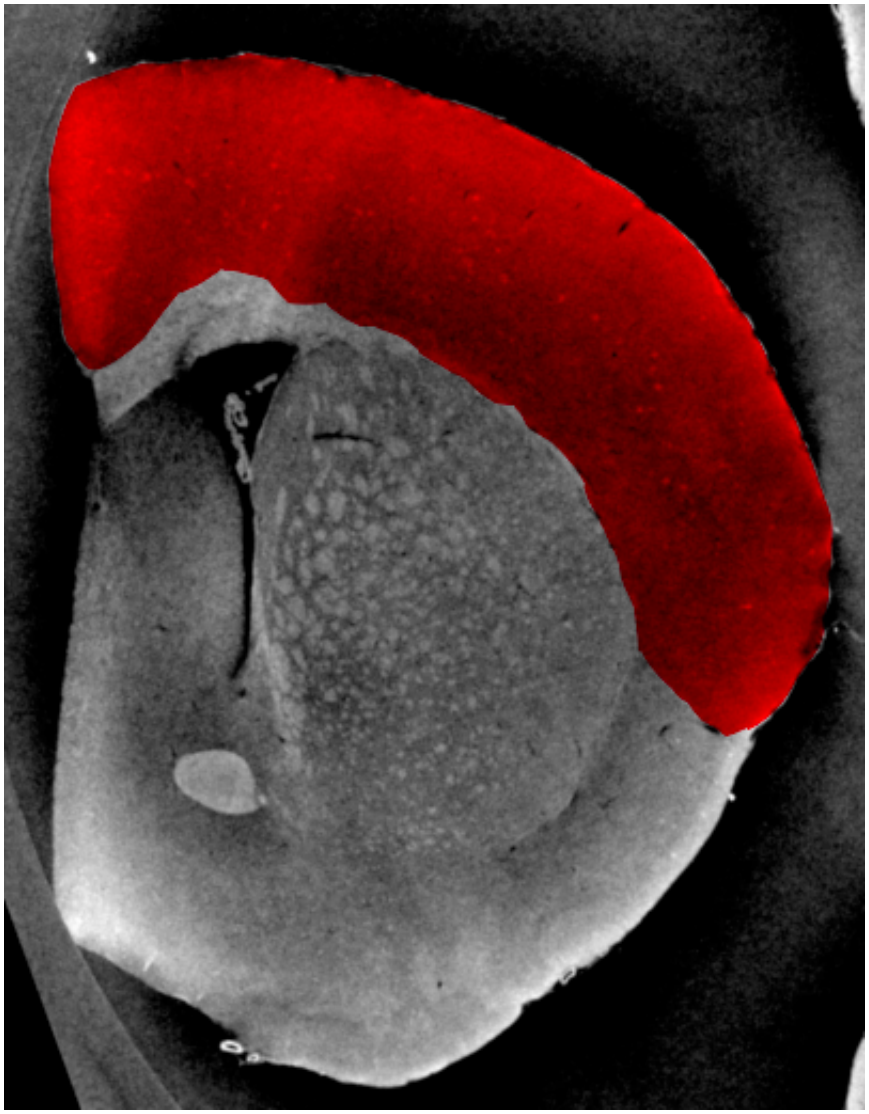
Courtesy of Alberto Alfonso

- Understand the progress of Alzheimer's disease as it relates to plaque formation in different regions of the

brain

- First identify different regions

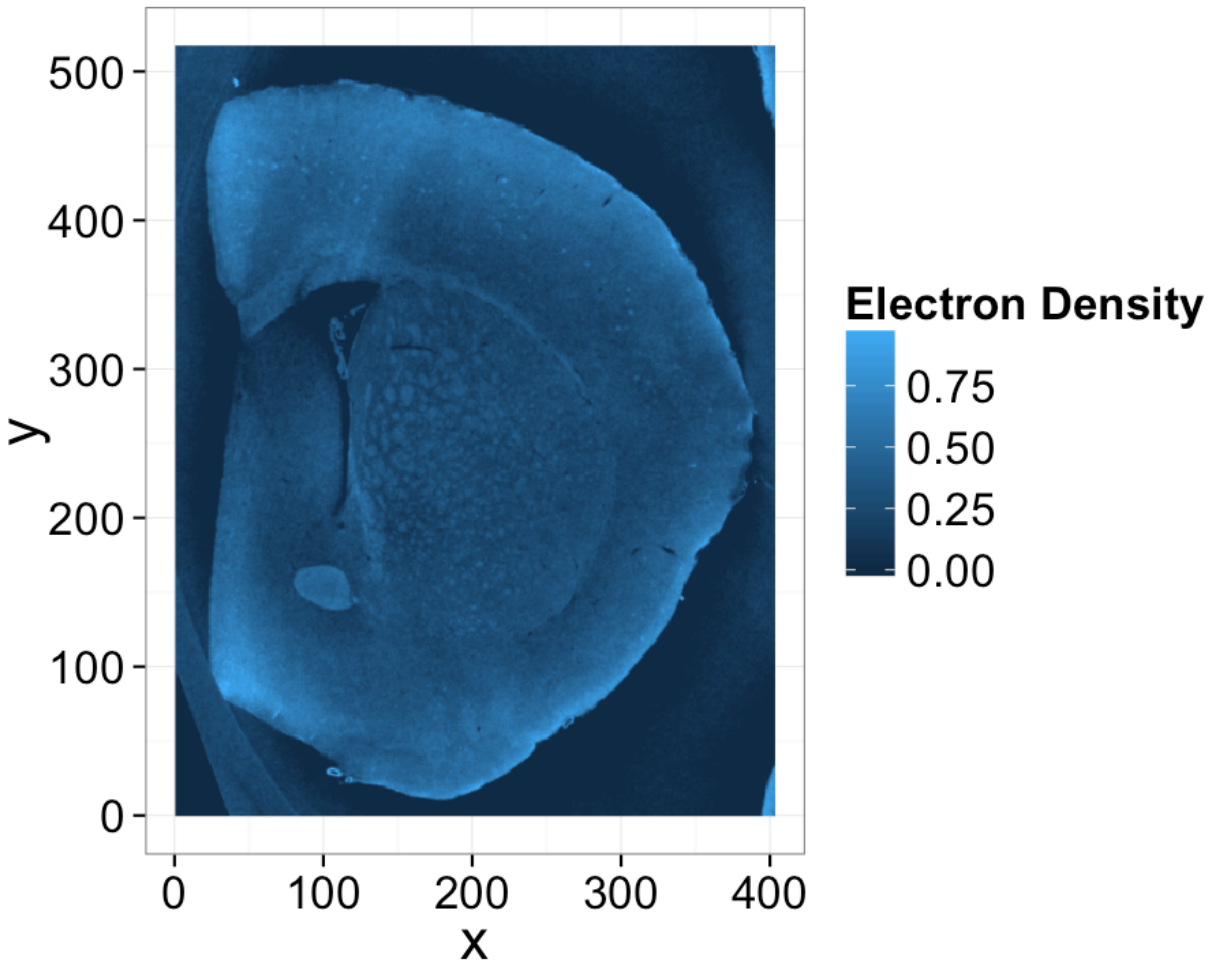
- Manually contoured



Quantifying Alzheimers: Segment the Cortex

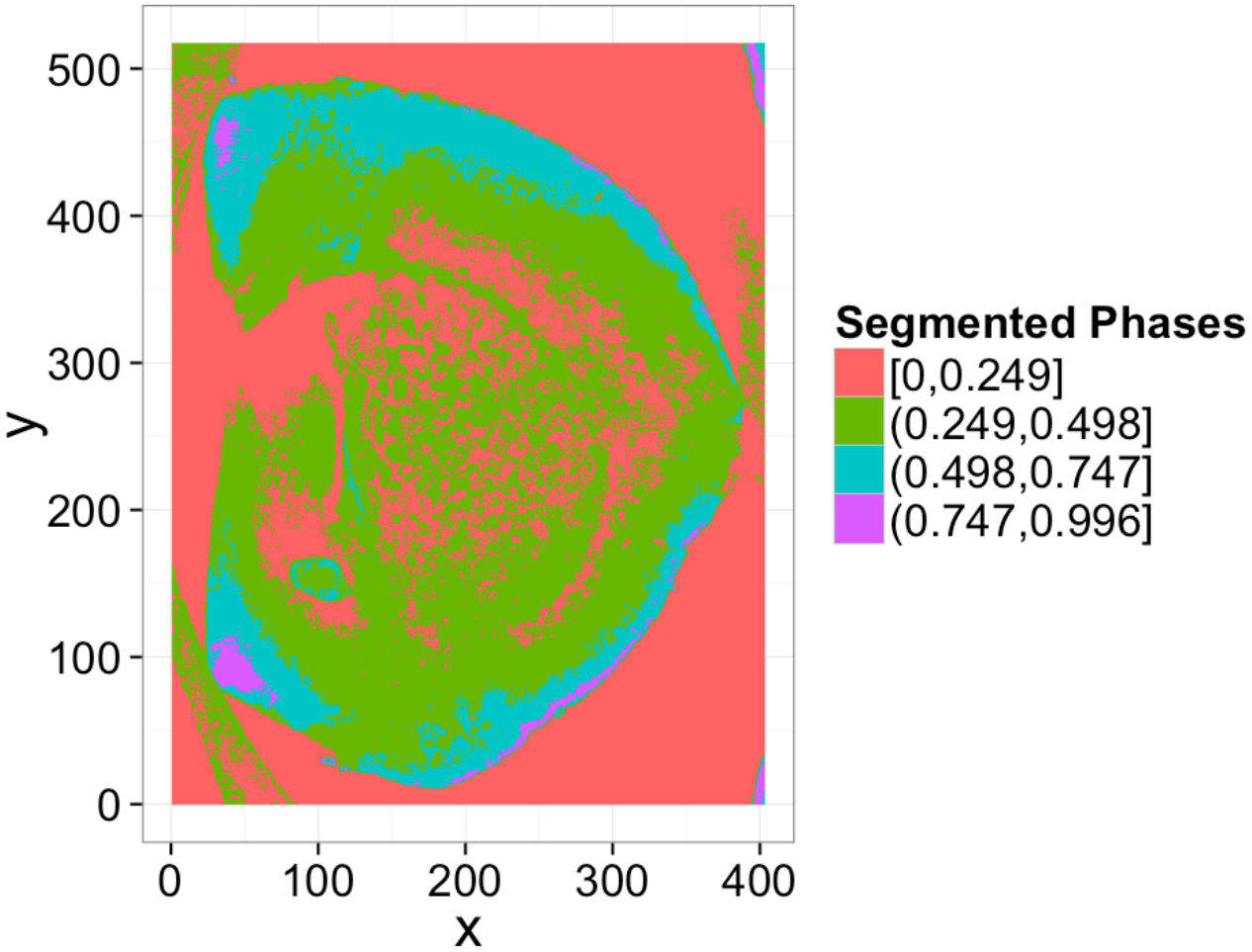
- The cortex is barely visible to the human eye
- Tiny structures hint at where cortex is located

+/- R Code



- A simple threshold is insufficient to finding the cortical structures
- Other filtering techniques are unlikely to magically fix this problem

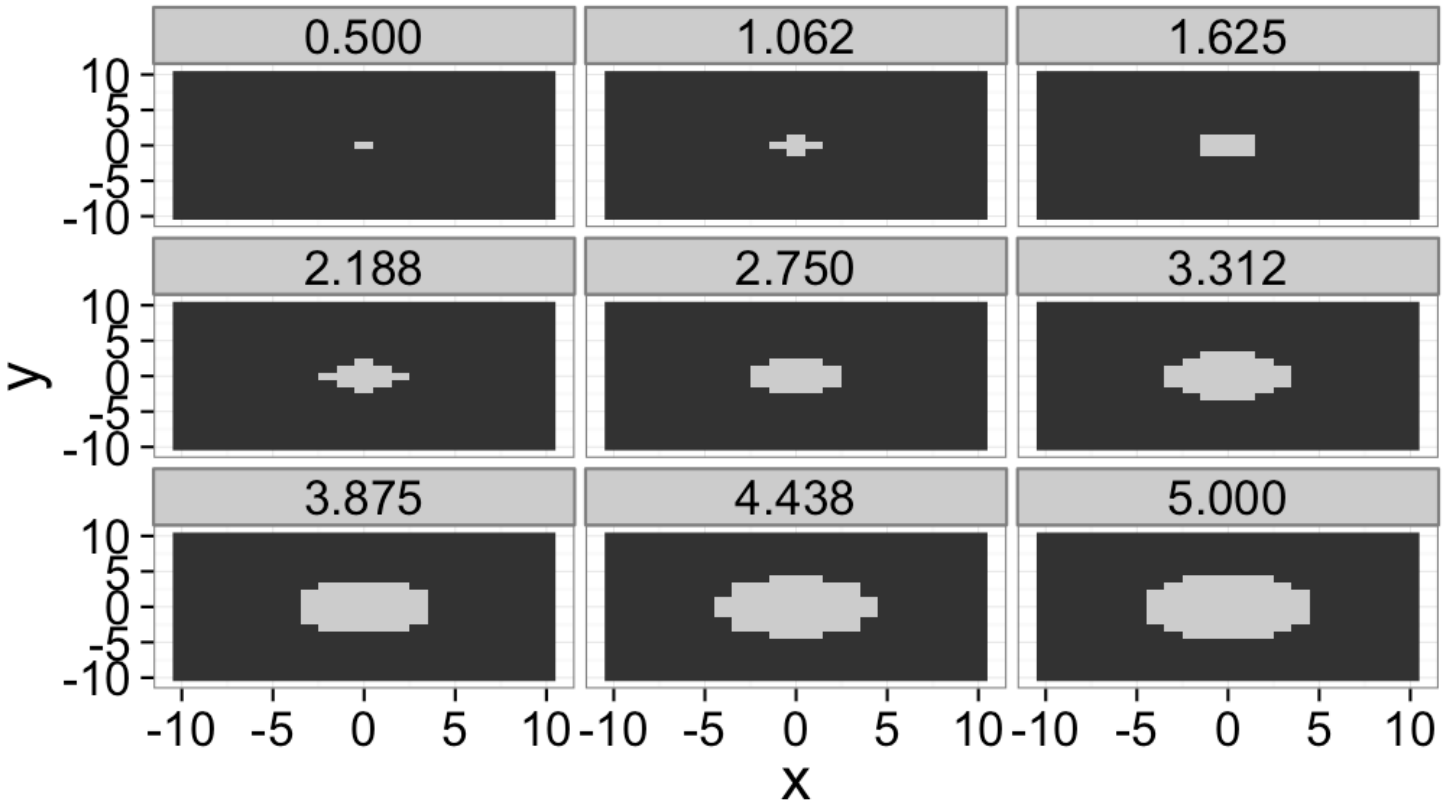
+/- R Code



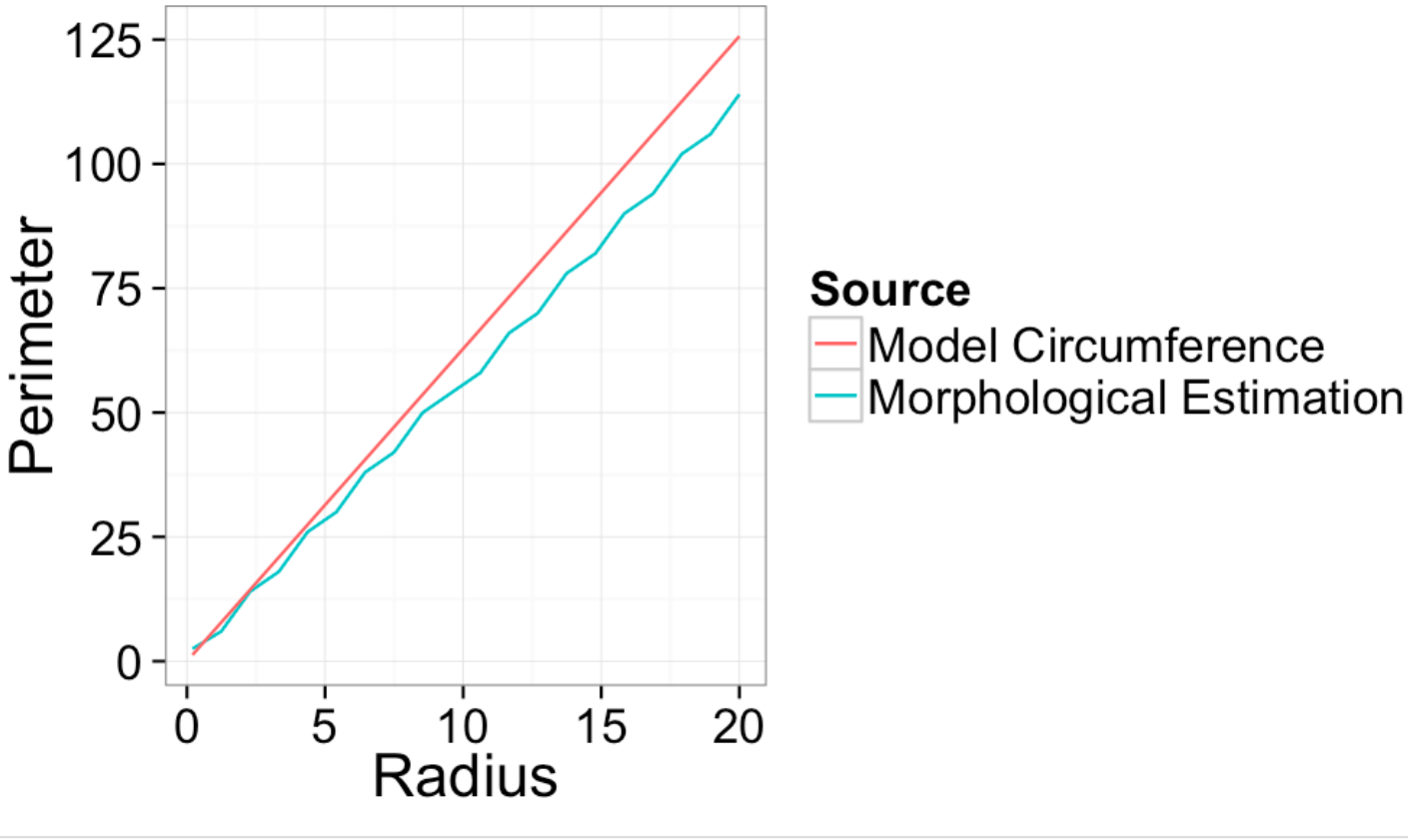
Surface Area / Perimeter

We see that the dilation and erosion affects are strongly related to the surface area of an object: the more surface area the larger the affect of a single dilation or erosion step.

+/- R Code



+/- R Code


125
100
75
50
25
0
Perimeter
0
5
10
15
20
Radius
Source
Model Circumference
Morphological Estimation