L8 15 Feb, Assign HW4, HW5

1) Download Moodle/Week 5/bullet dataset (binary)
2) Download Moodle/Week 5/HW4_Manipulate_segment_grains.nb
3) Download Moodle/Week 5/HW5_Connected_Components.nb
This lecture starts HW4, HW5 Manipulate, due Friday, 24 Feb.

Objective of HW4: (a) review Avizo lecture of week 3: "Watershed segmentation with Avizo Fire quantification module. (b) practice Module, functions, and Manipulate.

Objective of HW5: (a) using a label field matrix, (b) practice Map

Philosophy:

Programming is like chess. It takes practice, practice, practice. Also, feel free to ask for advice: “What’s the next move?”

The best programs that you write will be the programs that you can re-use next month, and re-use with understanding.

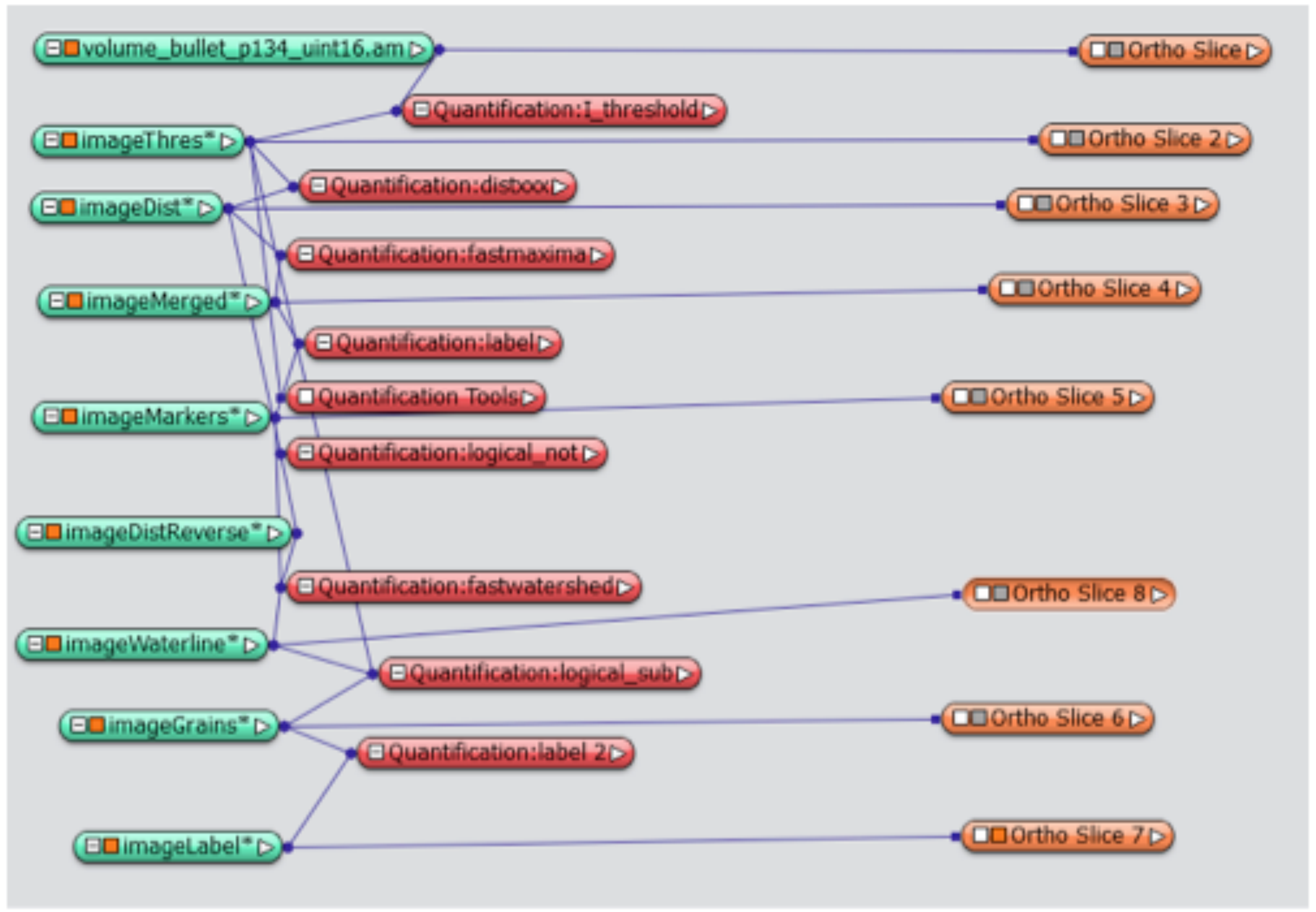
Grain segmentation uses a lot of image processing tools. The Avizo notes take 12 steps to generate the label field matrix and its colorized view.

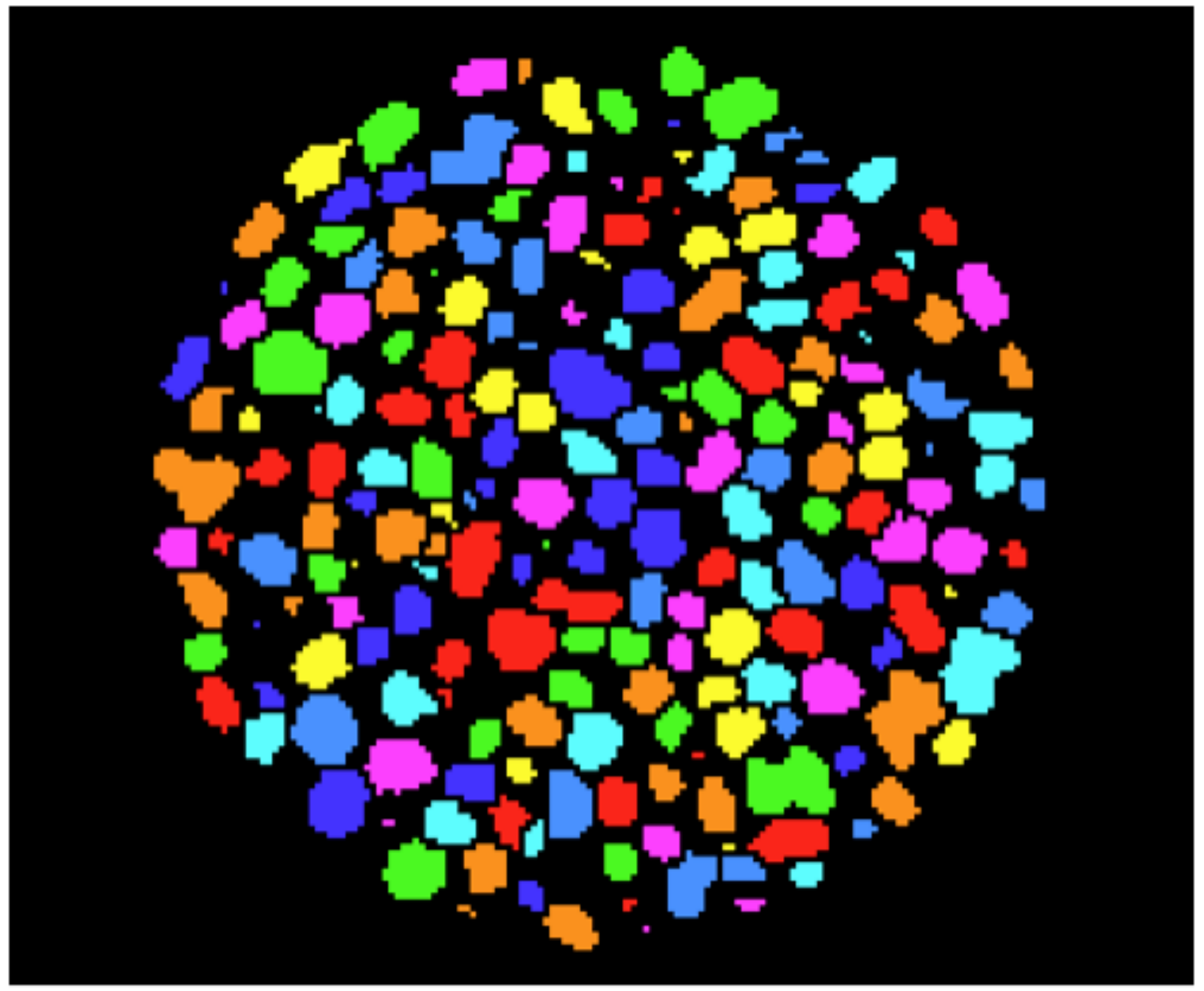
http://pcaster.net/viz_software/avizofire-watershed.html

AVIZO FIRE EXAMPLE

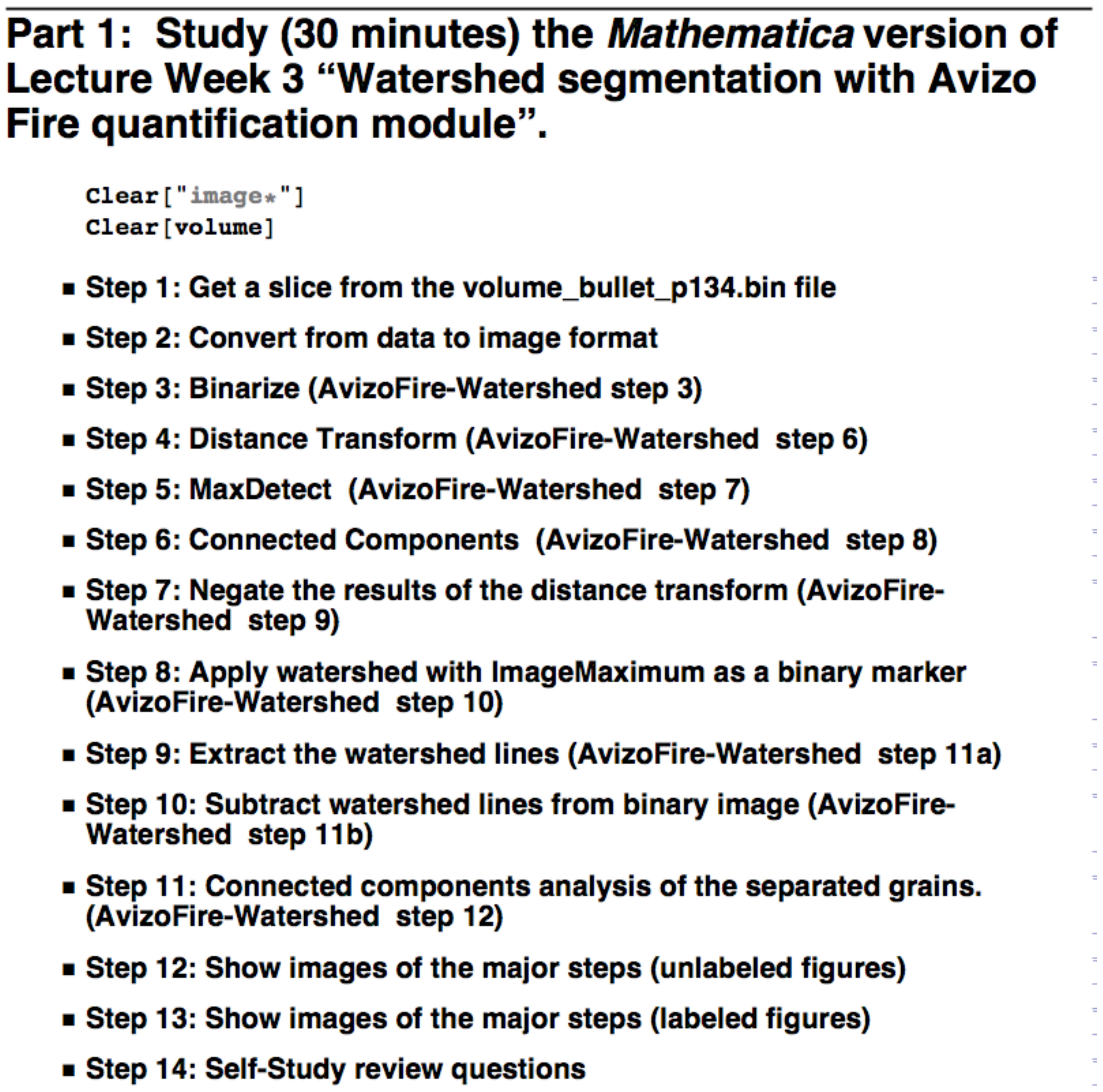
Watershed segmentation

Separating the propellant grains in bullet dataset





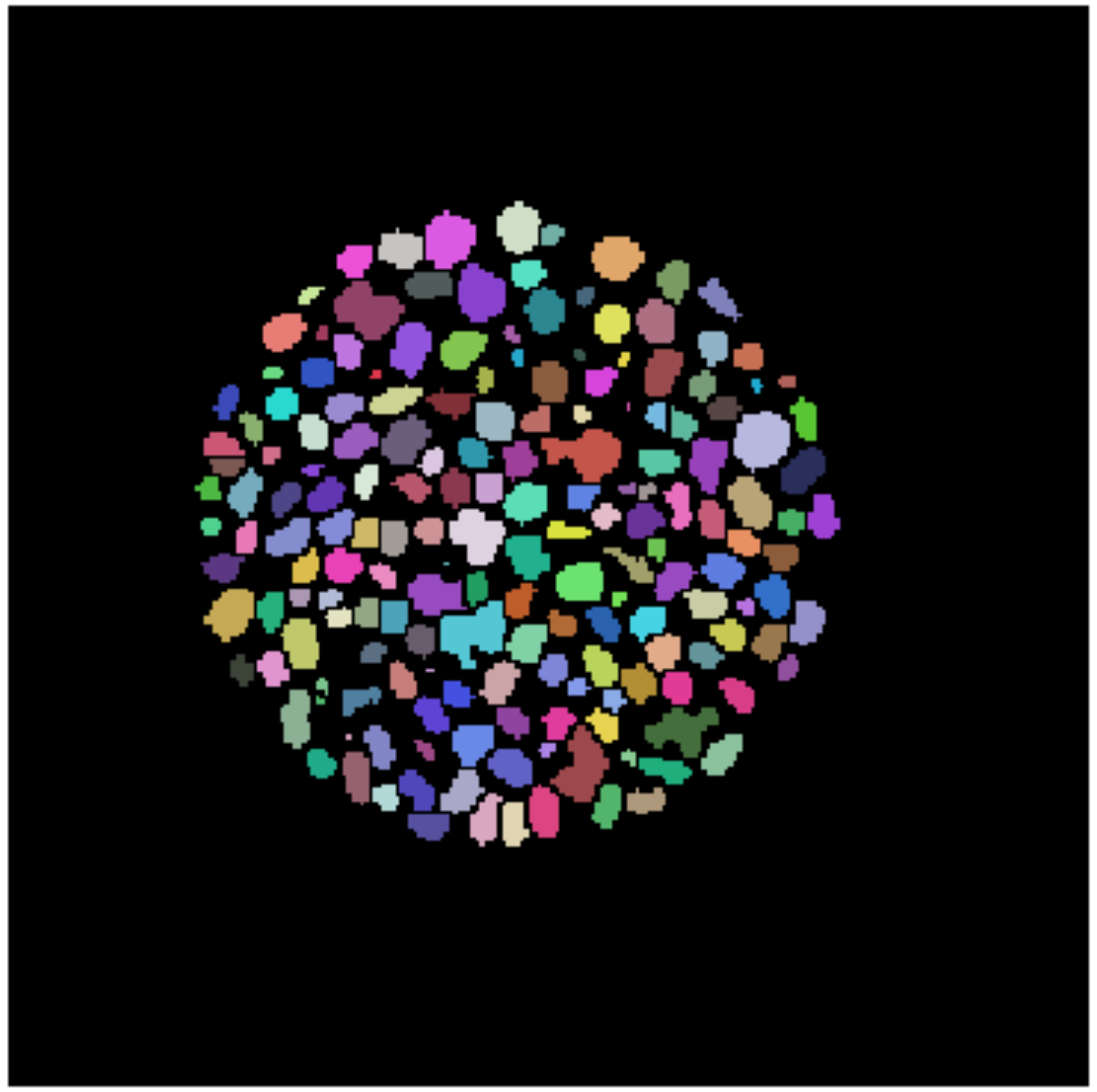
Grain segmentation uses a lot of image processing tools. The Mathematica code of HW4 also takes many steps to generate the label field matrix and its colorized view.



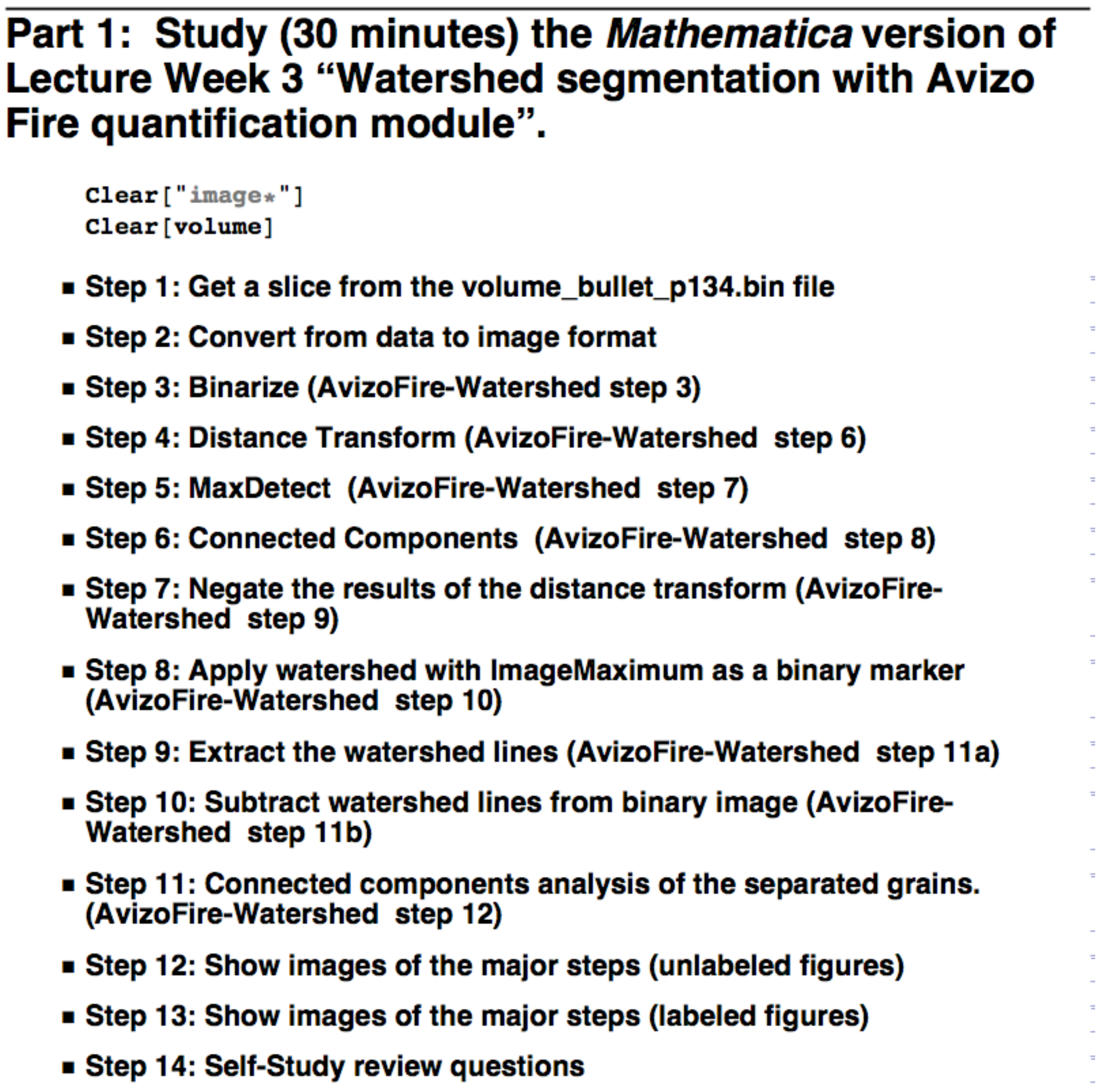
Part 1: Study (30 minutes) the *Mathematica* version of Lecture Week 3 "Watershed segmentation with Avizo Fire quantification module".

```
Clear["image*"]
Clear[volume]
```

- Step 1: Get a slice from the volume_bullet_p134.bin file
- Step 2: Convert from data to image format
- Step 3: Binarize (AvizoFire-Watershed step 3)
- Step 4: Distance Transform (AvizoFire-Watershed step 6)
- Step 5: MaxDetect (AvizoFire-Watershed step 7)
- Step 6: Connected Components (AvizoFire-Watershed step 8)
- Step 7: Negate the results of the distance transform (AvizoFire-Watershed step 9)
- Step 8: Apply watershed with ImageMaximum as a binary marker (AvizoFire-Watershed step 10)
- Step 9: Extract the watershed lines (AvizoFire-Watershed step 11a)
- Step 10: Subtract watershed lines from binary image (AvizoFire-Watershed step 11b)
- Step 11: Connected components analysis of the separated grains. (AvizoFire-Watershed step 12)
- Step 12: Show images of the major steps (unlabeled figures)
- Step 13: Show images of the major steps (labeled figures)
- Step 14: Self-Study review questions



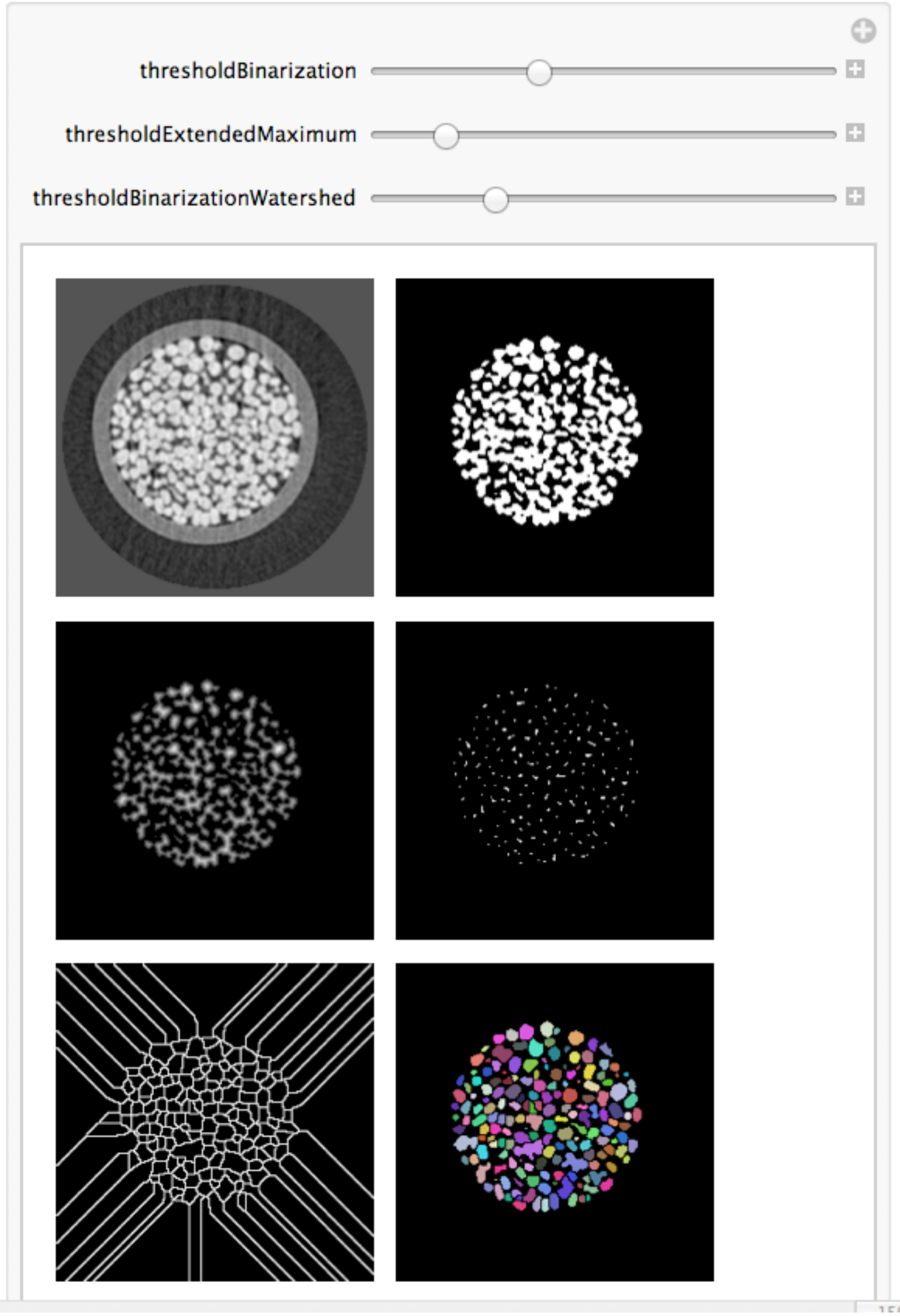
HW4. Apply Module, function, and Manipulate to Steps 3-11. The goal is (a) to practice these commands and (b) to have a friendlier segmentation program.



Part 1: Study (30 minutes) the *Mathematica* version of Lecture Week 3 “Watershed segmentation with Avizo Fire quantification module”.

```
Clear["image*"]
Clear[volume]
```

- Step 1: Get a slice from the volume_bullet_p134.bin file
- Step 2: Convert from data to image format
- Step 3: Binarize (AvizoFire-Watershed step 3)
- Step 4: Distance Transform (AvizoFire-Watershed step 6)
- Step 5: MaxDetect (AvizoFire-Watershed step 7)
- Step 6: Connected Components (AvizoFire-Watershed step 8)
- Step 7: Negate the results of the distance transform (AvizoFire-Watershed step 9)
- Step 8: Apply watershed with ImageMaximum as a binary marker (AvizoFire-Watershed step 10)
- Step 9: Extract the watershed lines (AvizoFire-Watershed step 11a)
- Step 10: Subtract watershed lines from binary image (AvizoFire-Watershed step 11b)
- Step 11: Connected components analysis of the separated grains. (AvizoFire-Watershed step 12)
- Step 12: Show images of the major steps (unlabeled figures)
- Step 13: Show images of the major steps (labeled figures)
- Step 14: Self-Study review questions



thresholdBinarization
thresholdExtendedMaximum
thresholdBinarizationWatershed

HW4. Apply Module, function, and Manipulate to Steps 3-11. The goal is (a) to practice these commands and (b) to have a friendlier segmentation program.

Comments about HW4 (page 1 of 4):

• Use volume_bullet_p134.bin The pixels range from 20,000 to 40,000. The code is expecting values in this range.

• Step 1 extracts a slice, with grains, from the volume. The slice is randomly chosen based on your student ID number (or any number you pick). Everyone has a slightly different HW problem.

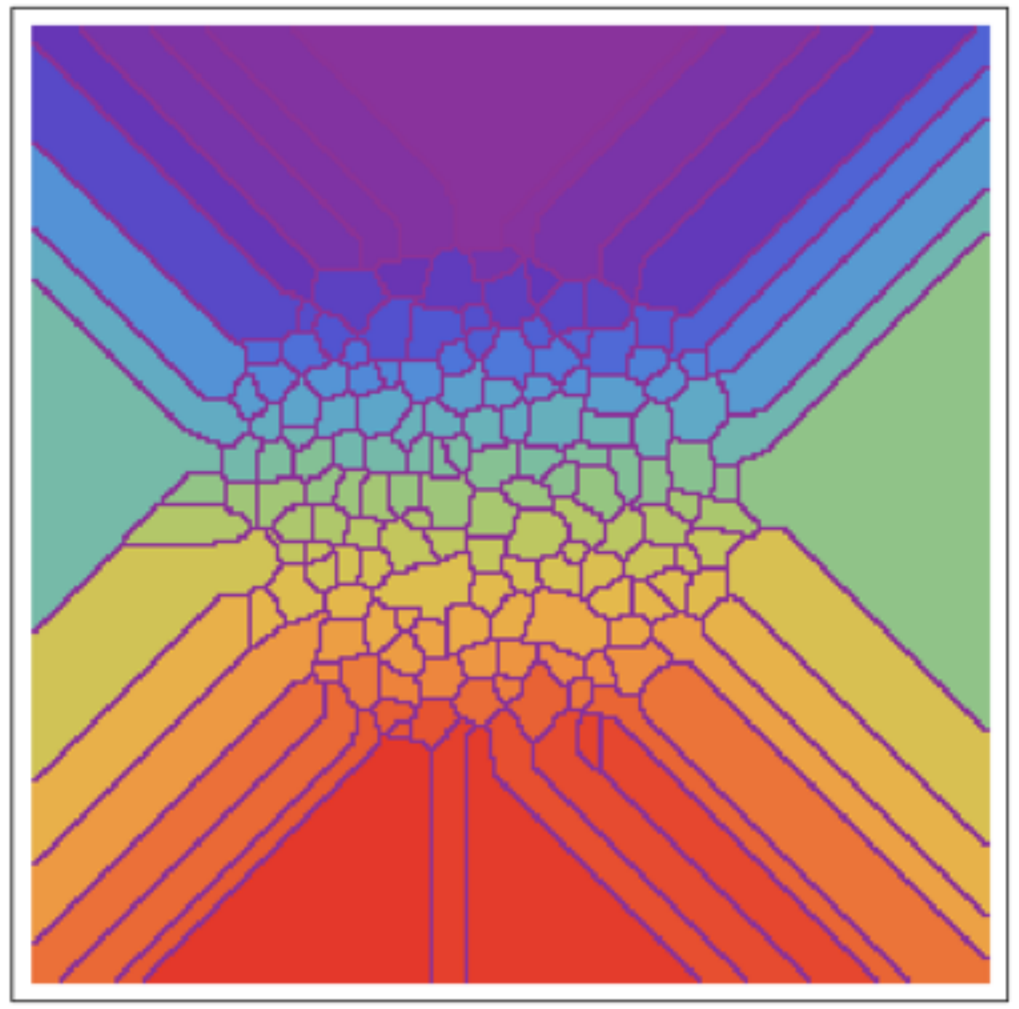
• Mathematica uses data type "image". A 2D array of numbers is converted into "image" in Step 2 with the command imageSlice = Image[myNewSlice, "Bit16"]

• Most of the Step 14 self-study questions are based on histograms of the "image". A function for plotting linear and log histograms is defined in Step 1.

• If the Part 1 code runs weirdly, clear all images and the volume and re-start at Step 1.

More comments about HW4 (page 2 of 4)

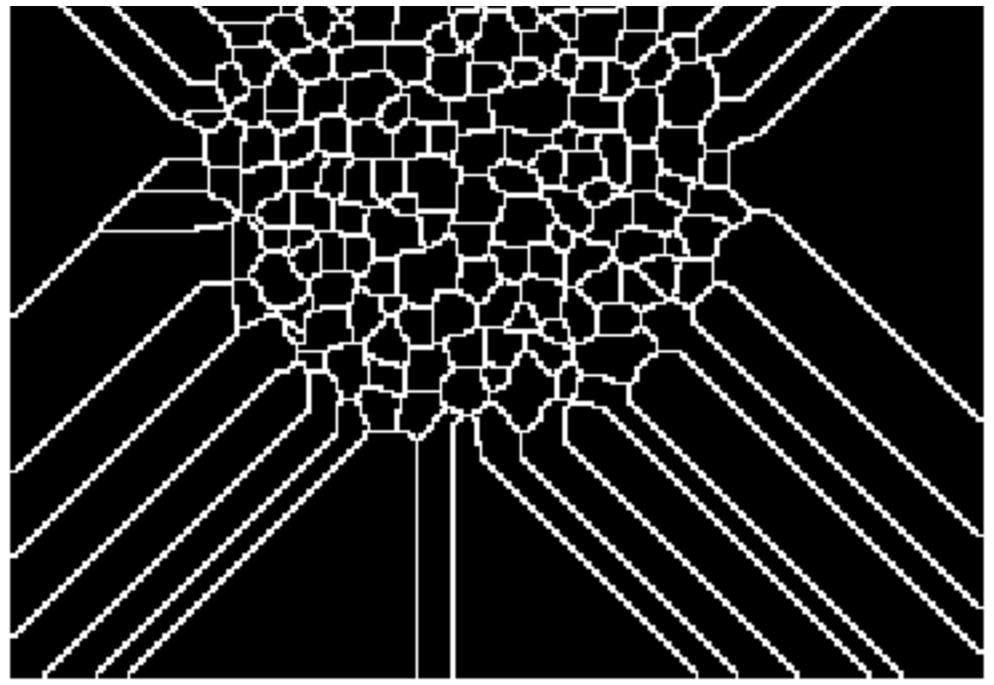
- Step 2: Explore the effect of ImageAdjust on image called imageSlice. Note how ImageAdjust changes the small histogram.
- Step 3: The Binarize command uses a threshold. **Great place for Manipulate**
- Step 5: The MaxDetect (the extended maximum algorithm) command uses a threshold. **Great place for Manipulate**
- Step 6: Note the values in the linear histogram. Why? This is covered in the self-study.
- Step 7: You could compare results of ImageHistogram[imageDistance] and ImageHistogram[imageDistanceNegate]
- Step 8: A pretty image:
  ArrayPlot[dataWatershedLines, ColorFunction -> "Rainbow"]
  The command WatersheComponents as two inputs:
  (a) the image, imageDistanceNegate
  (b) markers (or starting points), imageMaximum



More comments about HW4 (page 3 of 4)

• Step 9: This code needs improvement. The goal is white lines on a black background, but the code is way too sensitive to the value of the binarization threshold. Any suggestions for code improvement?

```
imageWatershedLines = Image[dataWatershedLines, "Bit16"];
imageWatershedLines = ColorNegate[Binarize[imageWatershedLines, 0.00001]]
```



• Step 10: Subtraction (sort of).

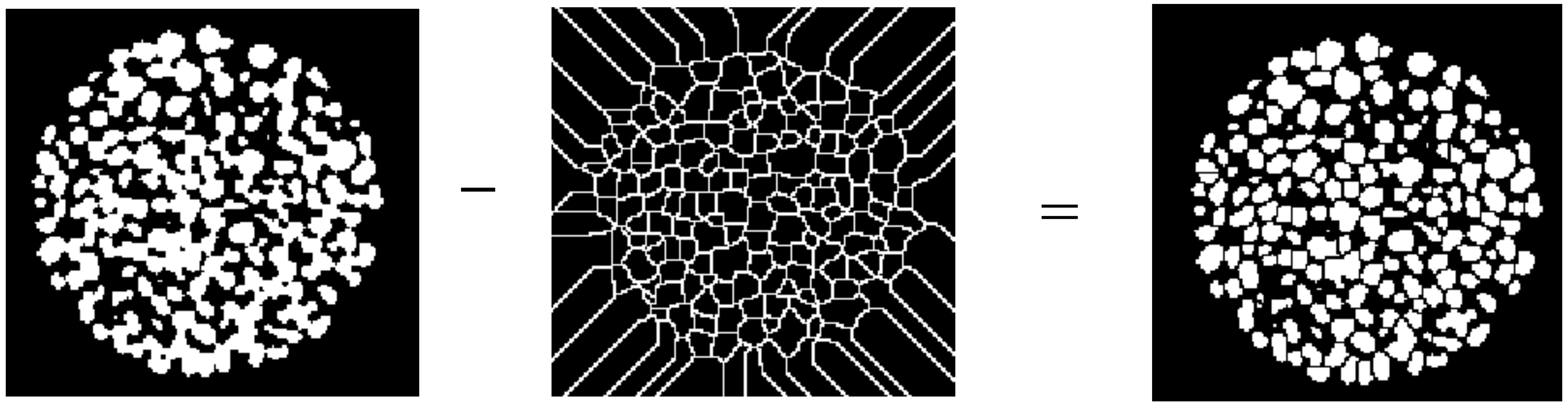
In image arithmetic, note that:

black - white = black

or

0 - 1 = 0 (not -1)

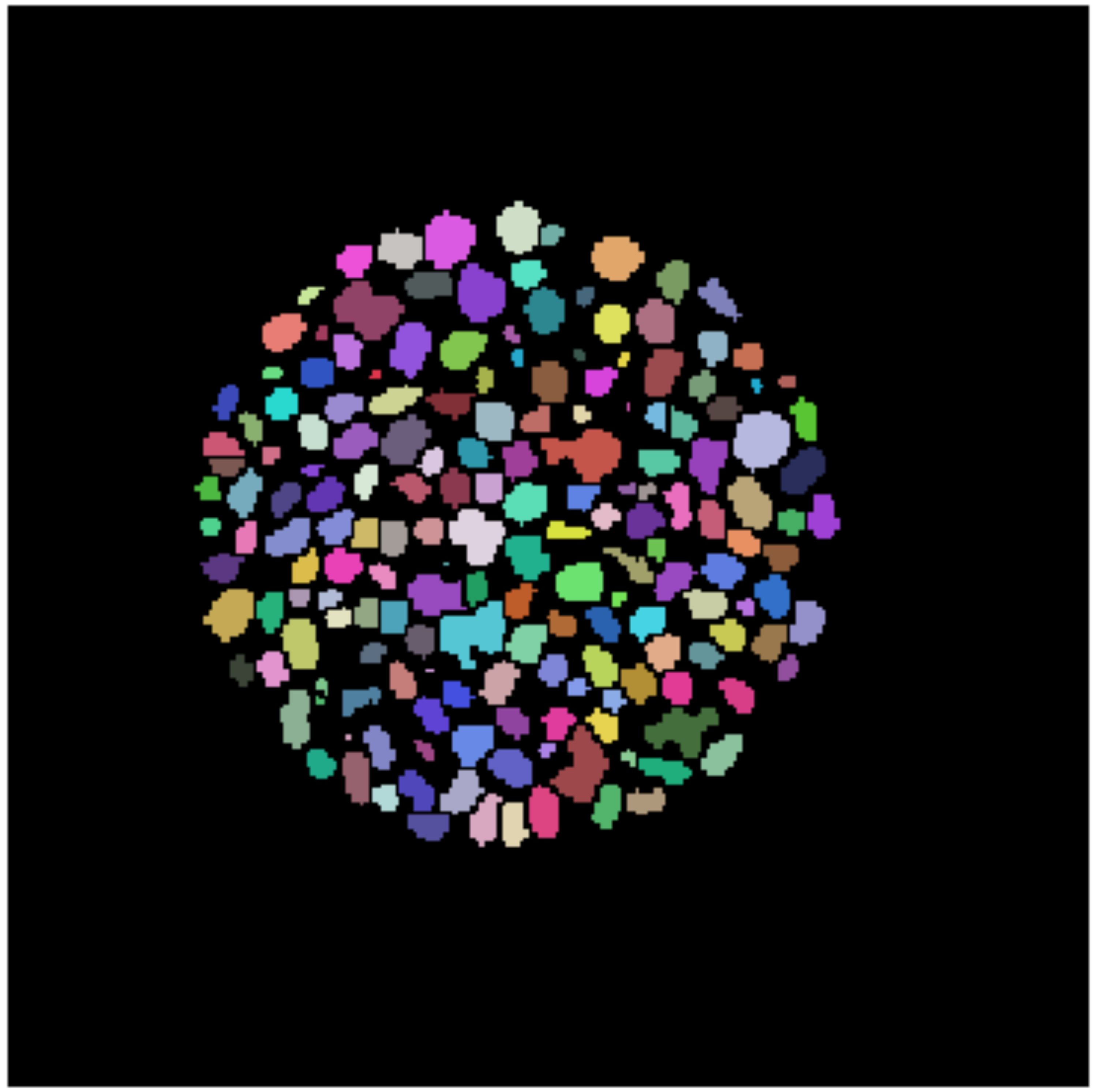
Anyway, we get a nice binary image.

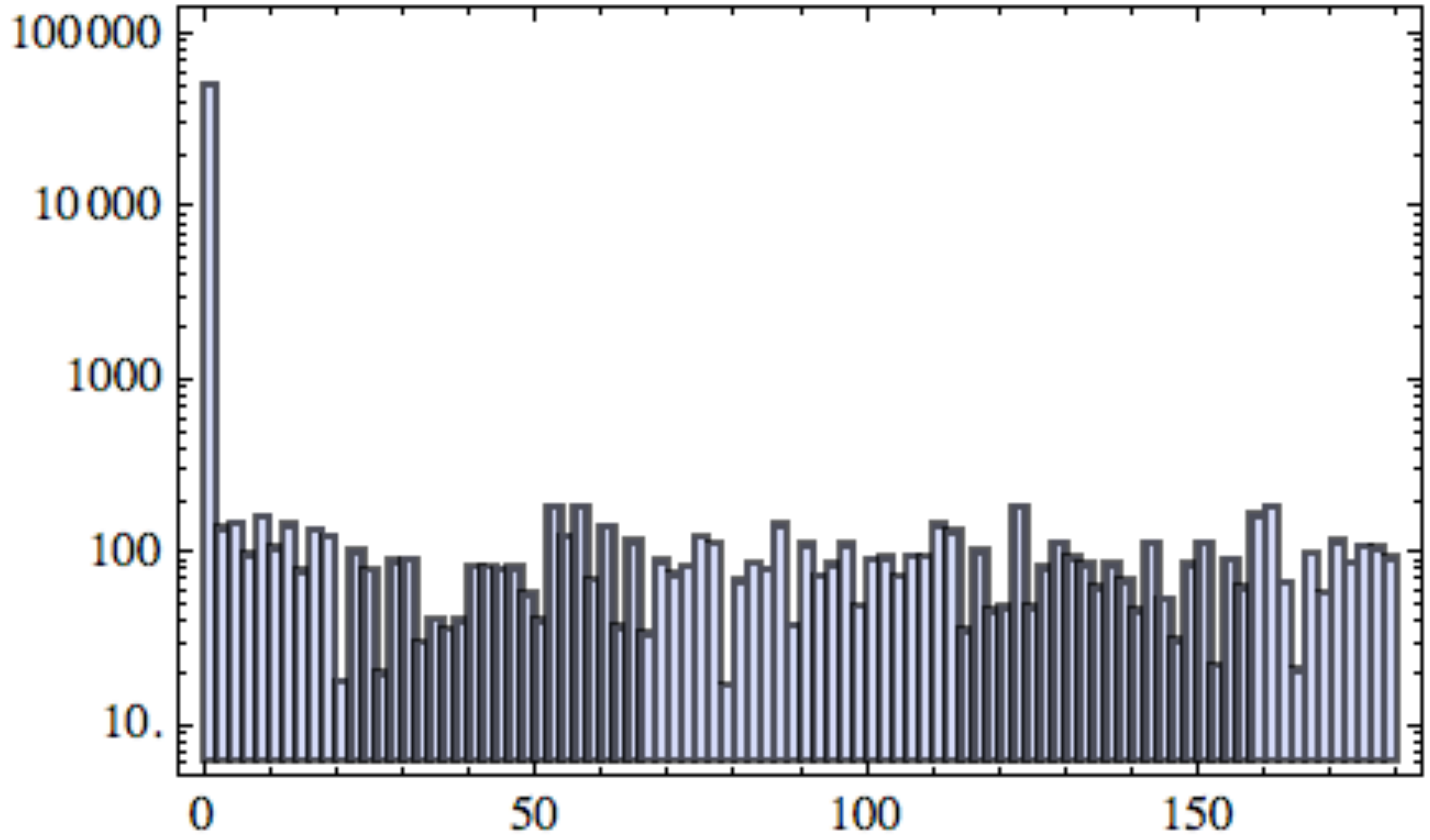


More comments about HW4 (page 4 of 4)

- Step 11: Connected components. The variable "dataComponents" is numbers (not image format). It's mostly 0 (background). Get comfortable with the range and distribution of numbers in this **label field matrix**. HW5 starts with this label field matrix and goes on.

```
{243, 243}
```

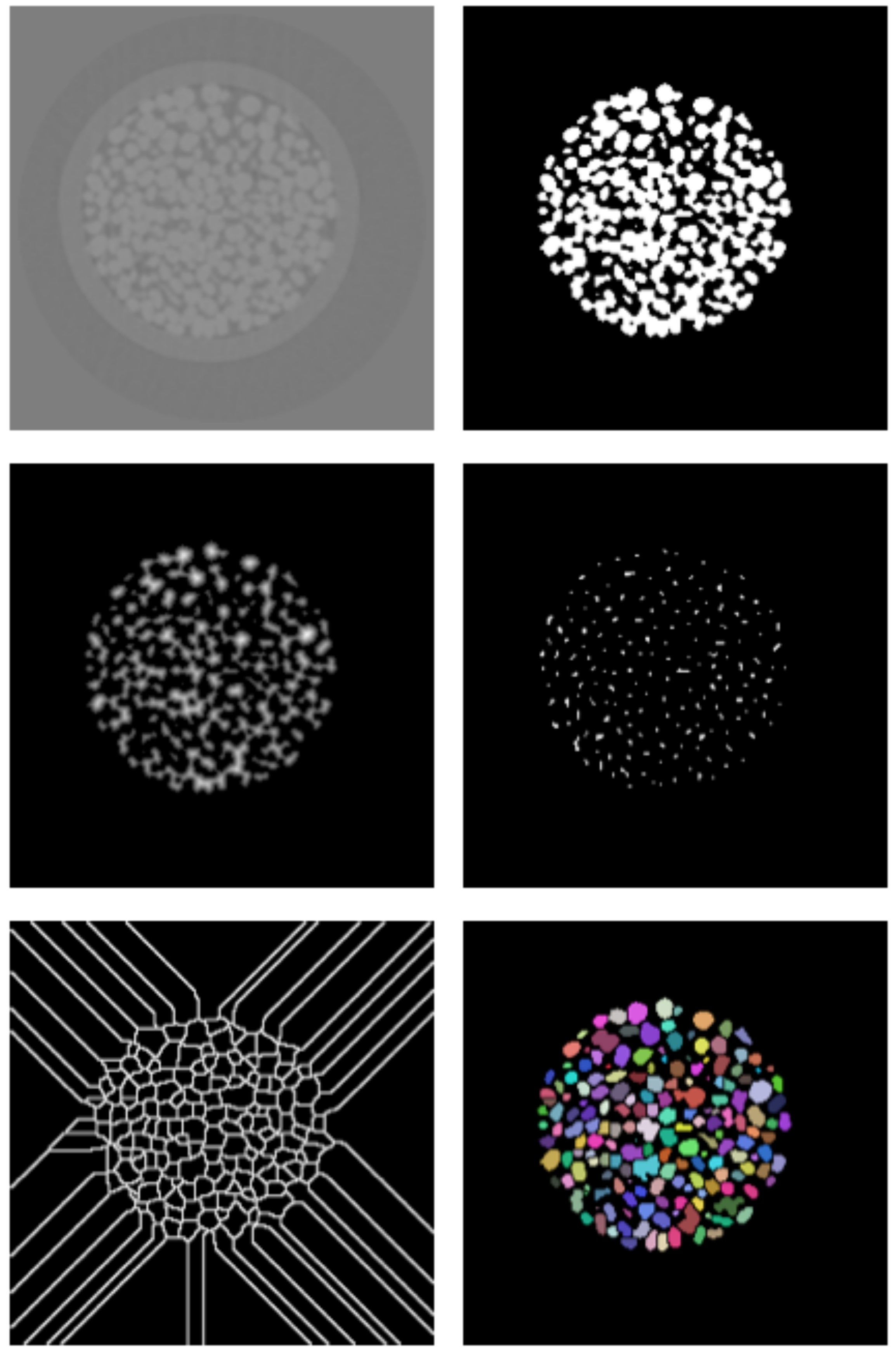




More comments about HW4 (page 5 of 4) (extras)

- Step 12 the figures

```
GraphicsGrid[{{imageSlice, imageBinary}, {imageDistance, imageMaximum},
  {imageWatershedLines, imageSegmentedGrains}}]
```

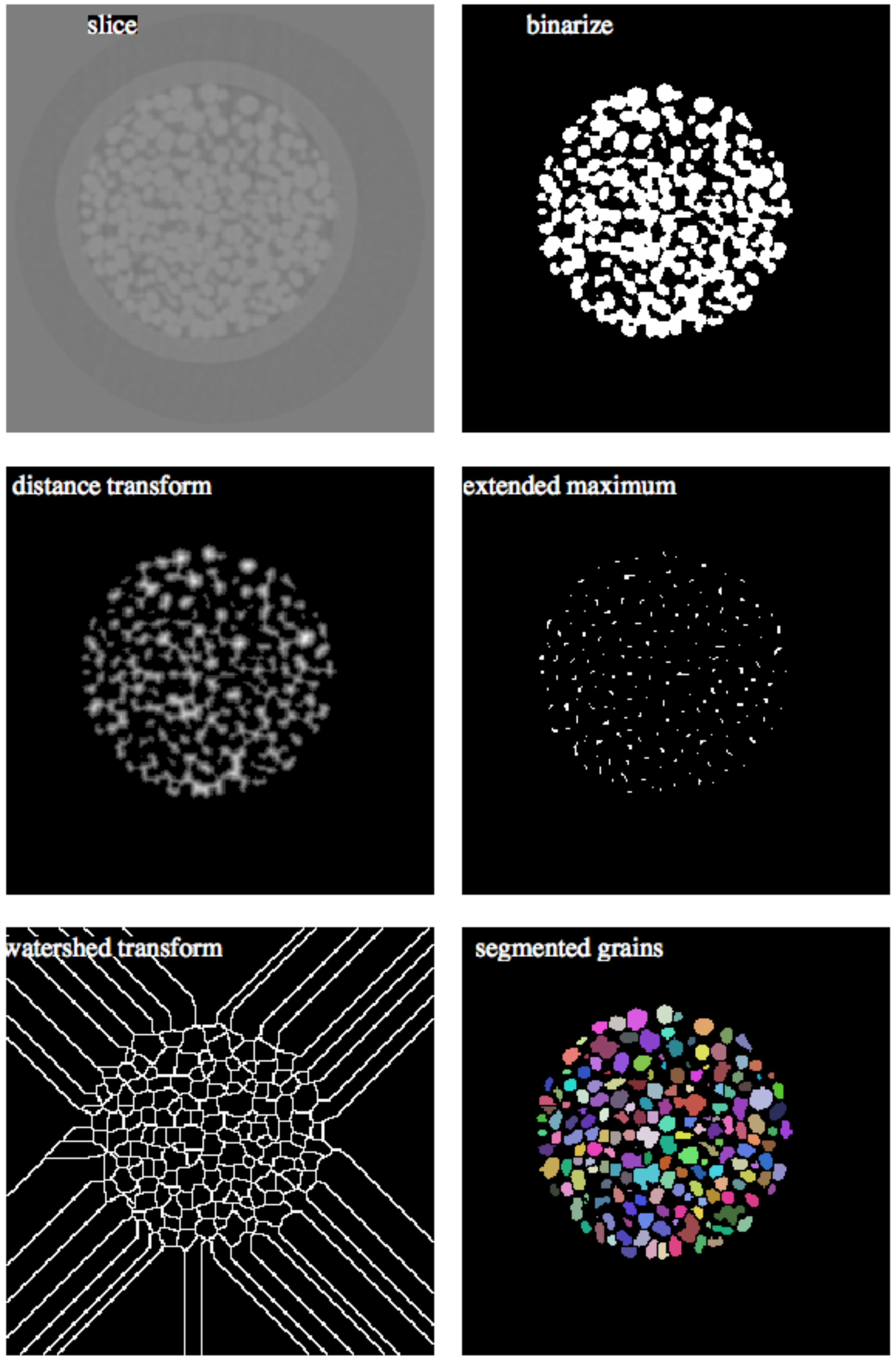


More comments about HW4 (page 6 of 4) (extras)

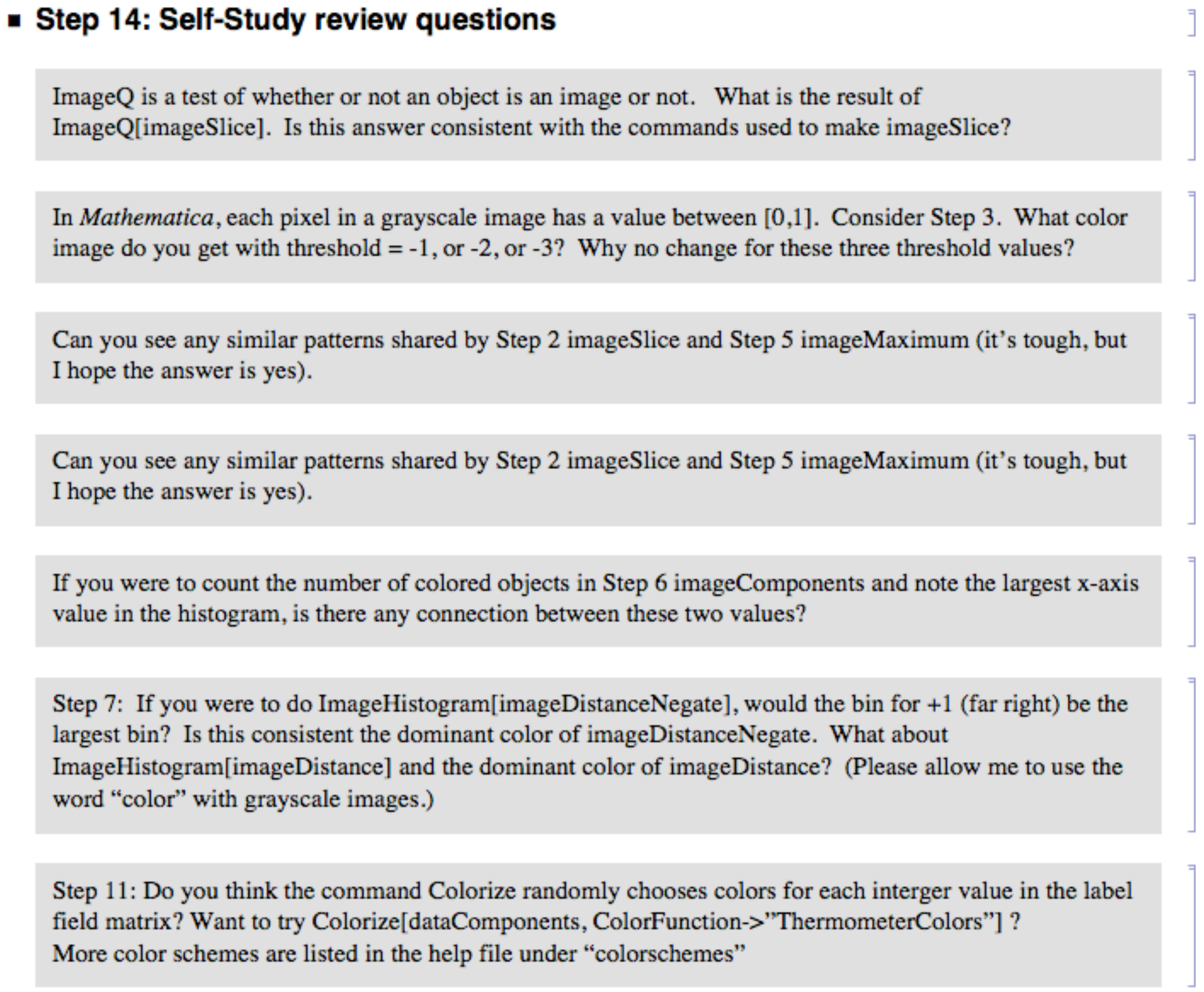
- Step 13 the labeled figures

```
gTextSlice = Text[Style["slice", 14, White, Background → Black]];
gTextBinary = Text[Style["binarize", 14, White, Background → Black]];
gTextDistance = Text[Style["distance transform", 14, White, Background → Black]];
gTextMaximum = Text[Style["extended maximum", 14, White, Background → Black]];
gTextWatershedLines = Text[Style["watershed transform", 14, White, Background → Black]];
gTextSegmented = Text[Style["segmented grains", 14, White, Background → Black]];

GraphicsGrid[{{
    Show[imageSlice, Epilog → Inset[gTextSlice, Scaled[{0.25, 0.95}]]],
    Show[imageBinary, Epilog → Inset[gTextBinary, Scaled[{0.25, 0.95}]]]
   },
   {Show[imageDistance, Epilog → Inset[gTextDistance, Scaled[{0.25, 0.95}]]],
    Show[imageMaximum, Epilog → Inset[gTextMaximum, Scaled[{0.25, 0.95}]]]
   },
   {Show[imageWatershedLines, Epilog → Inset[gTextWatershedLines, Scaled[{0.25, 0.95}]]],
    Show[imageSegmentedGrains, Epilog → Inset[gTextSegmented, Scaled[{0.25, 0.95}]]]
   }}, ImageSize → {500, 700}]
```



More comments about HW4 (page 7 of 4) (extras)



- **Step 14: Self-Study review questions**

ImageQ is a test of whether or not an object is an image or not. What is the result of ImageQ[imageSlice]. Is this answer consistent with the commands used to make imageSlice?

In *Mathematica*, each pixel in a grayscale image has a value between [0,1]. Consider Step 3. What color image do you get with threshold = -1, or -2, or -3? Why no change for these three threshold values?

Can you see any similar patterns shared by Step 2 imageSlice and Step 5 imageMaximum (it's tough, but I hope the answer is yes).

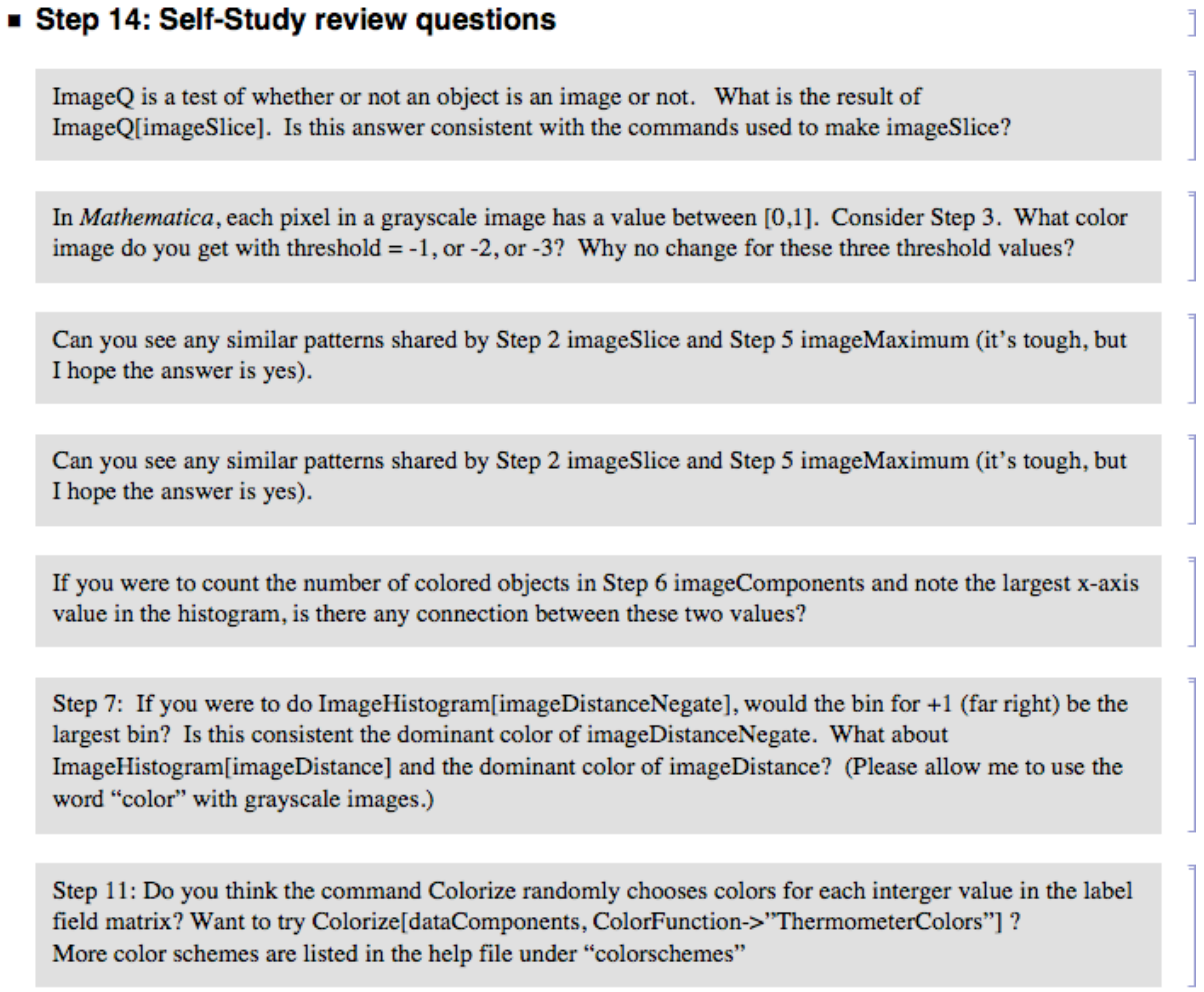
Can you see any similar patterns shared by Step 2 imageSlice and Step 5 imageMaximum (it's tough, but I hope the answer is yes).

If you were to count the number of colored objects in Step 6 imageComponents and note the largest x-axis value in the histogram, is there any connection between these two values?

Step 7: If you were to do ImageHistogram[imageDistanceNegate], would the bin for +1 (far right) be the largest bin? Is this consistent the dominant color of imageDistanceNegate. What about ImageHistogram[imageDistance] and the dominant color of imageDistance? (Please allow me to use the word "color" with grayscale images.)

Step 11: Do you think the command Colorize randomly chooses colors for each interger value in the label field matrix? Want to try Colorize[dataComponents, ColorFunction->"ThermometerColors"] ?
More color schemes are listed in the help file under "colorschemes"

More comments about HW4 (page 7 of 4) (extras)



- **Step 14: Self-Study review questions**

ImageQ is a test of whether or not an object is an image or not. What is the result of ImageQ[imageSlice]. Is this answer consistent with the commands used to make imageSlice?

In *Mathematica*, each pixel in a grayscale image has a value between [0,1]. Consider Step 3. What color image do you get with threshold = -1, or -2, or -3? Why no change for these three threshold values?

Can you see any similar patterns shared by Step 2 imageSlice and Step 5 imageMaximum (it's tough, but I hope the answer is yes).

Can you see any similar patterns shared by Step 2 imageSlice and Step 5 imageMaximum (it's tough, but I hope the answer is yes).

If you were to count the number of colored objects in Step 6 imageComponents and note the largest x-axis value in the histogram, is there any connection between these two values?

Step 7: If you were to do ImageHistogram[imageDistanceNegate], would the bin for +1 (far right) be the largest bin? Is this consistent the dominant color of imageDistanceNegate. What about ImageHistogram[imageDistance] and the dominant color of imageDistance? (Please allow me to use the word "color" with grayscale images.)

Step 11: Do you think the command Colorize randomly chooses colors for each interger value in the label field matrix? Want to try Colorize[dataComponents, ColorFunction->"ThermometerColors"] ?
More color schemes are listed in the help file under "colorschemes"