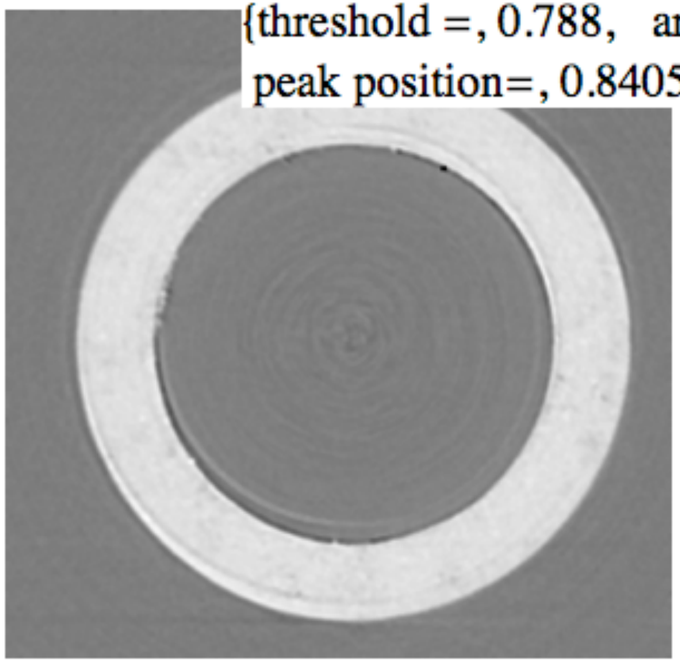
L11, 5 Mar, Segmentation: MAS Rotor

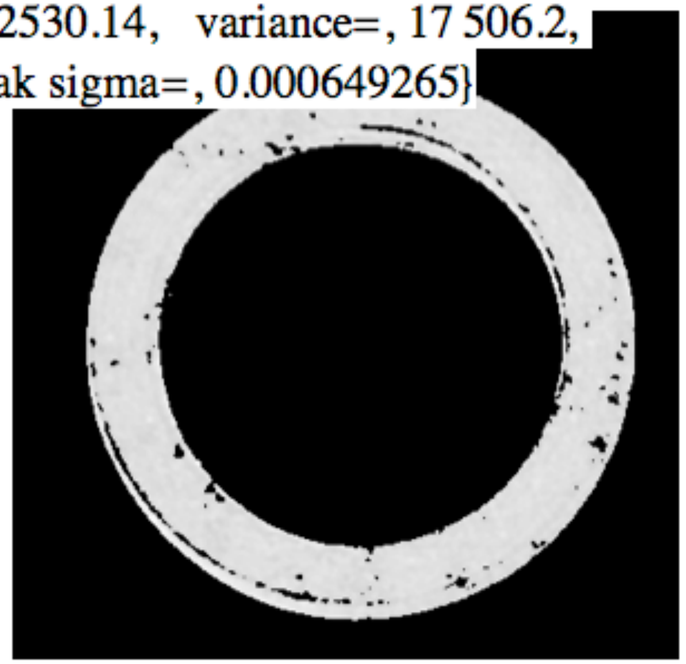
1) Download Moodle/Week 8/Pgm10_Segmentation_MAS_rotor.nb
2) optional. Download Moodle/Week 8/MAS_rotor_cropped.h5 (177MB)

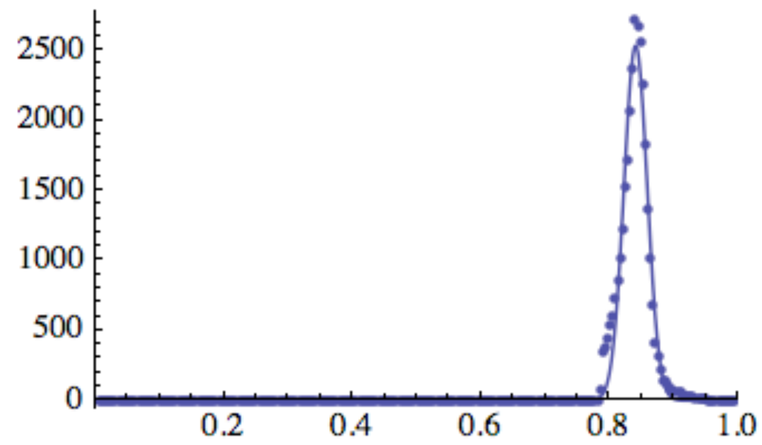
indexImage

thresholdHigh

{threshold =, 0.788, area=, 2530.14, variance=, 17 506.2,
 peak position=, 0.8405, peak sigma=, 0.000649265}



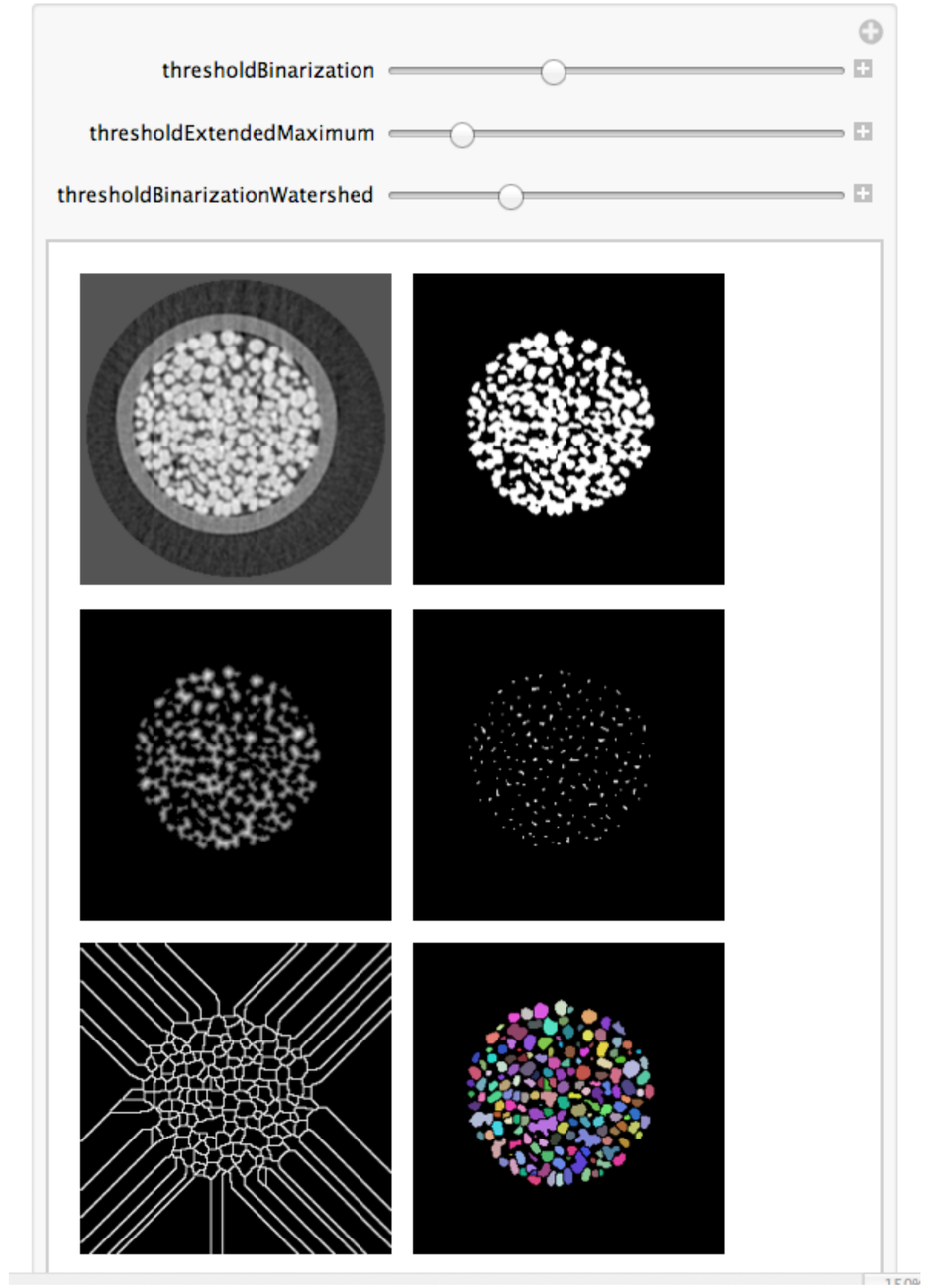




Philosophy

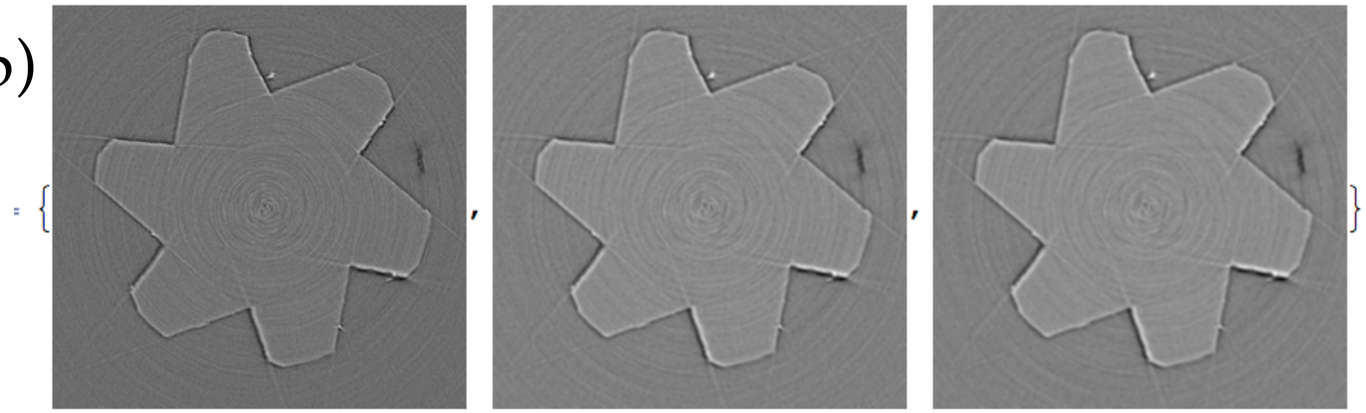
Our *a priori* knowledge about a sample can be used in segmentation.

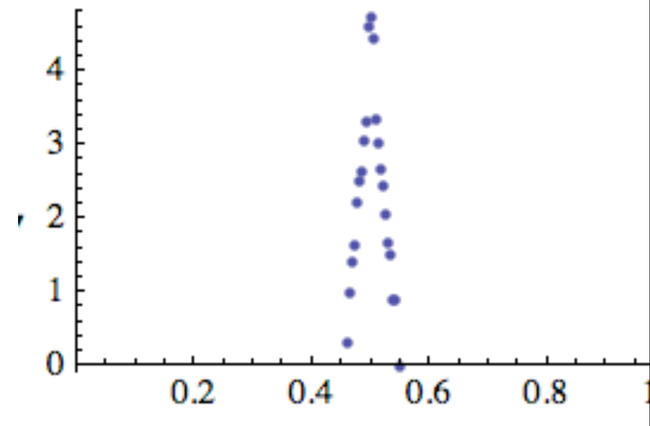
The MAS rotor has a Kel-F (plastic cap) and a zirconia (ZrO_2) cylinder. What does this tell us about the distribution of intensity values?



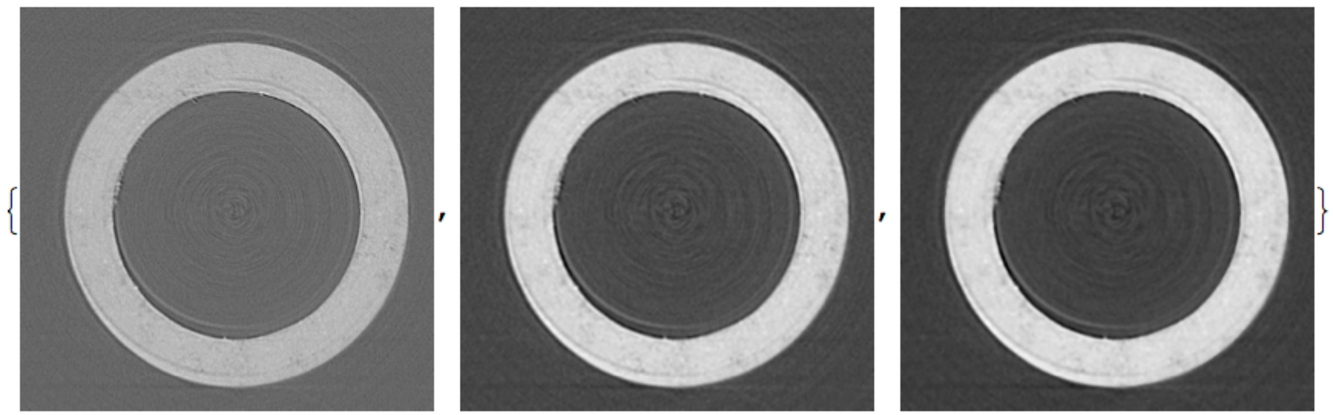
raw median filter + TV filter

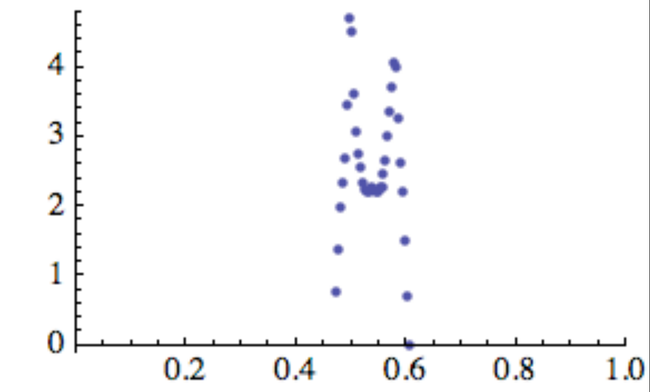
Kel-F (plastic cap)



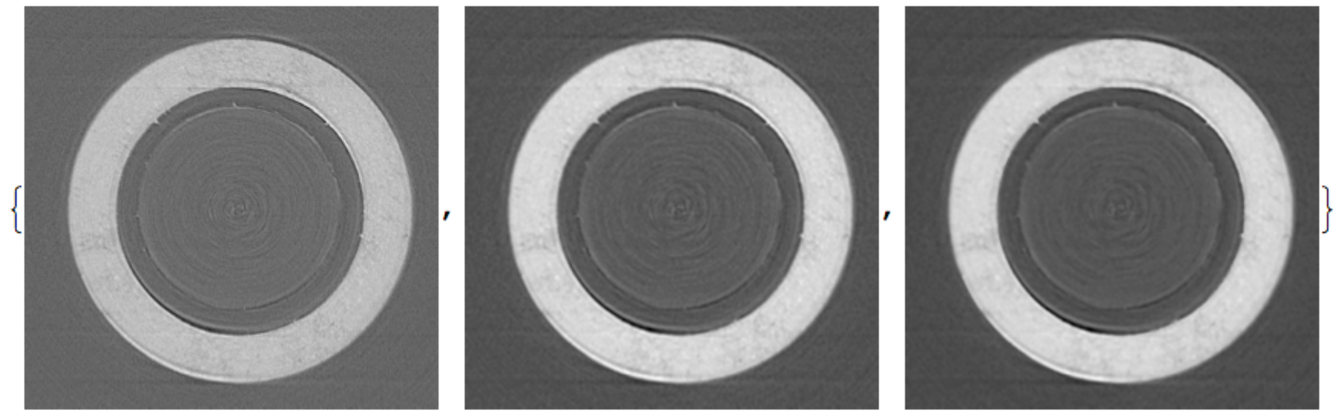


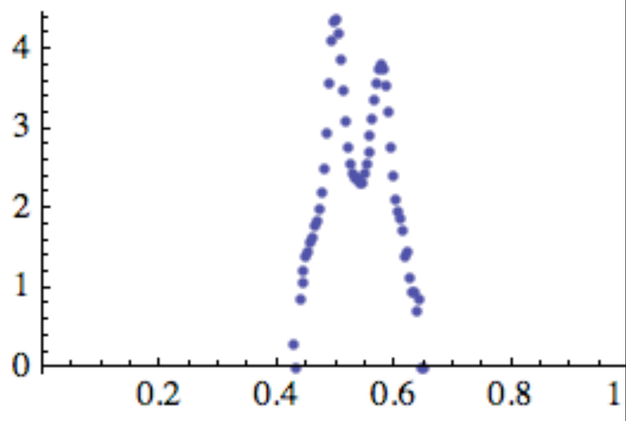
zirconia (ZrO_2) cylinder





Both Kel-F cap and zirconia cylinder





Kel-F is hard to segment. Zirconia is easy.

■ Step 4. Explore parameters for MorphologicalBinarization

```
In[3]:= Manipulate[Module[{},
    imageToBeTested = {imageKelFVTV, imageZrO2VTV, imageBoth}[[indexImage]];
    imageHighBinary = MorphologicalBinarize[imageToBeTested, {0.0, 0.02} + thresholdHigh];
    imageHigh = ImageMultiply[imageToBeTested, imageHighBinary];
    textString = {"threshold =", thresholdHigh};
    GraphicsRow[{ImageAdjust[imageToBeTested], imageHighBinary, imageHigh},
     Epilog → Inset[Text[Style[textString, 14, Background → White]], Scaled[{0.3, 0.9}]], ImageSize → 600] ]
   , {{indexImage, 1}, 1, 3, 1}
   , {{thresholdHigh, 0.5025}, 0.3, 0.6}, SynchronousUpdating → False]
```

indexImage

thresholdHigh

Out[8]=

Kel-F is hard to segment. Zirconia is easy.

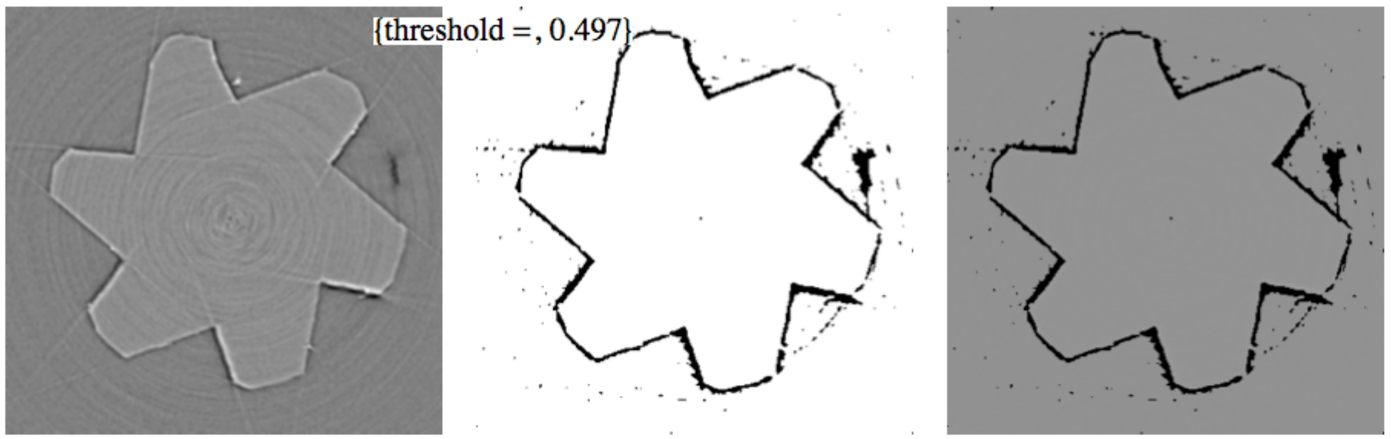
■ Step 4. Explore parameters for MorphologicalBinarization

```
]:= Manipulate[Module[{},
     imageToBeTested = {imageKelFVTV, imageZrO2VTV, imageBoth}[[indexImage]];
     imageHighBinary = MorphologicalBinarize[imageToBeTested, {0.0, 0.02} + thresholdHigh];
     imageHigh = ImageMultiply[imageToBeTested, imageHighBinary];
     textString = {"threshold =", thresholdHigh};
     GraphicsRow[{ImageAdjust[imageToBeTested], imageHighBinary, imageHigh},
      Epilog → Inset[Text[Style[textString, 14, Background → White]], Scaled[{0.3, 0.9}]], ImageSize → 600] ]
    , {{indexImage, 1}, 1, 3, 1}
    , {{thresholdHigh, 0.5025}, 0.3, 0.6}, SynchronousUpdating → False]
```

indexImage

thresholdHigh

8]=



1) Add ImageMultiply of original image X binary
2) Make a histogram of the result of ImageMultiply
3) Yields some information, but the range is compressed. Used ImageAdjust to expand range

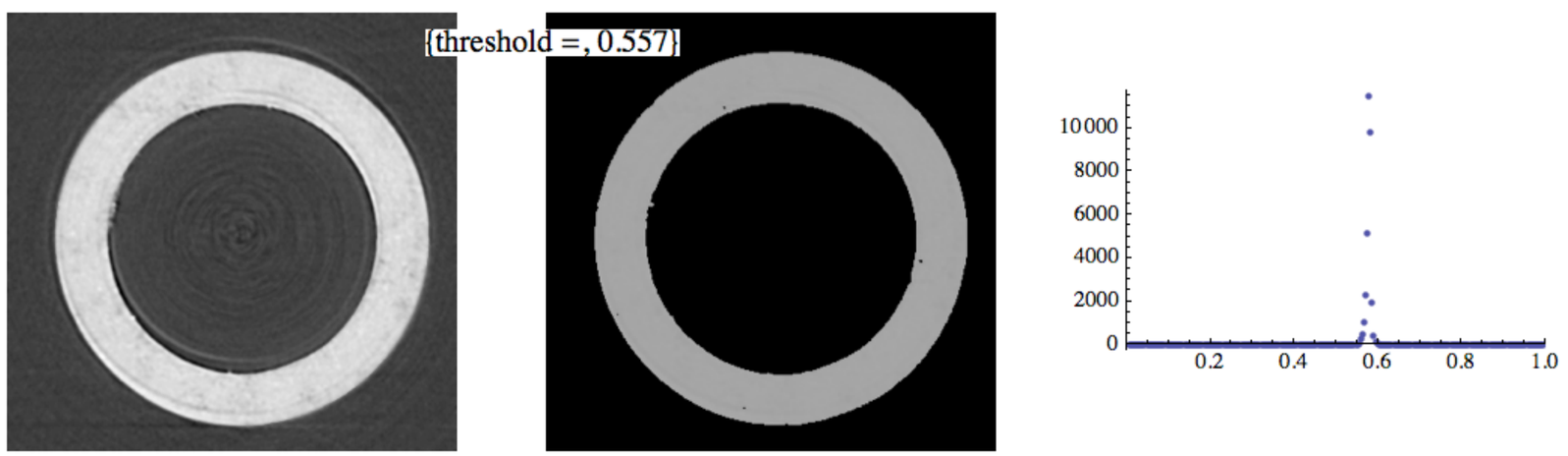
Step 5. Explore parameters for MorphologicalBinarization - add a histogram

```
]:= Manipulate[Module[{},
     imageToBeTested = {imageKelFVTV, imageZrO2VTV, imageBoth}[[indexImage]];
     imageHighBinary = MorphologicalBinarize[imageToBeTested, {0.0, 0.02} + thresholdHigh];
     imageHigh = ImageMultiply[imageToBeTested, imageHighBinary];
     temp = Rest[ImageLevels[imageHigh]];
     g1 = ListPlot[temp, PlotRange → {{0, 1}, All}];
     textString = {"threshold =", thresholdHigh};
     GraphicsRow[{ImageAdjust[imageToBeTested], imageHigh, Show[{g1}]},
      Epilog → Inset[Text[Style[textString, 14, Background → White]], Scaled[{0.3, 0.9}]], ImageSize -> 700] ]
    , {{indexImage, 1}, 1, 3, 1}
    , {{thresholdHigh, 0.5025}, 0.3, 0.9}, SynchronousUpdating → False]
```

indexImage

thresholdHigh

)]=



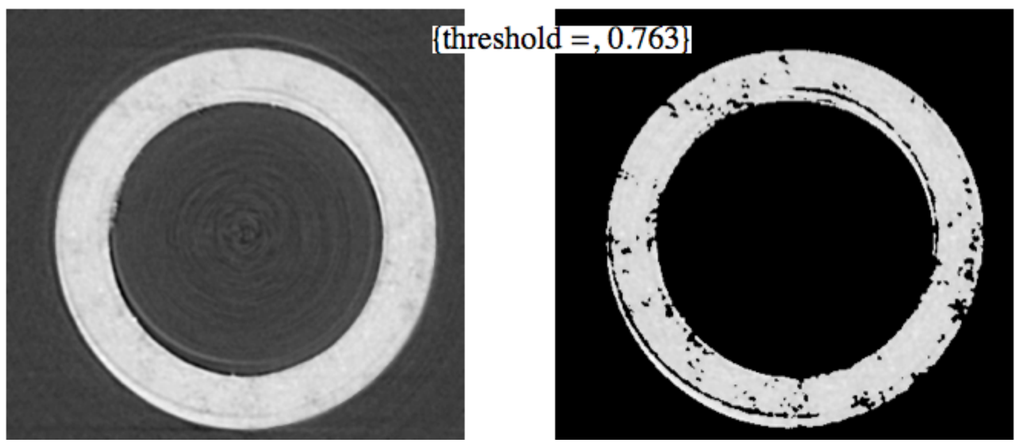
Zirconia cylinder
Binarization, ImageMultiply with original image, and Histogram of results.

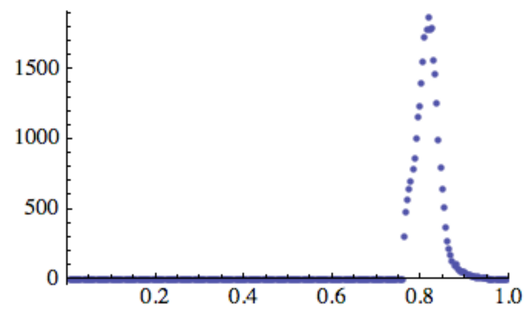
Step 6. Explore parameters for MorphologicalBinarization - add a histogram
(added an ImageAdjust for better range of histrogram values)
Try indexImage=2, thresholdHigh= 0.70 to 0.80

```
Manipulate[Module[{},
  imageToBeTested = ImageAdjust[{imageKelFVTV, imageZrO2VTV, imageBoth}[[indexImage]]];
  imageHighBinary = MorphologicalBinarize[imageToBeTested, {0.0, 0.02} + thresholdHigh];
  imageHigh = ImageMultiply[imageToBeTested, imageHighBinary];
  temp = Rest[ImageLevels[imageHigh]];
  g1 = ListPlot[temp, PlotRange → {{0, 1}, All}];
  textString = {"threshold =", thresholdHigh};
  GraphicsRow[{imageToBeTested, imageHigh, Show[{g1}]},
   Epilog → Inset[Text[Style[textString, 14, Background → White]], Scaled[{0.3, 0.9}]], ImageSize -> 700] ]
 , {{indexImage, 1}, 1, 3, 1}
 , {{thresholdHigh, 0.5025}, 0.2, 0.98}, SynchronousUpdating → False]
```

indexImage

thresholdHigh





Start thinking about prior knowledge.
What do we know about this peak?

Kel-F cap

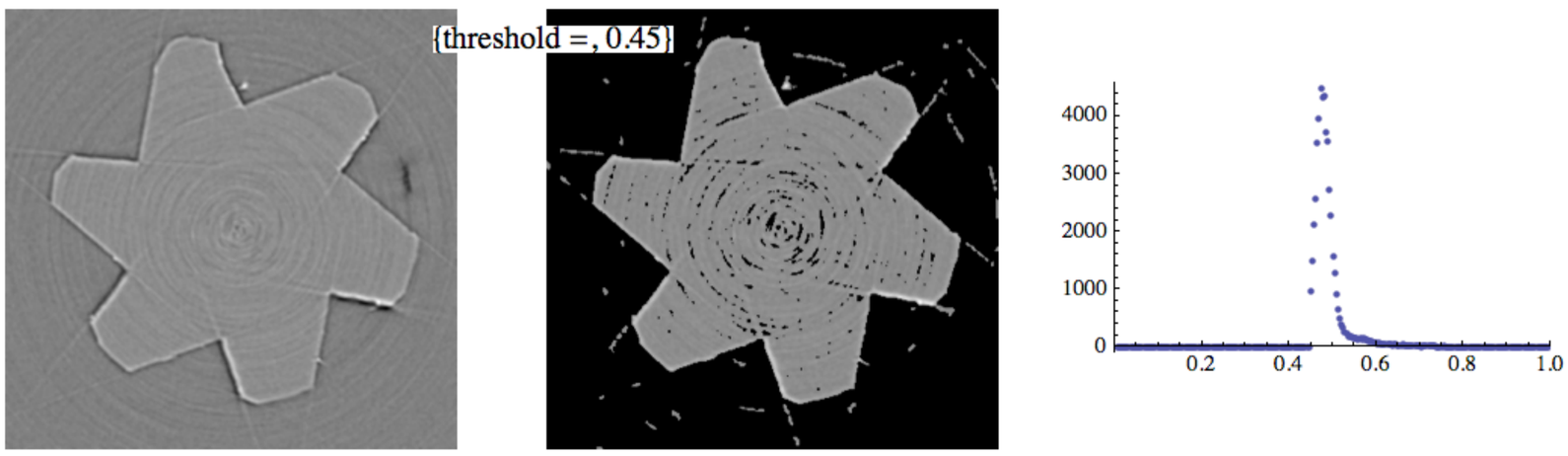
Binarization, ImageMultiply with original image, and Histogram of results.

Step 6. Explore parameters for MorphologicalBinarization - add a histogram (added an ImageAdjust for better range of histrogram values) Try indexImage=2, thresholdHigh= 0.70 to 0.80

```
Manipulate[Module[{},
  imageToBeTested = ImageAdjust[{imageKelFVTV, imageZrO2VTV, imageBoth}[[indexImage]]];
  imageHighBinary = MorphologicalBinarize[imageToBeTested, {0.0, 0.02} + thresholdHigh];
  imageHigh = ImageMultiply[imageToBeTested, imageHighBinary];
  temp = Rest[ImageLevels[imageHigh]];
  g1 = ListPlot[temp, PlotRange → {{0, 1}, All}];
  textString = {"threshold =", thresholdHigh};
  GraphicsRow[{imageToBeTested, imageHigh, Show[{g1}]},
   Epilog → Inset[Text[Style[textString, 14, Background → White]], Scaled[{0.3, 0.9}]], ImageSize -> 700] ]
 , {{indexImage, 1}, 1, 3, 1}
 , {{thresholdHigh, 0.5025}, 0.2, 0.98}, SynchronousUpdating → False]
```

indexImage

thresholdHigh

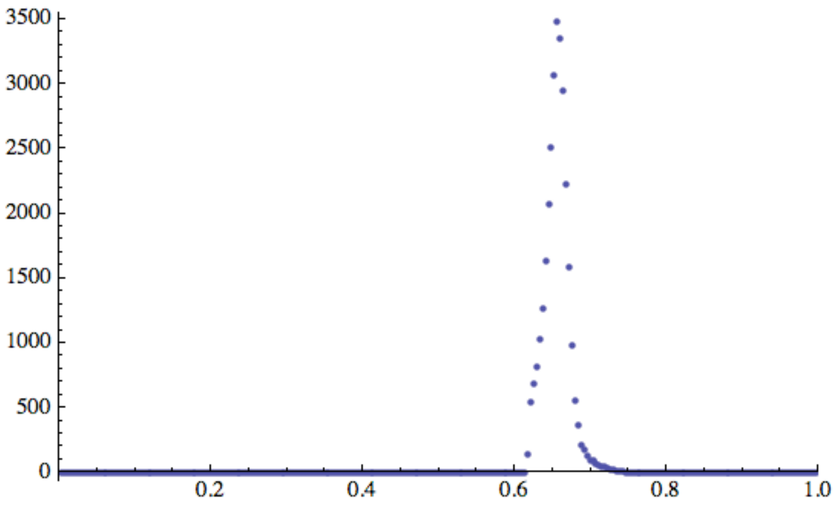


Start thinking about prior knowledge.
What do we know about this peak?

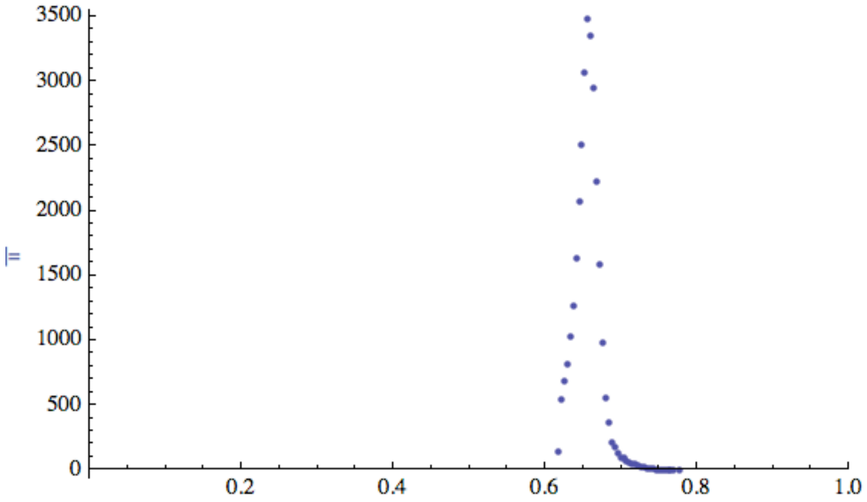
■ Step 7. Discuss peak fitting

```
)]:= indexImage = 2; thresholdHigh = 0.62;
     imageToBeTested = {imageKelFVTV, imageZrO2VTV, imageBoth}[[indexImage]];
     imageHighBinary = MorphologicalBinarize[imageToBeTested, {0.0, 0.02} + thresholdHigh];
     imageHigh = ImageMultiply[imageToBeTested, imageHighBinary];

i]:= temp = Rest[ImageLevels[imageHigh]];
     g1 = ListPlot[temp, PlotRange → {{0, 1}, All}]
```




```
5]= index = Flatten[Position[temp[[All, 2]], p_? (# > 0 &)]];
     tempFit = temp[[index, All]];
     g1b = ListPlot[tempFit, PlotRange → {{0, 1}, All}]
```




```
: tempFit
: {{0.785156, 247}, {0.789063, 990}, {0.792969, 1144}, {0.796875, 1222}, {0.800781, 1393}, {0.804688, 1550},
   {0.808594, 1729}, {0.8125, 1782}, {0.816406, 1870}, {0.820313, 1784}, {0.824219, 1791}, {0.828125, 1558},
   {0.832031, 1463}, {0.835938, 1256}, {0.839844, 996}, {0.84375, 797}, {0.847656, 649}, {0.851563, 516},
   {0.855469, 366}, {0.859375, 269}, {0.863281, 222}, {0.867188, 174}, {0.871094, 132}, {0.875, 98}, {0.878906, 102}
   {0.882813, 71}, {0.886719, 63}, {0.890625, 58}, {0.894531, 50}, {0.898438, 48}, {0.902344, 43}, {0.90625, 27},
   {0.910156, 34}, {0.914063, 24}, {0.917969, 23}, {0.921875, 23}, {0.925781, 14}, {0.929688, 21}, {0.933594, 10},
   {0.9375, 14}, {0.941406, 7}, {0.945313, 6}, {0.949219, 4}, {0.953125, 3}, {0.957031, 3}, {0.960938, 2},
   {0.964844, 2}, {0.972656, 1}, {0.976563, 1}, {0.980469, 1}, {0.984375, 1}, {0.988281, 1}, {0.996094, 1}}
```

```
answer = FindFit[tempFit, a Exp[-(x - b)^2/c], {a, b, c}, x]

{a → 3249.41, b → 0.655907, c → 0.000390006}

calculatedArea = a /. answer
estPeakPosition = b /. answer
estPeakSigma = c /. answer
```

My common mistakes with FindFit:

1) Coefficients and variable are black, not blue. Use Clear[a,b,c,x} to get blue variables.

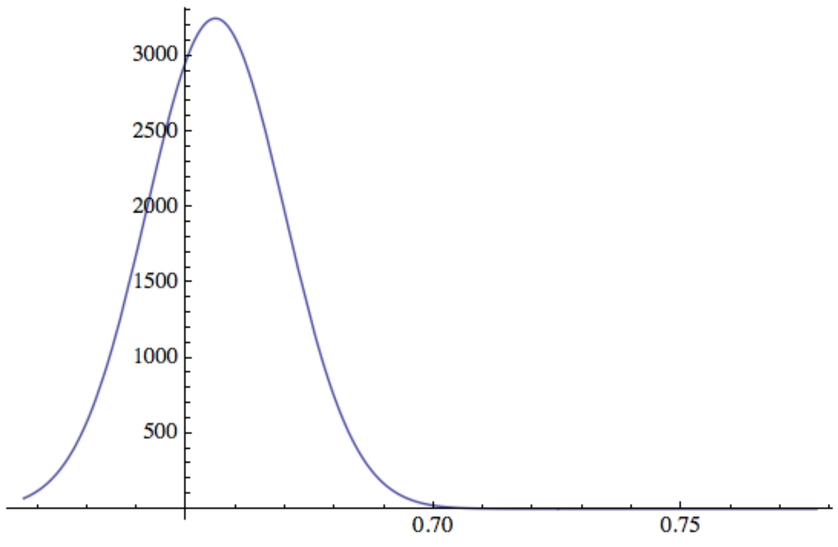
2) The data is not in the form of {{x1,y2}, {x2,y2},....} list. Use Transpose, Table, Map as necessary.

3) The answer is a list of “rules”. Use “/.” to extract a rule.

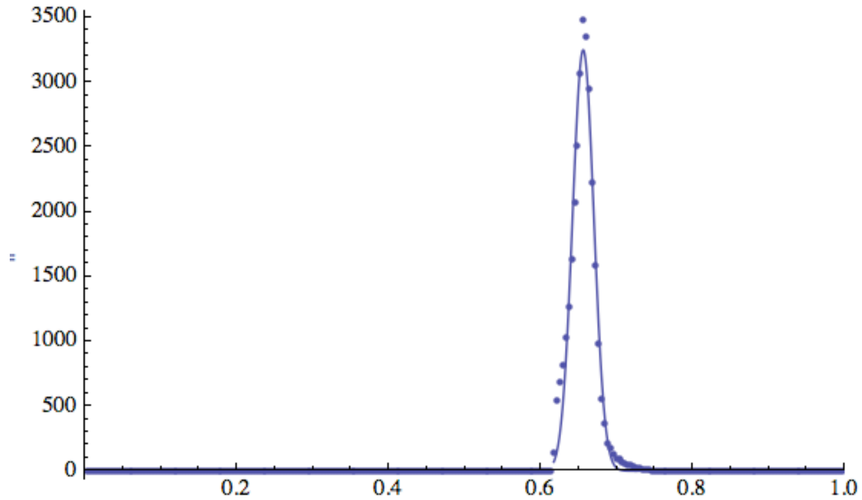
Mathematica calls “/.” the “ReplaceAll” command.

```
:= g2 = Plot[calculatedArea Exp[-(x - estPeakPosition)^2 / estPeakSigma], {x, Min[tempFit[[All, 1]]], Max[tempFit[[All, 1]]]}]
```

]=




```
= Show[{g1, g2}]
```



Residual: difference between y and y(calculated)
Variance: sum of the square of the residuals/n (here, n = number of points)

indexImage

thresholdHigh

{threshold =, 0.788, area=, 1845.59, variance=, 4610.99,
peak position=, 0.816792, peak sigma=, 0.000925183}