ScaDS.AI
DRESDEN LEIPZIG
CENTER FOR SCALABLE DATA ANALYTICS
AND ARTIFICIAL INTELLIGENCE

Image segmentation

Robert Haase

Using materials from Ryan Savill George (PoL, TU Dresden)

GEFÖRDERT VOM

Bundesministerium
für Bildung
und Forschung

Diese Maßnahme wird gefördert durch die Bundesregierung aufgrund eines Beschlusses des Deutschen Bundestages.
Diese Maßnahme wird mitfinanziert durch Steuermittel auf der Grundlage des von den Abgeordneten des Sächsischen Landtags beschlossenen Haushaltes.

Quiz (recap)

- Which of the following is a band-pass filter?

Gaussian

Median

Top-hat

Difference of Gaussian

Quiz (recap)

- Which of the following is a denoising filter?

Gaussian

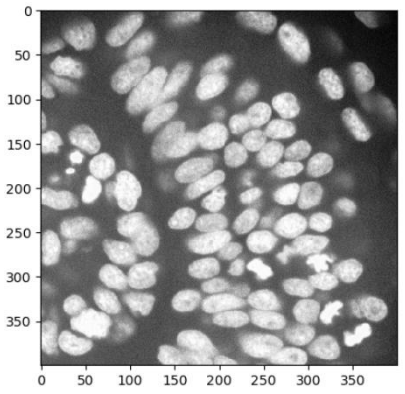
Median

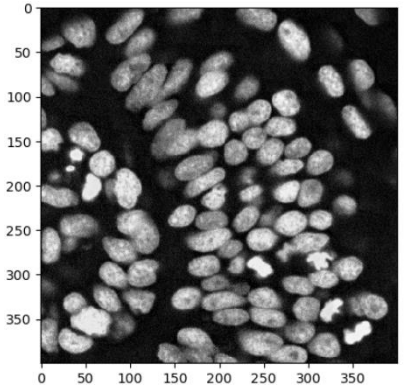
Top-hat

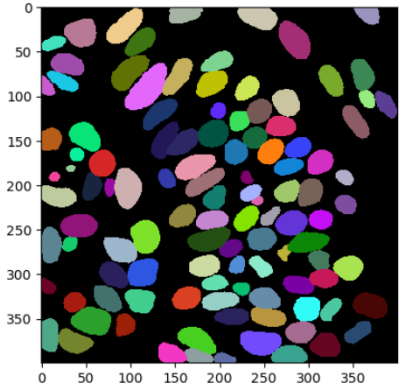
Difference of Gaussian

Lecture overview: Bio-image Analysis

- Image Data Analysis workflows
- Goal: **Quantify observations, substantiate conclusions with numbers**

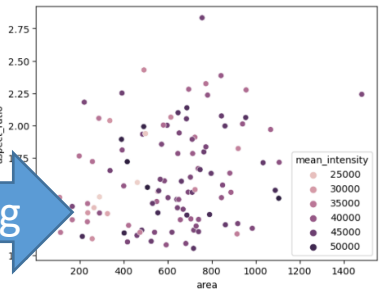






	area	mean_intensity	major_axis_length	minor_axis_length
0	594.0	40572.809764	28.611591	26.537947
1	645.0	43764.872868	33.511511	24.566916
2	1105.0	51970.561991	45.232031	31.456308
3	718.0	47015.487465	31.023274	29.5
4	791.0	49132.515803	36.382253	27.7
...	...	...	...	
105	238.0	30477.126050	20.252197	15.2
106	615.0	32886.154472	41.030760	19.8
107	110.0	33042.445455	14.366347	9.5
108	222.0	43304.180180	25.370549	11.6
109	167.0	43378.808383		

Plotting



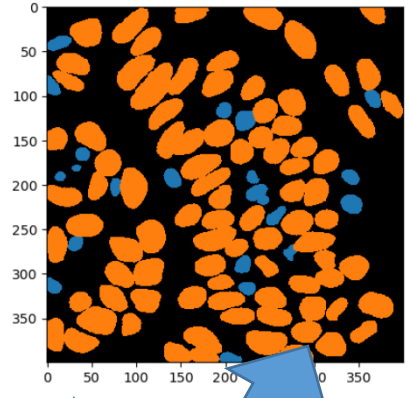


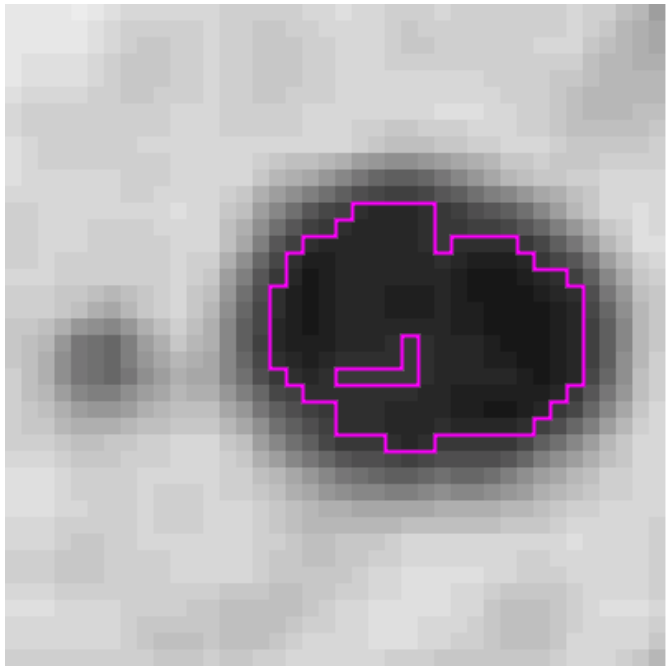
Image filtering → Image segmentation → Feature extraction → Object classification → Visualization

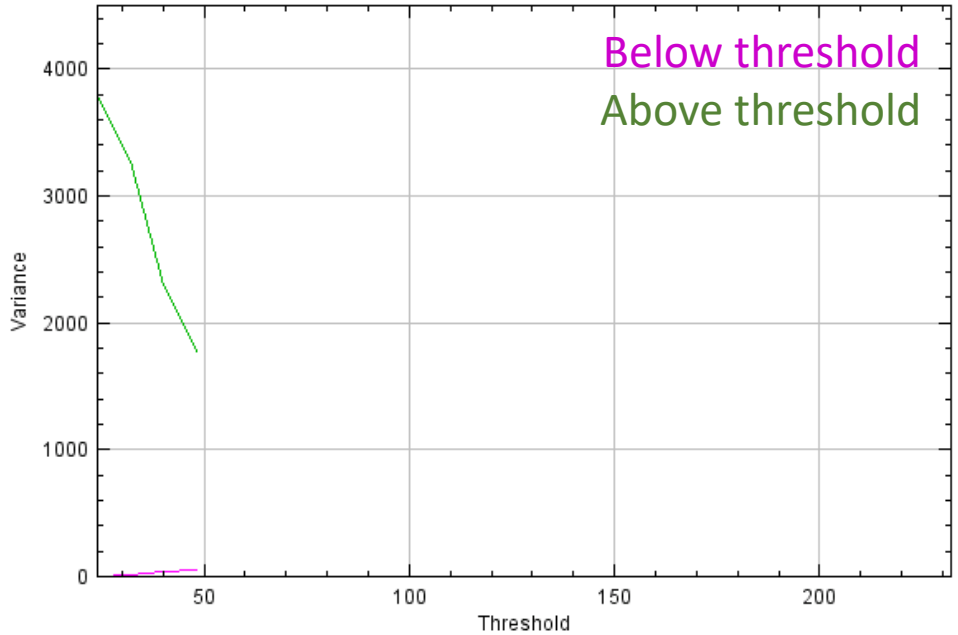
Thresholding: Otsu's method

- Searching for a threshold where the variance in both classes (above/below threshold) becomes minimal.

$$Var(I) = \sum_{i \in I} g_i - \bar{g}_I \qquad \bar{g}_I = \sum_{i \in I} \frac{g_i}{n_I}$$

$Var(I)$ … Variance in image I
g_i … grey value of a pixel i
$\bar{g}_I$ … mean grey value of the whole image I
n_I … number of pixels in Image I



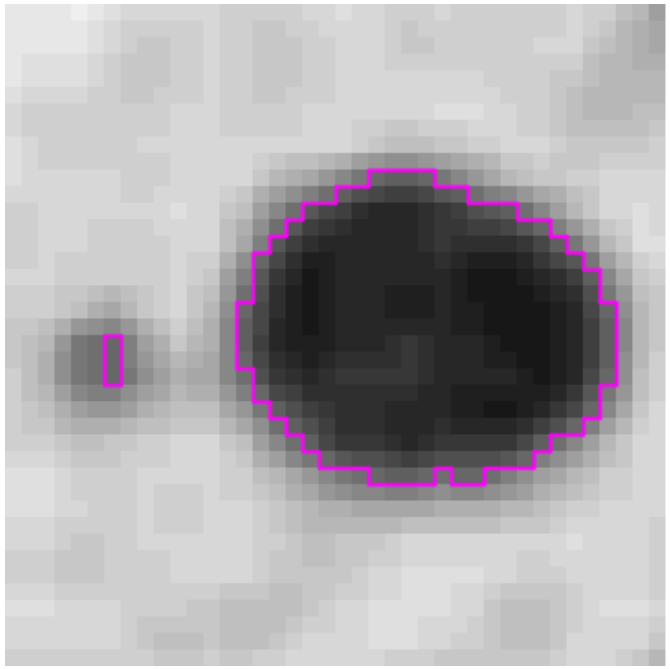


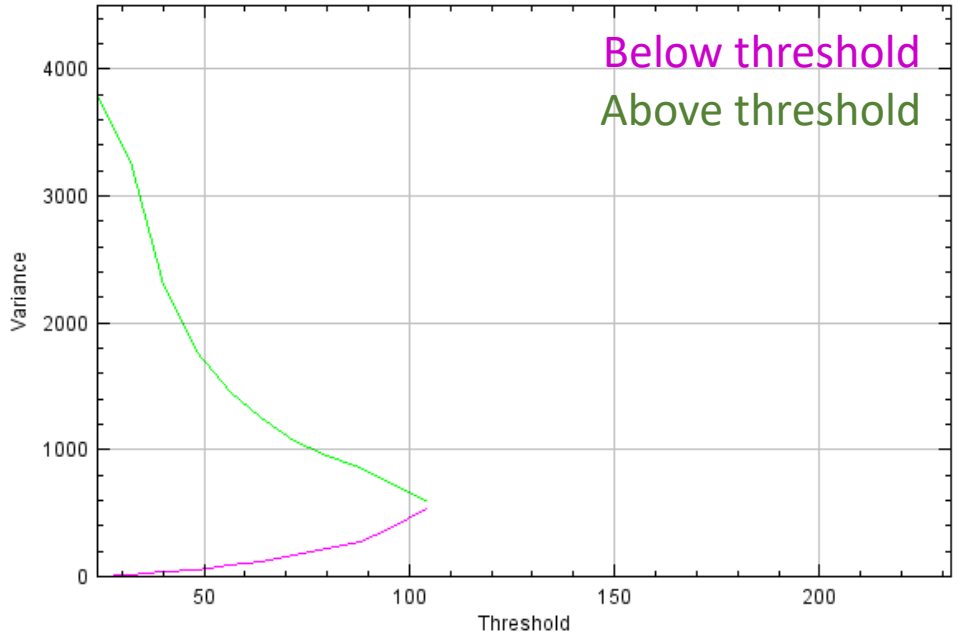
Thresholding: Otsu's method

- Searching for a threshold where the variance in both classes (above/below threshold) becomes minimal.

$$Var(I) = \sum_{i \in I} g_i - \bar{g}_I \qquad \bar{g}_I = \sum_{i \in I} \frac{g_i}{n_I}$$

$Var(I)$ … Variance in image I
g_i … grey value of a pixel i
$\bar{g}_I$ … mean grey value of the whole image I
n_I … number of pixels in Image I



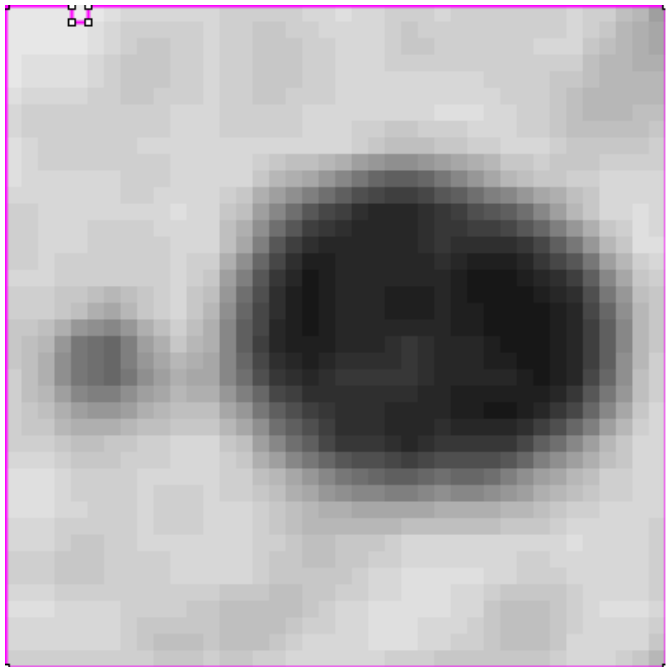


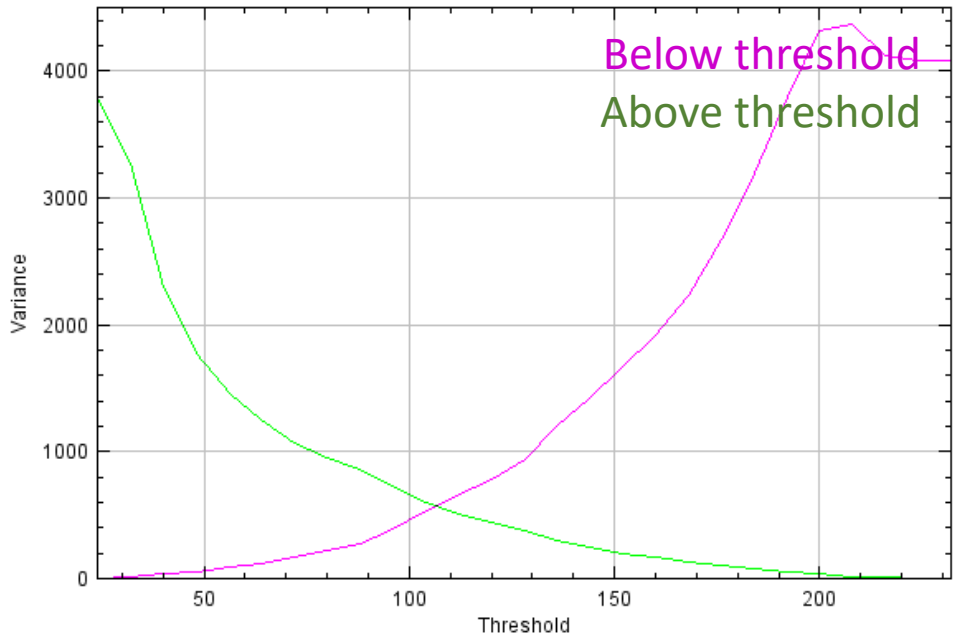
Thresholding: Otsu's method

- Searching for a threshold where the variance in both classes (above/below threshold) becomes minimal.

$$Var(I) = \sum_{i \in I} g_i - \bar{g}_I \qquad \bar{g}_I = \sum_{i \in I} \frac{g_i}{n_I}$$

$Var(I)$ … Variance in image I
g_i … grey value of a pixel i
$\bar{g}_I$ … mean grey value of the whole image I
n_I … number of pixels in Image I

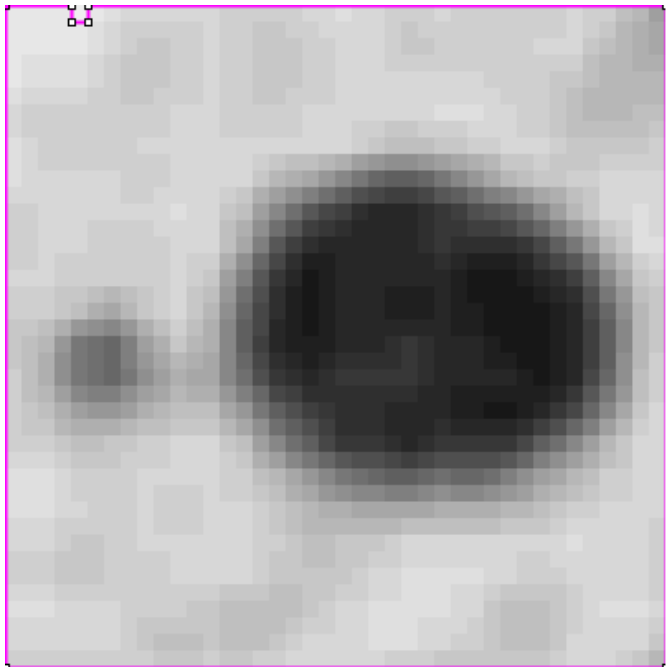


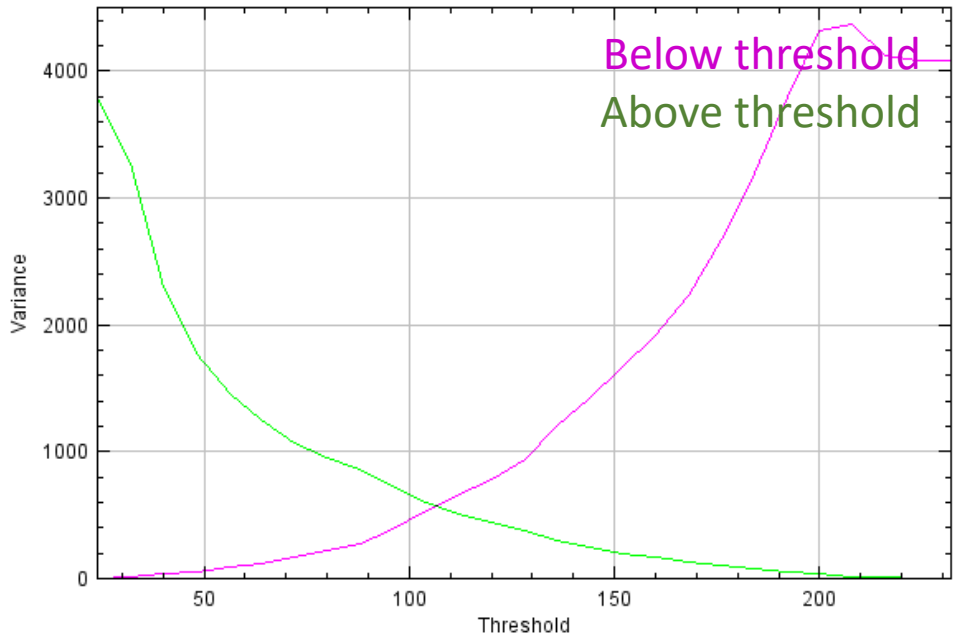


Thresholding: Otsu's method

- Searching for a threshold where the variance in both classes (above/below threshold) becomes minimal.
- Weighted (!) sum variance

$$Var'(I) = \frac{n_A}{n_I} Var(A) + \frac{n_B}{n_I} Var(B) \qquad I = A \cup B$$

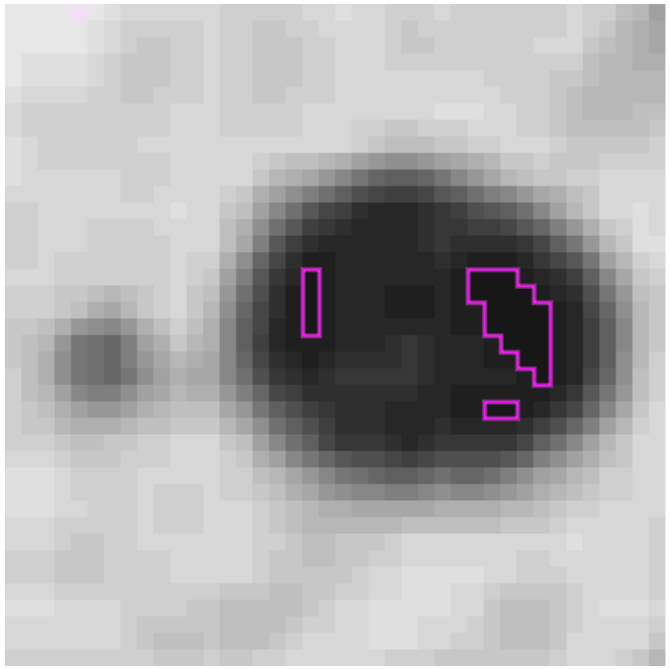


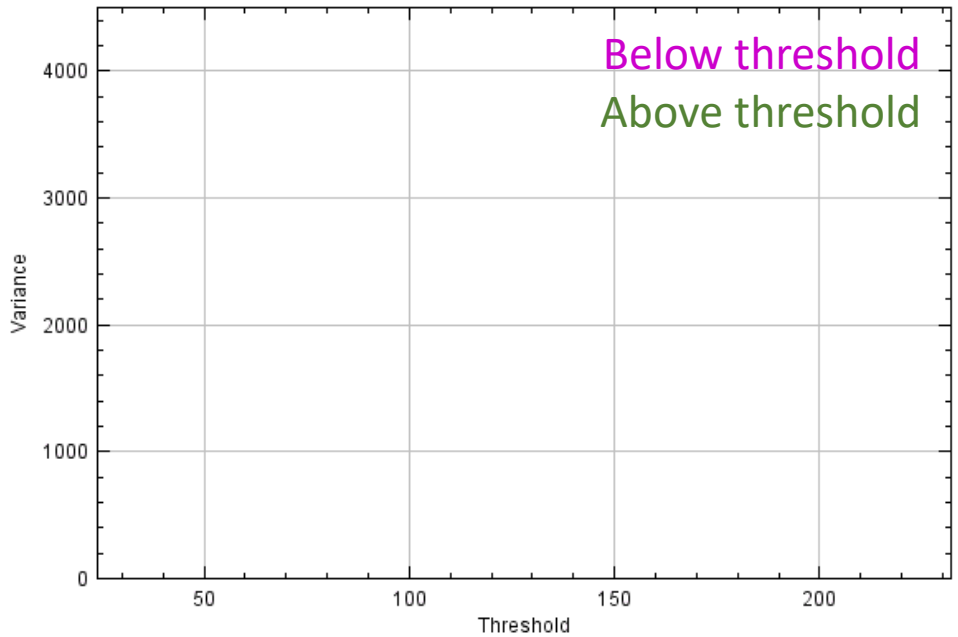


Thresholding: Otsu's method

- Searching for a threshold where the variance in both classes (above/below threshold) becomes minimal.
- Weighted (!) sum variance

$$Var'(I) = \frac{n_A}{n_I} Var(A) + \frac{n_B}{n_I} Var(B) \qquad I = A \cup B$$

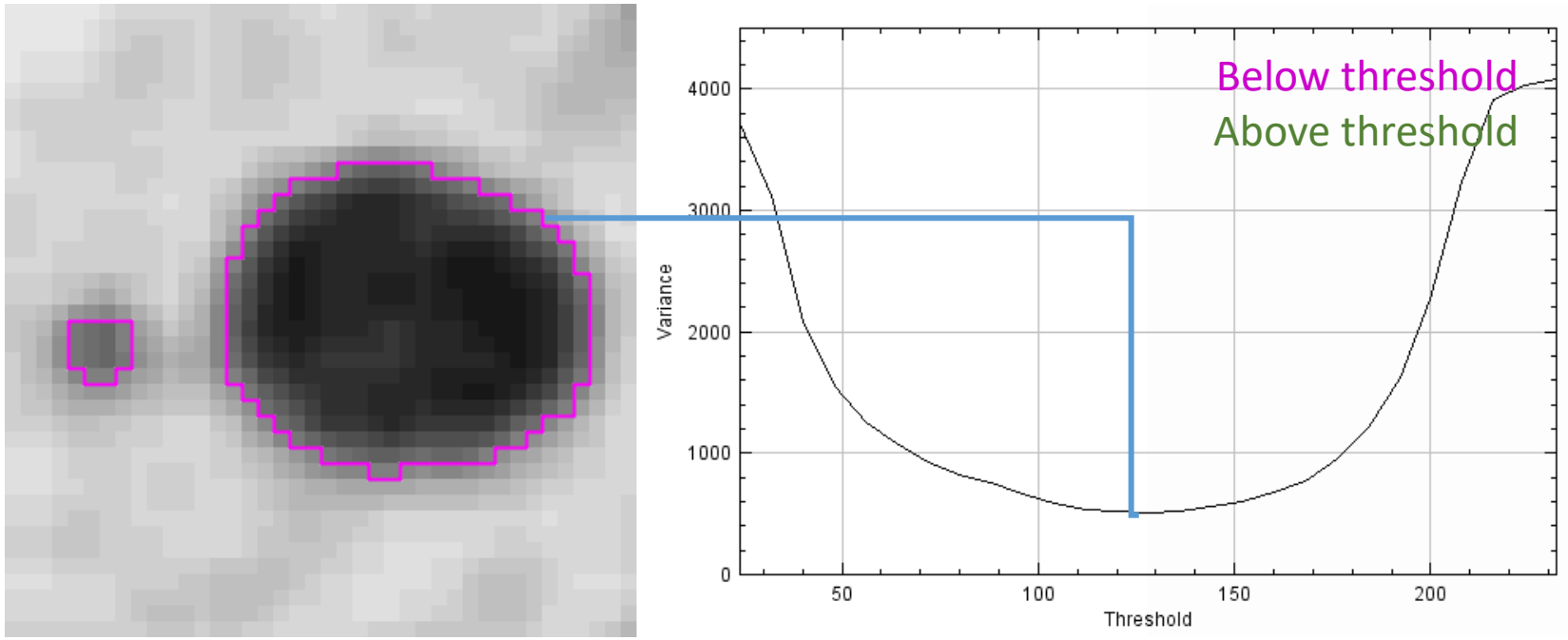




Thresholding: Otsu's method

- Searching for a threshold where the variance in both classes (above/below threshold) becomes minimal.
- Weighted (!) sum variance

$$Var'(I) = \frac{n_A}{n_I} Var(A) + \frac{n_B}{n_I} Var(B) \qquad I = A \cup B$$



Thresholding: Citing

- Cite the thresholding method of your choice properly

"We segmented the cell nuclei in the images using Otsu's thresholding method (Otsu et Al. 1979) implemented in scikit-image (van der Walt et al. 2014)."

IEEE TRANSACTIONS ON SYSTEMS, MAN, AND CYBERNETICS, VOL. SMC-9, NO. 1, JANUARY 1979

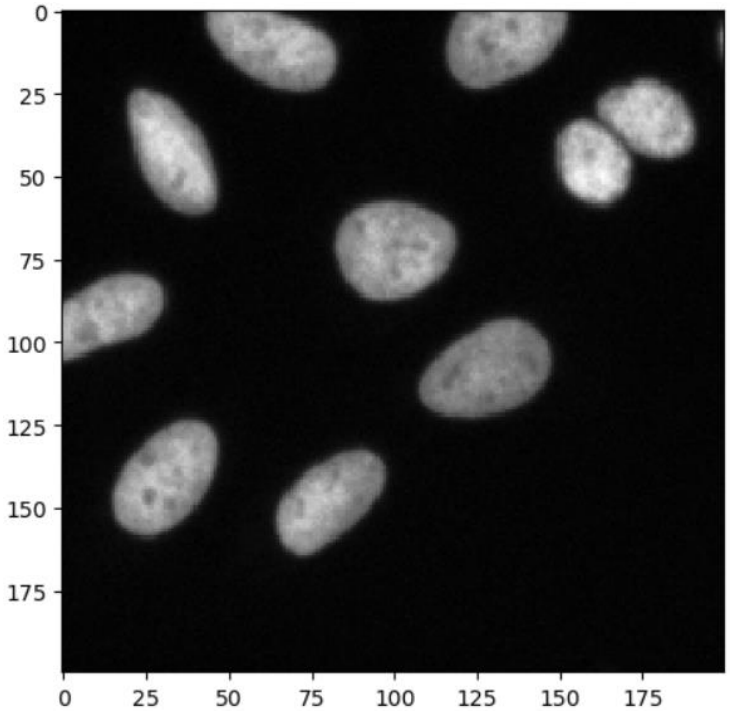
A Threshold Selection Method from Gray-Level Histograms

NOBUYUKI OTSU

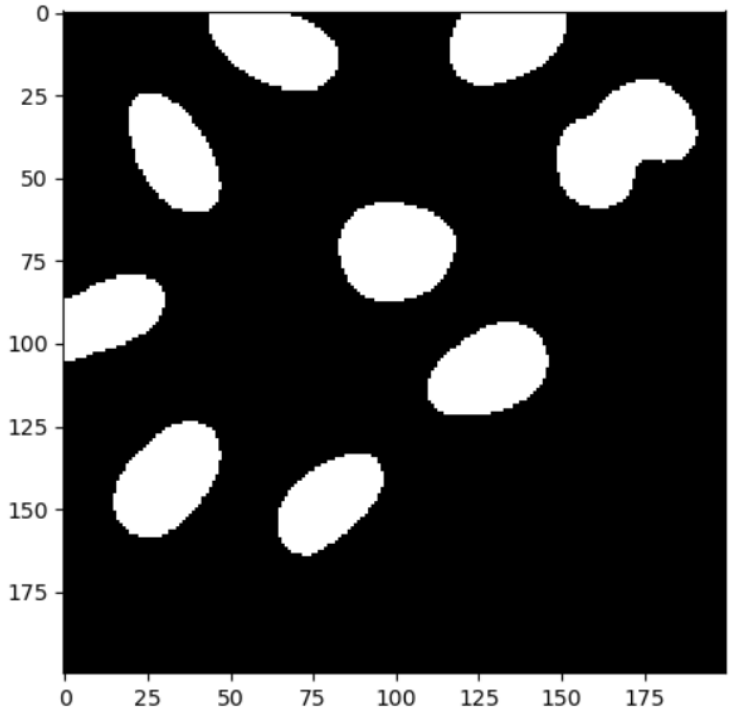
Abstract—**A nonparametric and unsupervised method of automatic threshold selection for picture segmentation is presented. An optimal threshold is selected by the discriminant criterion, namely, so as to maximize the separability of the resultant classes in gray levels. The procedure is very simple, utilizing only the zeroth- and the first-order cumulative moments of the gray-level histogram. It is straightforward to extend the method to multithreshold problems. Several experimental results are also presented to support the validity of the method.**

Terminology

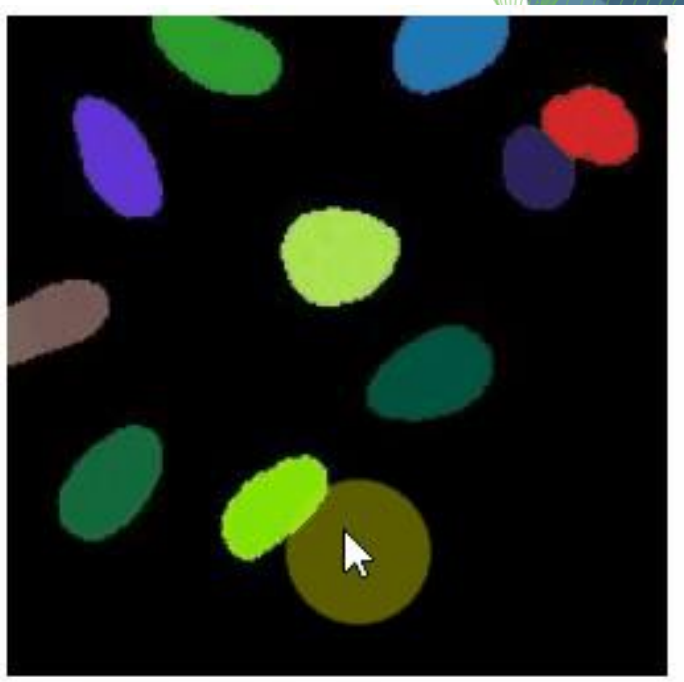
Intensity image



Binary image

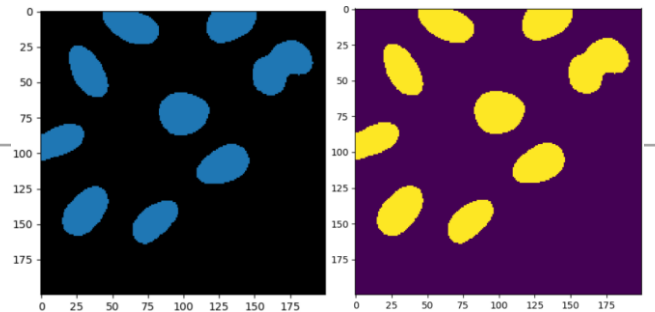


Label image



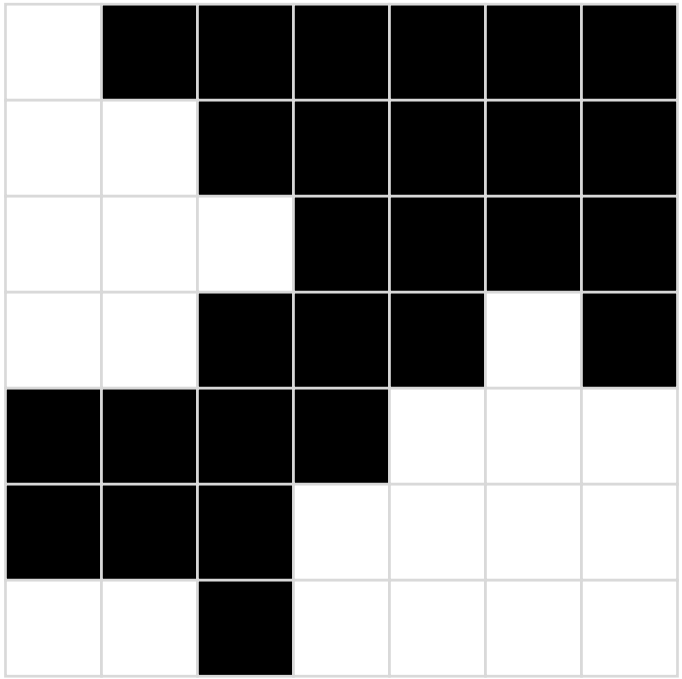
[y=152, x=92] = 0

No matter how they are displayed



Connected component labelling

- In order to allow the computer differentiating objects, connected component analysis (CCA) is used to mark pixels belonging to different objects with different numbers
- Background pixels are marked with 0.
- The maximum intensity of a labelled map corresponds to the number of objects.



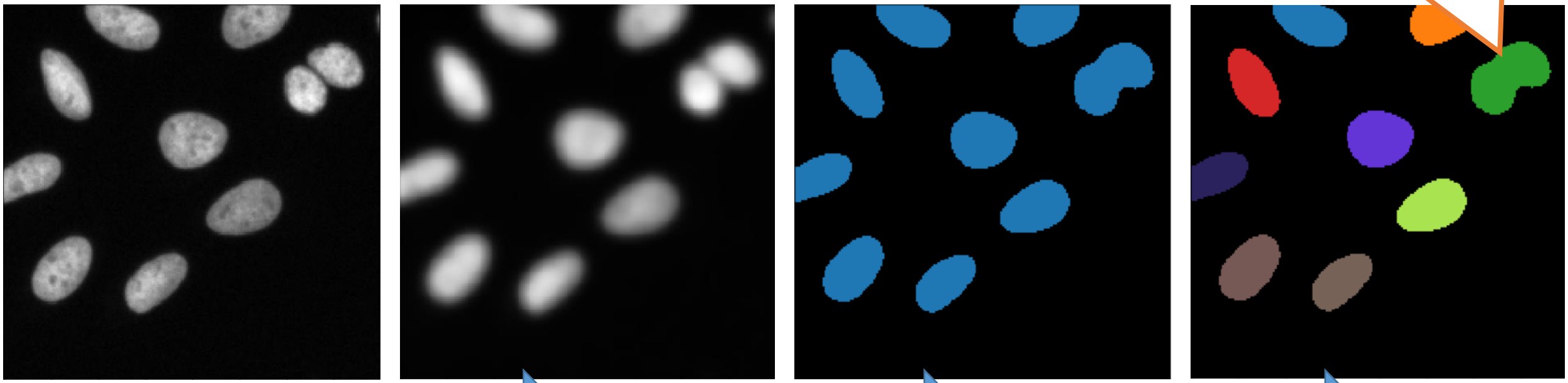
CCA

1	0	0	0	0	0	0
1	1	0	0	0	0	0
1	1	1	0	0	0	0
1	1	0	0	0	3	0
0	0	0	0	3	3	3
0	0	0	3	3	3	3
2	2	0	3	3	3	3

Common image segmentation workflows

- Presumably the most common segmentation algorithm used for fluorescent microscopy images:
  - Gaussian blur, Otsu's Threshold, Connected Component Labeling

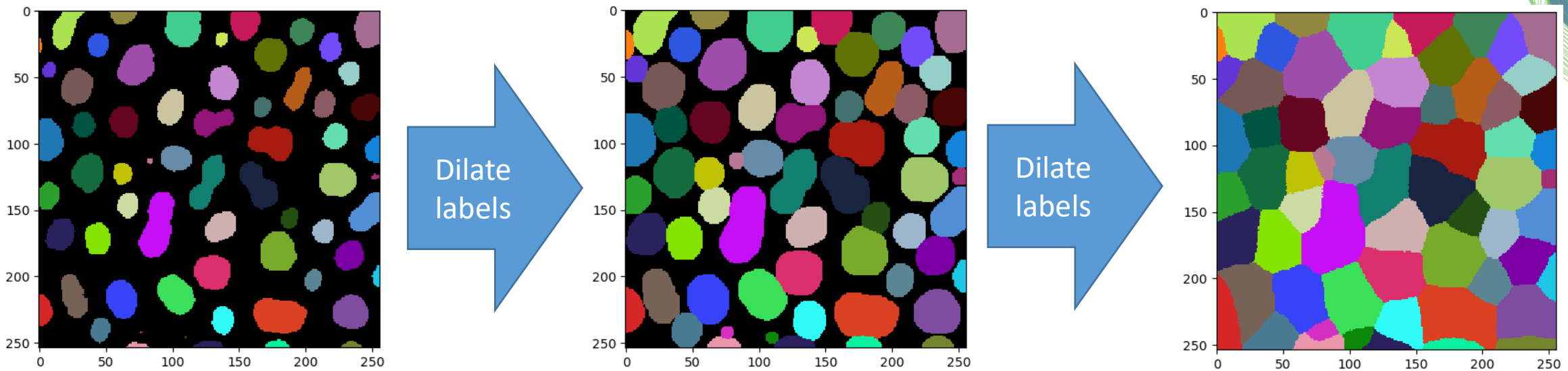
Limitation: Dense objects



Denoising → Binarization → Labeling

Voronoi-Tesselation

- For separating objects using spatial constraints (not intensity-based)

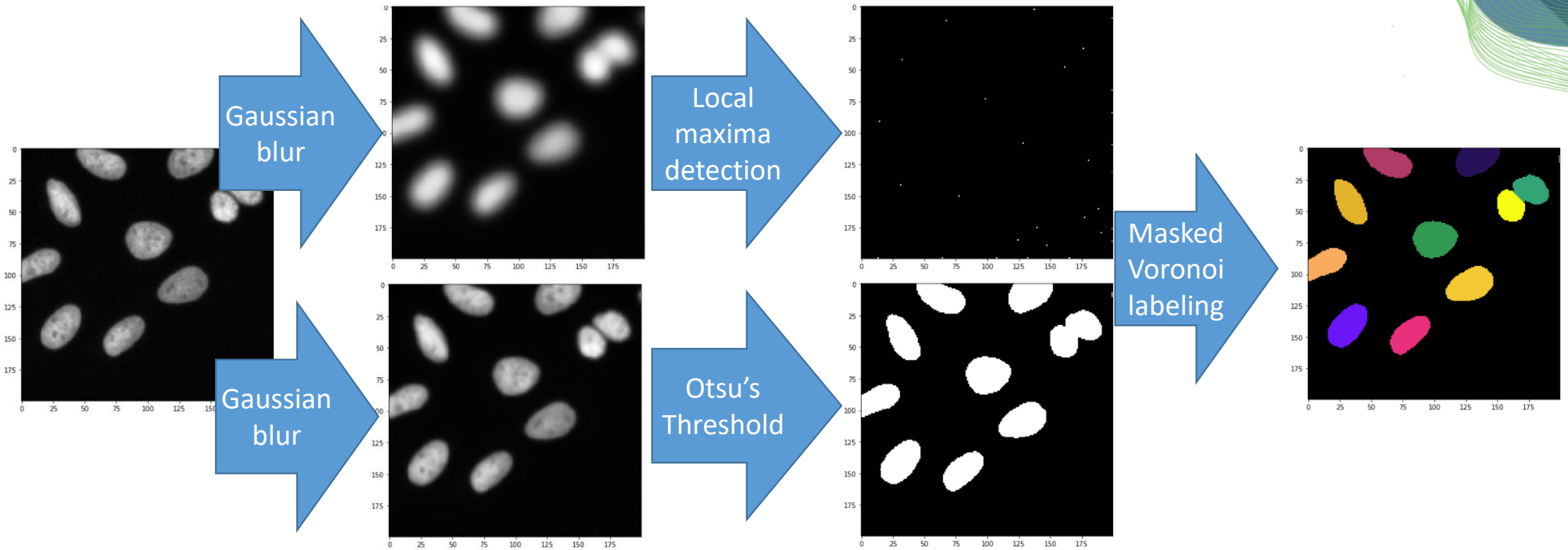


Label-image

Voronoi-label-image

Common image segmentation workflows

- Combination of Gaussian blur, Otsu's Threshold and Voronoi-labeling

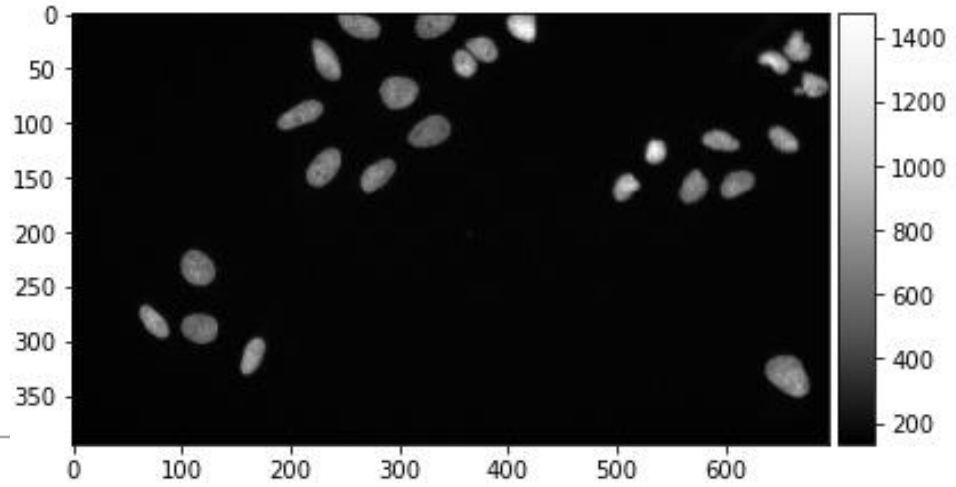


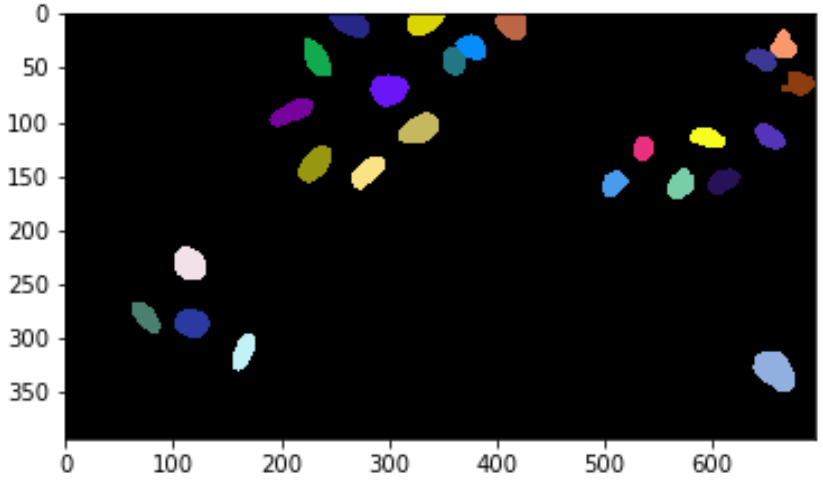
Voronoi-Otsu-Labeling

- Gaussian-Blur
- Otsu-Thresholding
- Spot-detection
- Watershed on the binary image

… in a single line of code:

```
segmented = nsbatwm.voronoi_otsu_labeling(input_image,
                                          spot_sigma=5,
                                          outline_sigma=1
                                         )
segmented
```





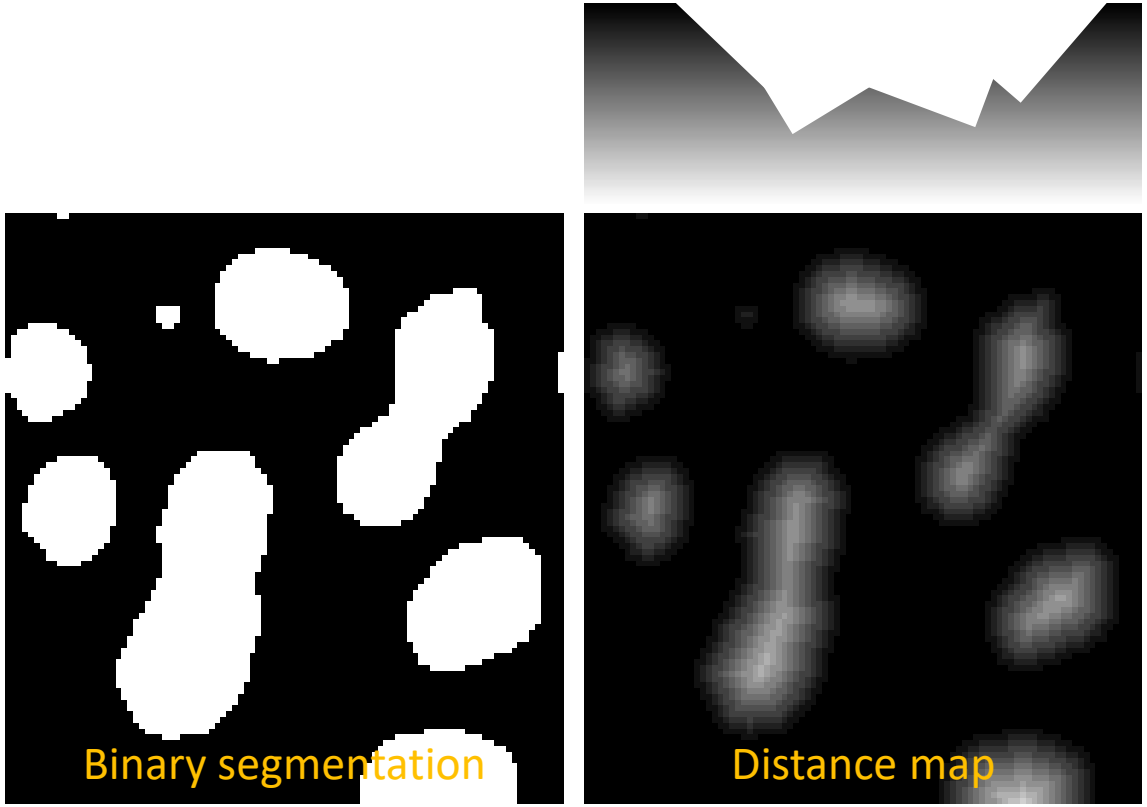
nsbatwm made image

shape	(395, 695)
dtype	int32
size	1.0 MB
min	0
max	25

https://haesleinhuepf.github.io/BioImageAnalysisNotebooks/20_image_segmentation/11_voronoi_otsu_labeling.html
https://www.youtube.com/watch?v=evgRgDfVuEc

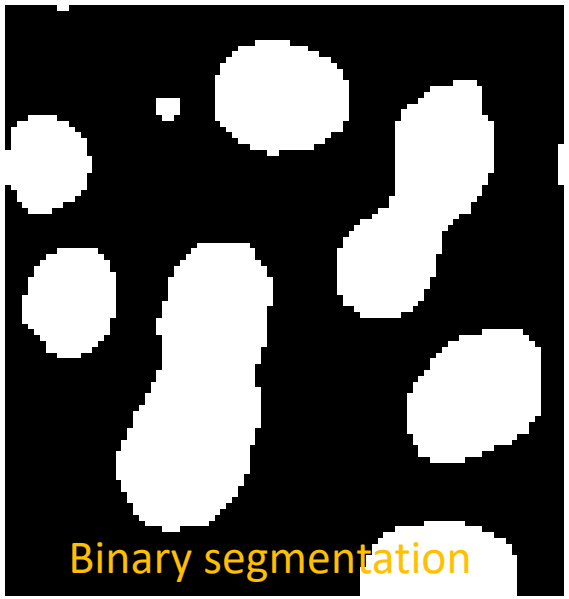
Watershed

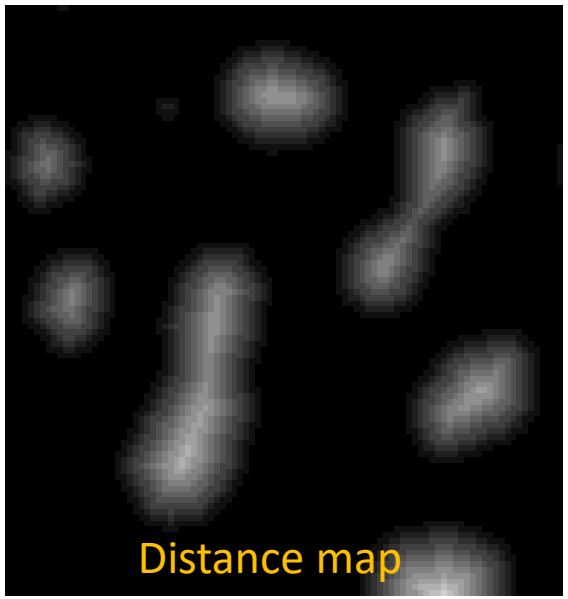
- The watershed algorithm for binary images allows cutting one object into tow where it’s reasonable.



Watershed

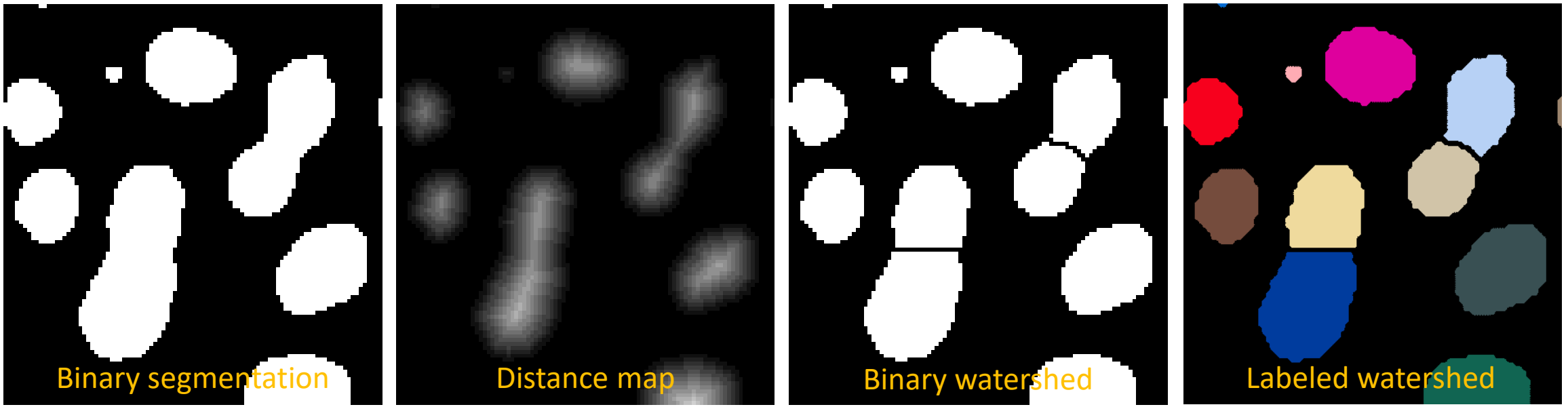
- The watershed algorithm for binary images allows cutting one object into tow where it’s reasonable.





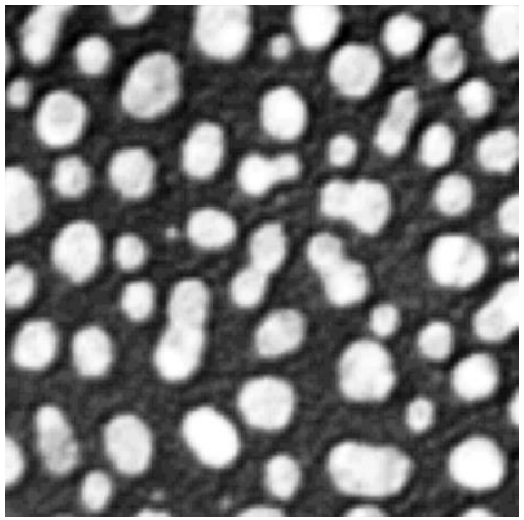
Watershed

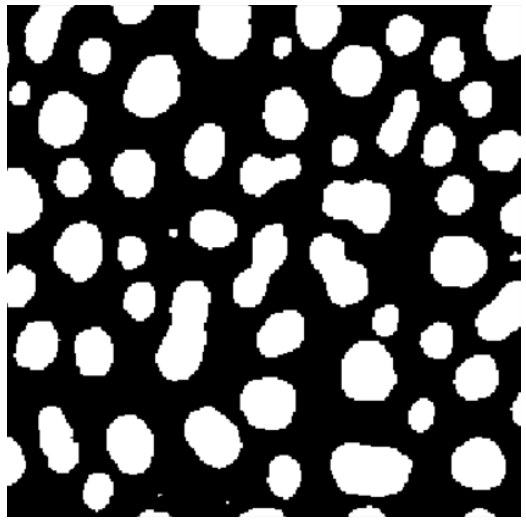
- The watershed algorithm for binary images allows cutting one object into two where it’s reasonable.
- The distance-maps are typicall made from binary images. It does not take the original image into account!



Watershed use-cases

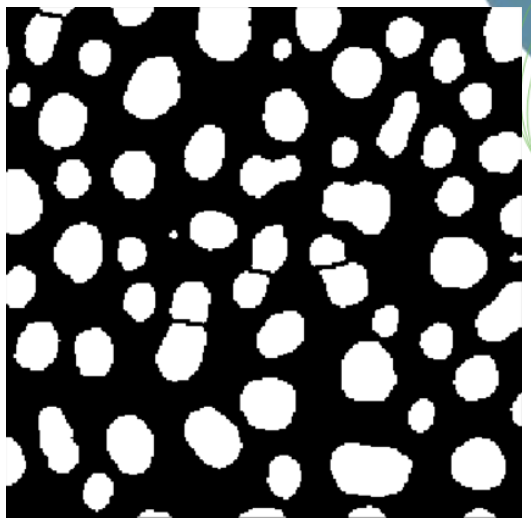
- Split dense objects



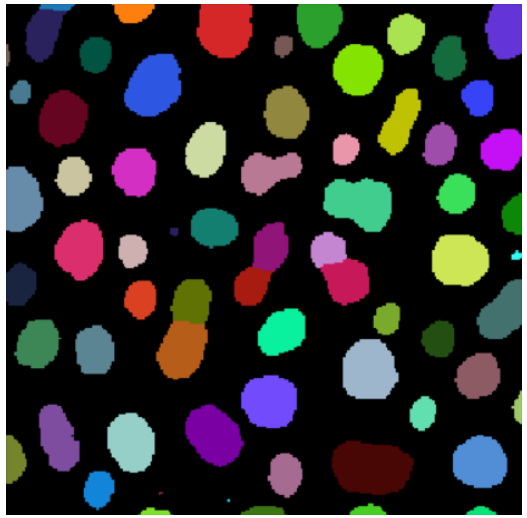


Binarization

Watershed

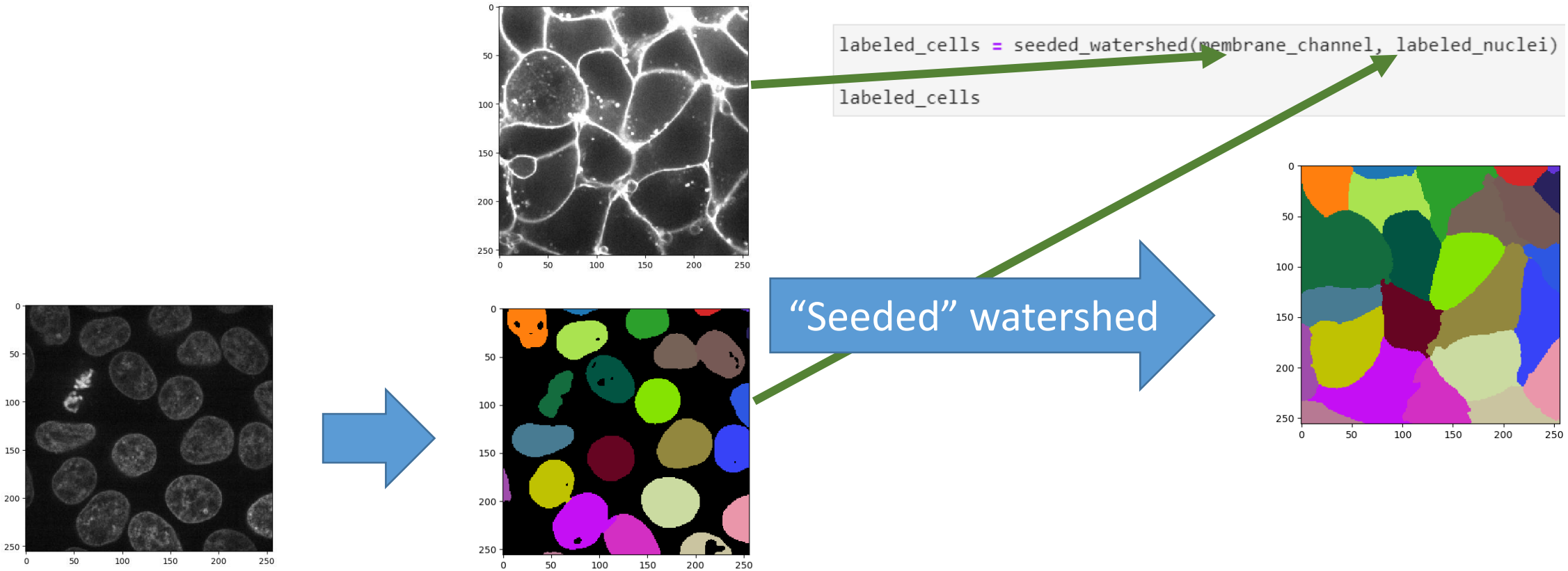


Watershed



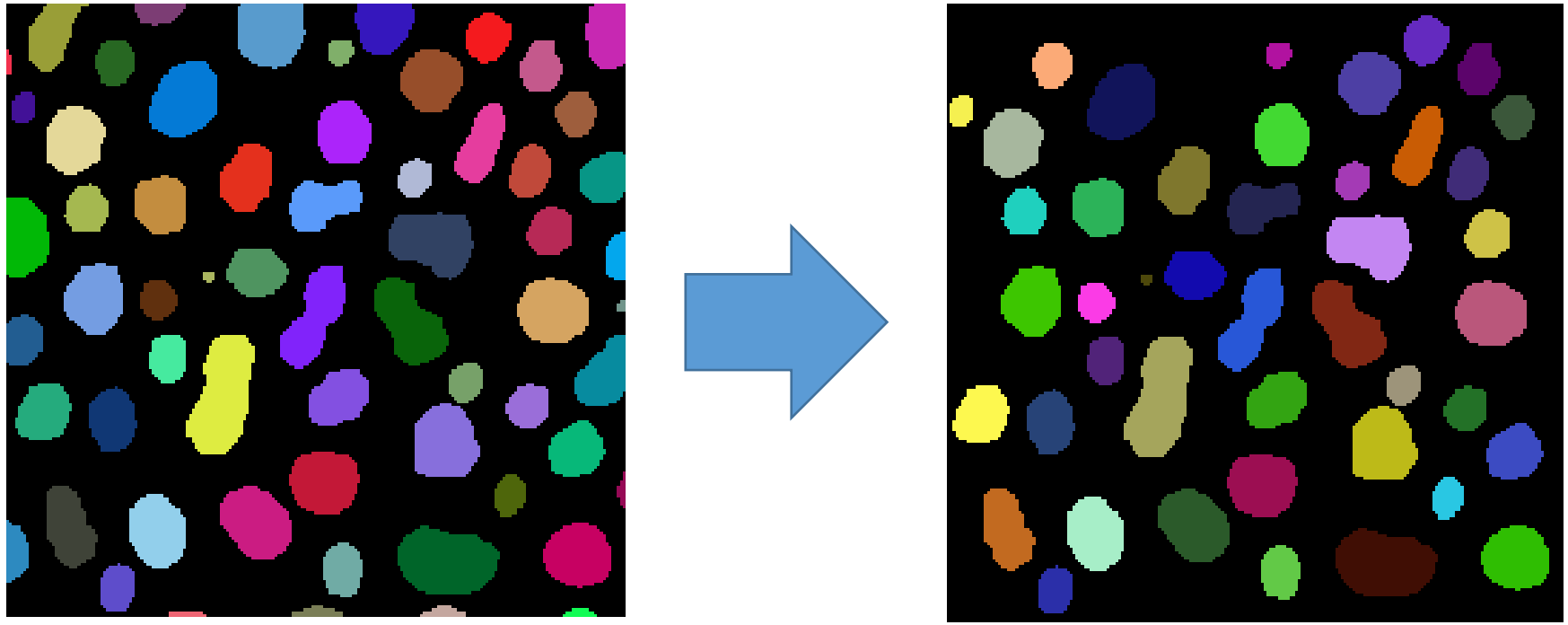
Watershed use-cases

- Seeded watershed: Flood regions from pre-defined seeds
- Example: Flood cells from nuclei positions



Label post-processing / selections

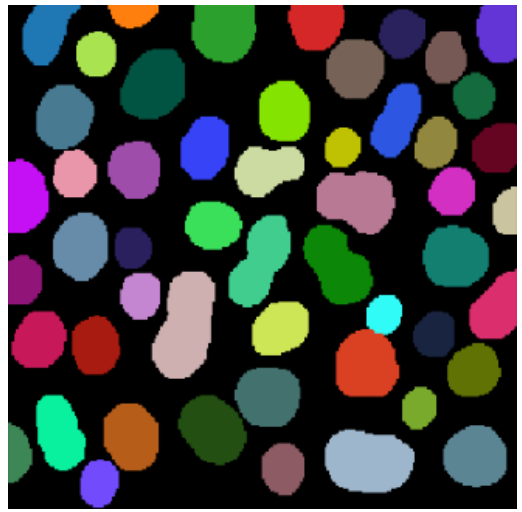
- Remove objects at the image border
- Their measurements (shape, size) would be misleading anyway



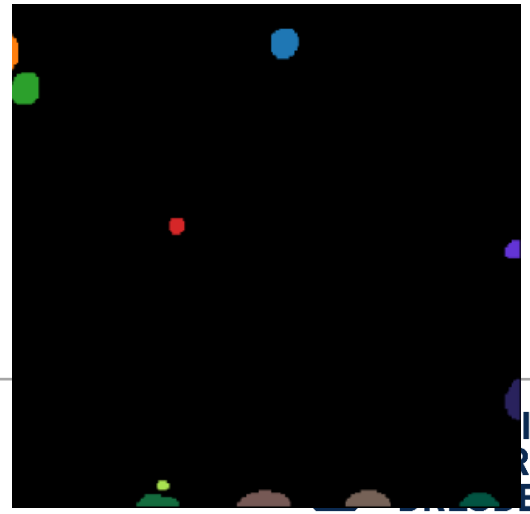
Label post-processing / selections

- Excluding small / large objects
- Common correction-step in case segmentations contain noise-related small particles

Exclude small objects



Exclude large objects



```
small_labels_only = cle.exclude_large_labels(label_image, minimum_size=size_threshold)
```

ScaDS.AI
DRESDEN LEIPZIG

CENTER FOR SCALABLE DATA ANALYTICS AND ARTIFICIAL INTELLIGENCE

Napari

Robert Haase

Funded by

Bundesministerium für Bildung und Forschung

SACHSEN

Diese Maßnahme wird gefördert durch die Bundesregierung aufgrund eines Beschlusses des Deutschen Bundestages. Diese Maßnahme wird mitfinanziert durch Steuermittel auf der Grundlage des von den Abgeordneten des Sächsischen Landtags beschlossenen Haushaltes.

Chan Zuckerberg Initiative

Napari

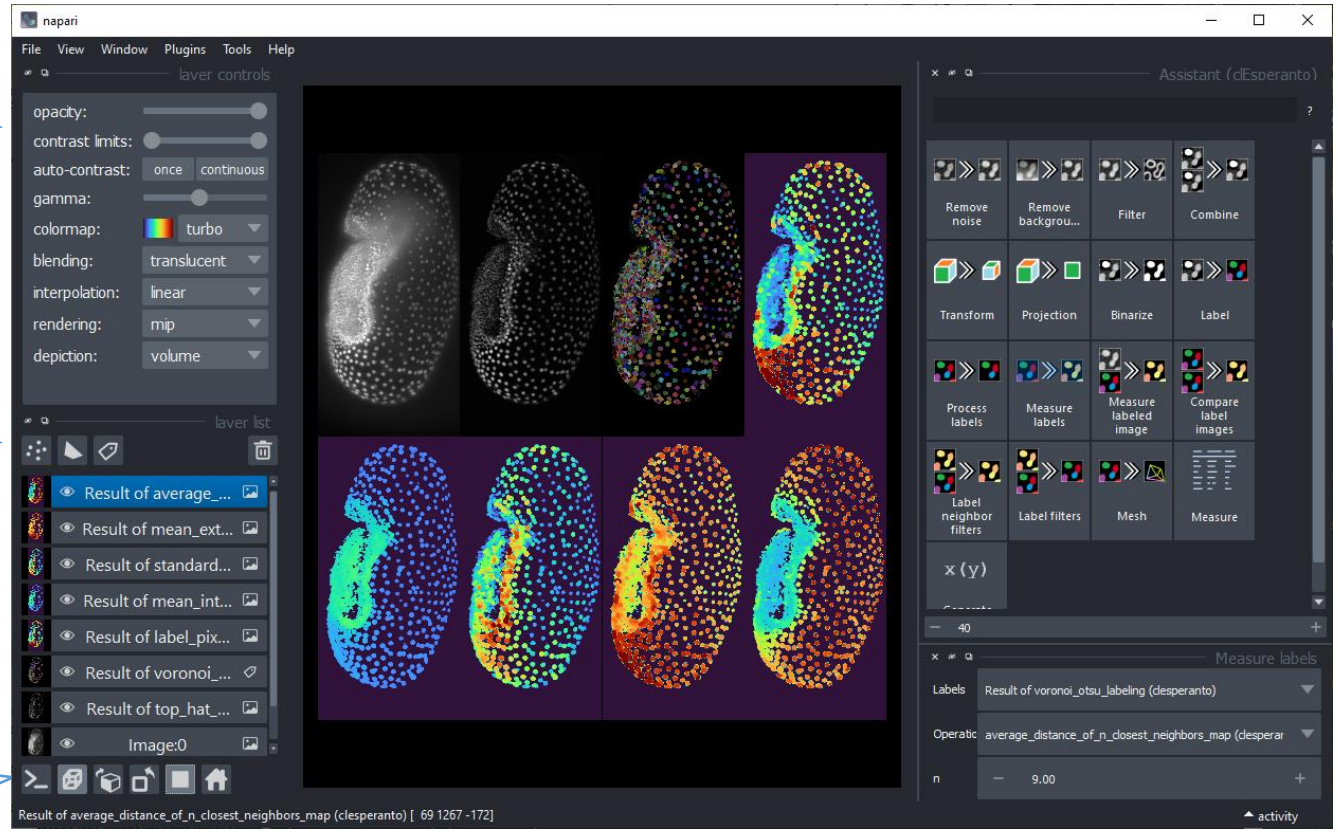
- A viewer for n-dimensional image data written in Python

Napari – Graphical User Interface

View configuration / tools

Layers

Viewer controls

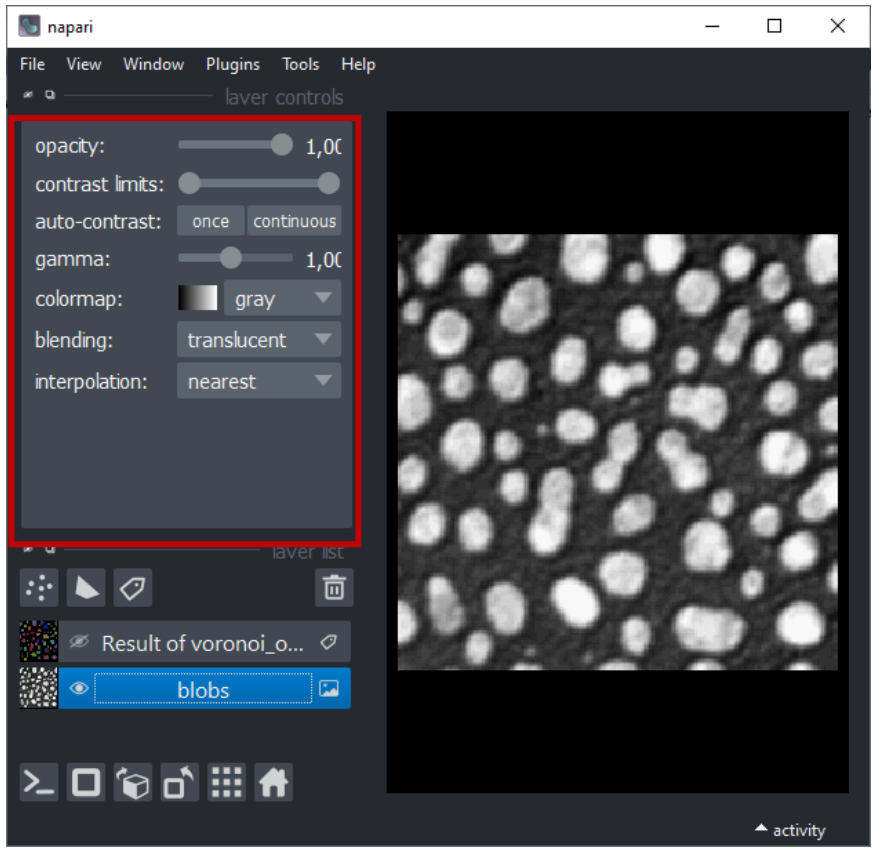


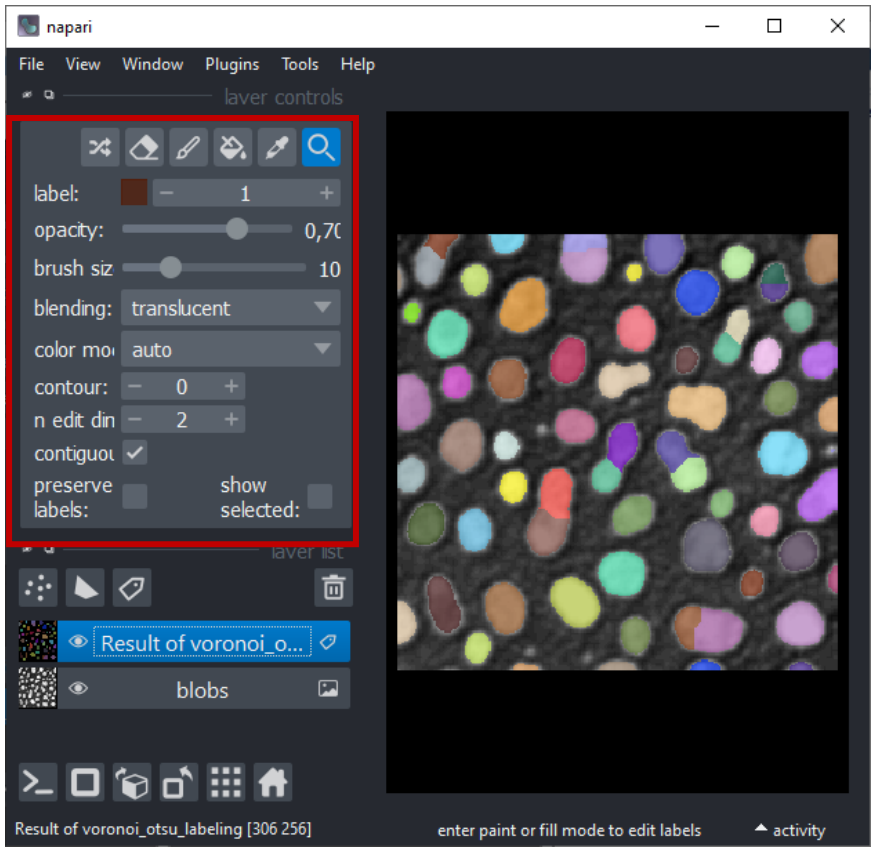
Dock widgets (custom plugins)

Function widgets (custom plugins)

Napari – Graphical User Interface

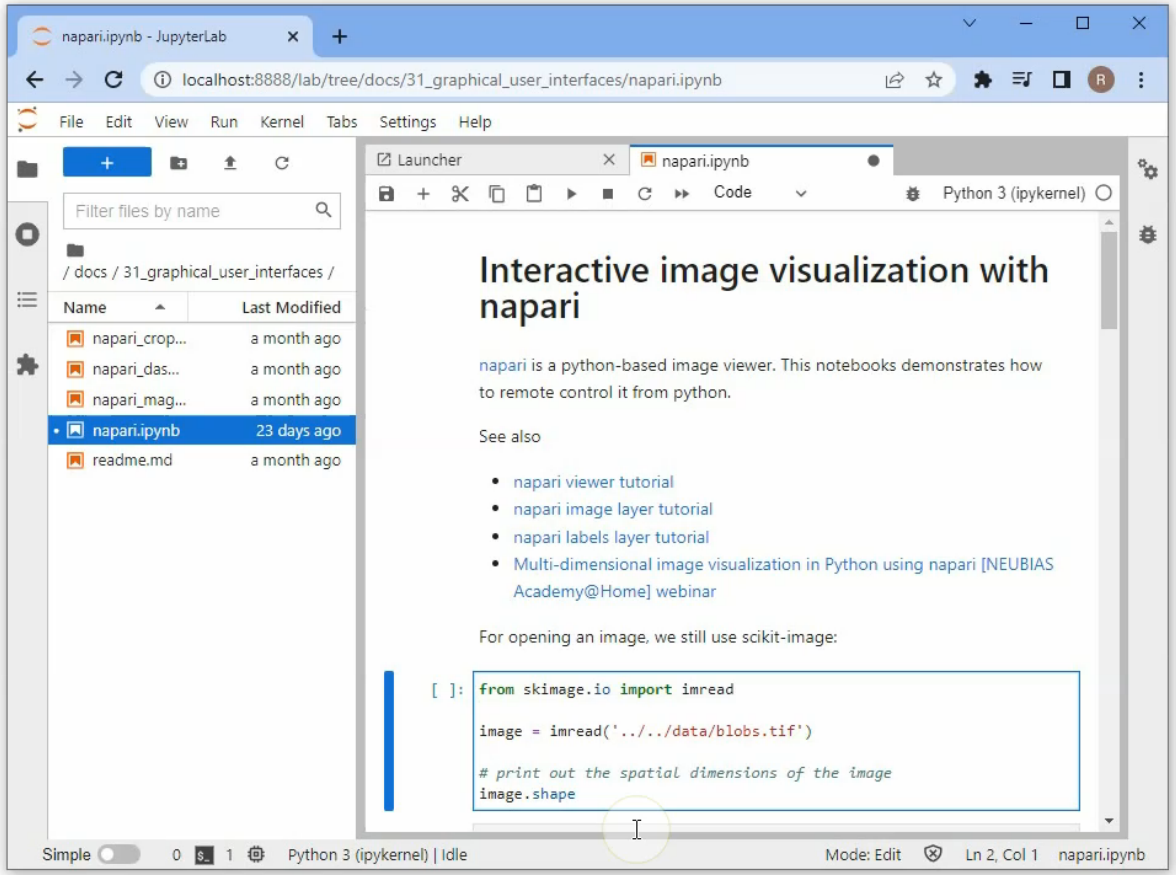
- Context / data type dependent tools





Napari – Python Scripting

- Mixing interactivity and reproducibility



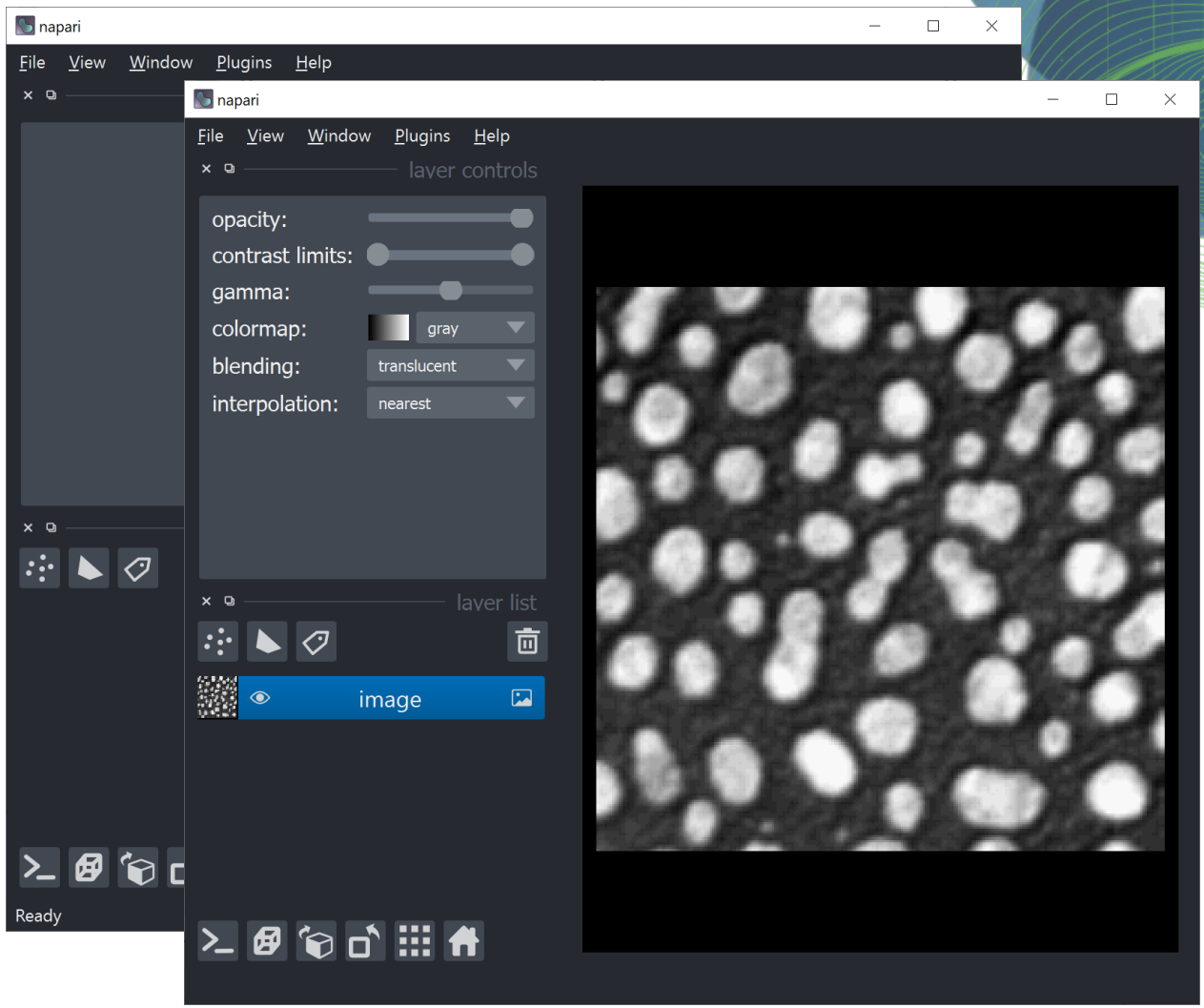
Napari – Python Scripting

- Initialization

```
import napari

# Create an empty viewer
viewer = napari.Viewer()
```

- Adding images

```
viewer.add_image(image)
```



Napari – Python Scripting

- Removing layers

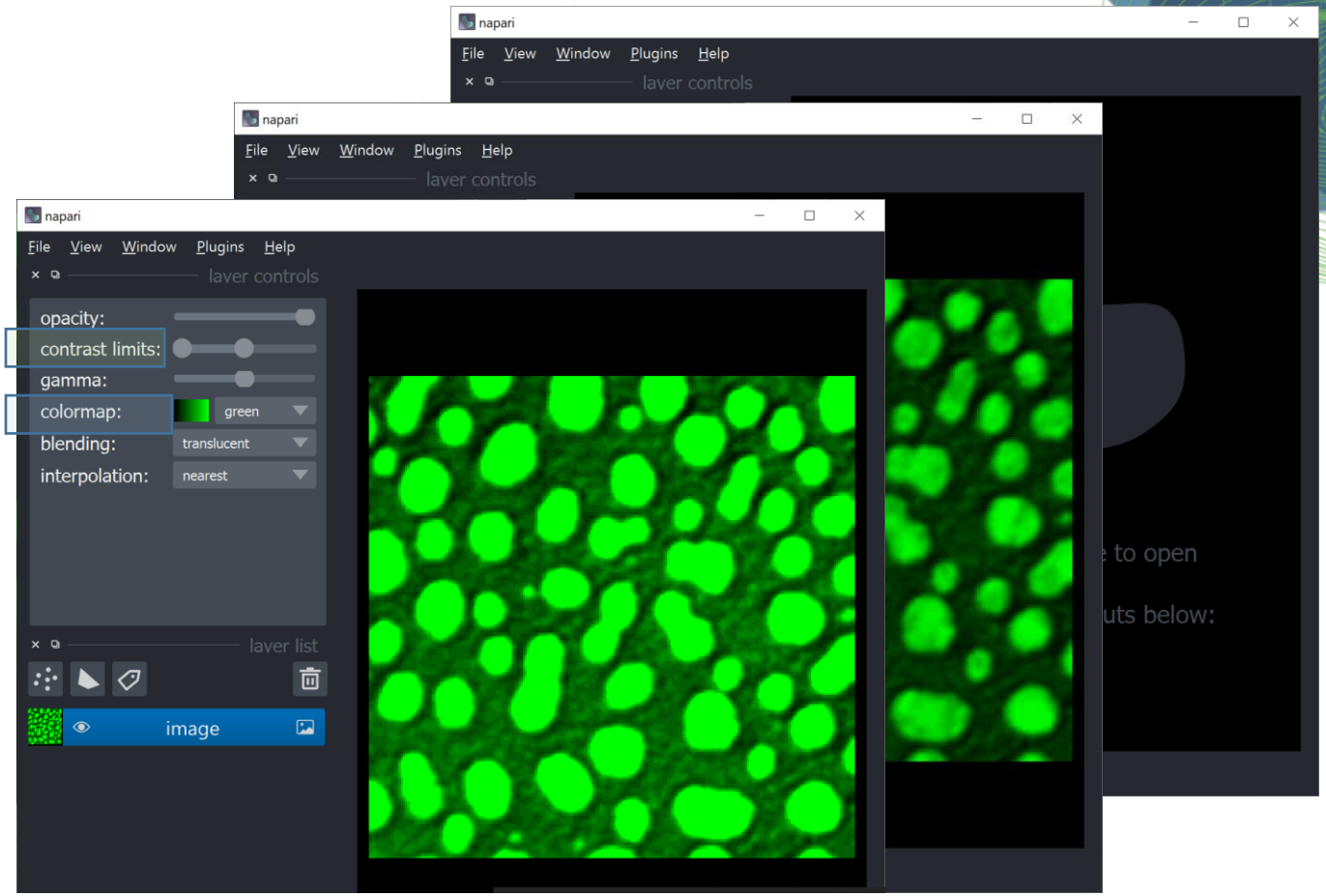
```
for l in viewer.layers:
    viewer.layers.remove(l)
```

- Modify visualization while adding layers

```
viewer.add_image(image,
                 colormap='green')
```

- Modify layers after adding

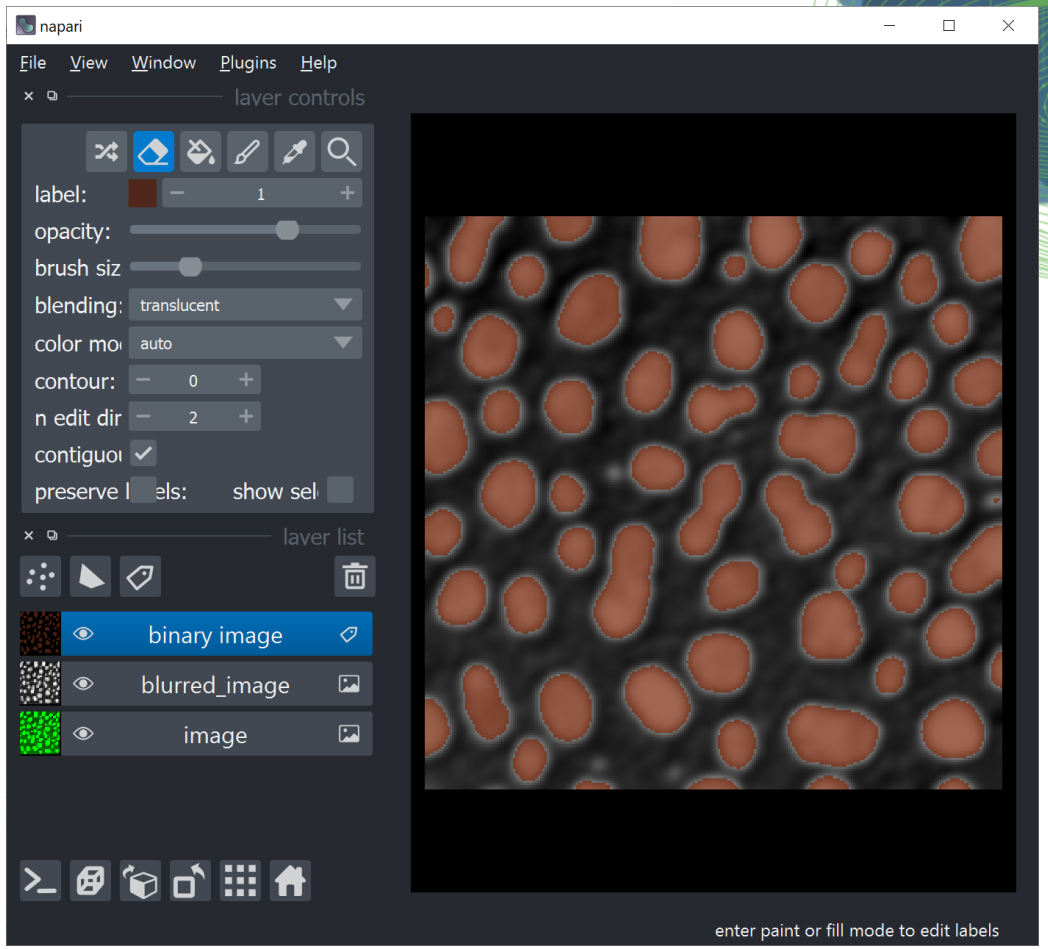
```
layer = viewer.add_image(image)
layer.colormap = 'green'
layer.contrast_limits = (0, 128)
```



Napari – Python Scripting

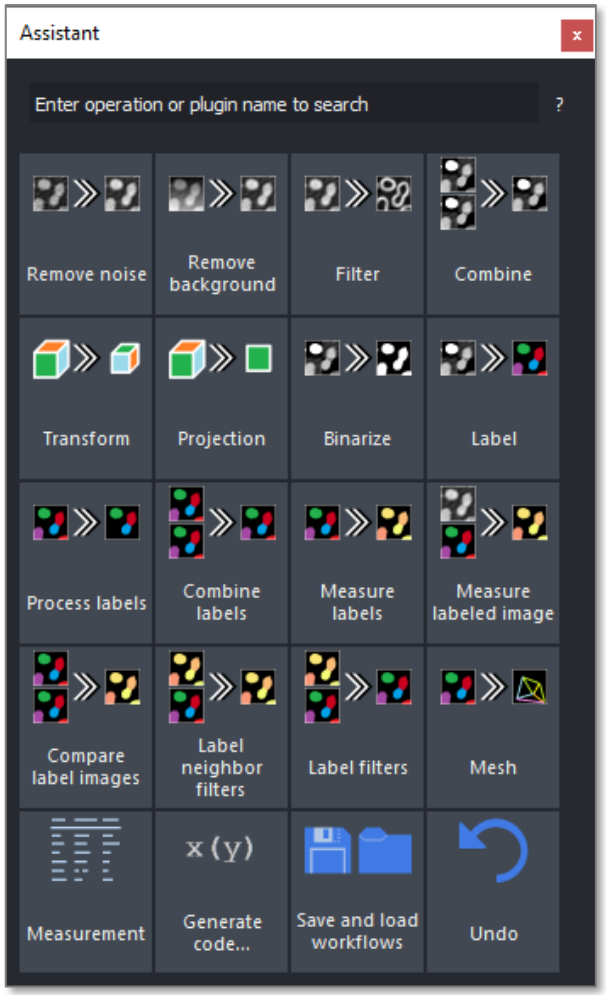
- Binary images and label images visualized as label layers

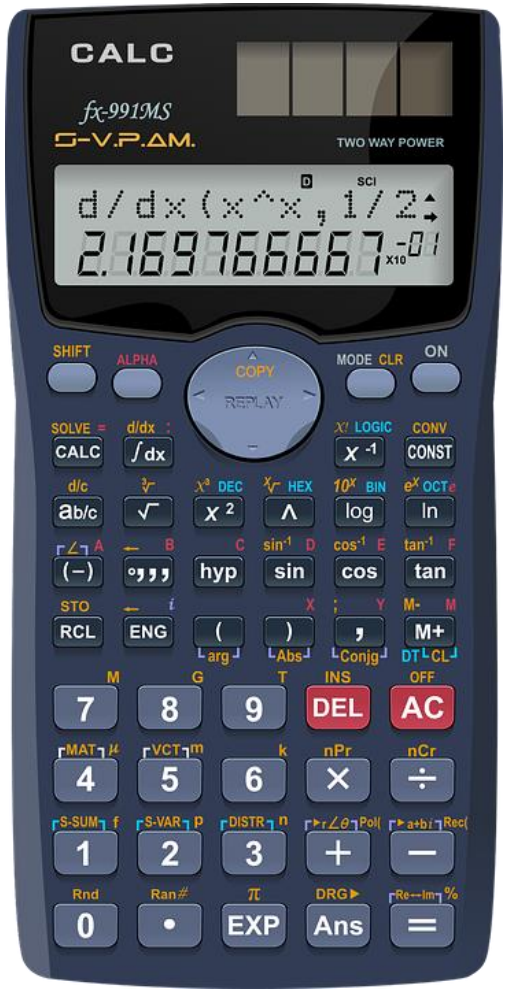
```
from skimage.filters import threshold_otsu
threshold = threshold_otsu(blurred_image)
binary_image = blurred_image > threshold

# Add a new labels layer containing an image
viewer.add_labels(binary_image)
```



The Napari Assistant

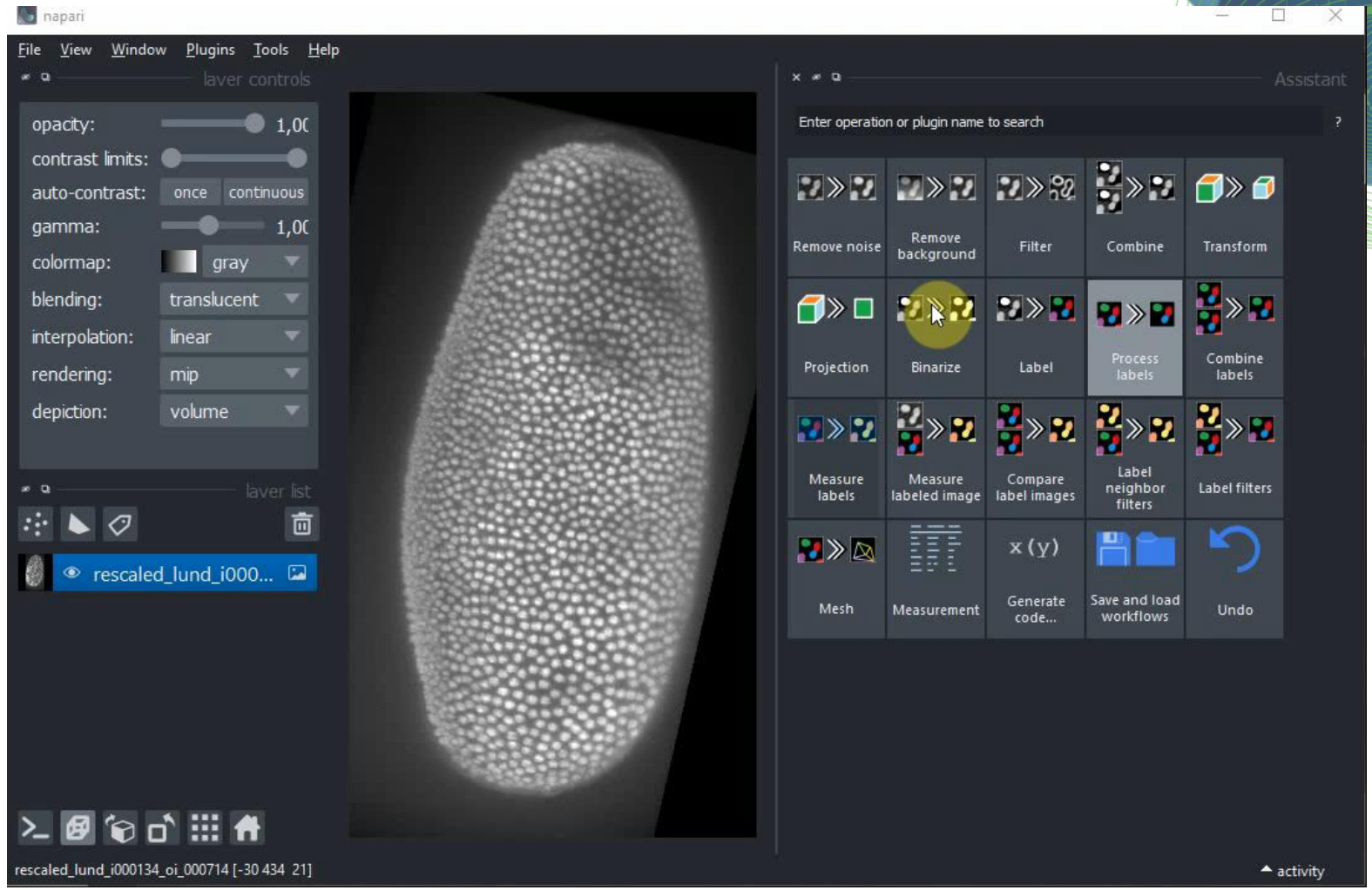
- A pocket-calculator-like interface to build image analysis workflows





The Napari Assistant

- Classical image processing operations + advanced tools
- Saving&loading supported
- Undo [redo]
- Hints for next steps
- …

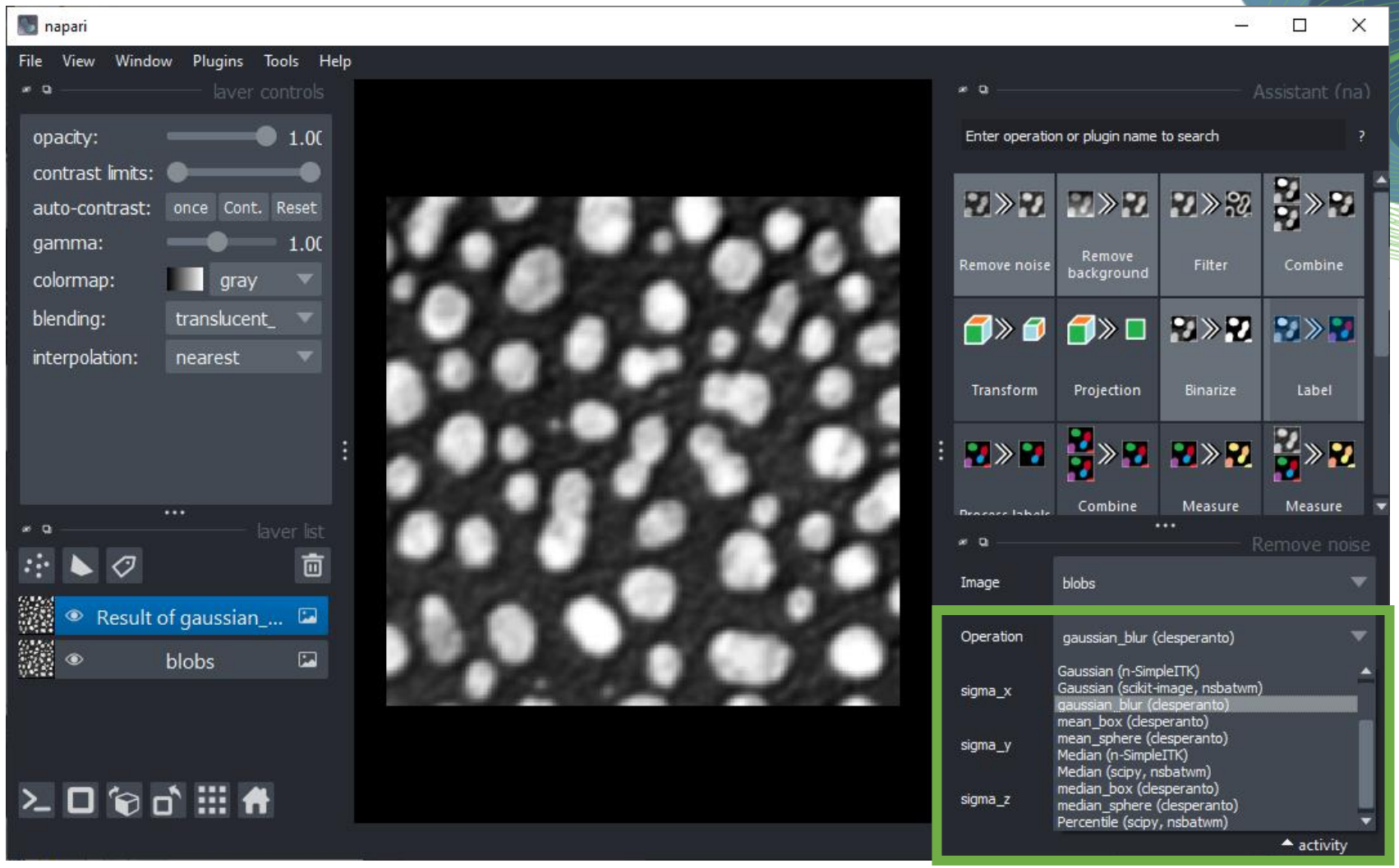


Big thanks to:

Ryan Savill
@RyanSavill4

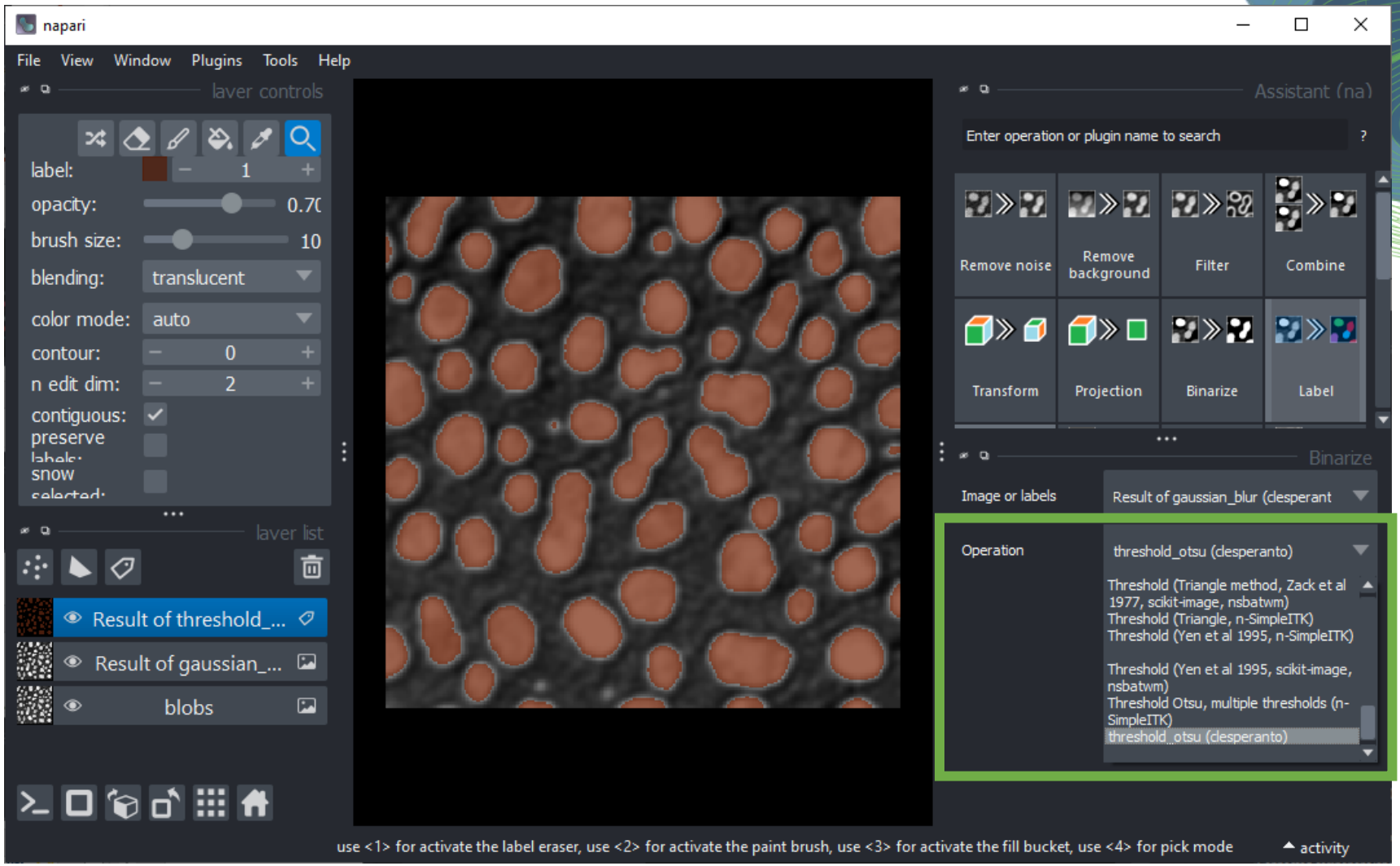
Workflow building

- Try different algorithms, e.g. for removing noise
- Find them in the pulldown



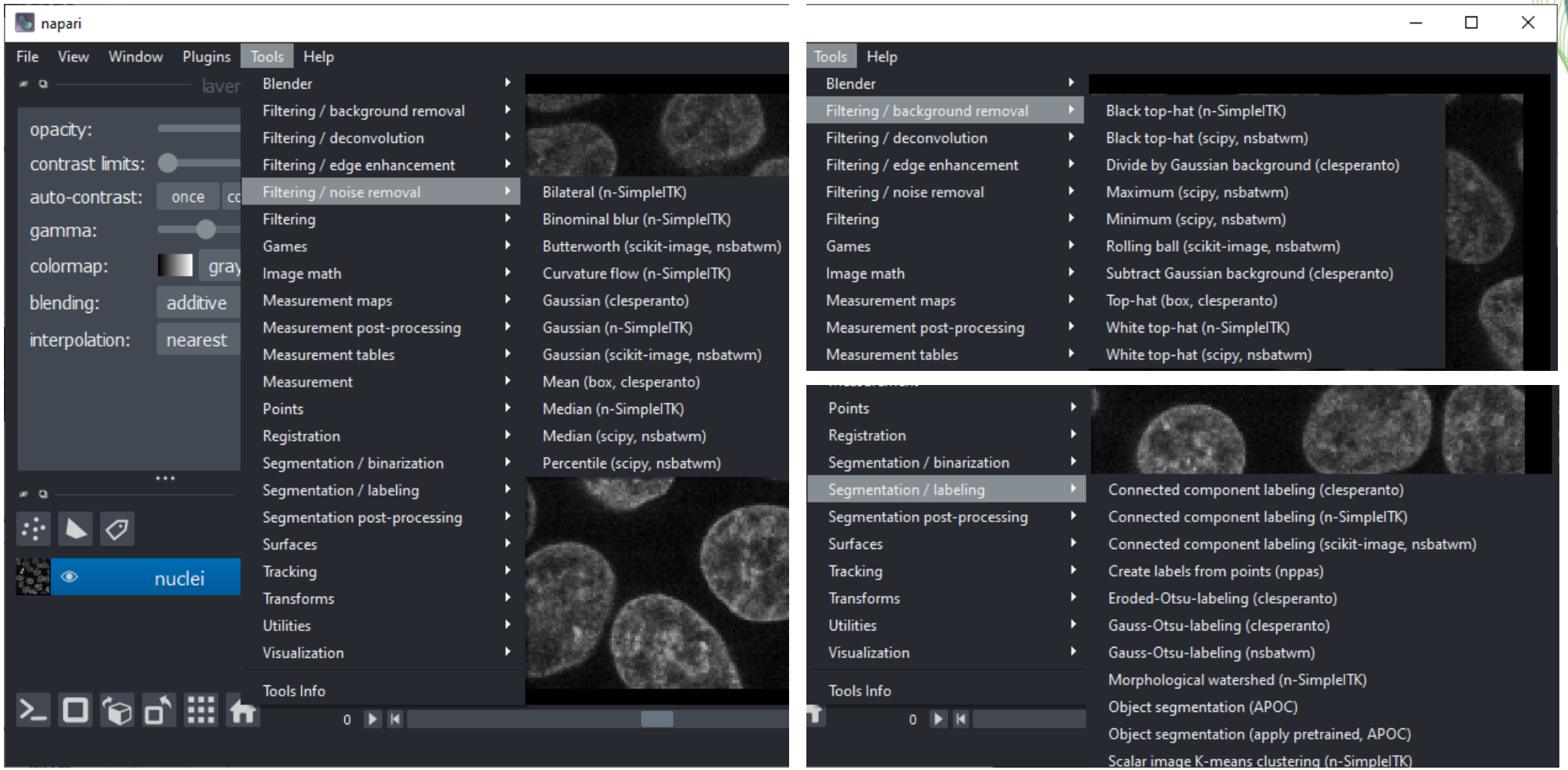
Workflow building

- Try different binarization algorithms



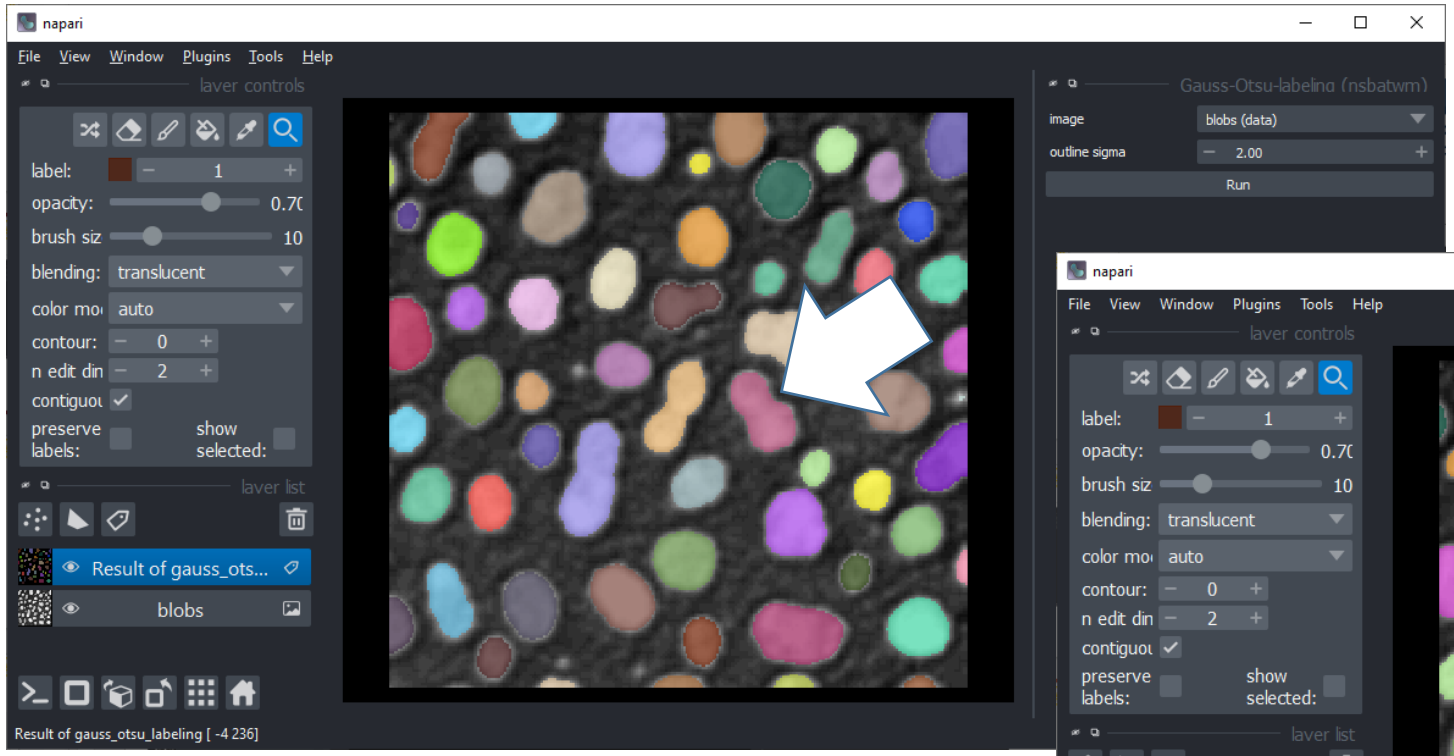
The Tools menu

- Organized in categories



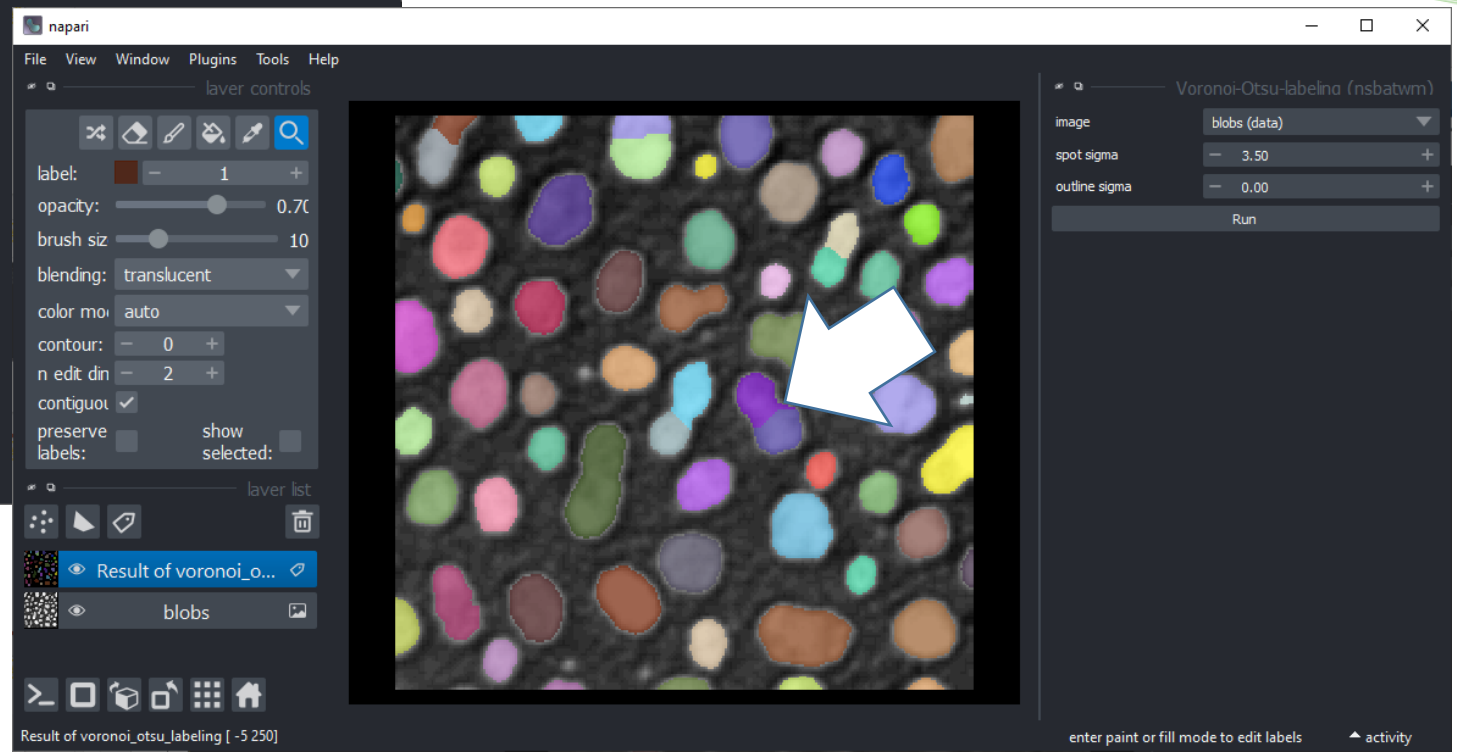
Short-cuts: Voronoi-Otsu-Labeling

Also check out the Tools > Segmentation / labeling menu



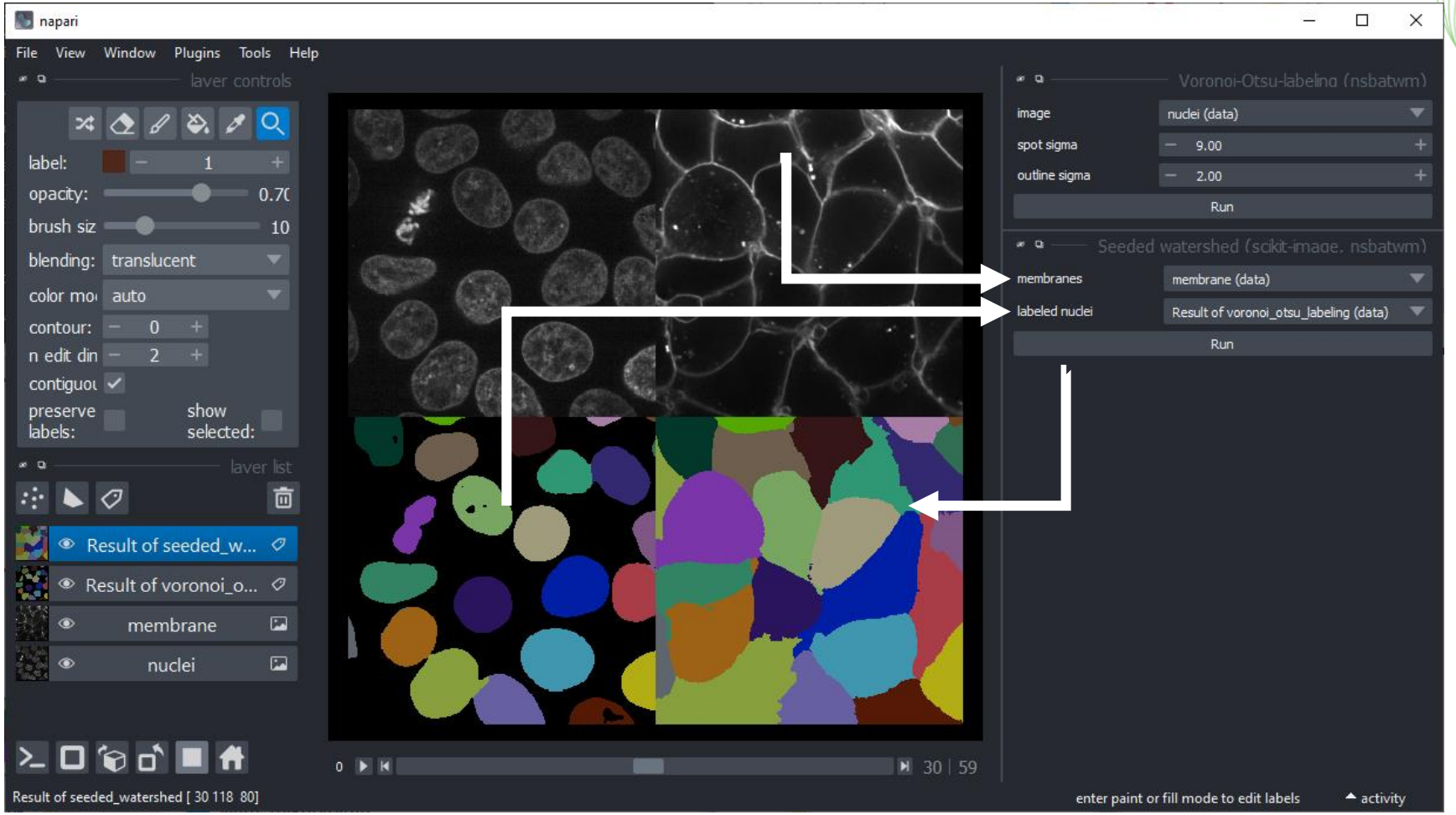
Gauss-Otsu-Labeling

Voronoi-Otsu-Labeling



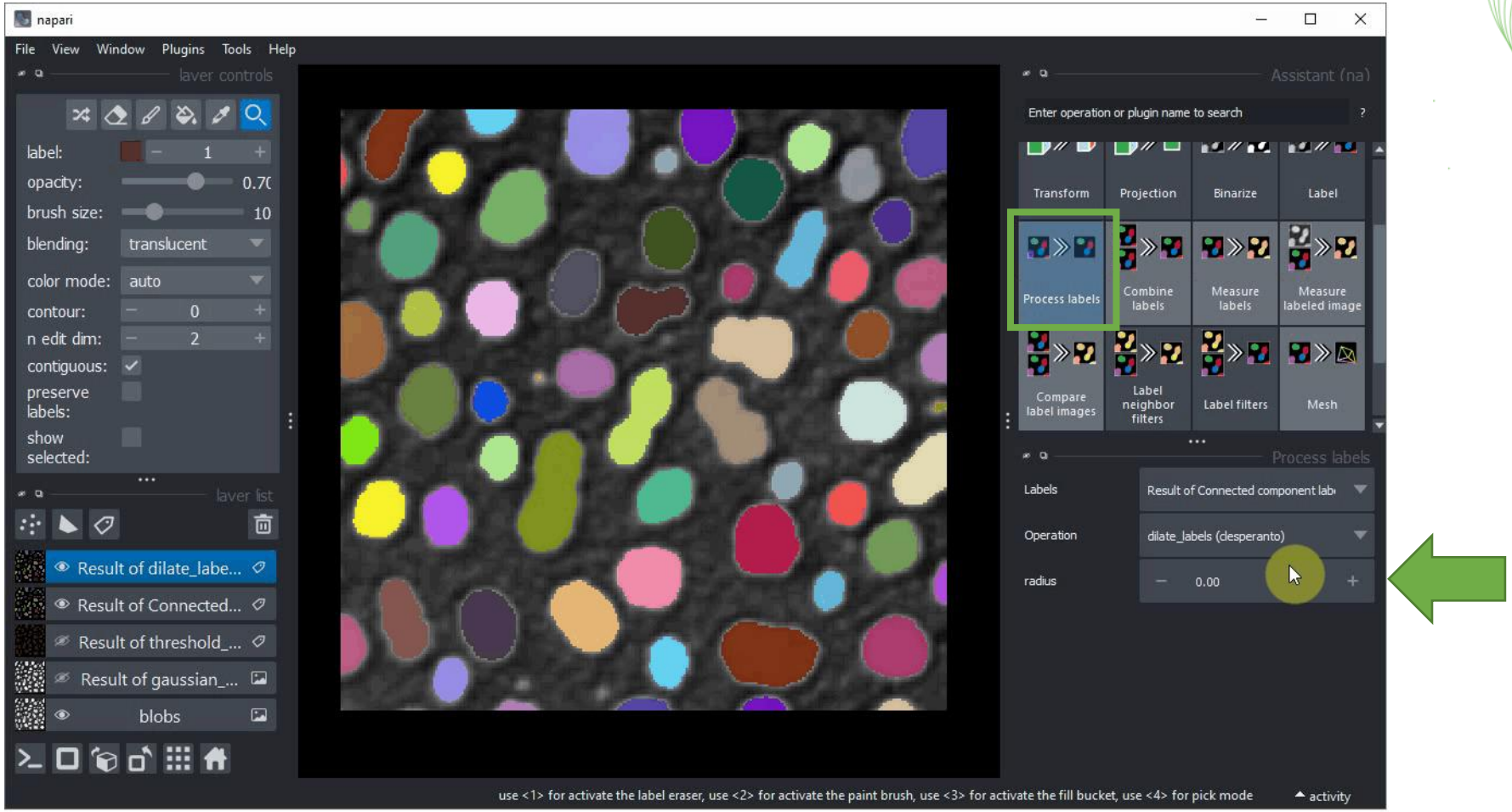
Watershed

Also check out the Tools > Segmentation / labeling menu



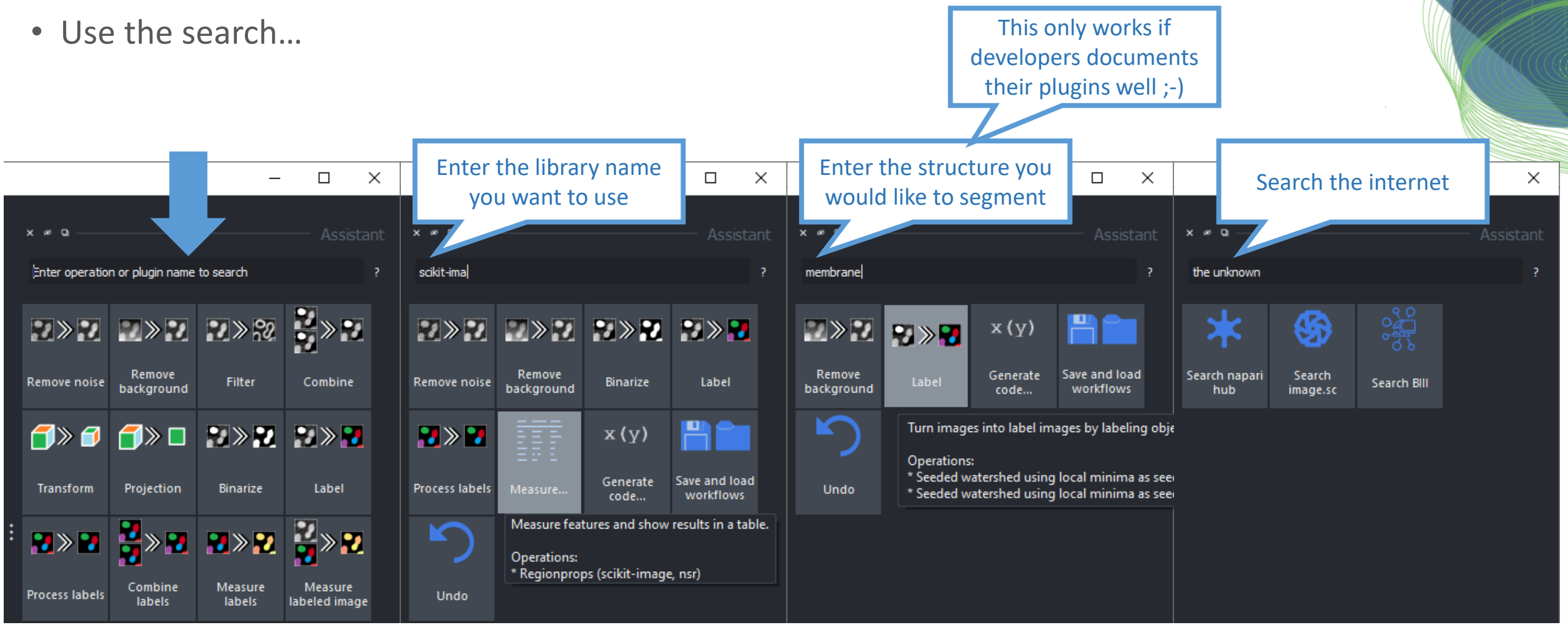
Label erosion, dilation, opening, closing, …

- In Napari Assistant: Process labels

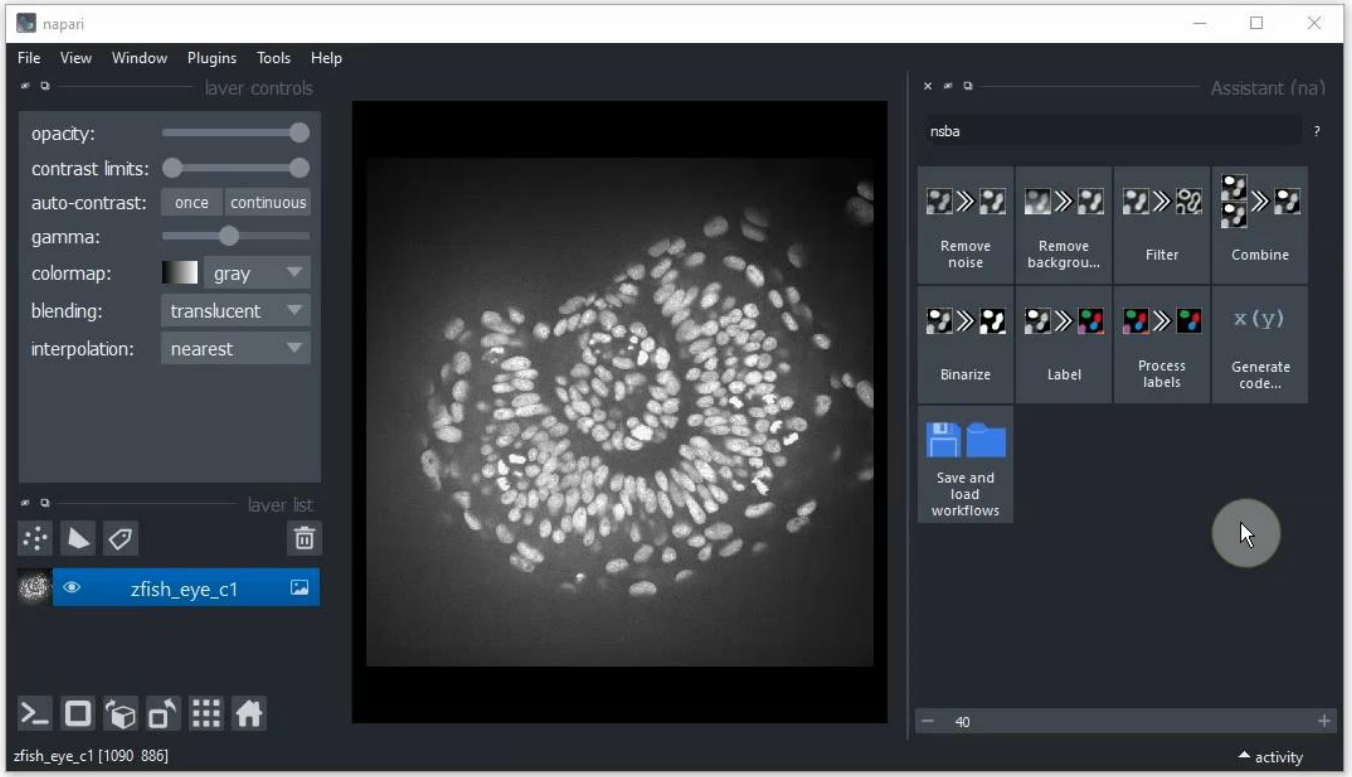


Browse operations

- Use the search…



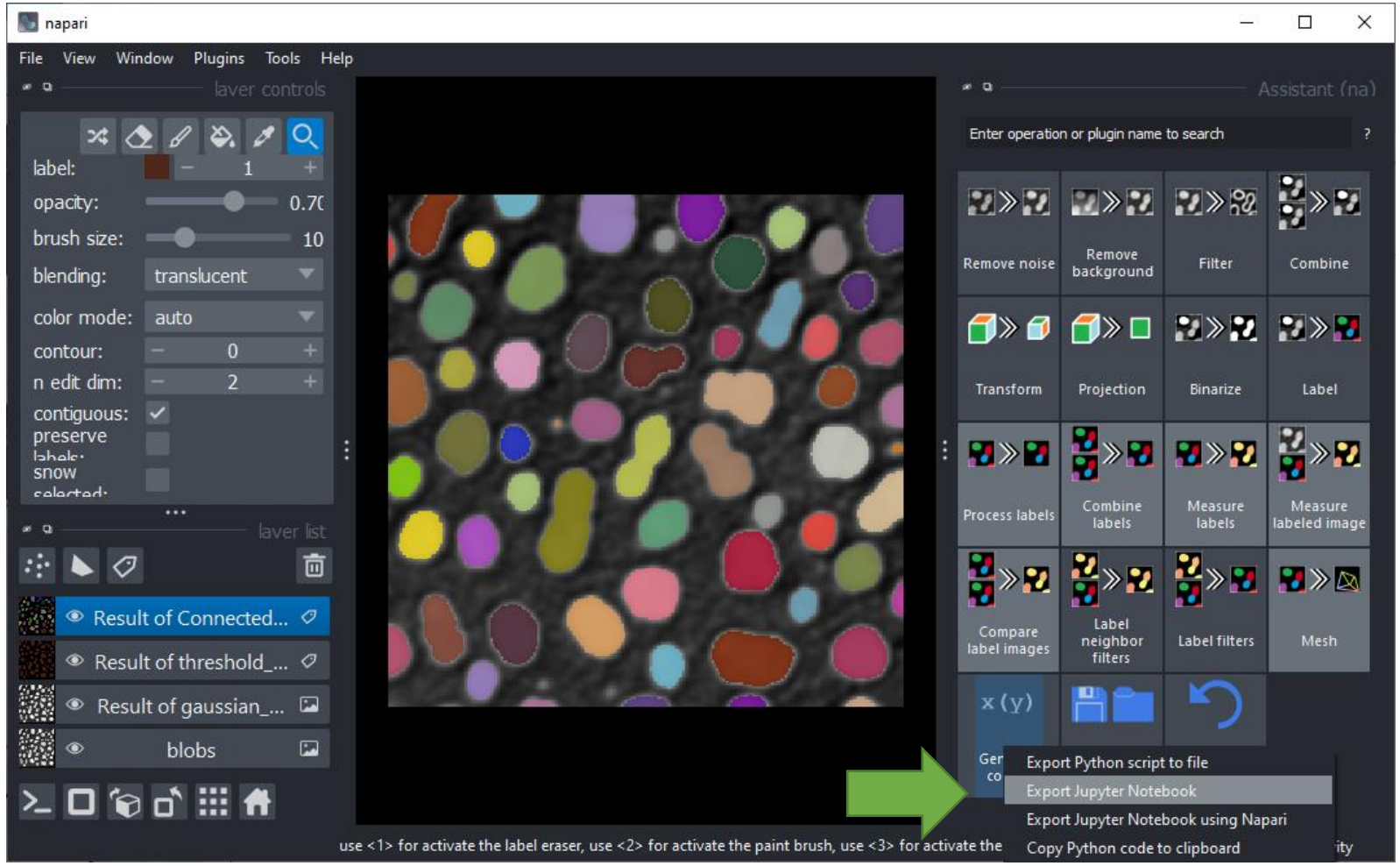
Export code to Jupyter Notebooks

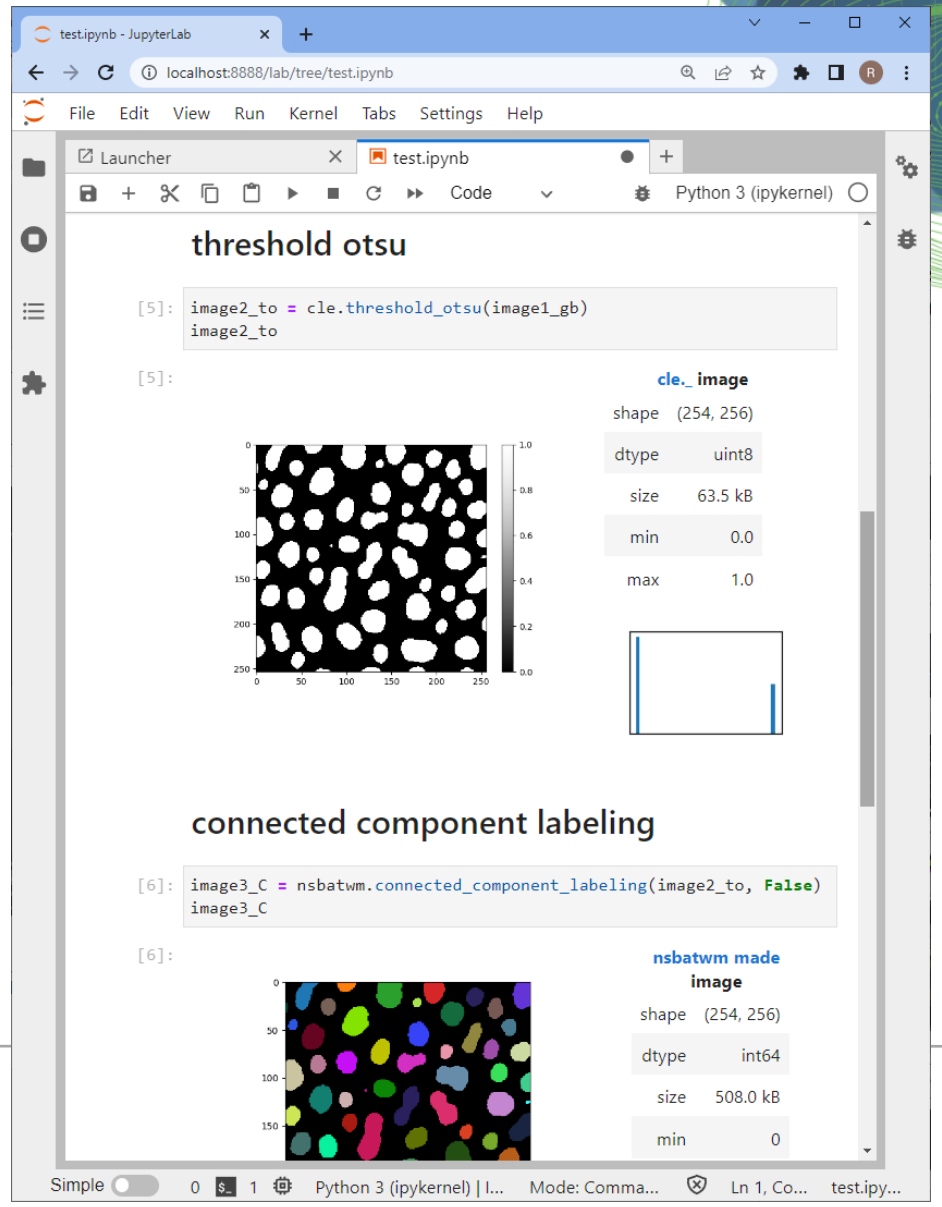


https://github.com/haesleinhuepf/napari-assistant

Image data source: Mauricio Rocha Martins, Norden lab, MPI CBG (now at IGC Oeiras)

Export code to Jupyter Notebooks





ScaDS.AI DRESDEN LEIPZIG

Robert Haase
@haesleinhuepf
BIDS Lecture 4/14
April 23rd 2024

https://github.com/haesleinhuepf/napari-assistant#code-generation

ScaDS.AI
DRESDEN LEIPZIG
CENTER FOR SCALABLE DATA ANALYTICS
AND ARTIFICIAL INTELLIGENCE

Exercises

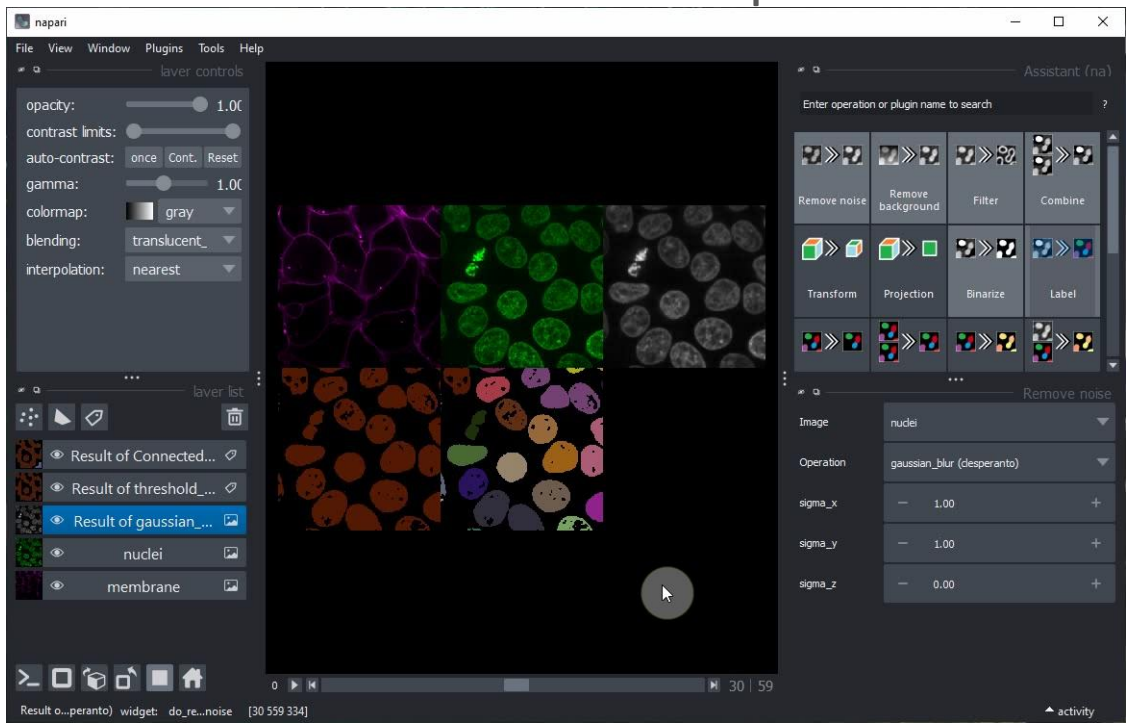
Robert Haase

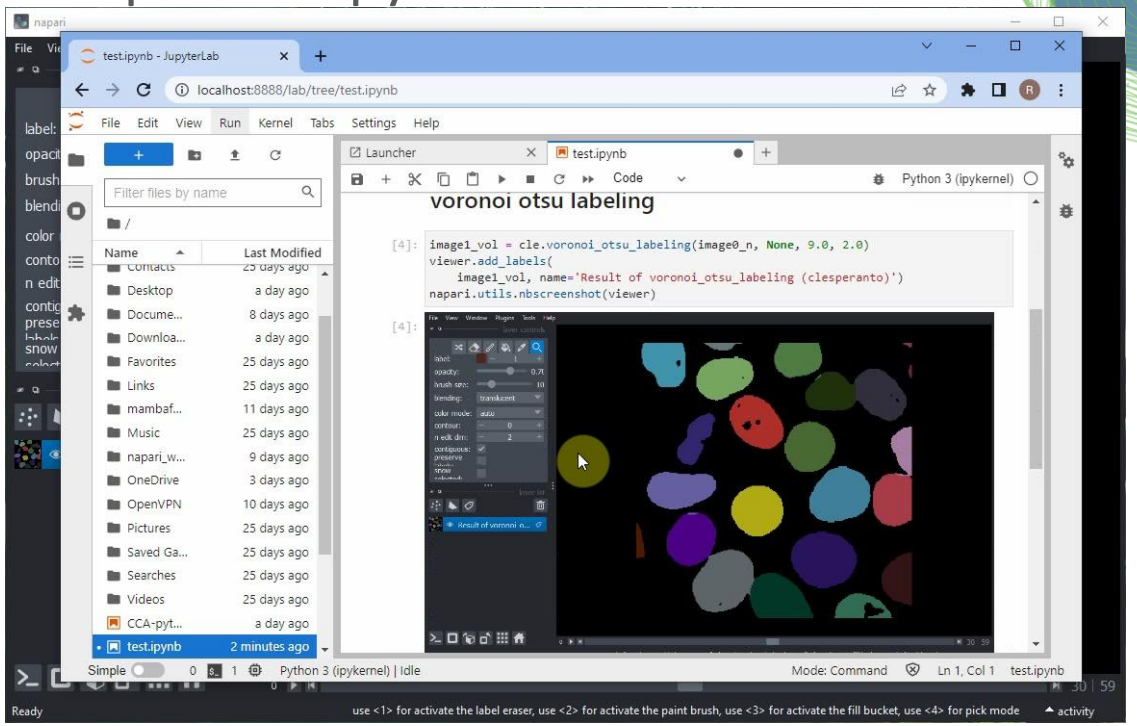
GEFÖRDERT VOM
Bundesministerium
für Bildung
und Forschung

SACHSEN
Diese Maßnahme wird gefördert durch die Bundesregierung aufgrund eines Beschlusses des Deutschen Bundestages.
Diese Maßnahme wird mitfinanziert durch Steuermittel auf der Grundlage des von den Abgeordneten des Sächsischen Landtags beschlossenen Haushaltes.

Napari - Exercises

- Start napari from the terminal!
- Follow the instructions to set up a workflow and export a Jupyter notebook





https://github.com/ScaDS/BIDS-lecture-2024/blob/main/04b_napari_notebooks/napari-assistant.md

https://github.com/ScaDS/BIDS-lecture-2024/blob/main/04b_napari_notebooks/notebook_export.md

Napari - Exercises

- Start using napari from Python

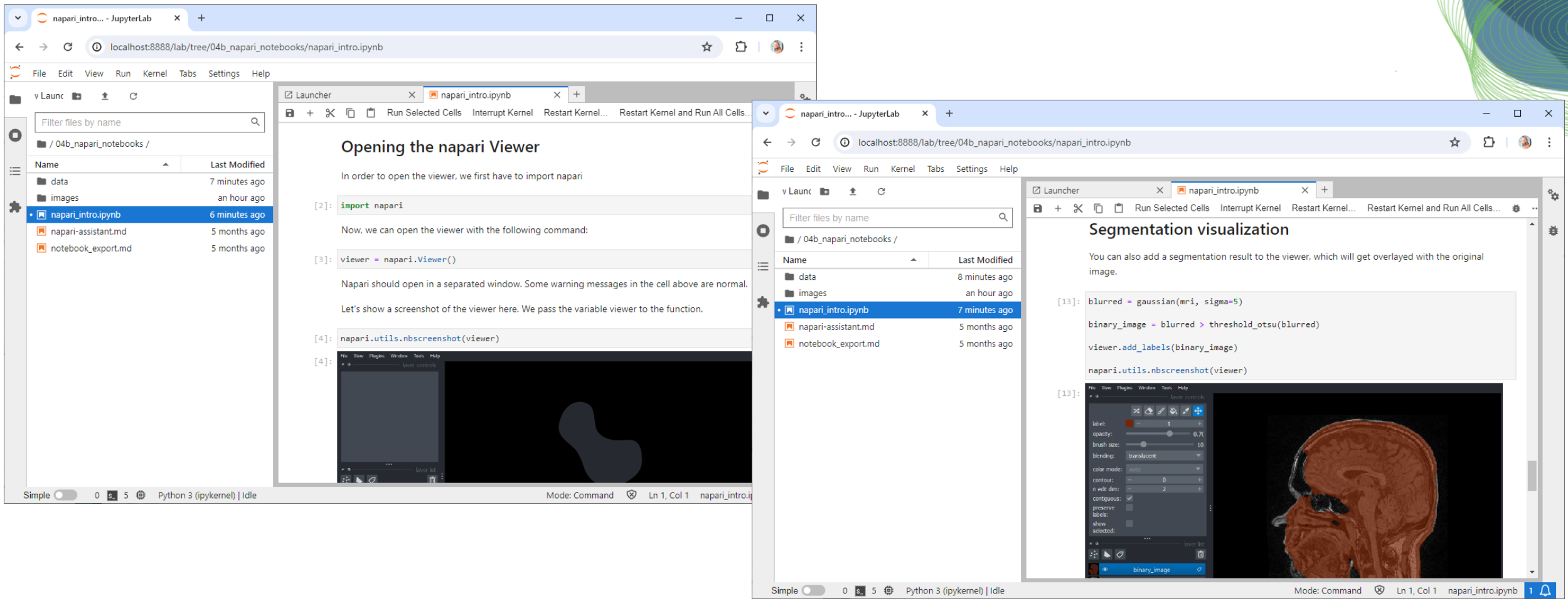


Image segmentation exercises

- Try out segmentation algorithms and apply them to other datasets