PoL
Physics of Life
TU Dresden

CENTER FOR
SYSTEMS BIOLOGY
DRESDEN

Recap day 1

Marcelo Zoccoler, Robert Haase

@haesleinhuepf

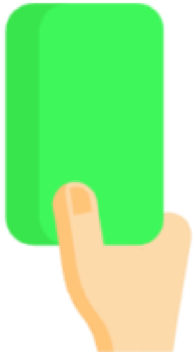
October 2022

Quiz

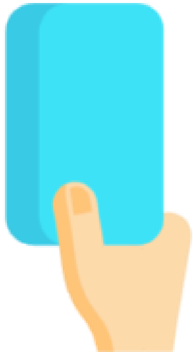
- What’s the first command you should run in a python/data analysis project any after starting a terminal?

conda install …

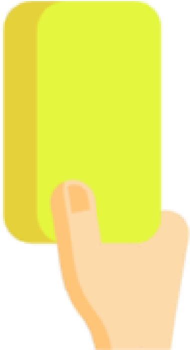
pip install …



conda activate …



jupyter lab

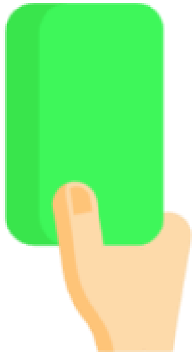


Quiz

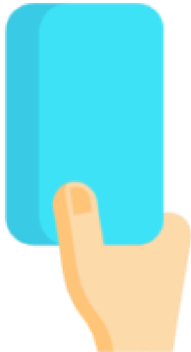
- What characterizes a bio-image analysis workflow that is reproducible?

It outputs numbers

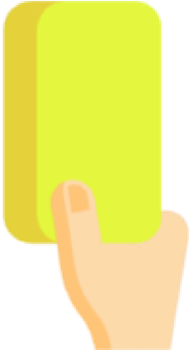
It is fully automated



It is well documented

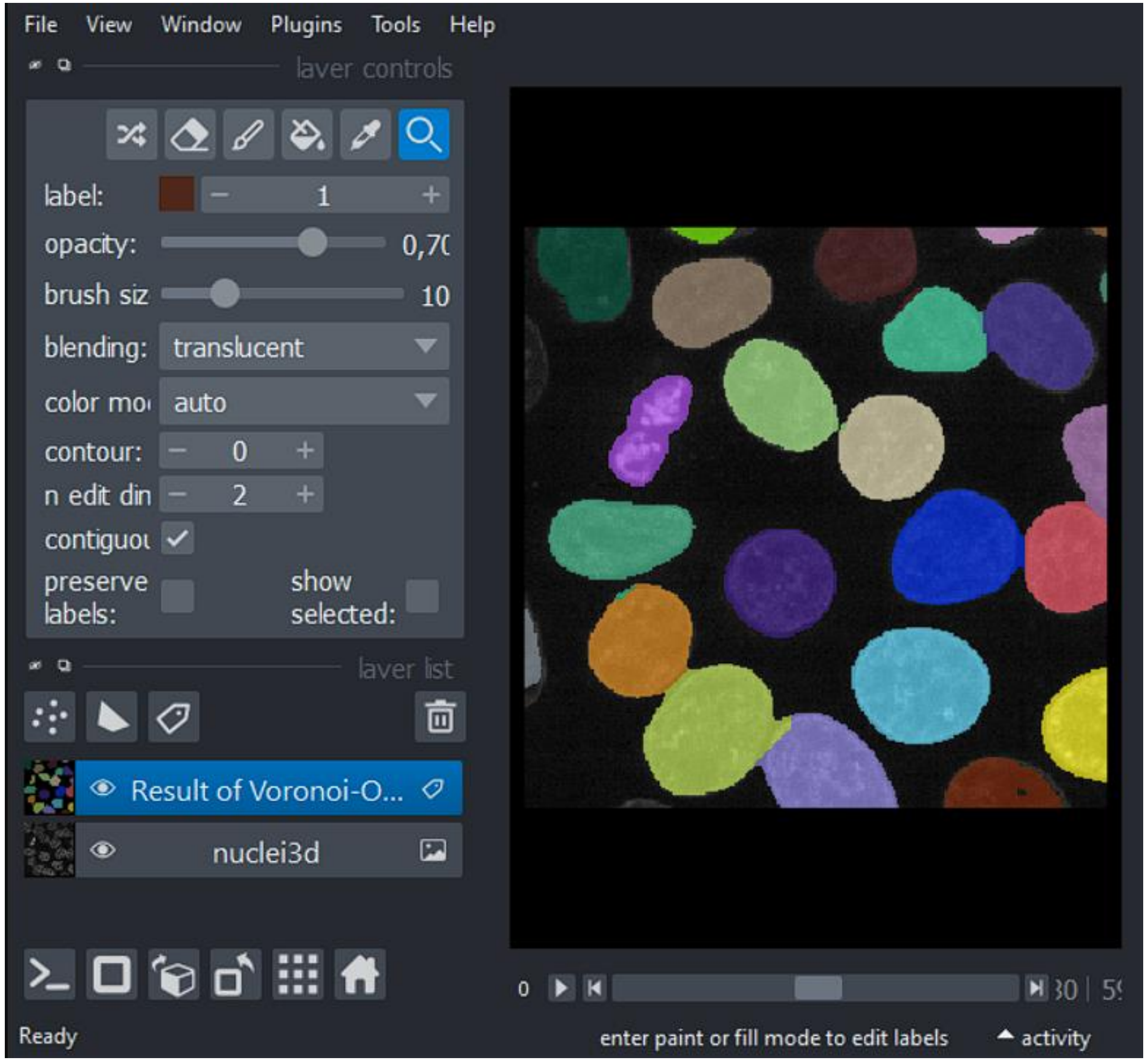


It visualizes results



Napari

How do I get the image from napari?



label: 1

opacity: 0,7(

brush siz 10

blending: translucent

color mo auto

contour: 0

n edit din 2

contiguou ✓

preserve labels: □

show selected: □

Result of Voronoi-O...

nuclei3d

Ready

enter paint or fill mode to edit labels

activity

image = viewer.image[0]

image = viewer.layers[nulcei3d]

image = viewer.layers["nulcei3d"]

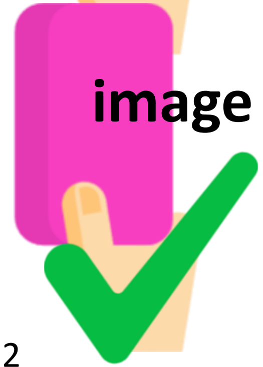
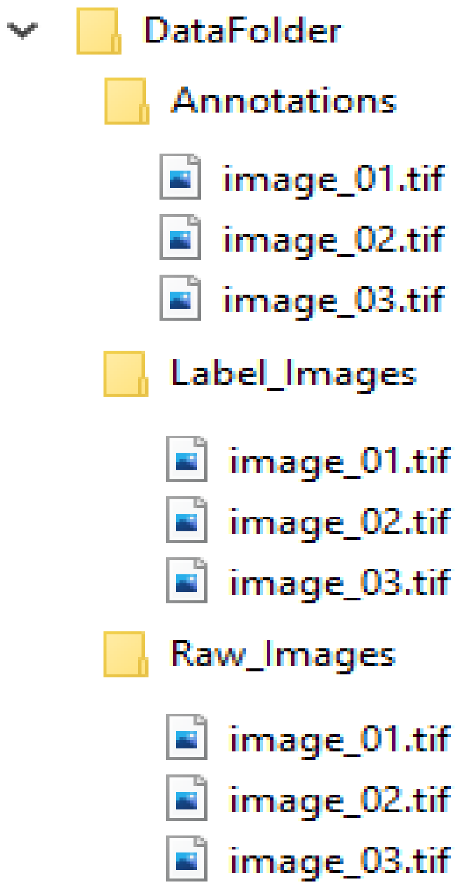
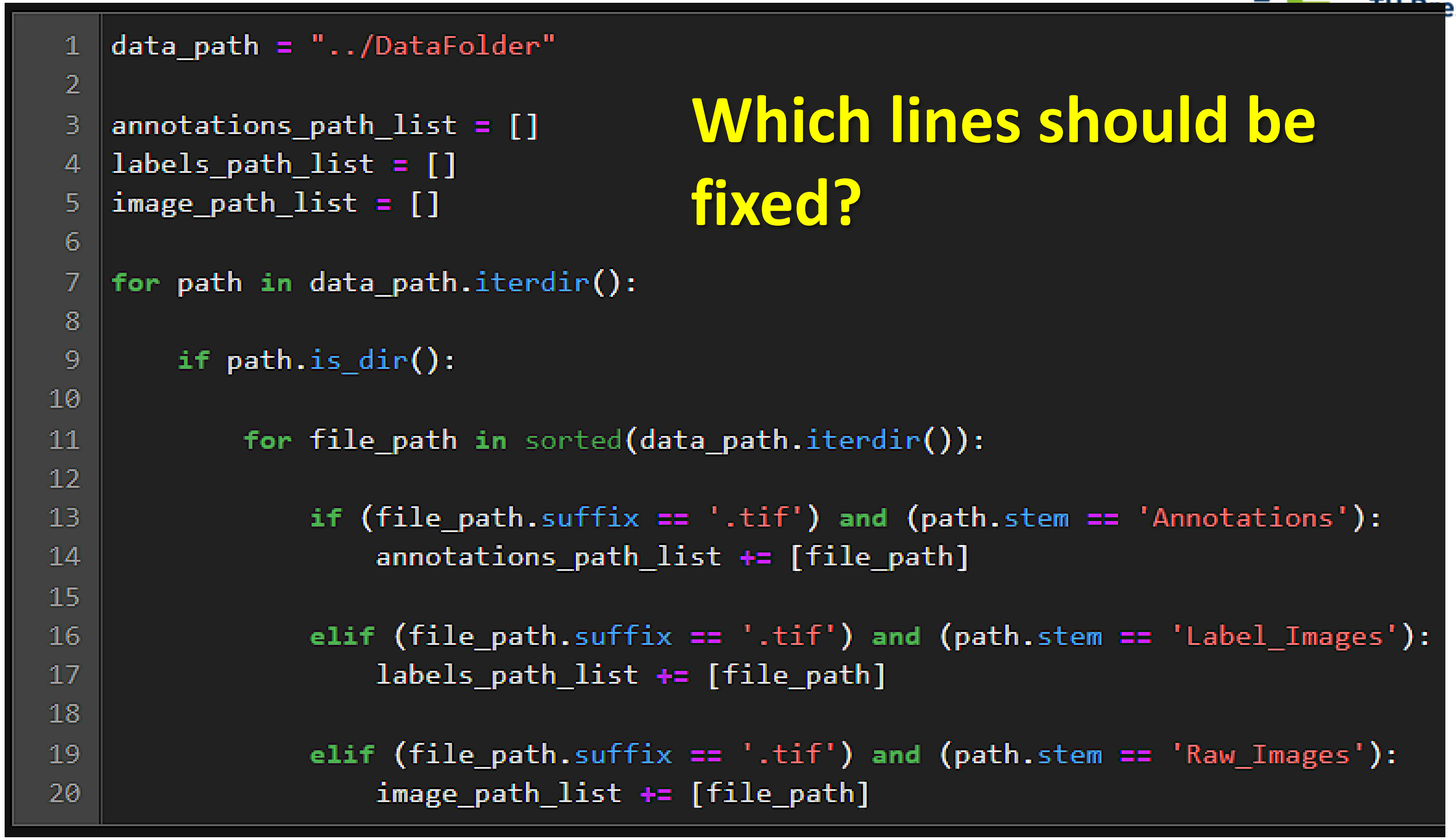


image = viewer.layers[0].data

Folder Structures



- DataFolder
  - Annotations
    - image_01.tif
    - image_02.tif
    - image_03.tif
  - Label_Images
    - image_01.tif
    - image_02.tif
    - image_03.tif
  - Raw_Images
    - image_01.tif
    - image_02.tif
    - image_03.tif



```
 1  data_path = "../DataFolder"
 2
 3  annotations_path_list = []
 4  labels_path_list = []
 5  image_path_list = []
 6
 7  for path in data_path.iterdir():
 8
 9      if path.is_dir():
10
11          for file_path in sorted(data_path.iterdir()):
12
13              if (file_path.suffix == '.tif') and (path.stem == 'Annotations'):
14                  annotations_path_list += [file_path]
15
16              elif (file_path.suffix == '.tif') and (path.stem == 'Label_Images'):
17                  labels_path_list += [file_path]
18
19              elif (file_path.suffix == '.tif') and (path.stem == 'Raw_Images'):
20                  image_path_list += [file_path]
```

Which lines should be fixed?

1 and 11

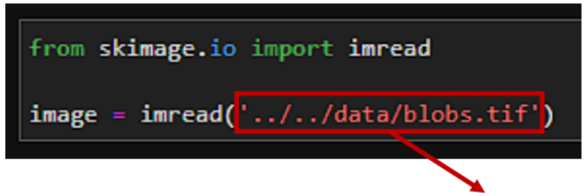
9 and 19

1 and 19

11 and 19

Backslash slide fix

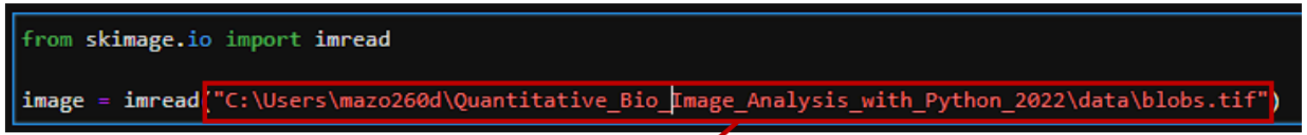
In Python, images can often be read by providing a path to a “imread” function. The path can be a relative path or an absolute path.



```
from skimage.io import imread

image = imread('../../data/blobs.tif')
```

Relative path to image

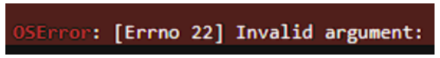


```
from skimage.io import imread

image = imread("C:\Users\mazo260d\Quantitative_Bio_Image_Analysis_with_Python_2022\data\blobs.tif")
```

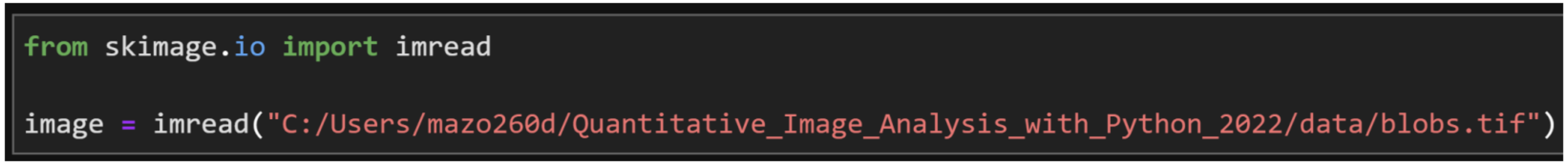
Absolute path to image

Backslash (‘\’) is a special character. In Windows, this may lead to errors.



```
OSError: [Errno 22] Invalid argument:
```

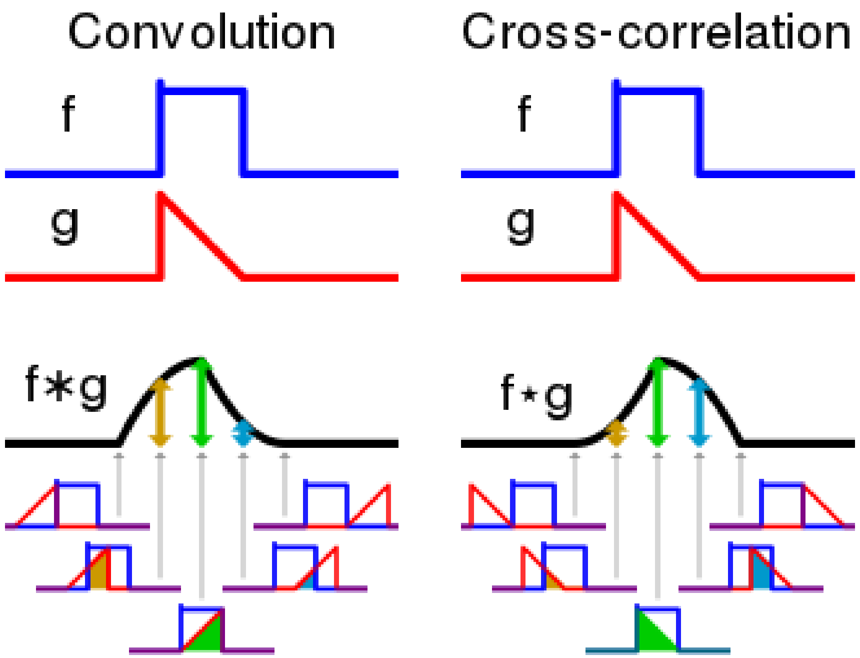
Replace backslashes (‘\’) by forward slashes (‘/’)



```
from skimage.io import imread

image = imread("C:/Users/mazo260d/Quantitative_Image_Analysis_with_Python_2022/data/blobs.tif")
```

Correlation vs Convolution



Cmglee, CC BY-SA 3.0 <https://creativecommons.org/licenses/by-sa/3.0>, via Wikimedia Commons

https://github.com/scipy/scipy/blob/1023d9207fdc1430a8ba196f1a1616ac3c264acf/scipy/ndimage/_filters.py

```
def _correlate_or_convolve(input, weights, output, mode, cval, origin,
                           convolution):
```

Cross-Correlation

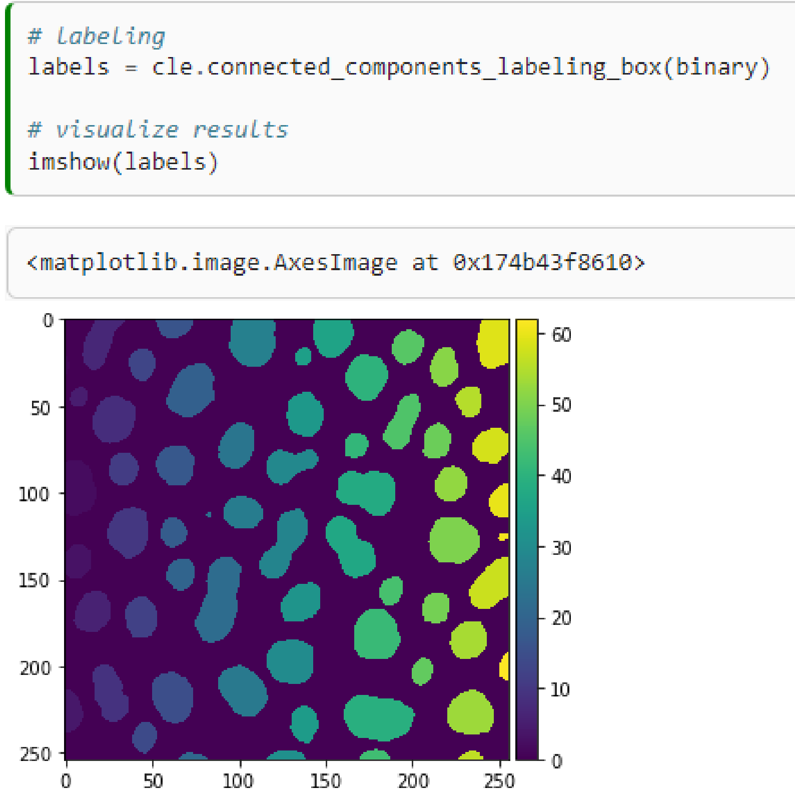
$$F \circ I(x) = \sum_{i=-N}^{N} F(i) I(x+i)$$

Convolution

$$F * I(x) = \sum_{i=-N}^{N} F(i) I(x-i)$$

Why are there multiple functions that do the same?

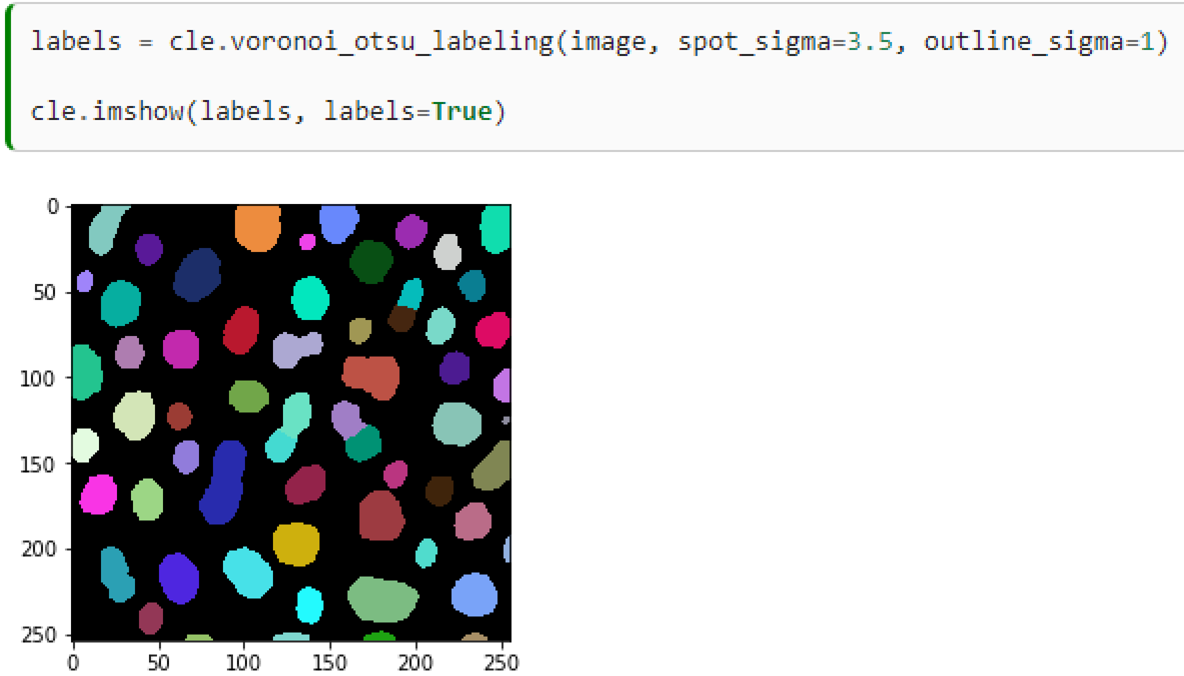
- Diversity is cool.
- Also community needs differ.
- Example: matplotlibs's imshow was not developed specifically for the life-sciences. cle.imshow was.



```
# labeling
labels = cle.connected_components_labeling_box(binary)

# visualize results
imshow(labels)
```

```
<matplotlib.image.AxesImage at 0x174b43f8610>
```



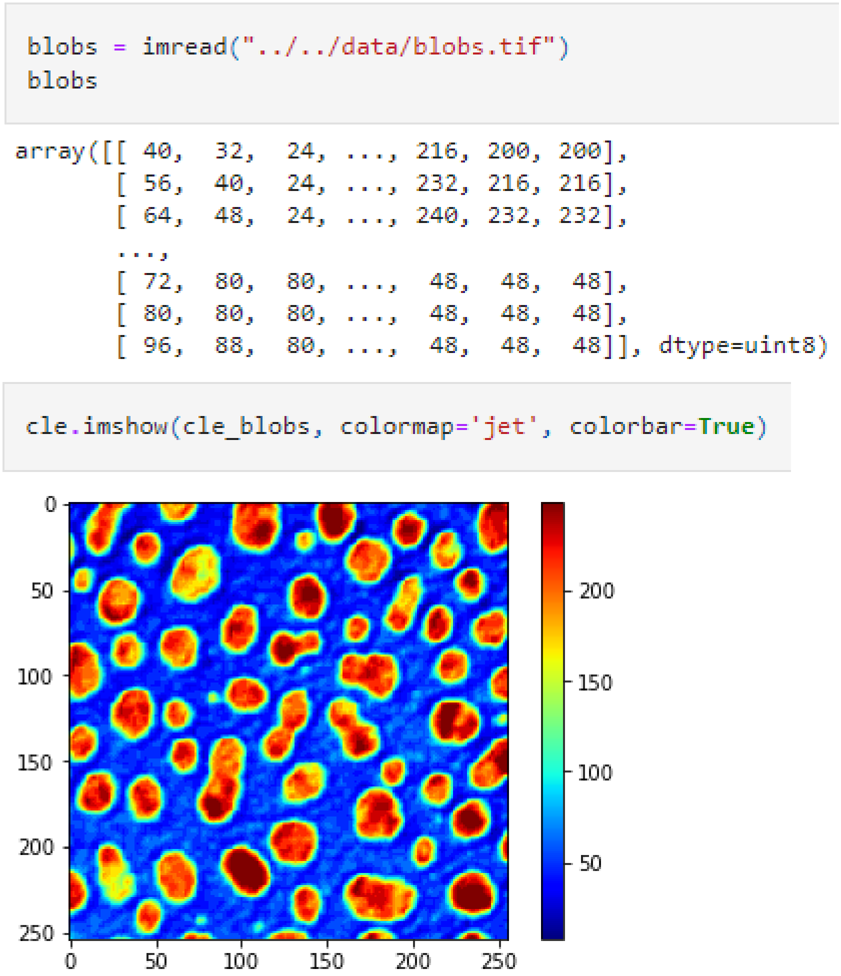
```
labels = cle.voronoi_otsu_labeling(image, spot_sigma=3.5, outline_sigma=1)

cle.imshow(labels, labels=True)
```

https://haesleinhuepf.github.io/BioImageAnalysisNotebooks/15_gpu_acceleration/clesperanto.html

Why are there multiple functions that do the same?

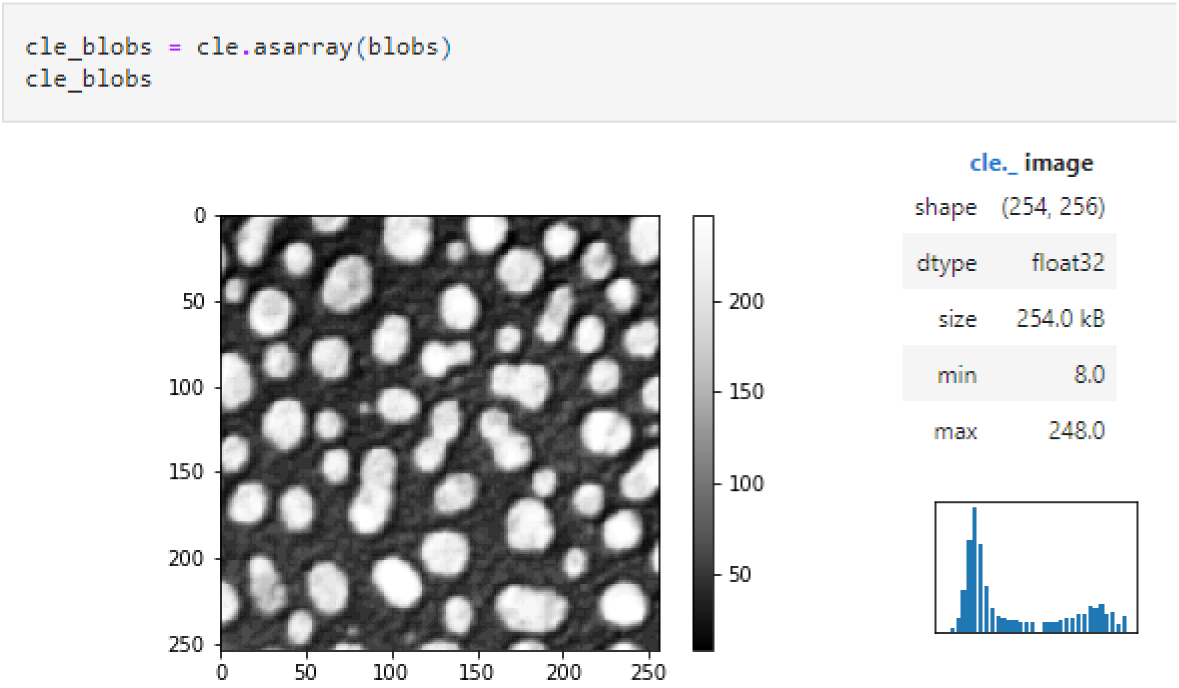
- Hint: The tools we develop (such as clesperanto) can be modified by us. Let us know what you need and we make it happen!



```
blobs = imread("../../data/blobs.tif")
blobs
```

```
array([[ 40,  32,  24, ..., 216, 200, 200],
       [ 56,  40,  24, ..., 232, 216, 216],
       [ 64,  48,  24, ..., 240, 232, 232],
       ...,
       [ 72,  80,  80, ...,  48,  48,  48],
       [ 80,  80,  80, ...,  48,  48,  48],
       [ 96,  88,  80, ...,  48,  48,  48]], dtype=uint8)
```

```
cle.imshow(cle_blobs, colormap='jet', colorbar=True)
```



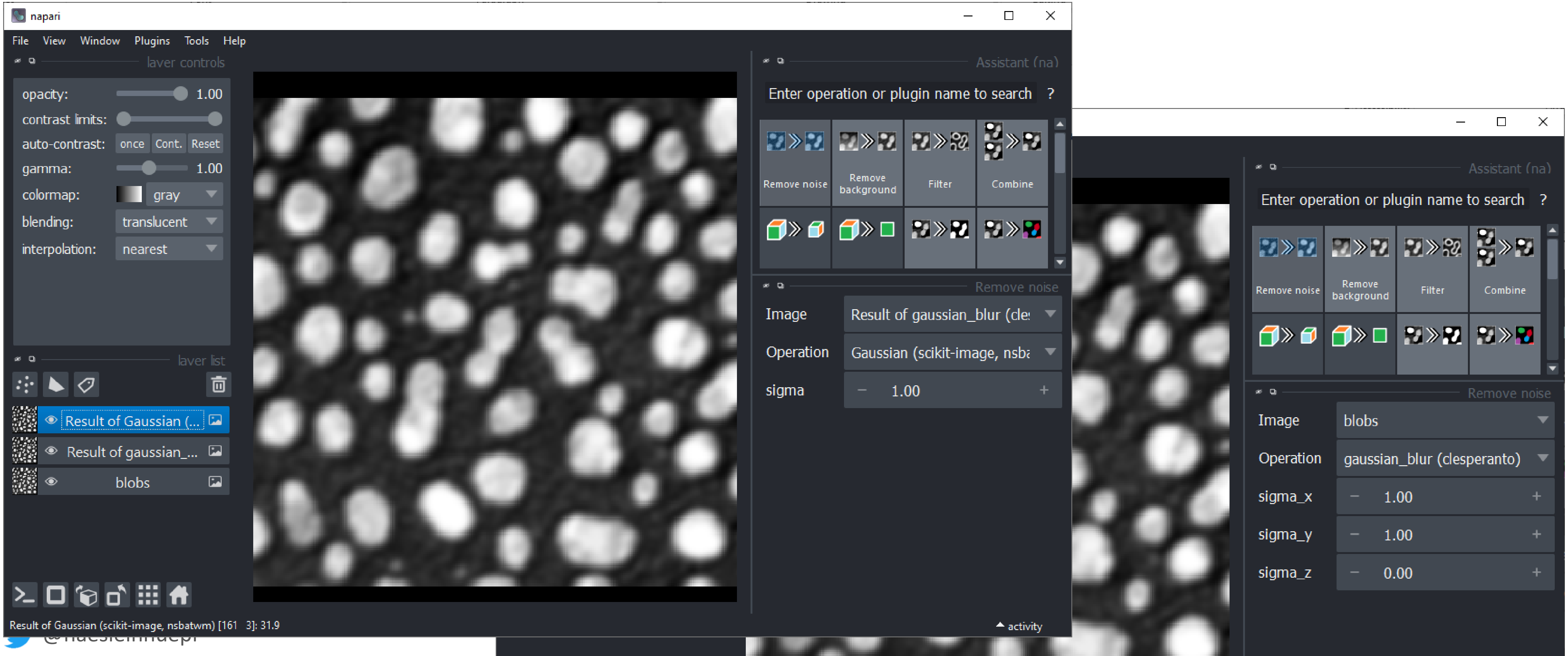
```
cle_blobs = cle.asarray(blobs)
cle_blobs
```

cle._ image	
shape	(254, 256)
dtype	float32
size	254.0 kB
min	8.0
max	248.0

https://github.com/clEsperanto/pyclesperanto_prototype/blob/master/demo/interoperability/jupyter.ipynb

Why are there multiple functions that do the same?

- Example Gaussian blur:
  - Some are fast, some are slow.
  - Some run on all computers, some need a graphics card.



opacity: 1.00
contrast limits:
auto-contrast: once Cont. Reset
gamma: 1.00
colormap: gray
blending: translucent
interpolation: nearest

Result of Gaussian (...
Result of gaussian_...
blobs

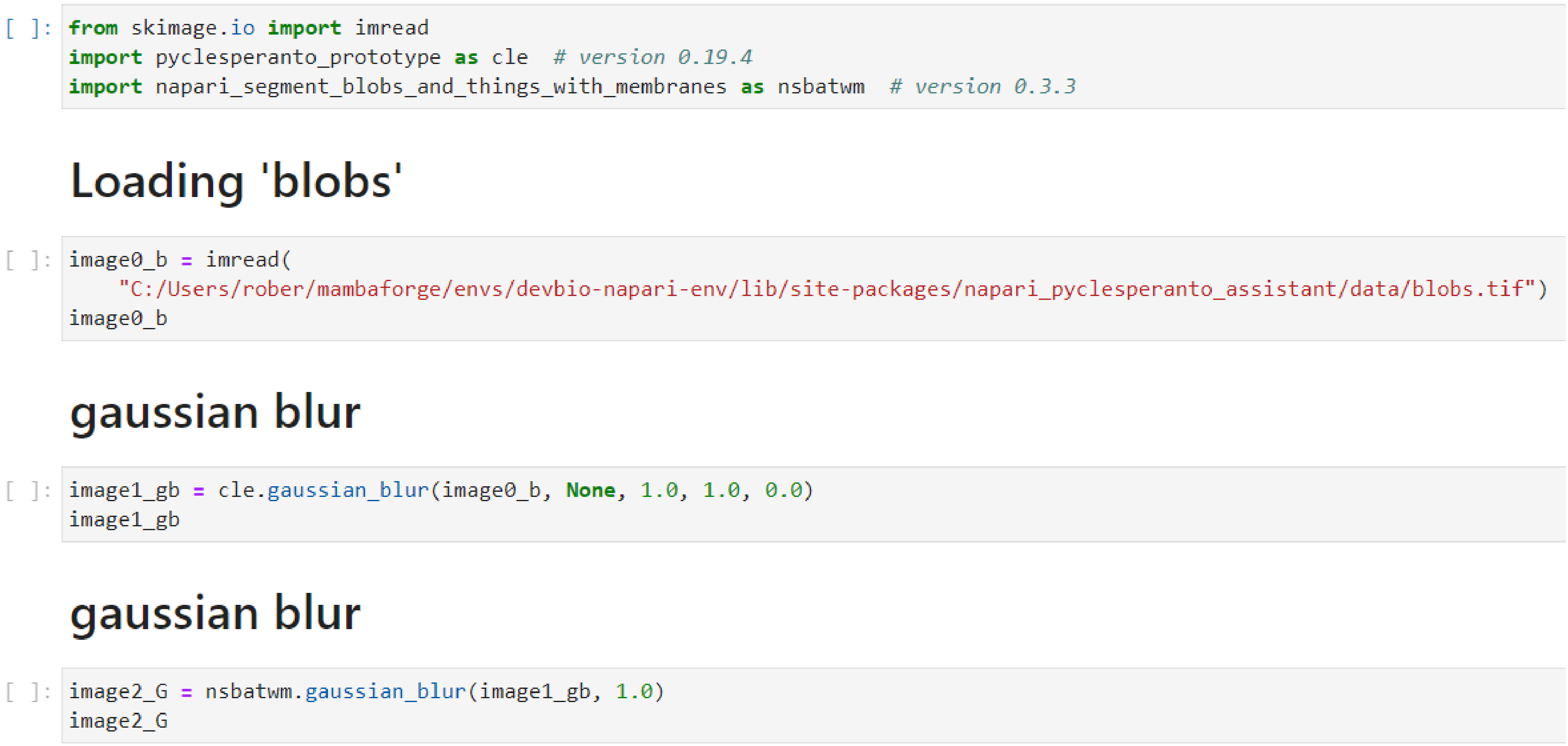
Remove noise
Image: Result of gaussian_blur (cle
Operation: Gaussian (scikit-image, nsba
sigma: 1.00

Result of Gaussian (scikit-image, nsbatwm) [161 3]: 31.9

Remove noise
Image: blobs
Operation: gaussian_blur (clesperanto)
sigma_x: 1.00
sigma_y: 1.00
sigma_z: 0.00

Why are there multiple functions that do the same?

- Read the documentation of the different variants and make your choice. Try to stick to similar operations between projects.
- Also: Generate notebooks



```
[ ]: from skimage.io import imread
     import pyclesperanto_prototype as cle  # version 0.19.4
     import napari_segment_blobs_and_things_with_membranes as nsbatwm  # version 0.3.3
```

Loading 'blobs'

```
[ ]: image0_b = imread(
         "C:/Users/rober/mambaforge/envs/devbio-napari-env/lib/site-packages/napari_pyclesperanto_assistant/data/blobs.tif")
     image0_b
```

gaussian blur

```
[ ]: image1_gb = cle.gaussian_blur(image0_b, None, 1.0, 1.0, 0.0)
     image1_gb
```

gaussian blur

```
[ ]: image2_G = nsbatwm.gaussian_blur(image1_gb, 1.0)
     image2_G
```