PoL Physics of Life TU Dresden

CENTER FOR SYSTEMS BIOLOGY DRESDEN

Image filtering

Robert Haase

With material from

Mauricio Rocha Martins, Norden lab, MPI CBG

Dominic Waithe, Oxford University

Nuno P Martins, IGC Lisbon

Paulo Aguiar, INEB, Porto

Sebastian Tosi, IRB Barcelona

neubias network of european bioimage analysts

Benoit Lombardot, Scientific Computing Facility, MPI CBG

Alex Bird, Dan White, MPI CBG

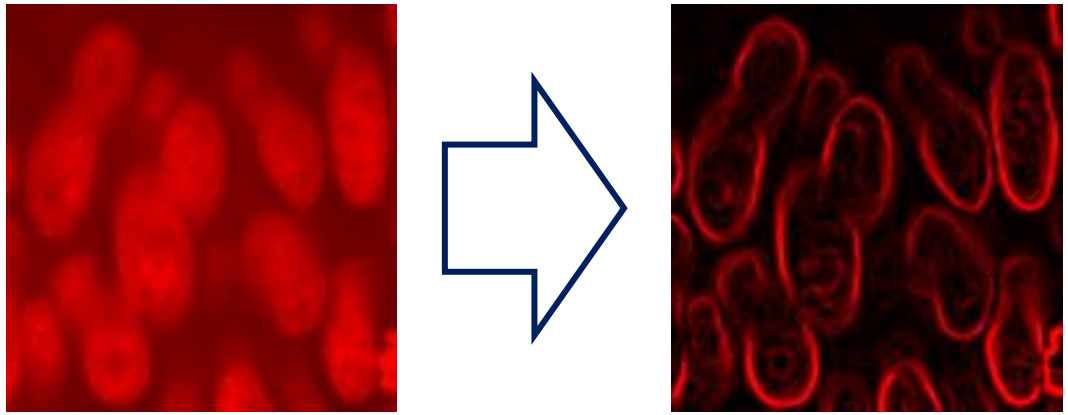
April 2021

@haesleinhuepf

Lecture overview

Today

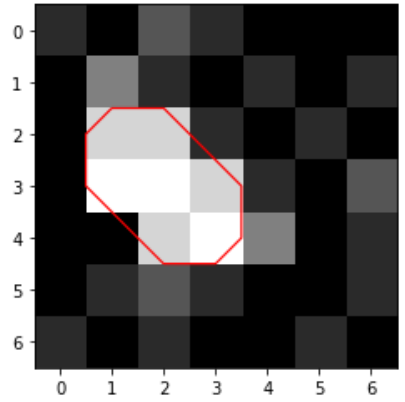
- Filters
- Image math



- Image filtering with Fiji
- Image filtering with Python
- Exercises

```
import matplotlib.pyplot as plt

# create a new plot
fig, axes = plt.subplots(1,1)

# add two images
axes.imshow(image, cmap=plt.cm.gray)
axes.contour(binary_image, [0.5], linewidths=1.2, colors='r')
```

```
<matplotlib.contour.QuadContourSet at 0x206dc159310>
```



Filters

- An image processing filter is an operation on an image.
- It takes an image and produces a new image out of it.
- Filters change pixel values.

- There is no “best” filter. Which filter fits your needs, depends on the context.
- Filters do not do magic. They can not make things visible which are not in the image.

- Application examples
  - Noise-reduction
  - Artefact-removal
  - Contrast enhancement
  - Correct uneven illumination

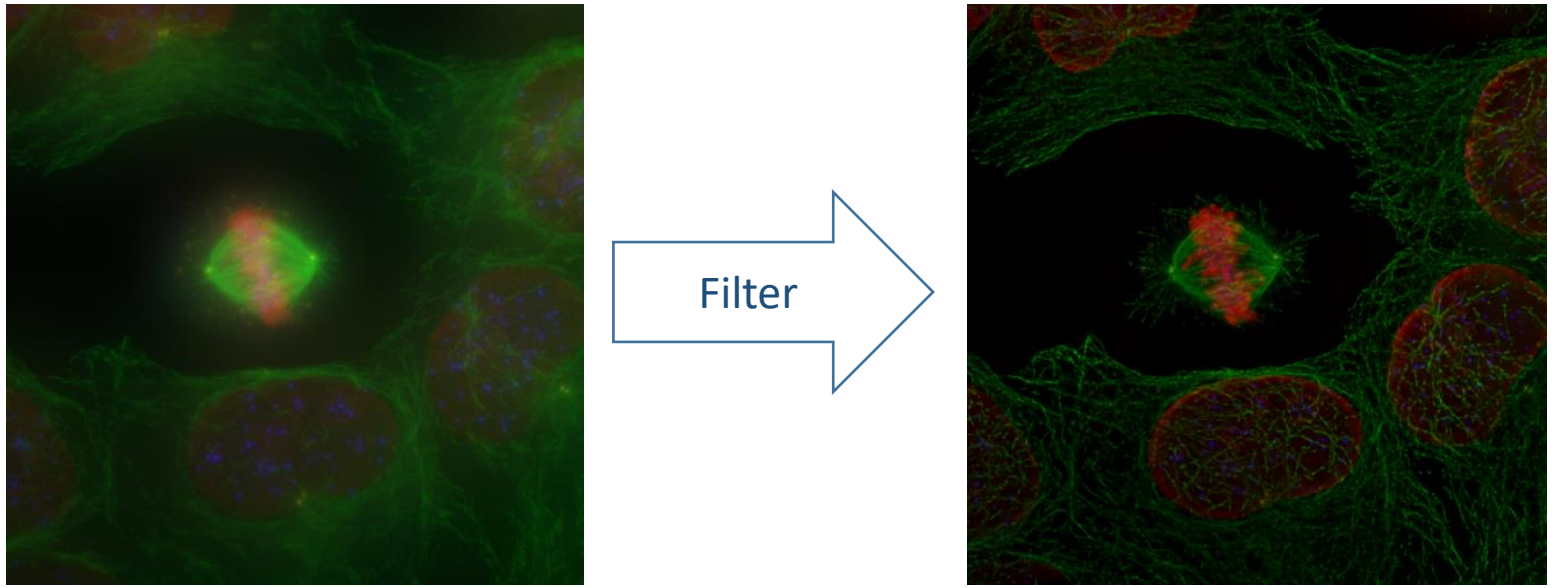


Image source: Alex Bird / Dan White MPI CBG

Effects harming image quality

- Noise is a general term for unwanted modifications that a signal may suffer during capture, storage, transmission, processing, or conversion.
- In microscopy, image quality suffers from
  - shot noise: Statistical variation of the photons arriving at the camera
  - dark noise: Statistical variation of how many electrons are generated if a photon arrives in a camera pixel (temperature dependent).
  - read out noise: introduced by the electronics, especially the pixel and the analog-digital-converter
  - Physical/optical effects: aberrations, defocus
  - Biological/physiological/structural effects: motion, diffusion

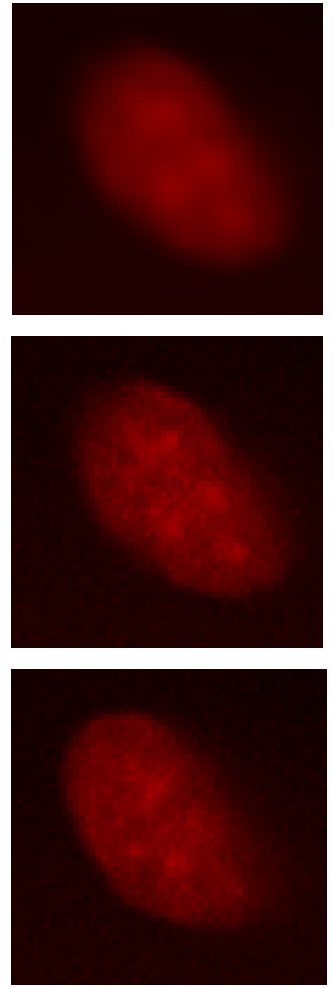


Image source: Mauricio Rocha Martins (Norden/Myers lab, MPI CBG)

Noise

- When dealing with noise in microscopic image processing, we mostly mean the noise visible in the images.

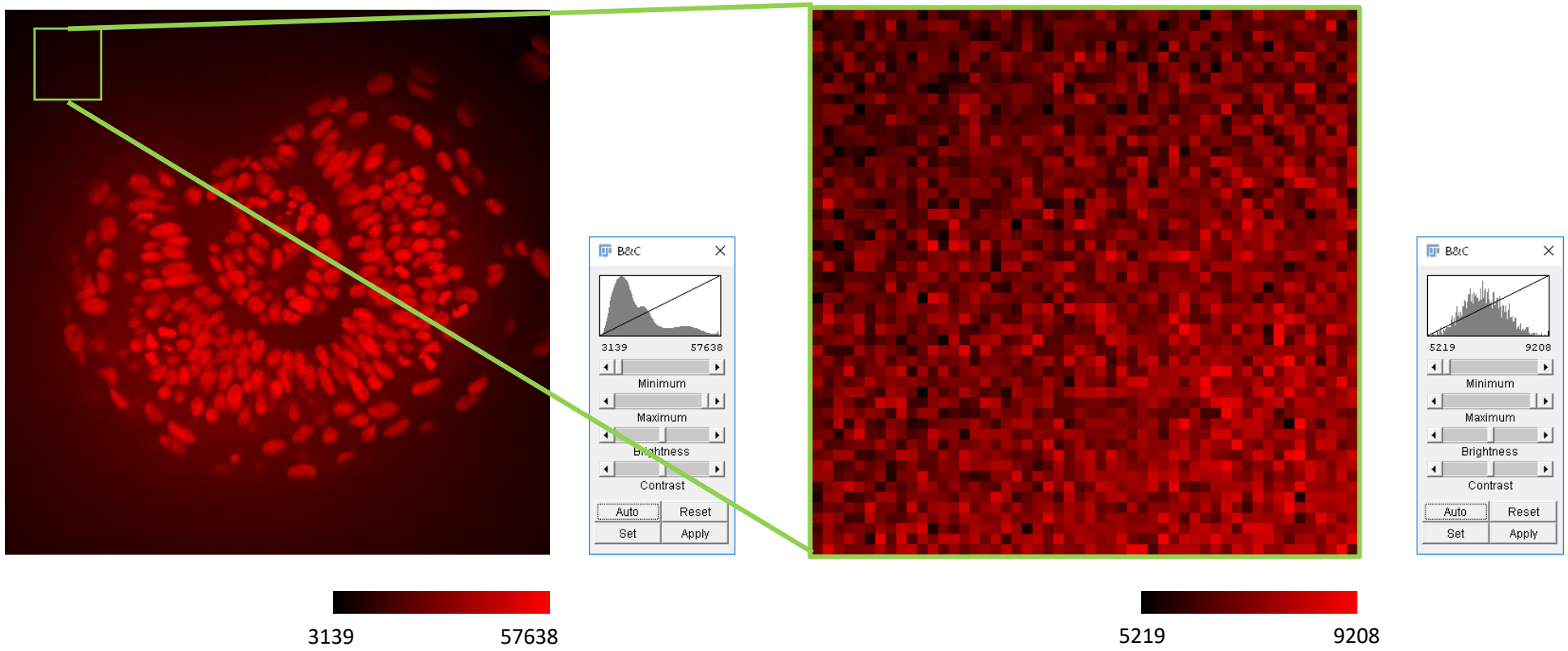


Image source: Mauricio Rocha Martins (Norden/Myers lab, MPI CBG)

Image correction / noise removal

- We need to remove the noise to help the computer *interpreting* the image

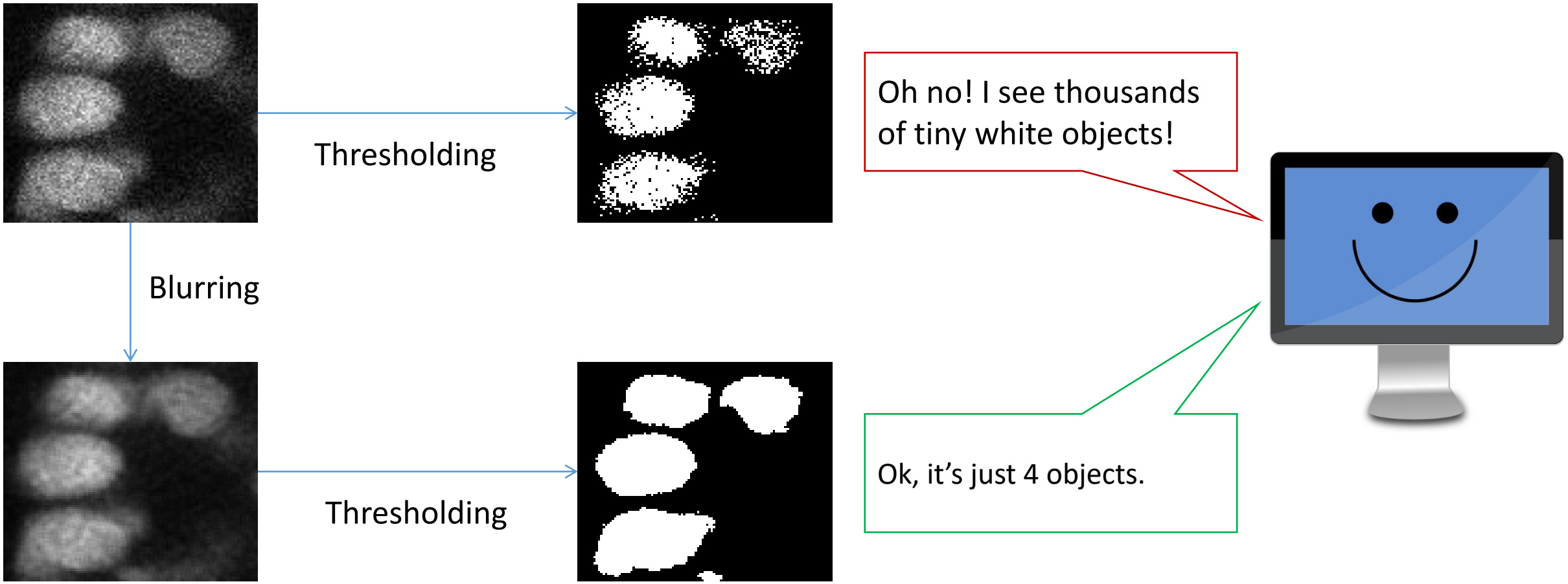
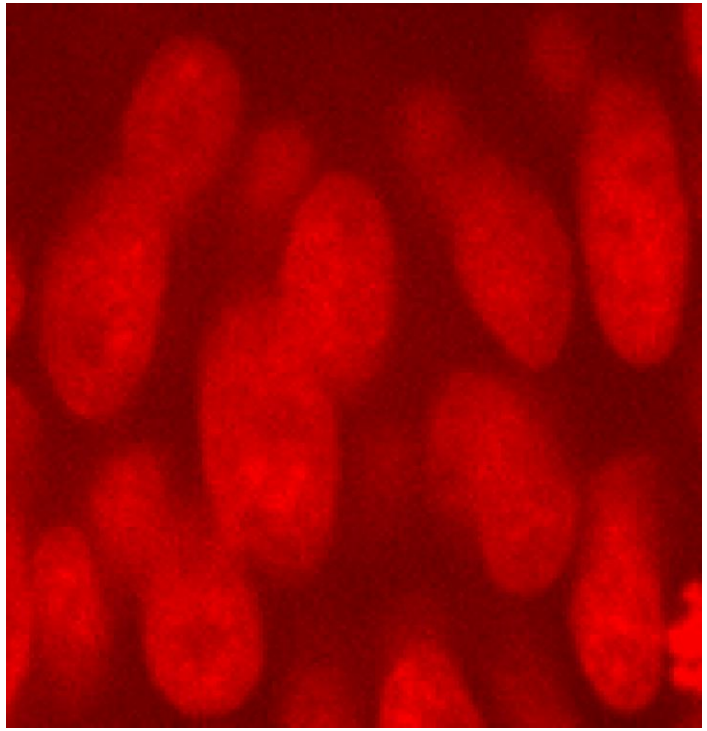
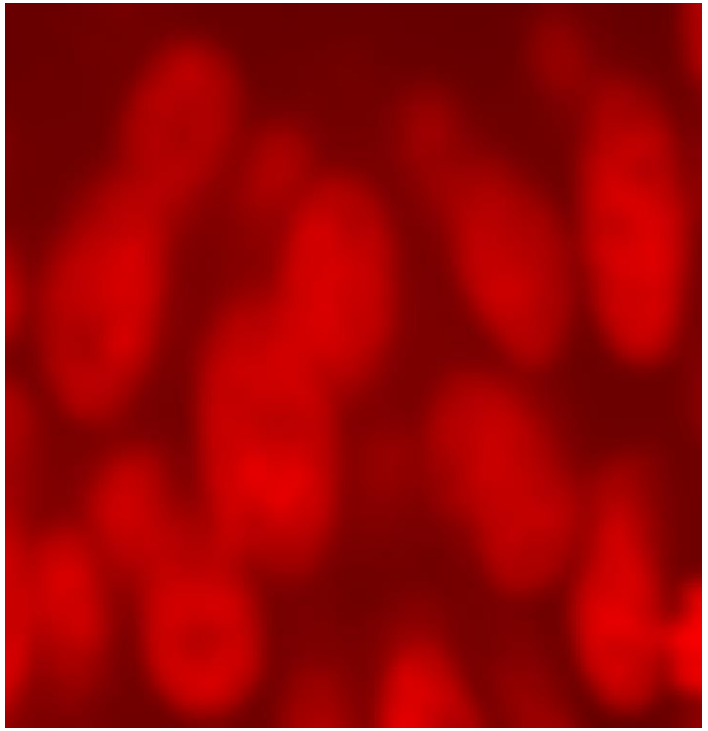


Image correction / noise removal

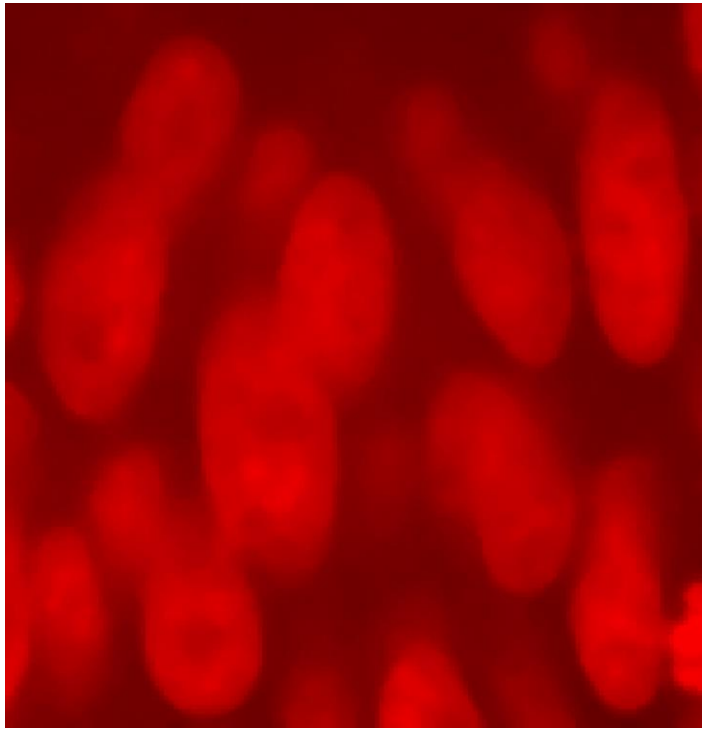
- Noise removal using *filters*



Original image



Gaussian blur filtered

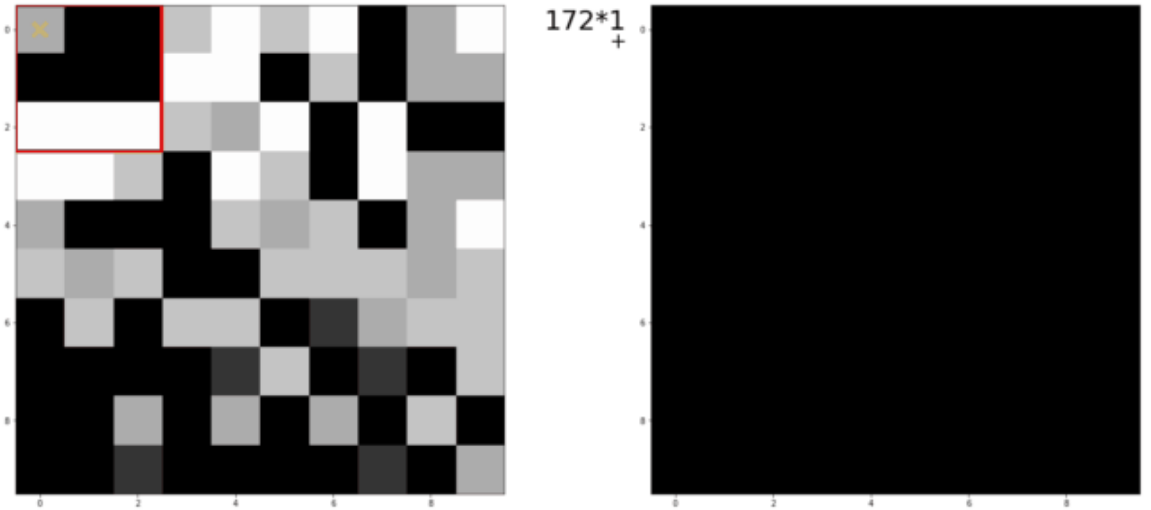


Median filtered

Image source: Mauricio Rocha Martins (Norden/Myers lab, MPI CBG)

Linear Filters

- Linear filters replace each pixel value with a linear combination of surrounding pixels
  - Basically, linear filtering is a Convolution
  - It needs a kernel (weight template)
  - Result: new image where each pixel is replaced by the weighted sum of pixel values in the neighbourhood.



- Kernels are matrices describing a linear filter

Mean filter, 3x3 kernel

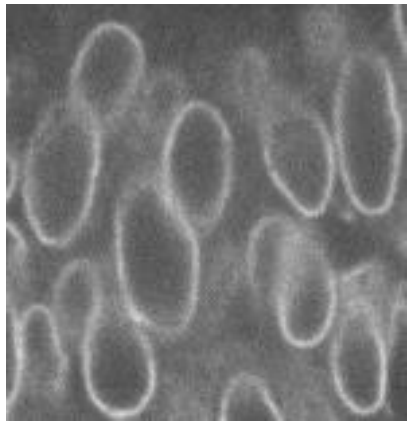
$$\begin{bmatrix} 1/9 & 1/9 & 1/9 \\ 1/9 & 1/9 & 1/9 \\ 1/9 & 1/9 & 1/9 \end{bmatrix}$$

Animation source: Dominic Waithe, Oxford University
https://github.com/dwaithe/generalMacros/tree/master/convolution_ani

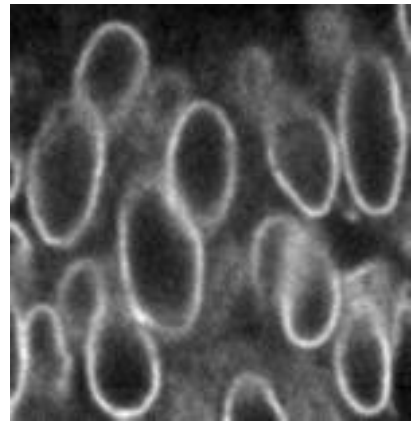
Linear filters

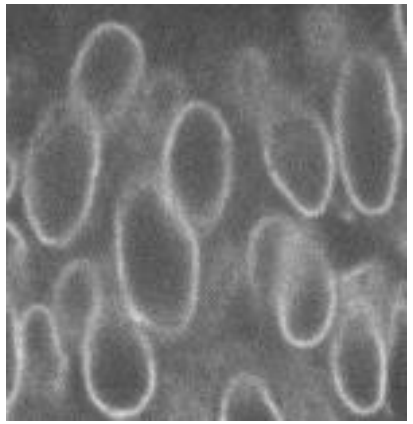
- Terminology:
  - “We convolve an image with a kernel.”
  - Convolution operator: *

- Examples
  - Mean
  - Gaussian blur
  - Sobel-operator
  - Laplace-filter

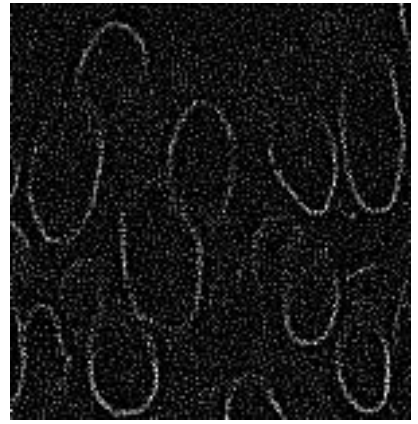


$$*\begin{bmatrix} 1 & 1 & 1 \\ 1 & 8 & 1 \\ 1 & 1 & 1 \end{bmatrix}=$$





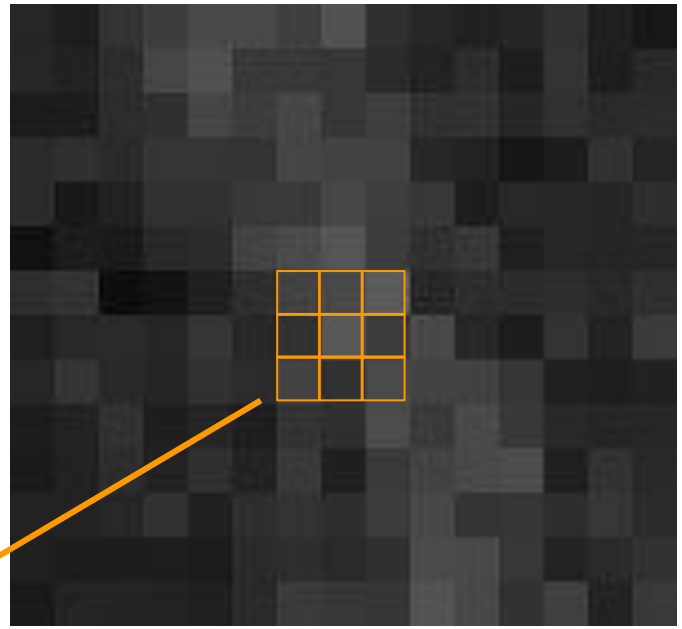
$$*\begin{bmatrix} -1 & -1 & -1 \\ -1 & 8 & -1 \\ -1 & -1 & -1 \end{bmatrix}=$$

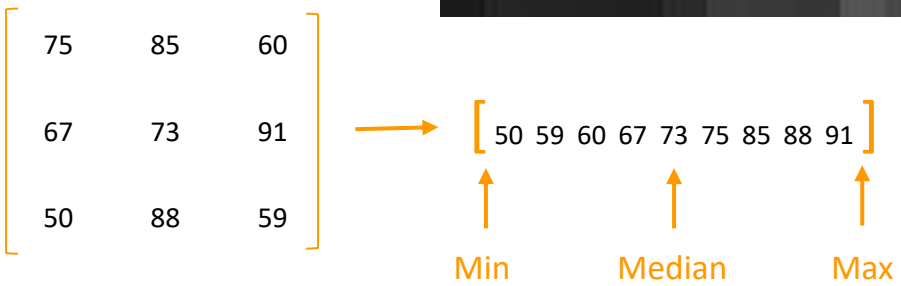


Nonlinear Filters

- Non linear filters also replace pixel value inside as rolling window but in a non linear function.

- Examples: order statistics filters
  - Min
  - Median
  - Max
  - Variance
  - Standard deviation





PoL
Physics of Life
TU Dresden

CENTER FOR
SYSTEMS BIOLOGY
DRESDEN

Image math

Robert Haase

With material from

Mauricio Rocha Martins, Norden lab, MPI CBG

April 2021

@haesleinhuepf

Background removal

- Differentiating objects is easier if their background intensity is equal.

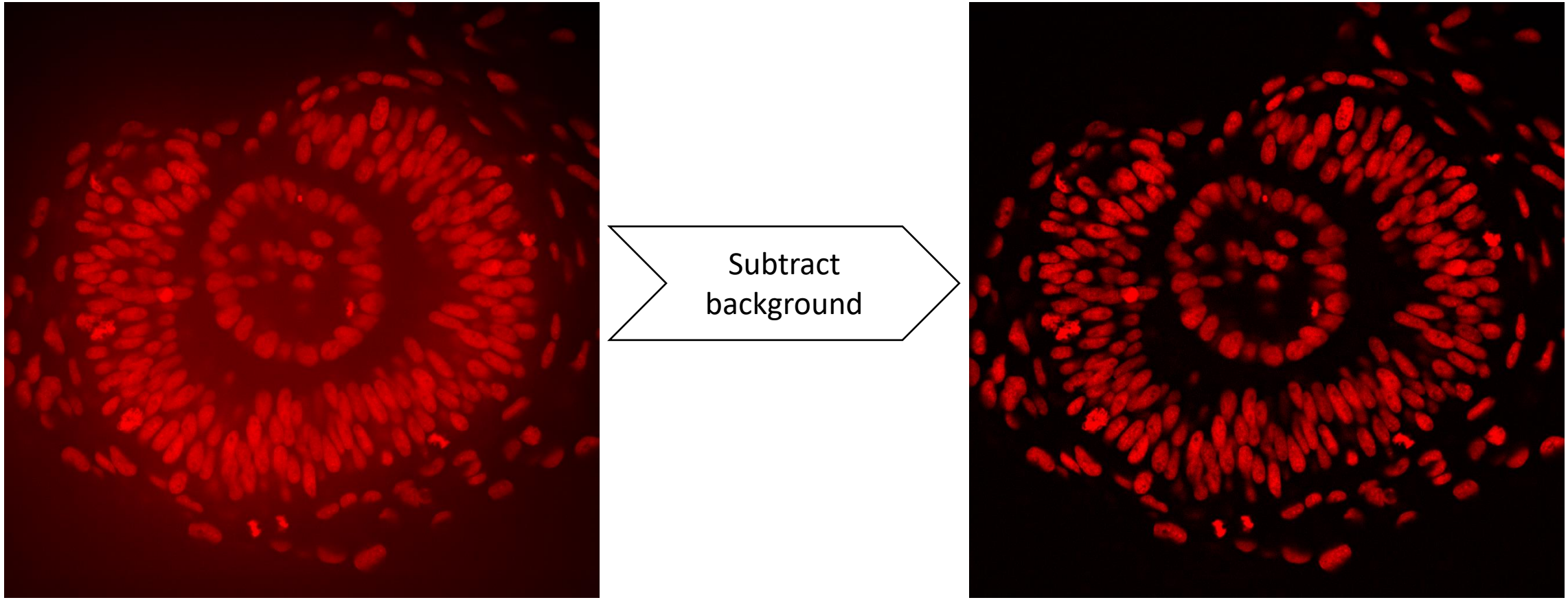
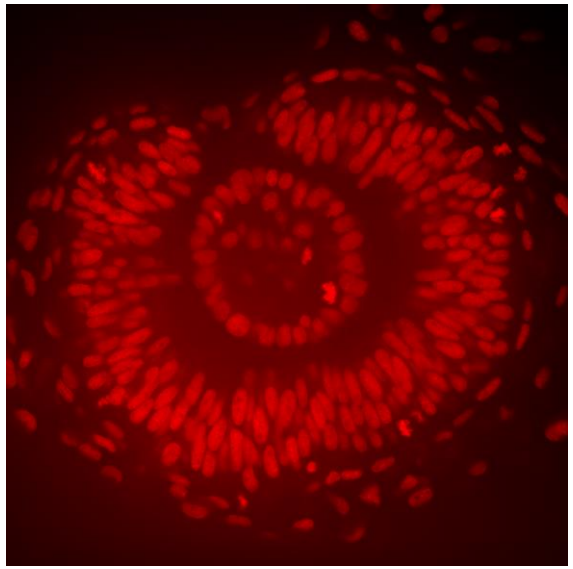


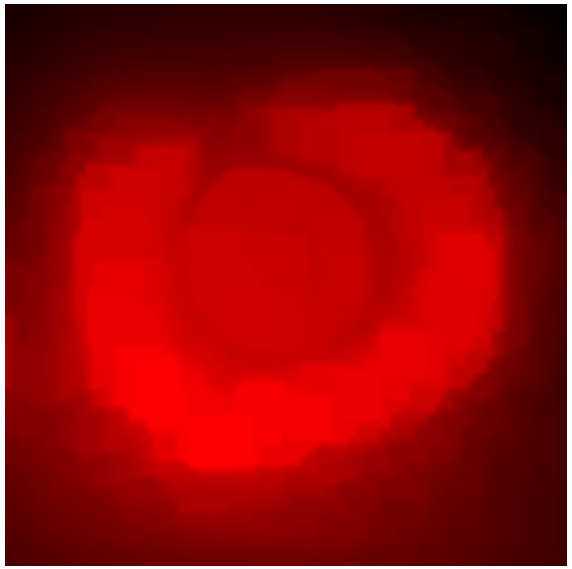
Image source: Mauricio Rocha Martins (Norden/Myers lab, MPI CBG)

Background removal

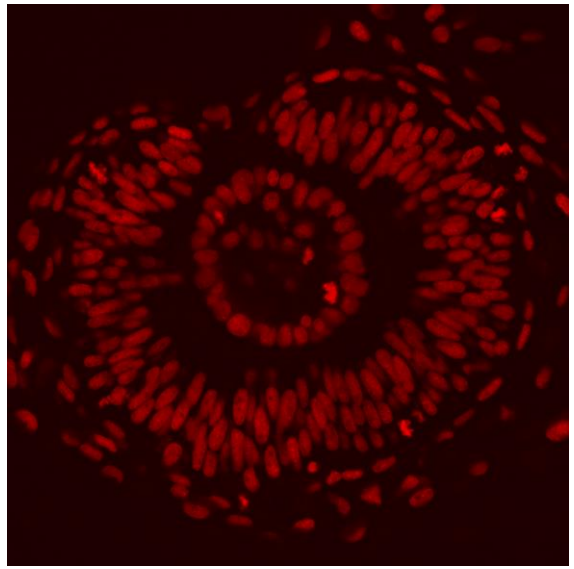
- Depending on the effect we want to correct for, it might make sense to divide an image by its background.



Original image



Extracted background



Subtracted background

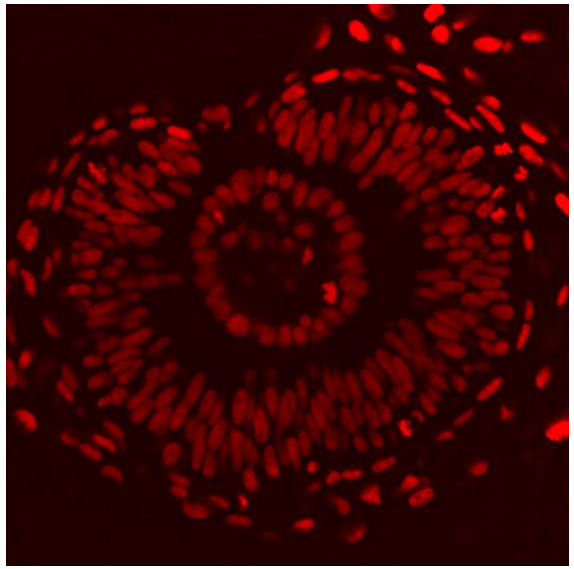
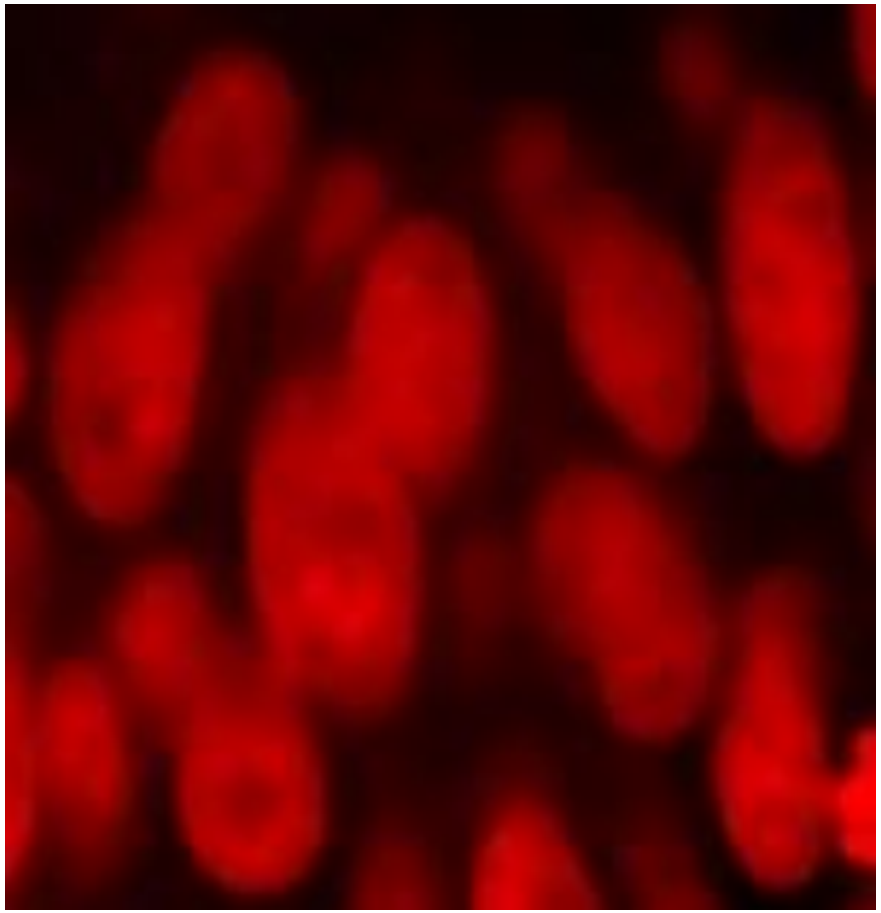


Image divided by background

Image source: Mauricio Rocha Martins (Norden/Myers lab, MPI CBG)

Mathematical operations on images

- It might be hard to differentiate objects from single images…



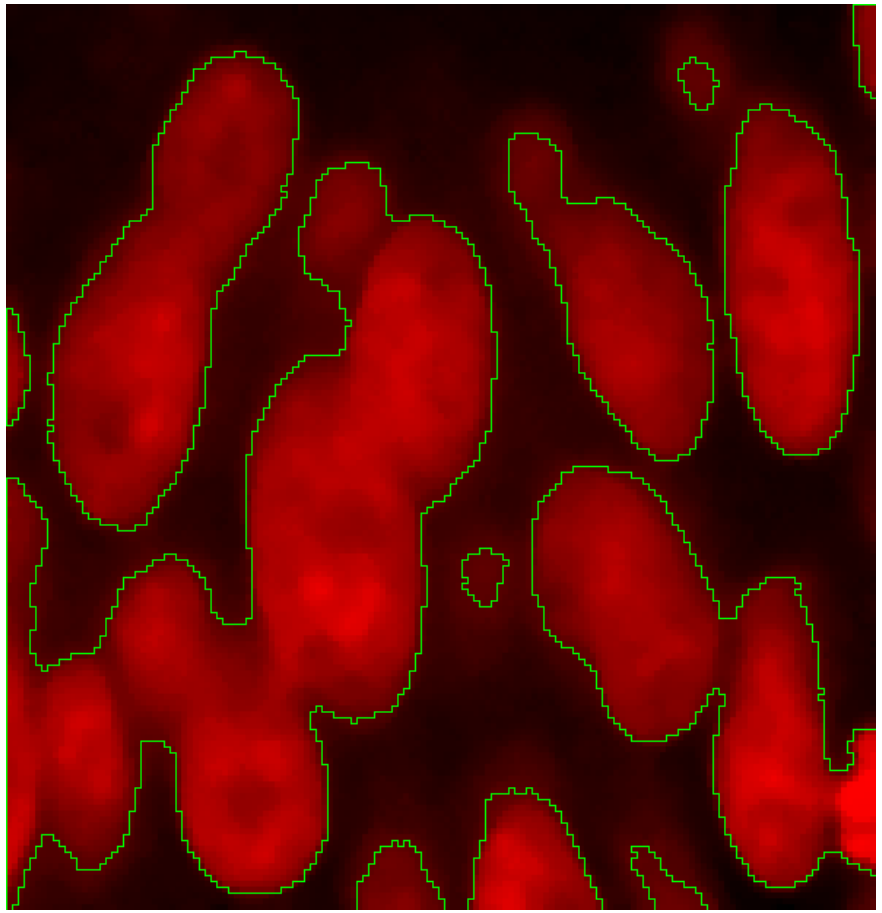


Image source: Mauricio Rocha Martins (Norden/Myers lab, MPI CBG)

Mathematical operations on images

- Segmentation of nuclei can be improved by subtracting a channel visualizing nuclei envelope from the nuclei channel.

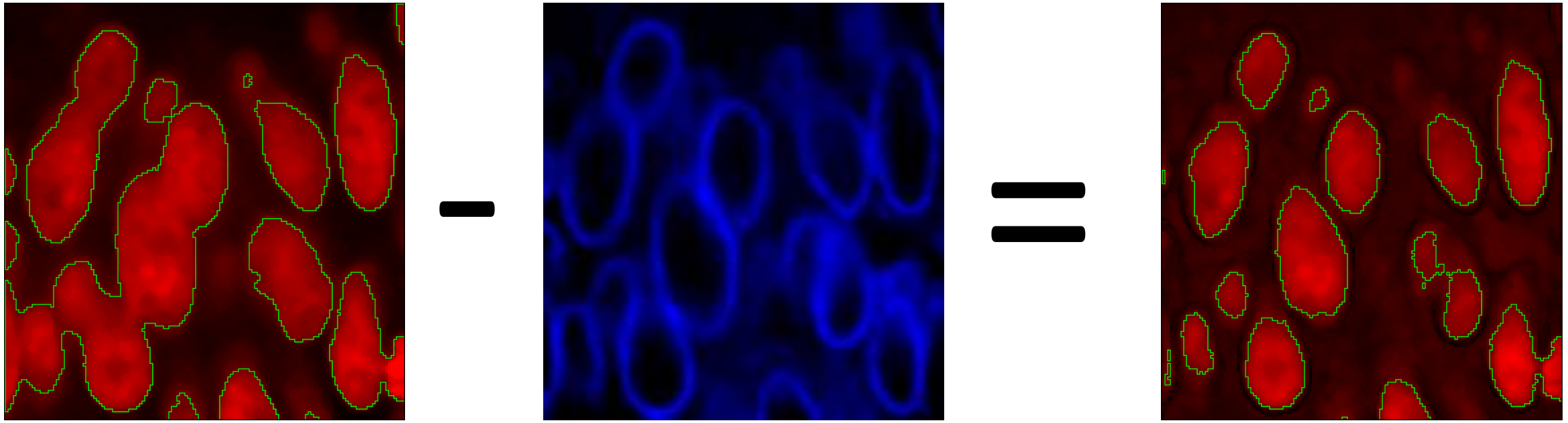
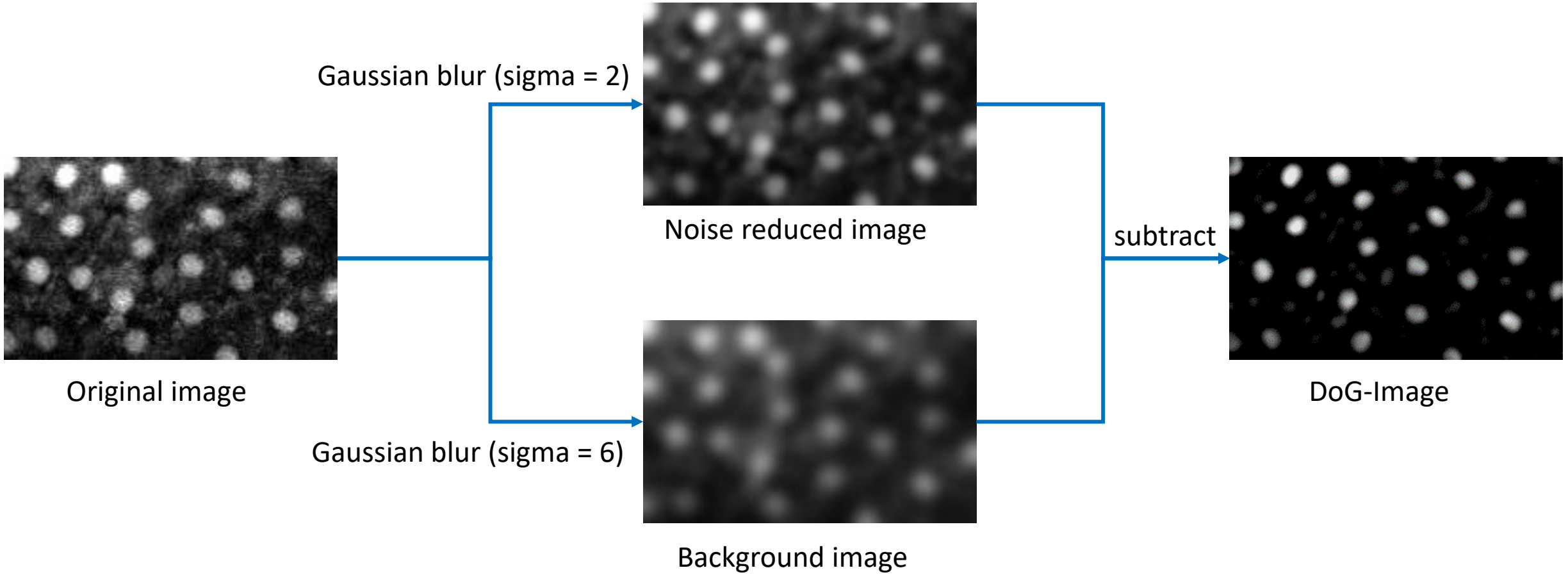


Image source: Mauricio Rocha Martins (Norden/Myers lab, MPI CBG)

Difference-of-Gaussian (DoG)

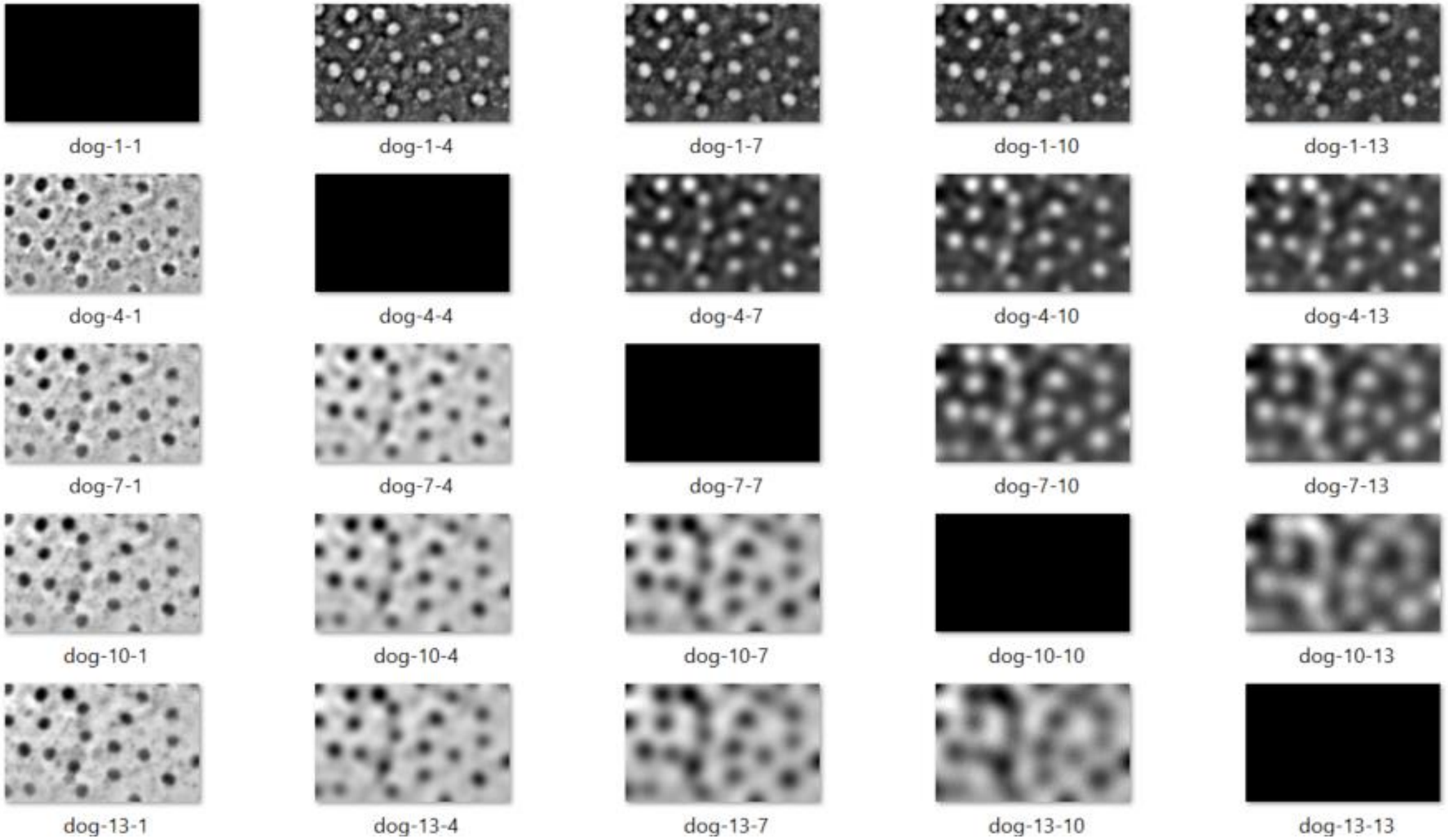
- Improve image in order to detect bright objects.



Difference-of-Gaussian (DoG)

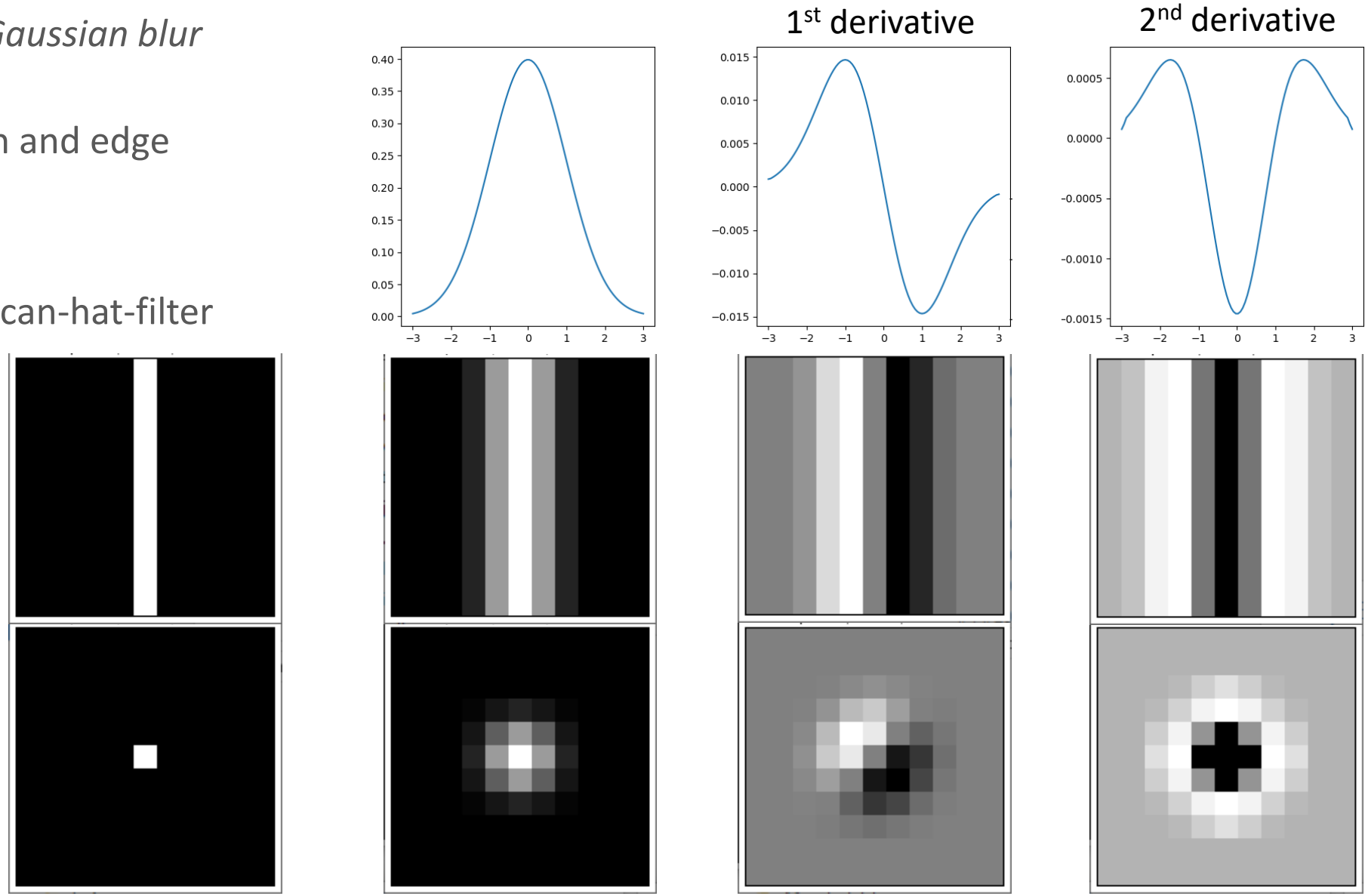
PoL Physics of Life TU Dresden

- Example DoG images



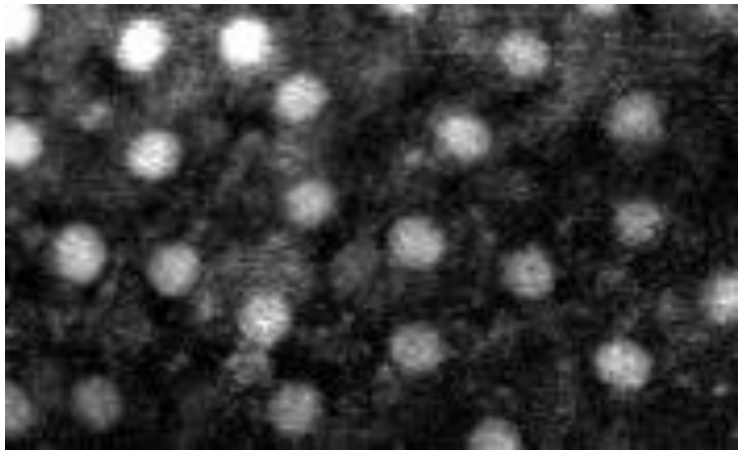
Laplace-filter

- *Second derivative of a Gaussian blur filter*
- Used for edge-detection and edge enhancement
- Also known as the Mexican-hat-filter

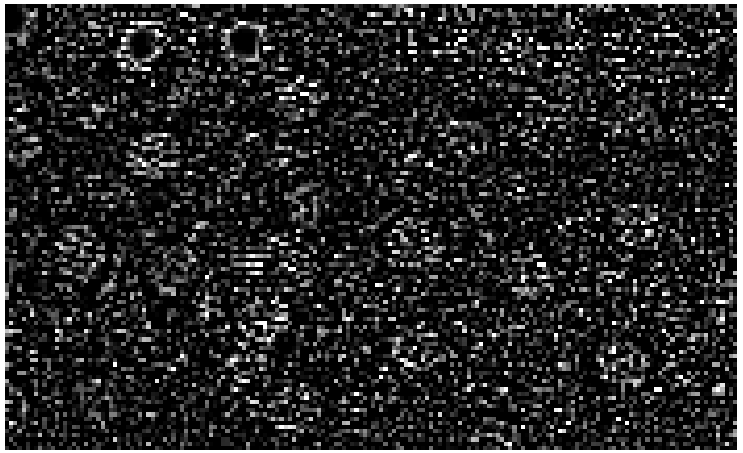


Laplacian-of-Gaussian (LoG)

Laplace filter



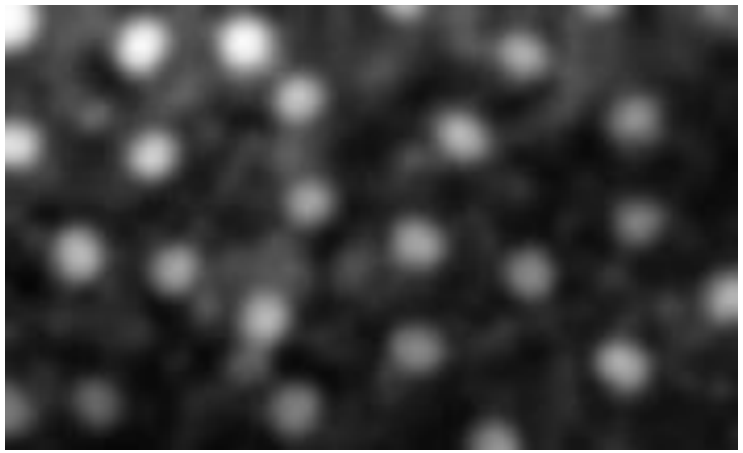
$$*\begin{bmatrix} -1 & -1 & -1 \\ -1 & 8 & -1 \\ -1 & -1 & -1 \end{bmatrix} =$$



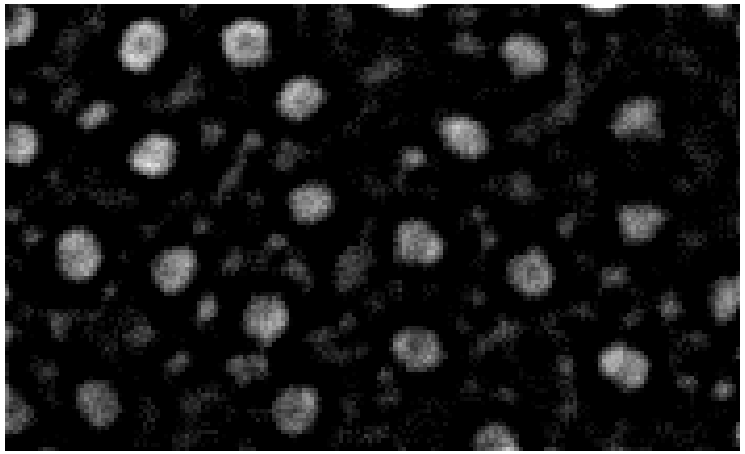
Laplace filtered image

Gaussian blur

Laplacian of Gaussian filter



$$*\begin{bmatrix} -1 & -1 & -1 \\ -1 & 8 & -1 \\ -1 & -1 & -1 \end{bmatrix} =$$



LoG image

PoL Physics of Life TU Dresden

CENTER FOR SYSTEMS BIOLOGY DRESDEN

Image Filtering in Fiji

Robert Haase

With material from

Mauricio Rocha Martins, Norden lab, MPI CBG

Dominic Waithe, Oxford University

Nuno P Martins, IGC Lisbon

Paulo Aguiar, INEB, Porto

Sebastian Tosi, IRB Barcelona

neubias network of european bioimage analysts

Benoit Lombardot, Scientific Computing Facility, MPI CBG

Alex Bird, Dan White, MPI CBG

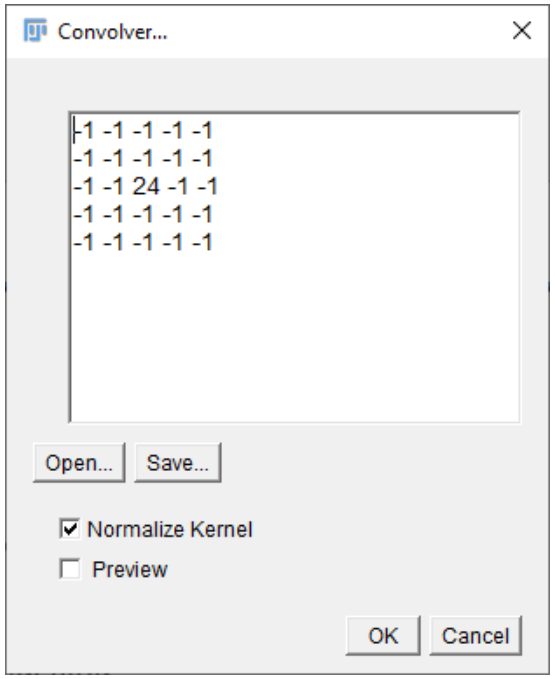
April 2020

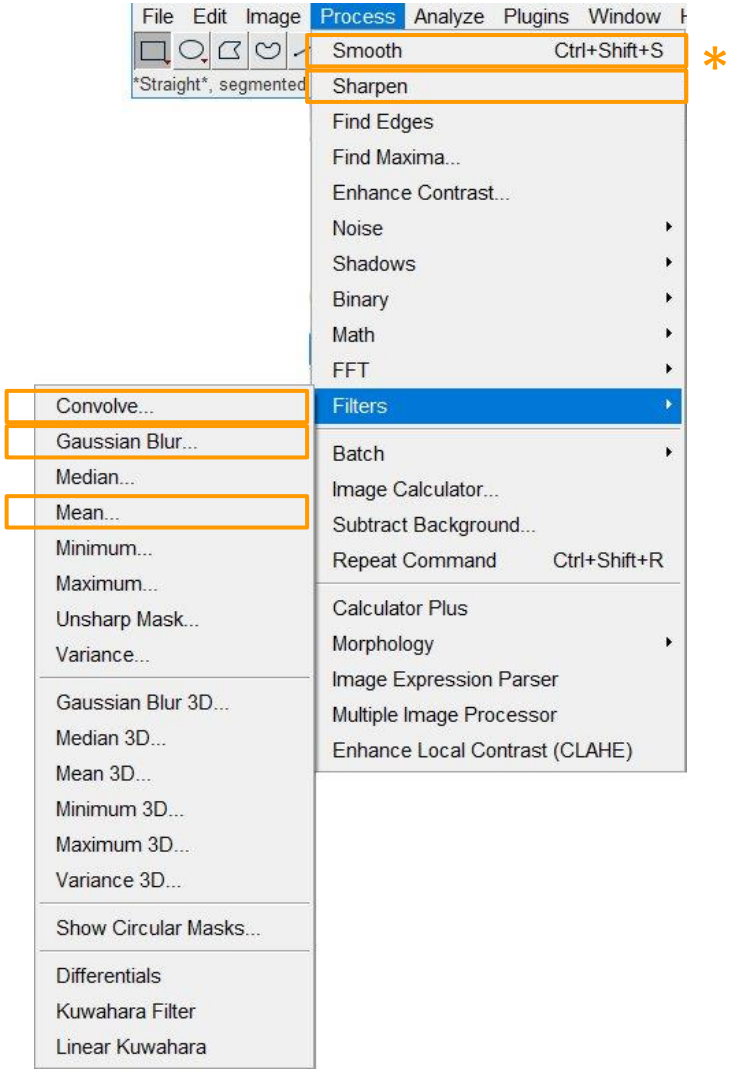
@haesleinhuepf

Linear Filters

Linear filters:

- Convolve…
  - Allows us to specify our kernel

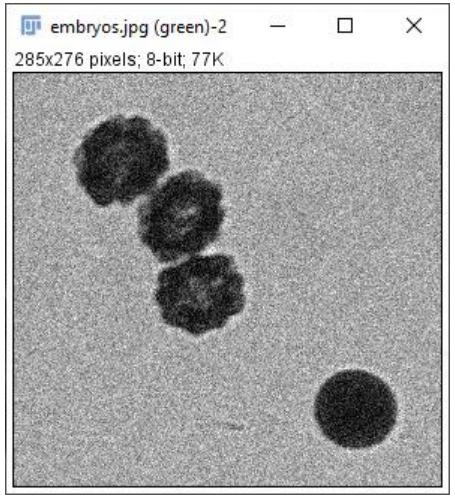


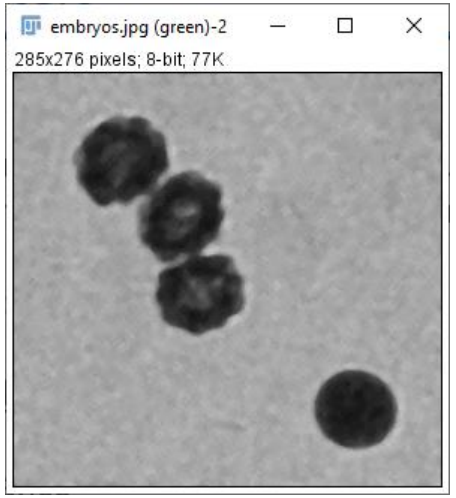


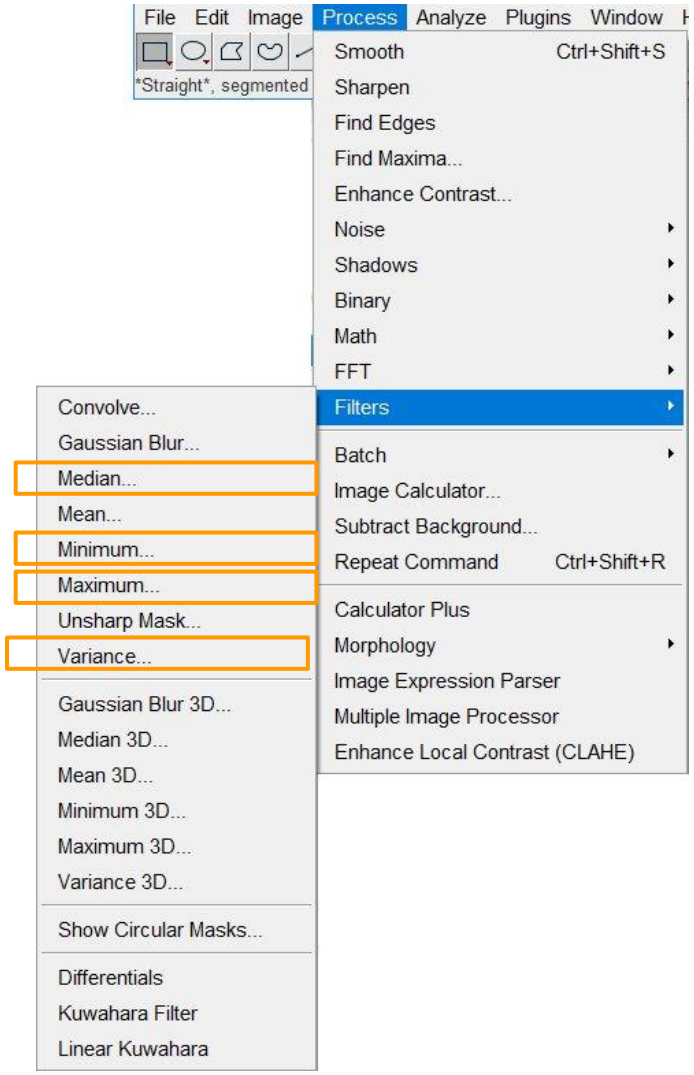
- Gaussian blur
- Mean
  - * Smooth is a 3x3 Mean filter

Nonlinear Filters

- Non linear filters also replace pixel value inside as rolling window but in a non linear function.
- Examples:
  - Min
  - Median
  - Max
  - Variance



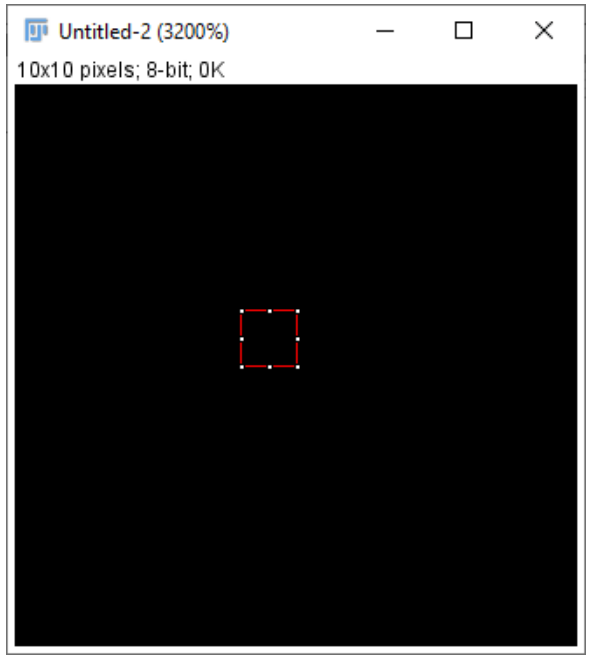


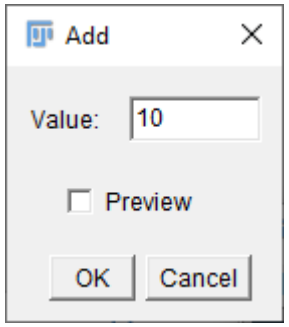


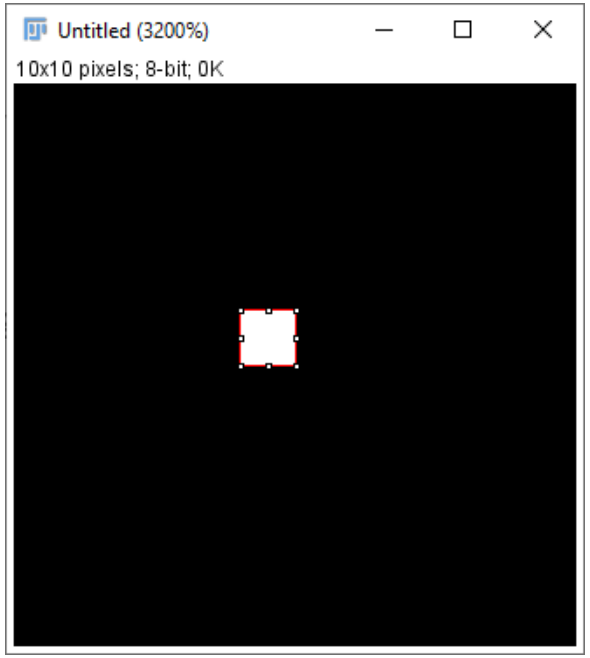
Image/pixel math

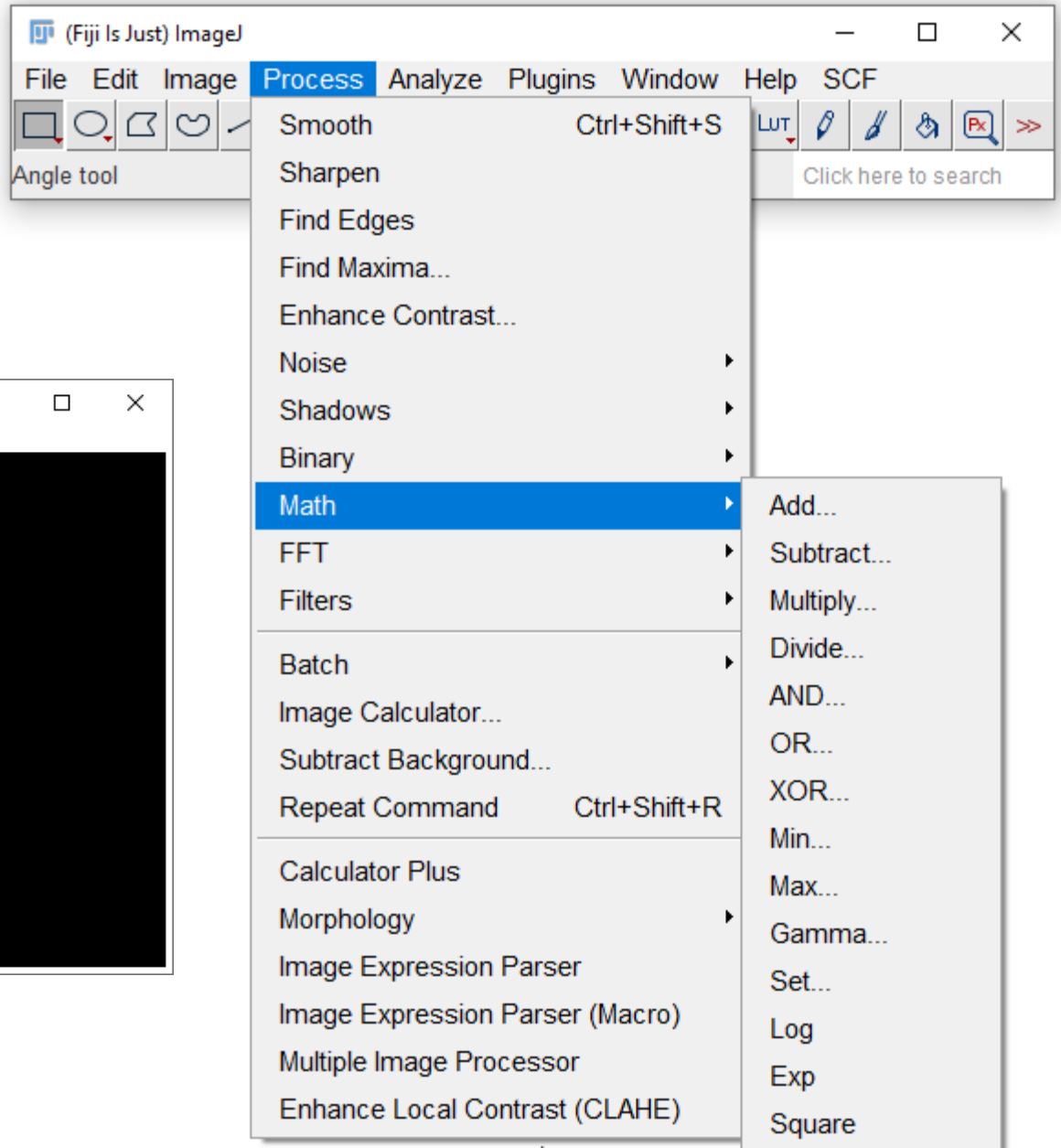
- All basic math operations for pixels are available in ImageJ/Fiji.

- If you want to simulate a binary image: Select some black pixels and add 255 to make them white.



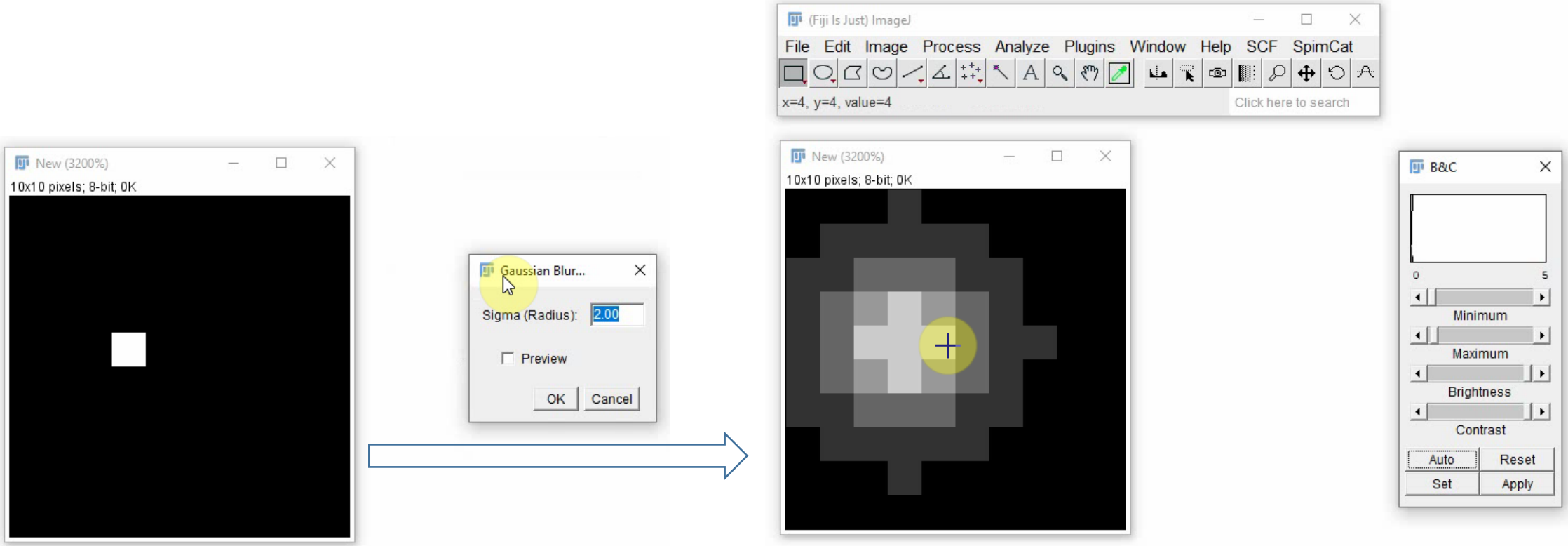






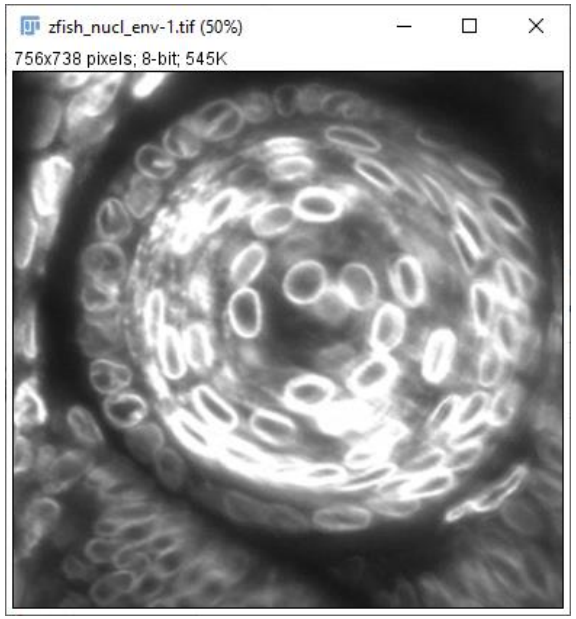
Hint: Image/pixel math

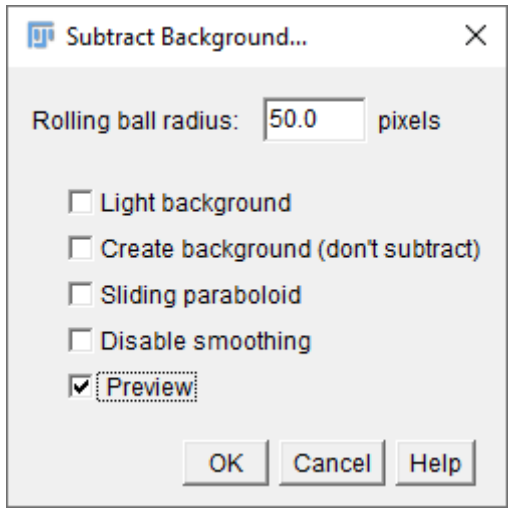
- Use images with single pixels > 0 and apply filters to understand what the filters do

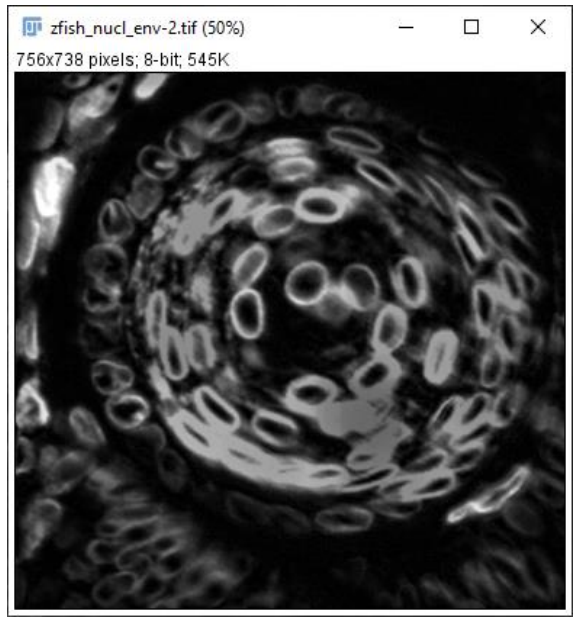


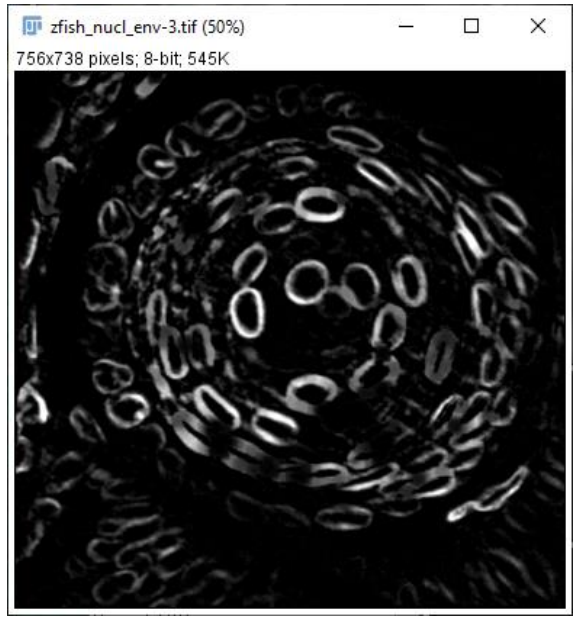
Background removal

- Process > Subtract background…
  - The rolling ball radius should be a bit larger than the objects you are analyzing.
  - Try the preview checkbox while changing parameters!



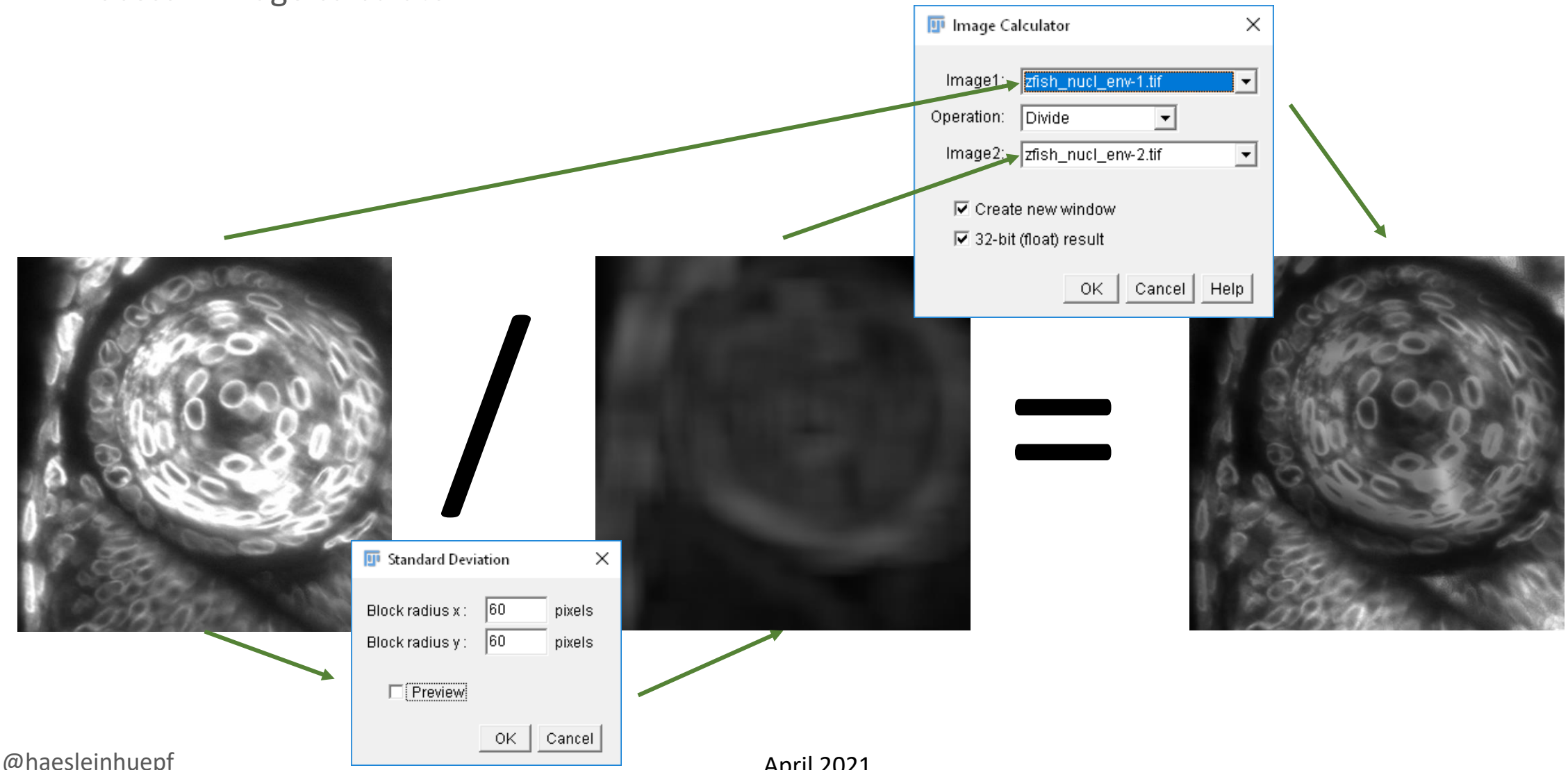






Background removal

- Plugins > Integral image filters > Standard deviation
- Process > Image calculator



Mathematical operations on images

- Use the *Image Calculator* to subtract, add, multiply or divide images.

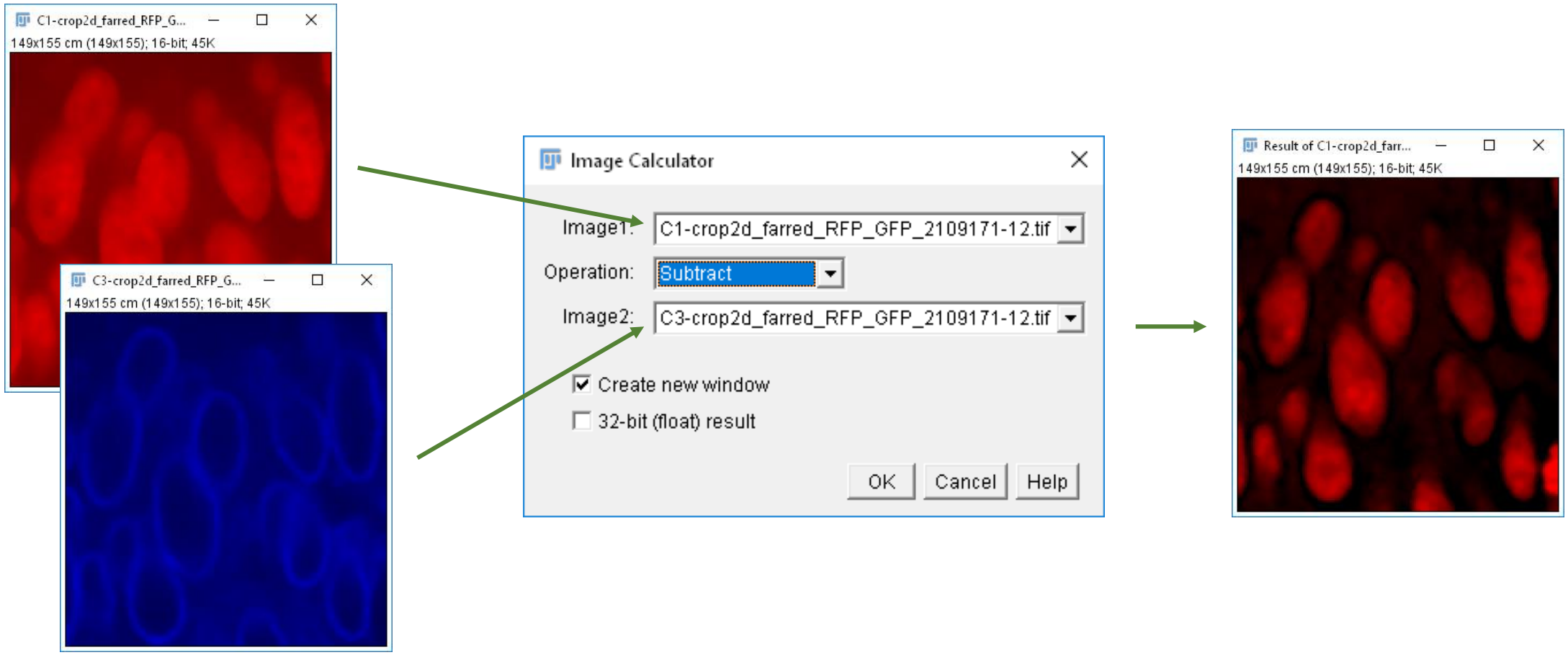


Image source: Mauricio Rocha Martins (Norden/Myers lab, MPI CBG)

PoL
Physics of Life
TU Dresden

CENTER FOR
SYSTEMS BIOLOGY
DRESDEN

Image Filtering in Python using scikit-image

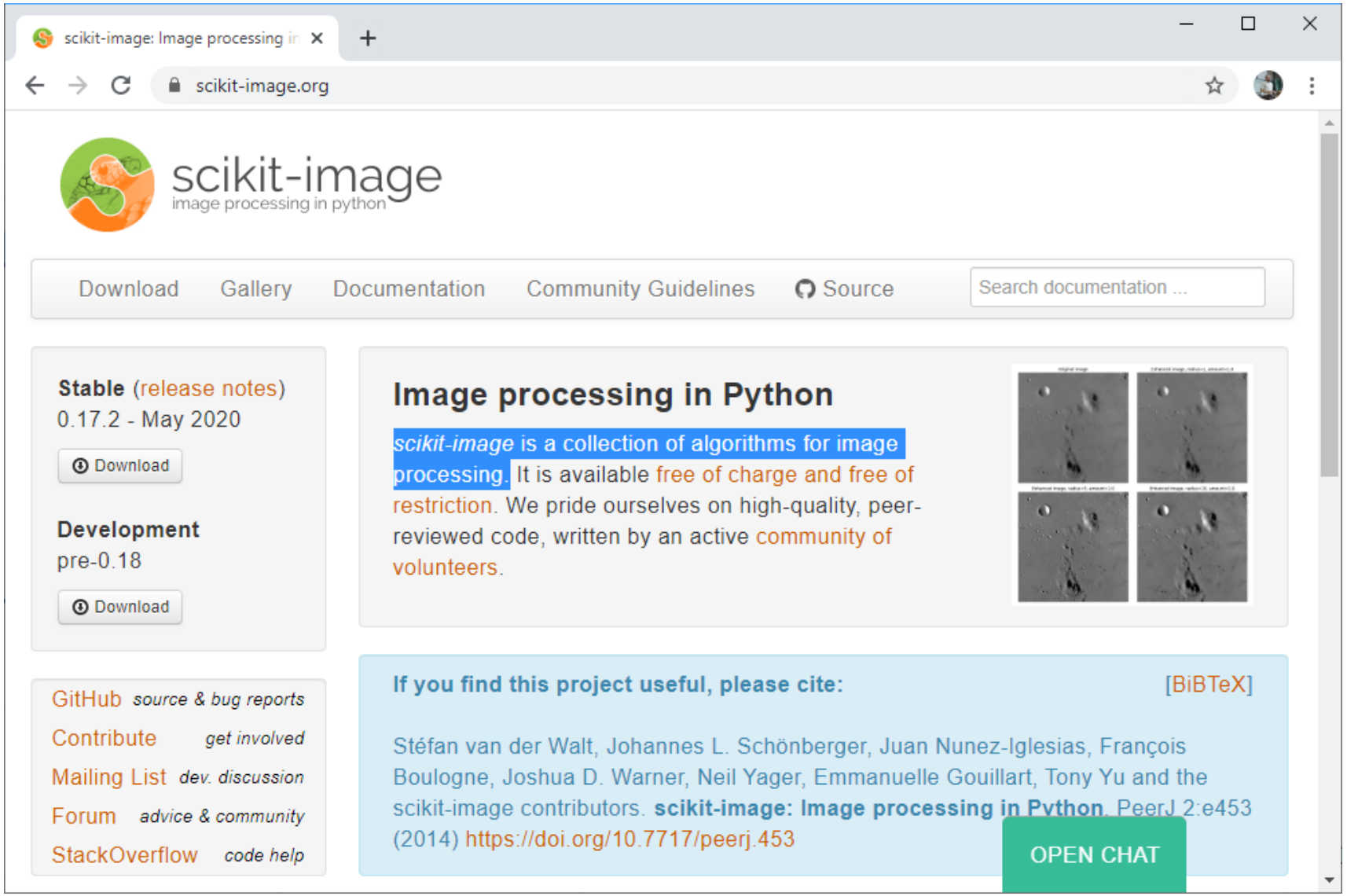
Robert Haase

April 2021

scikit-image

- *scikit-image* is a collection of algorithms for image processing.

```
conda install scikit-image
```



Working with images in python

PoL Physics of Life TU Dresden

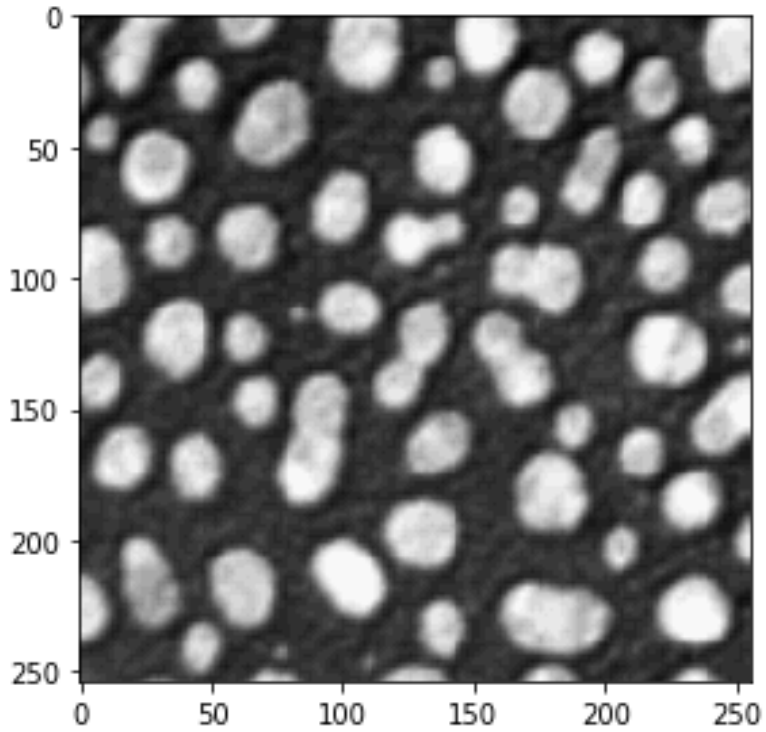
- Open images

```
from skimage.io import imread

image = imread("blobs.tif")
```

- Visualize images

```
from skimage.io import imshow

imshow(image)
```

```
<matplotlib.image.AxesImage at 0x245e7e5b220>
```



Filtering images

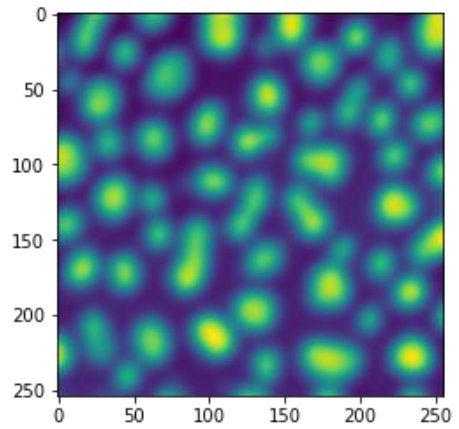
- Explore available filters (lectures 02, 07) and combine them

```
from skimage import filters

filters.
```

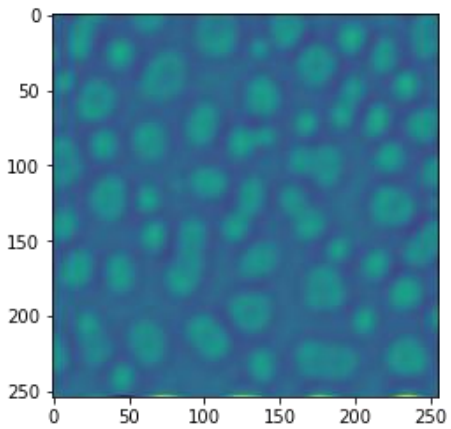
```
LPIFilter2D
median
meijering
prewitt
prewitt_h
prewitt_v
rank
rank_order
ridges
roberts
```

```
from skimage import filters

# Gaussian blur
gaussian_blurred_image = filters.gaussian(image, 5)
plt.imshow(gaussian_blurred_image)
plt.show()

# LoG
laplacian_of_gaussian = filters.laplace(gaussian_blurred_image)
plt.imshow(laplacian_of_gaussian)
plt.show()

# another Gaussian blur
gaussian_blurred_image_2 = filters.gaussian(image, 10)
plt.imshow(gaussian_blurred_image_2)
plt.show()

# DoG
difference_of_gaussian = gaussian_blurred_image - gaussian_blurred_image_2
plt.imshow(difference_of_gaussian)
plt.show()
```

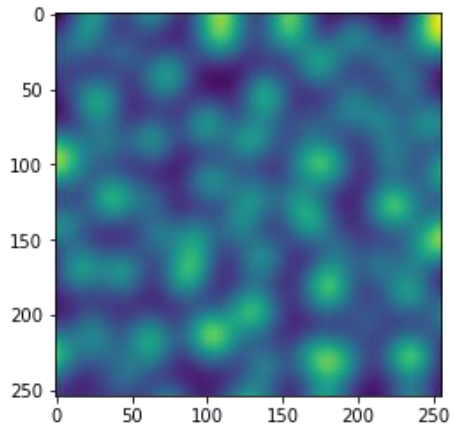
Gaussian blur (sigma=5)



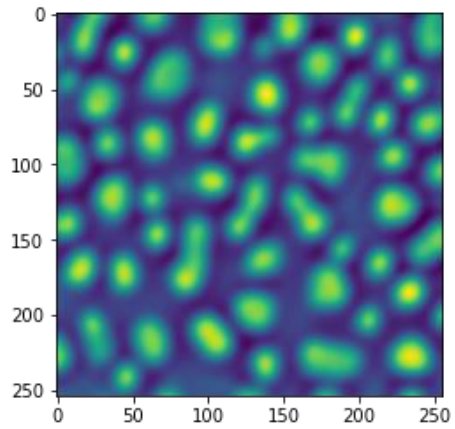
LoG



Gaussian blur (sigma=10)



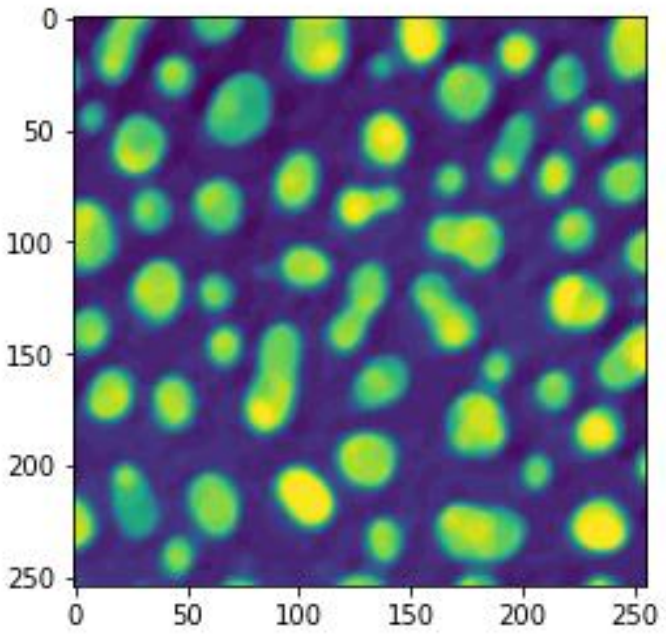
DoG



Filtering images: structuring elements

- Some filters ask for structuring elements

```
from skimage.morphology import disk

median_filtered = filters.median(image, disk(5))
plt.imshow(median_filtered)
plt.show()
```



Structuring element

Module: filters — skimage v0.17.

scikit-image.org/docs/stable/api/skimage.filters.html#skimage.filters.median

median

skimage.filters. **median**(*image*, *selem=None*, *out=None*, *mode='nearest'*, *cval=0.0*, *behavior='ndimage'*) [source]

Return local median of an image.

Parameters

image : array-like

Input image.

selem : ndarray, optional

If `behavior=='rank'`, `selem` is a 2-D array of 1's and 0's. If `behavior=='ndimage'`, `selem` is a N-D array of 1's and 0's with the same number of dimension than `image`. If None, `selem` will be a N-D array with 3 elements for each dimension (e.g., vector, square, cube, etc.)

out : ndarray, (same dtype as image), optional

If None, a new array is allocated.

mode : {'reflect', 'constant', 'nearest', 'mirror','wrap'}, optional

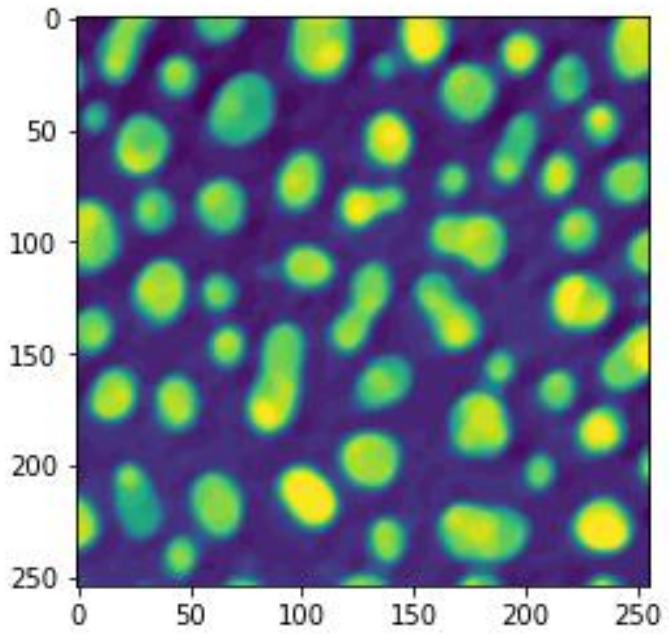
The mode parameter determines how the array borders are handled, where `cval` is the value when mode is equal to 'constant'. Default is 'nearest'.

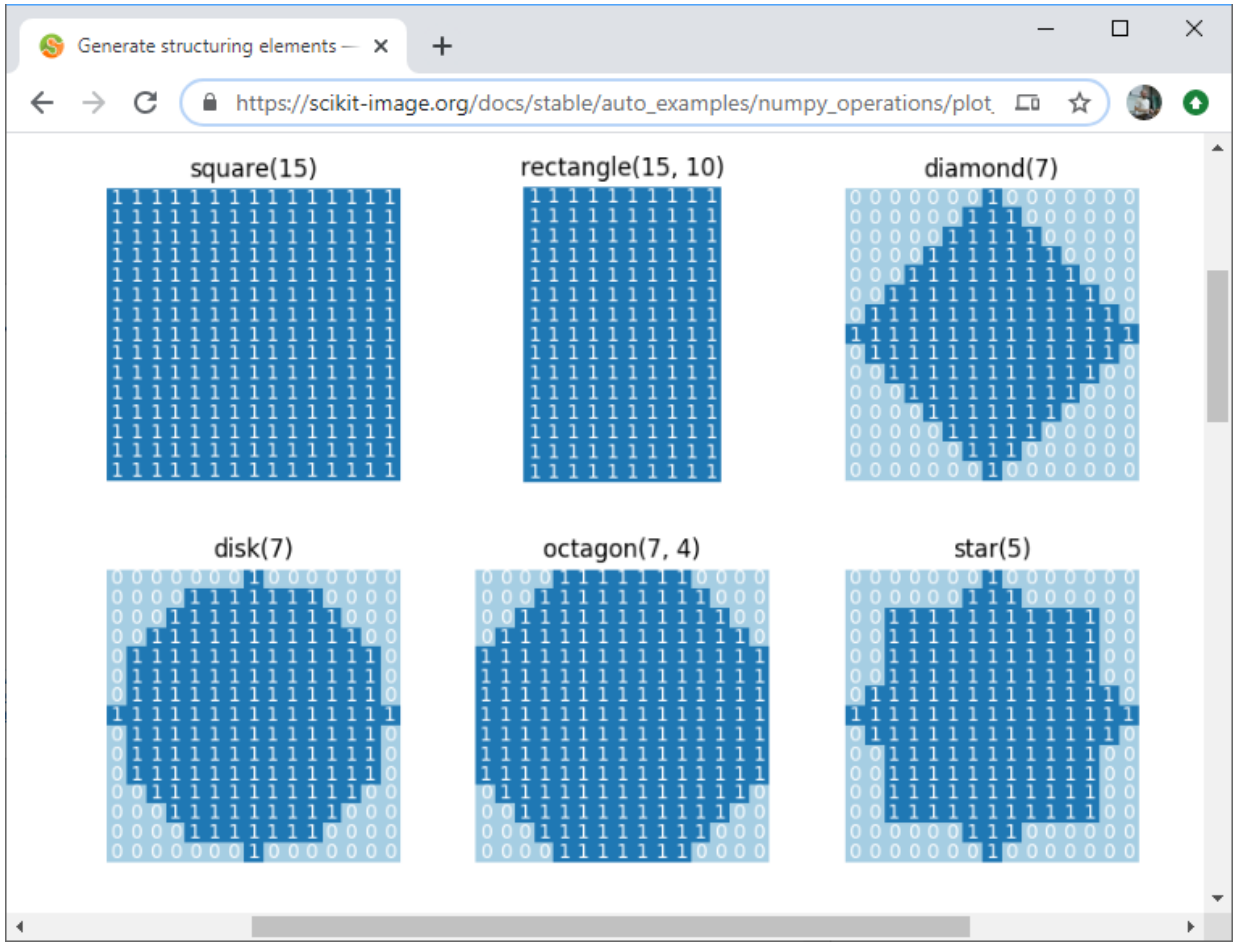
New in version 0.15: `mode` is used when `behavior='ndimage'`

Filtering images: structuring elements

- Some filters ask for structuring elements

```
from skimage.morphology import disk

median_filtered = filters.median(image, disk(5))
plt.imshow(median_filtered)
plt.show()
```





https://scikit-image.org/docs/stable/auto_examples/numpy_operations/plot_structuring_elements.html#sphx-glr-auto-examples-numpy-operations-plot-structuring-elements-py

Image visualization: subplots

- Use matplotlib to put images side-by-side

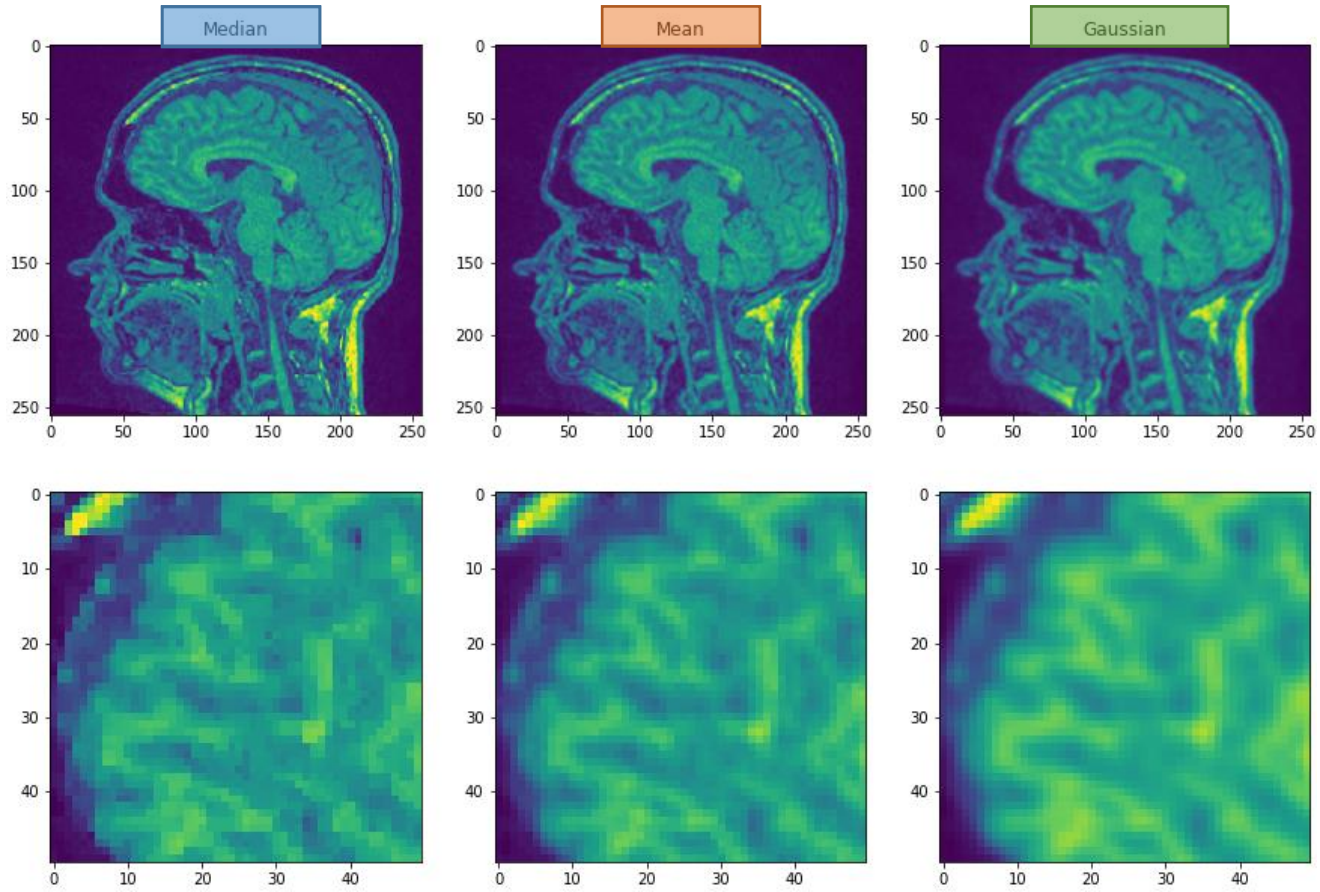
```
median_filtered = filters.median(noisy_mri, disk(1))
mean_filtered = filters.rank.mean(noisy_mri, disk(1))
gaussian_filtered = filters.gaussian(noisy_mri, sigma=1)

fig, axs = plt.subplots(2, 3, figsize=(15,10))

# first row
axs[0, 0].imshow(median_filtered)
axs[0, 0].set_title("Median")
axs[0, 1].imshow(mean_filtered)
axs[0, 1].set_title("Mean")
axs[0, 2].imshow(gaussian_filtered)
axs[0, 2].set_title("Gaussian")

# second row
axs[1, 0].imshow(median_filtered[50:100, 50:100])
axs[1, 1].imshow(mean_filtered[50:100, 50:100])
axs[1, 2].imshow(gaussian_filtered[50:100, 50:100])
```

rows

columns

row

column



https://matplotlib.org/stable/api/_as_gen/matplotlib.pyplot.subplots.html

PoL
Physics of Life
TU Dresden

CENTER FOR
SYSTEMS BIOLOGY
DRESDEN

Image visualization in python using napari

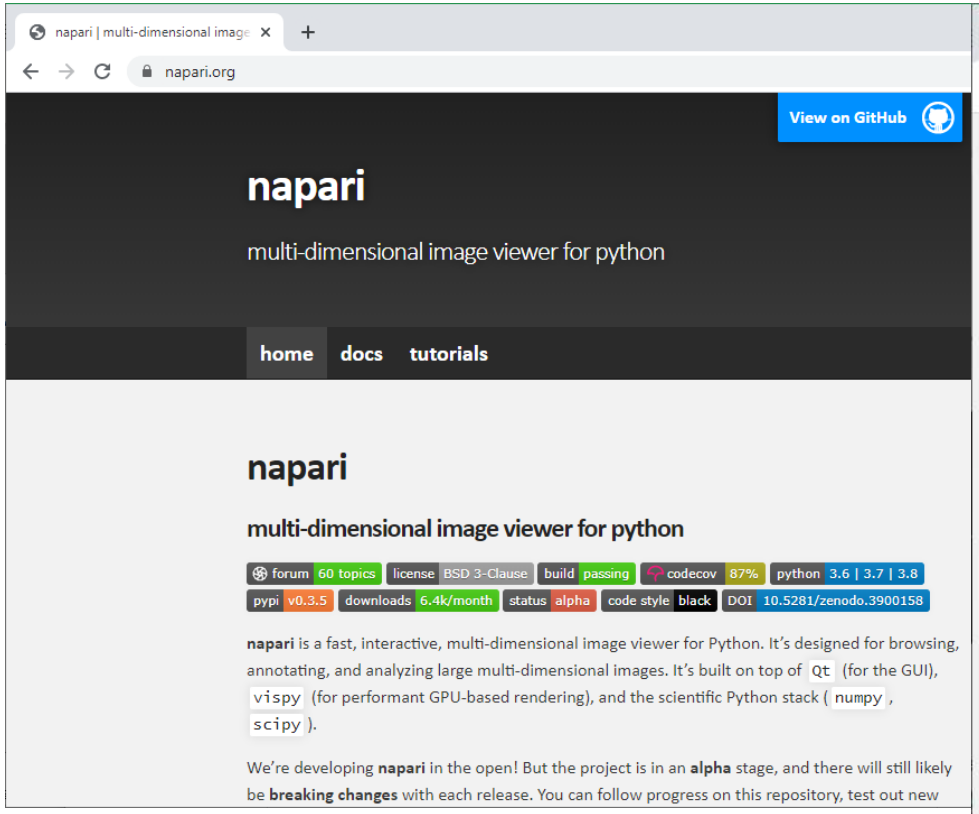
Robert Haase

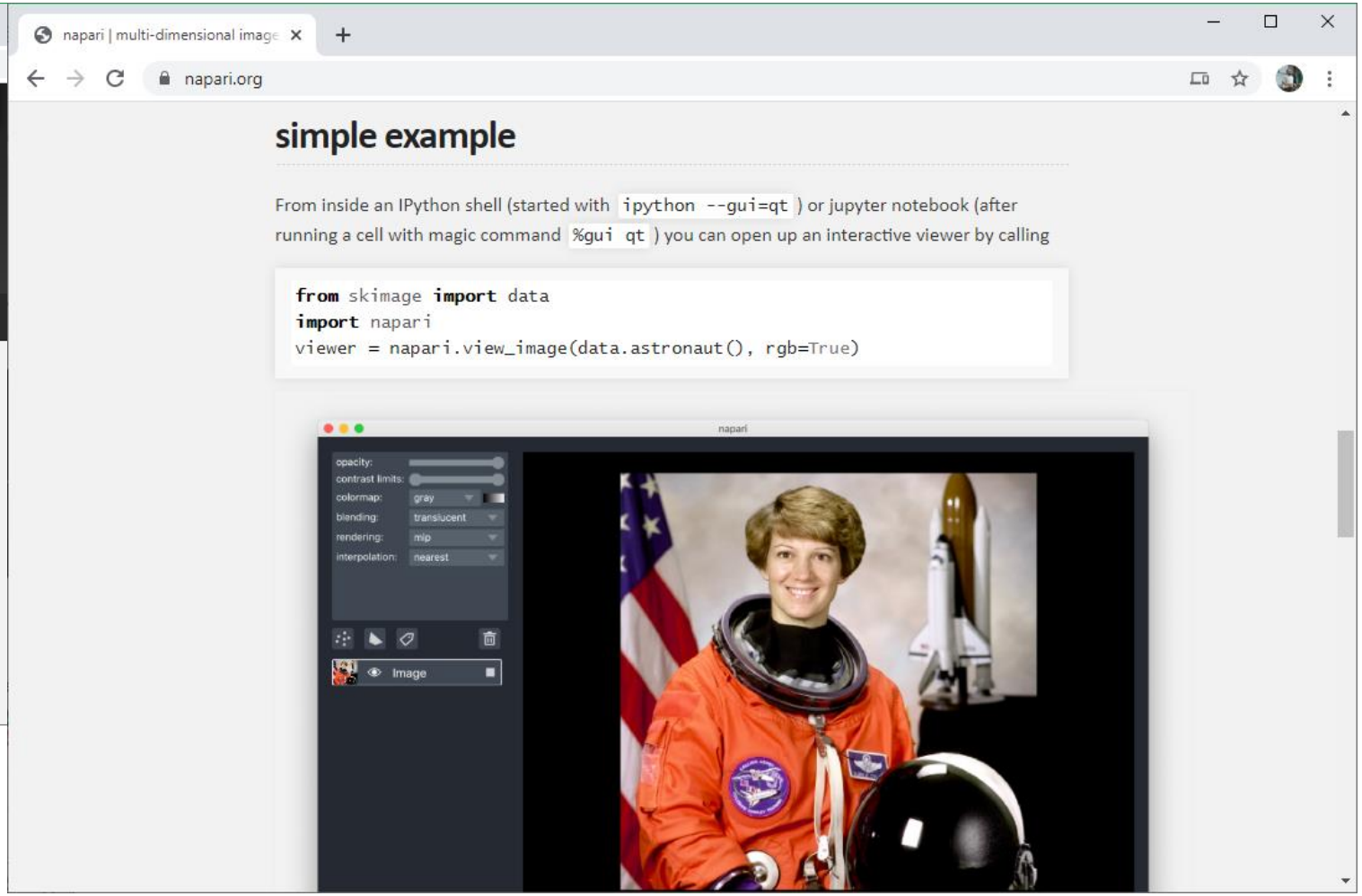
April 2021

@haesleinhuepf

napari

- Multi-dimensional image viewer in python
- https://napari.org/





napari

- Initialisation

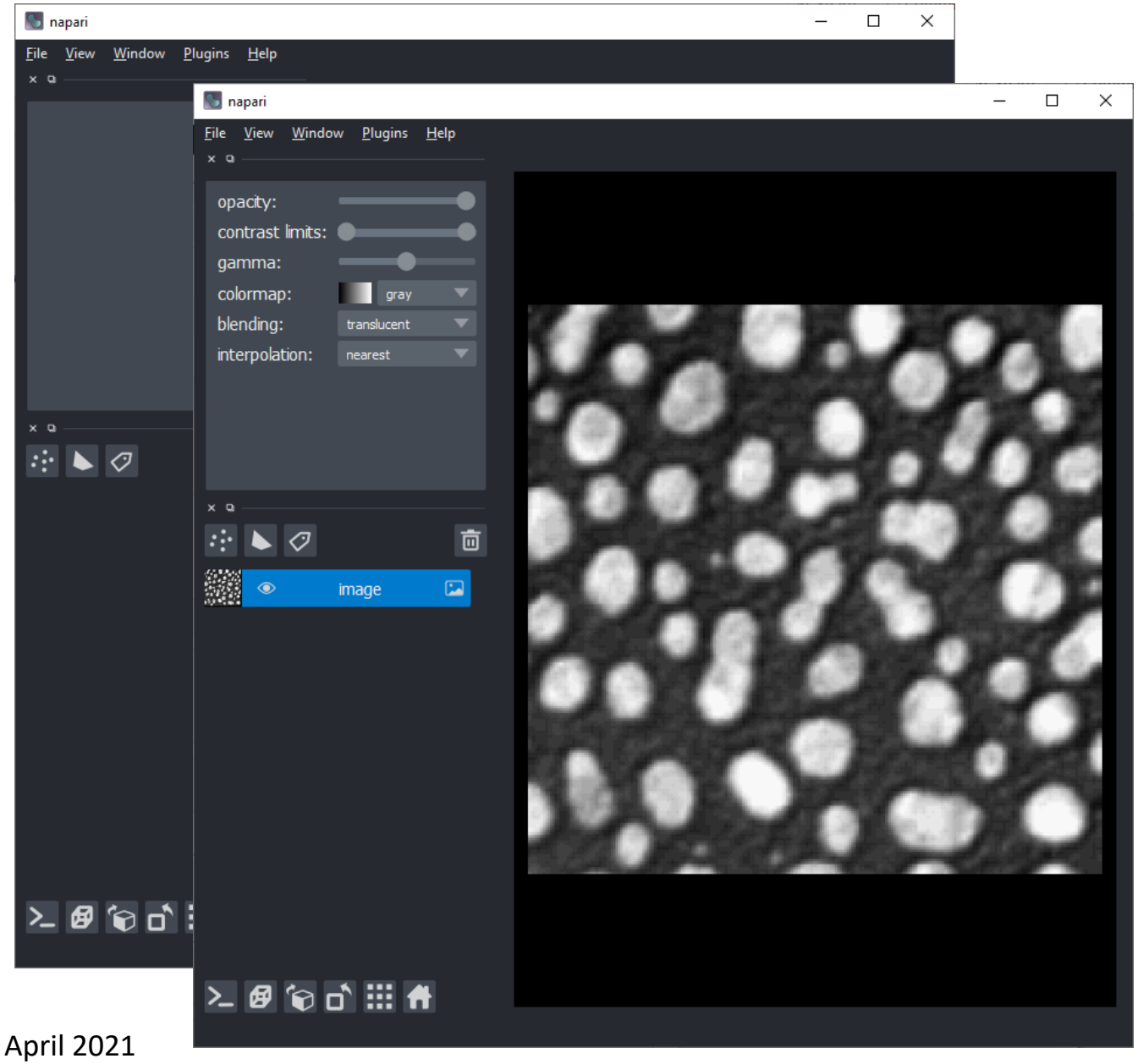
```
%gui qt
```

- Open a window

```
import napari

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

- Add an image (layer)

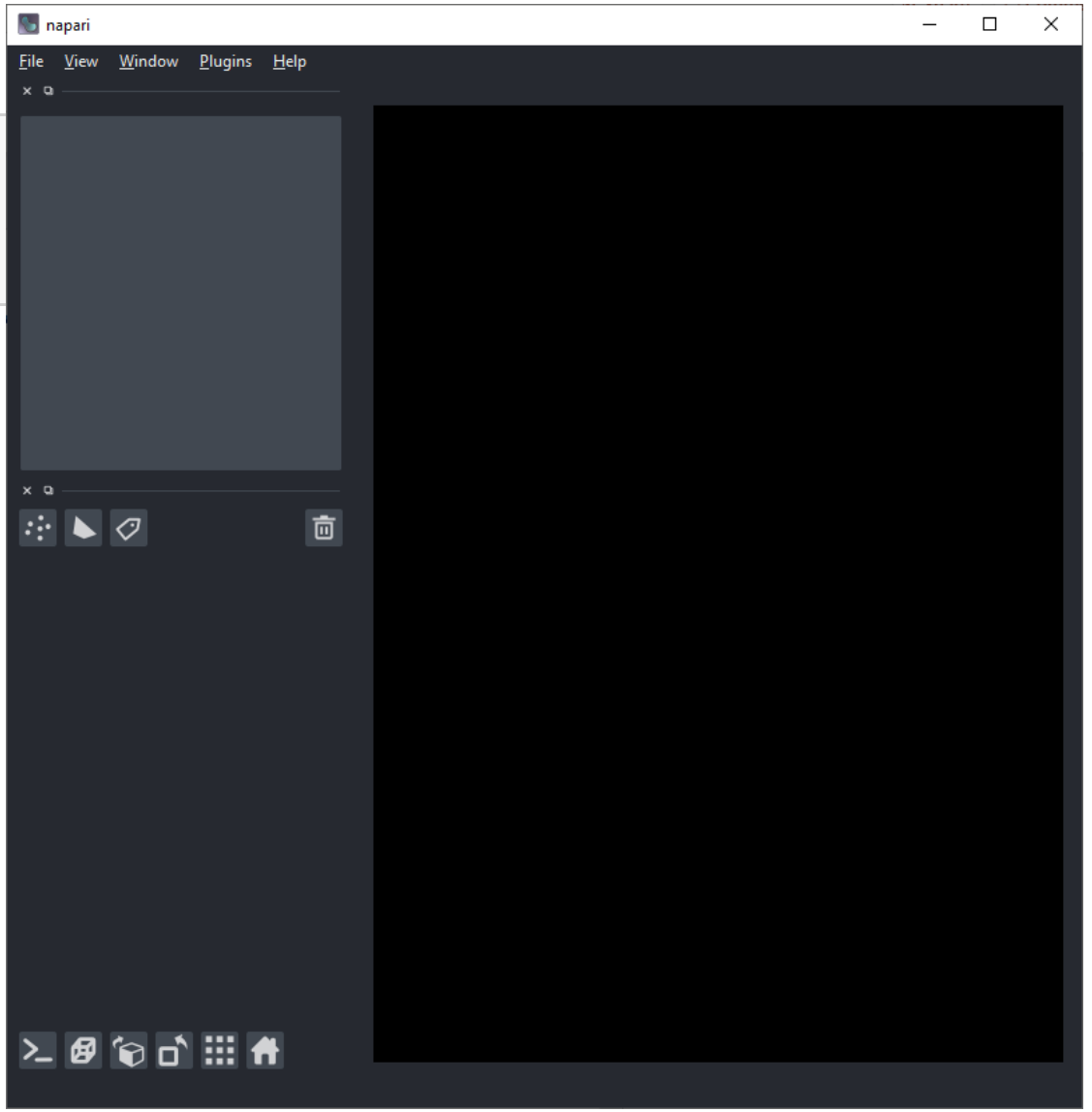
```
# Add a new layer containing an image
viewer.add_image(image);
```



Removing layers

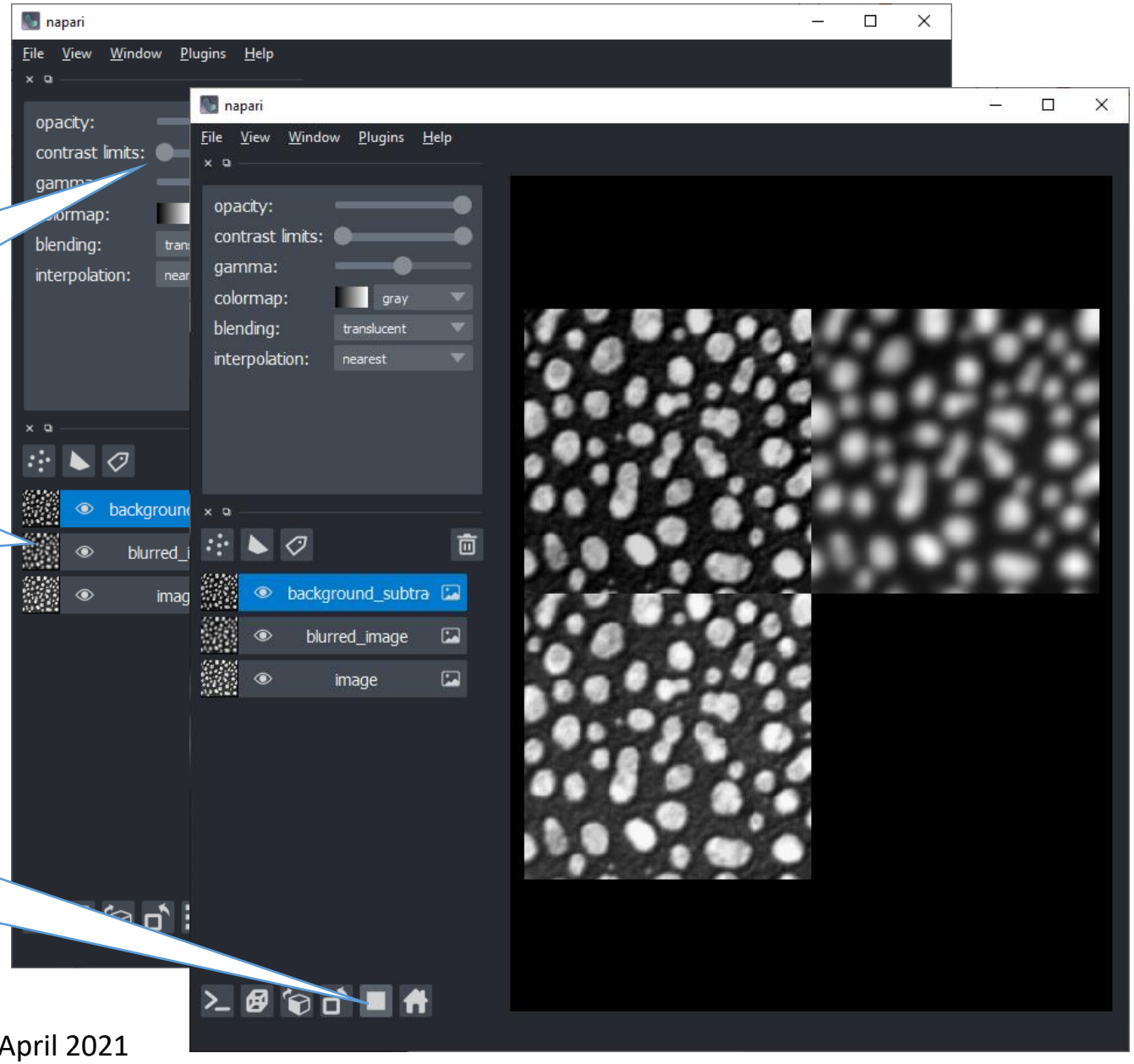
- Remove all layers from napari

```
# Remove all layers to start from scratch
for l in viewer.layers:
    viewer.layers.remove(l)
```



Add multiple images as layers

- Add layers to napari to visualize intermediate processing results on top of each other or side by side



When to use what? (opinionated slide)

- ImageJ / Fiji
  - Established platform for 20 years
  - Right tool for quick exploration of image data
  - Classical algorithms
  - Hundreds of plugins available
  - Not compatible with python
  - Workflow development limited to ImageJ compatible functionality
  - Pixel-by-pixel comparison complicated

- napari
  - Under development
  - Limited functionality (yet)
  - Small number of plugins available, e.g. for deep-learning
  - Compatible with python
  - Workflows can include any kind of data analysis tools from the python ecosystem
  - Layer-concept makes comparing images easy

Summary

- Image filtering
  - Noise removal
  - Background subtraction
- Tools
  - ImageJ / Fiji
  - Python
    - Scikit-image
    - napari

Coming up next

- Image Segmentation
  - Thresholding
  - Mask refinement
  - Connected component analysis