ScaDS.AI DRESDEN LEIPZIG

CENTER FOR SCALABLE DATA ANALYTICS AND ARTIFICIAL INTELLIGENCE

NFDI4 BIOIMAGE

NATIONAL RESEARCH DATA MANAGEMENT INFRASTRUCTURE FOR MICROSCOPY AND BIOIMAGE ANALYSIS

GloBIAS Global BioImage Analysts' Society

GLOBAL BIOIMAGE ANALYST'S SOCIETY

Reproducible Bio-Image Analysis using Python, Napari, Jupyter and AI

Robert Haase

https://scads.github.io/embo-lsm-bia-2025

Funded by Bundesministerium für Forschung, Technologie und Raumfahrt

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.

Reproducibility: Python

- Technical: I can rerun the workflow because it is well documented.
- Inter-personal: I understand a workflow and can explain all steps.

```python
from skimage.io import imread, imsave
import stackview
import pyclesperanto as cle
import numpy as np

image = imread('data/lund.tif')

# Apply top hat filter to the image
processed_image = np.asarray(cle.top_hat_box(image, radius_x=10, radius_y=10, radius_z=0))
# Segment the image using voronoi otsu labeling
label_image = cle.voronoi_otsu_labeling(processed_image, spot_sigma=2, outline_sigma=2)

print(label_image.max())
```

Output: 285

We cannot follow the workflow, because we do not see intermediate results.

Technical: ✅

Inter-personal: ❌

Reproducibility: Python + Jupyter

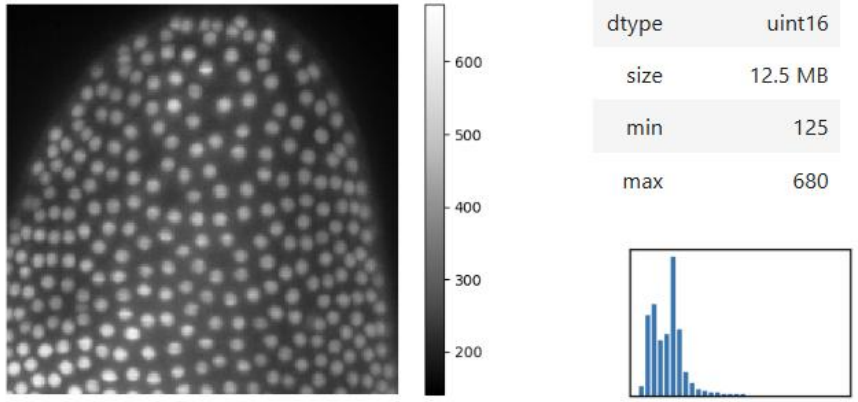
- Technical: I can rerun the workflow because it is well documented.
- Inter-personal: I understand a workflow and can explain all steps.

```
[2]: image = imread('data/lund.tif')

     image.shape
```

```
[2]: (100, 256, 256)
```

```
[3]: stackview.insight(image)
```

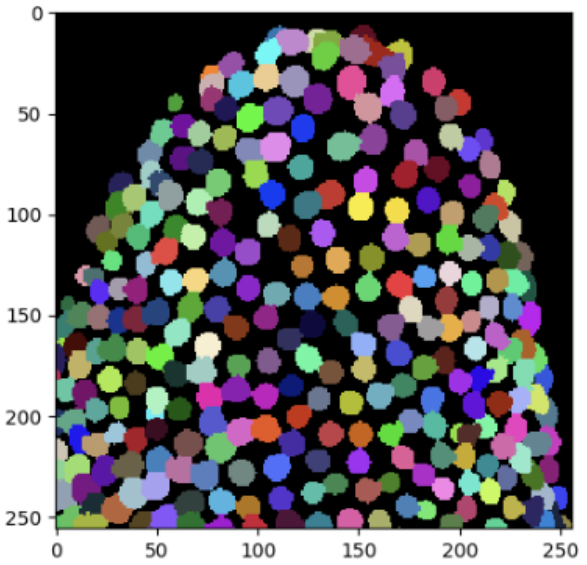
[3]:



shape	(100, 256, 256)
dtype	uint16
size	12.5 MB
min	125
max	680


```
[12]: # Segment the image using voronoi otsu labeling
      label_image = cle.voronoi_otsu_labeling(processed_image, spot_sigma=2, outline_sigma=2)
      label_image
```

[12]:



cle._ image

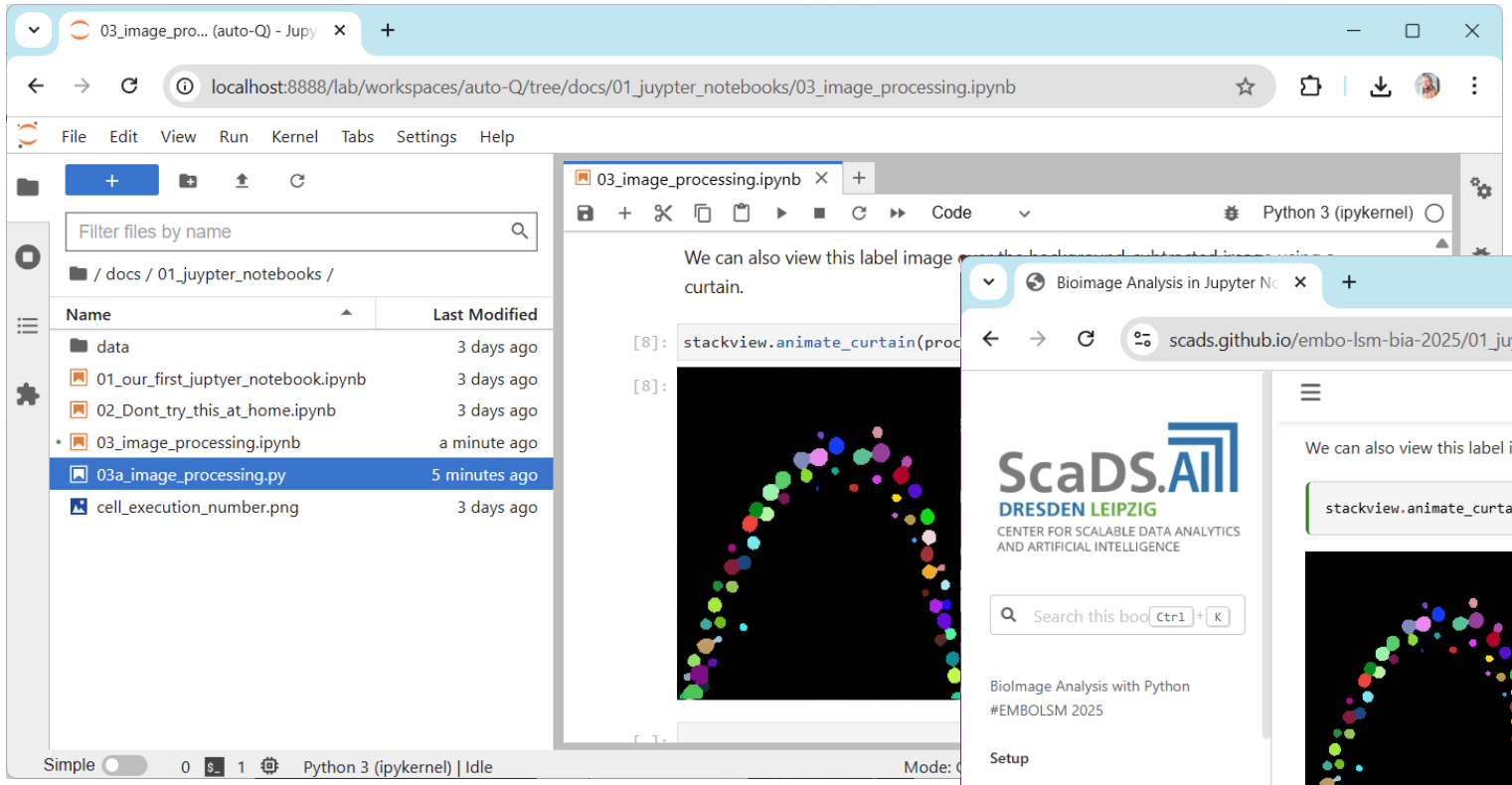
shape	(100, 256, 256)
dtype	uint32
size	25.0 MB
min	0.0
max	285.0

Technical: ✅

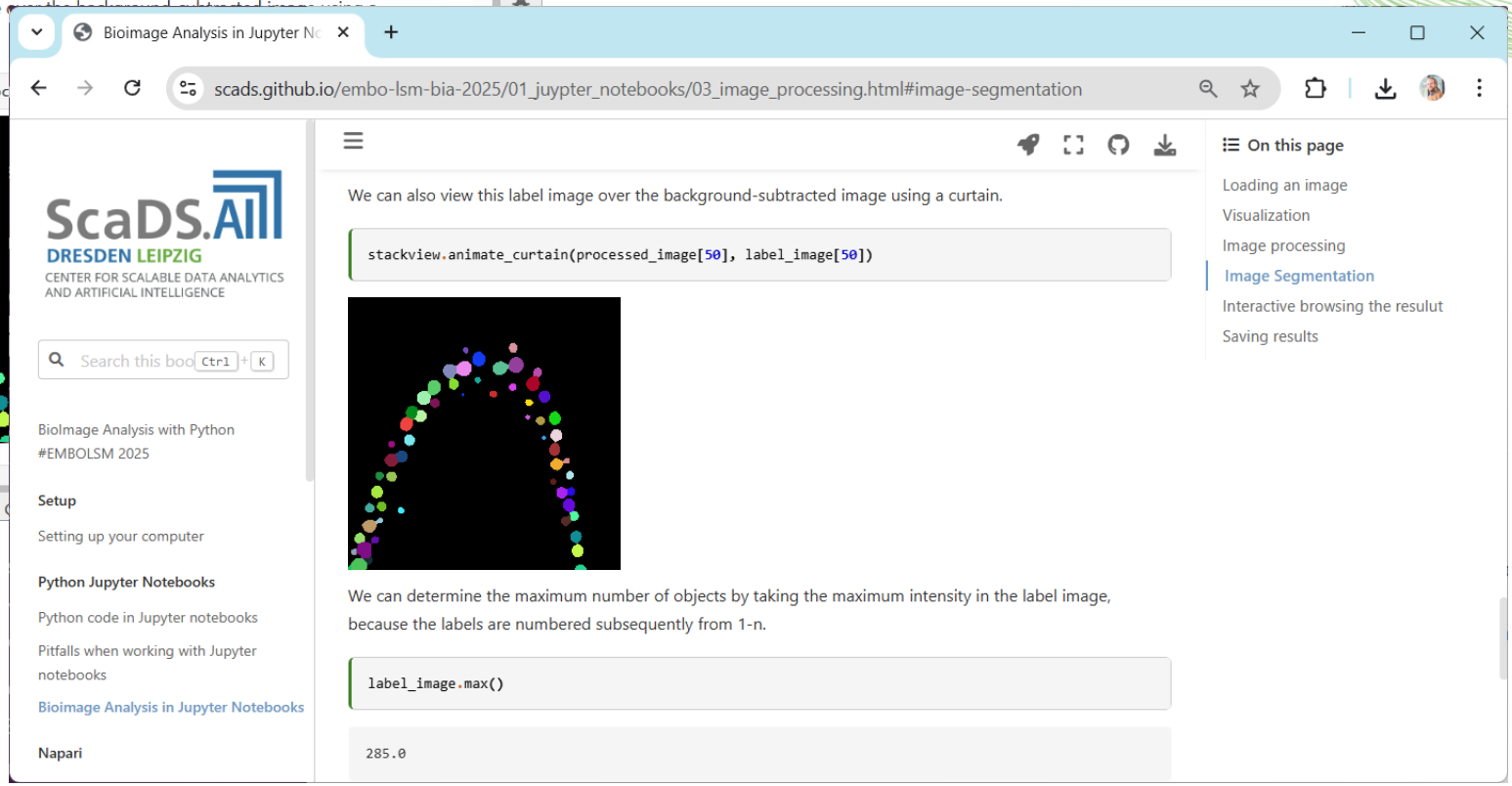
Inter-personal: ✅

Reproducibility : Python + Jupyter

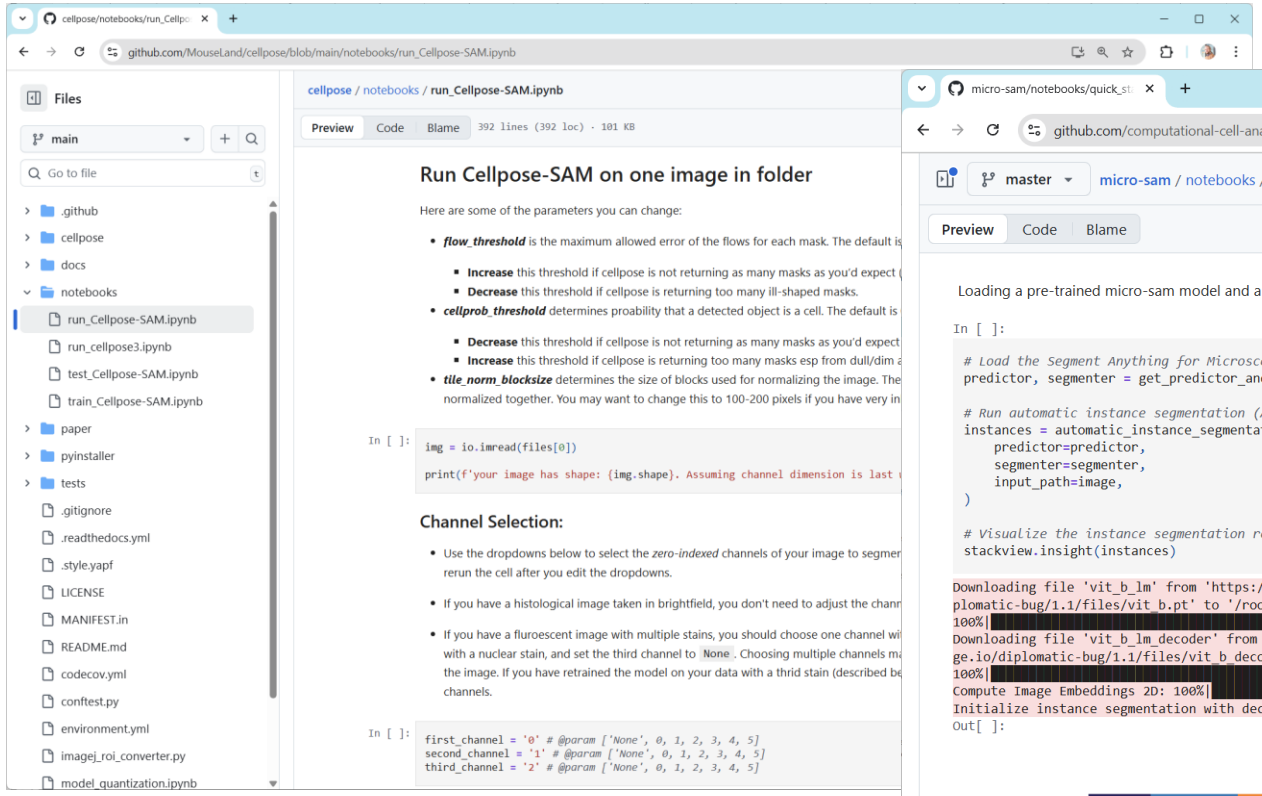
Other's view when you share the workflow online



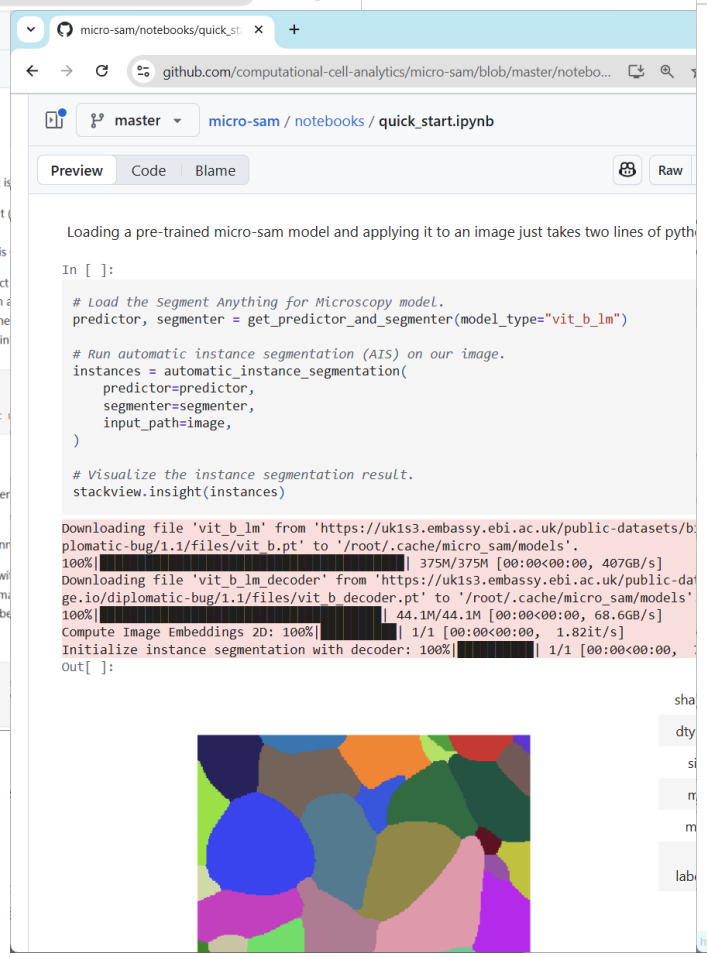
View on your laptop



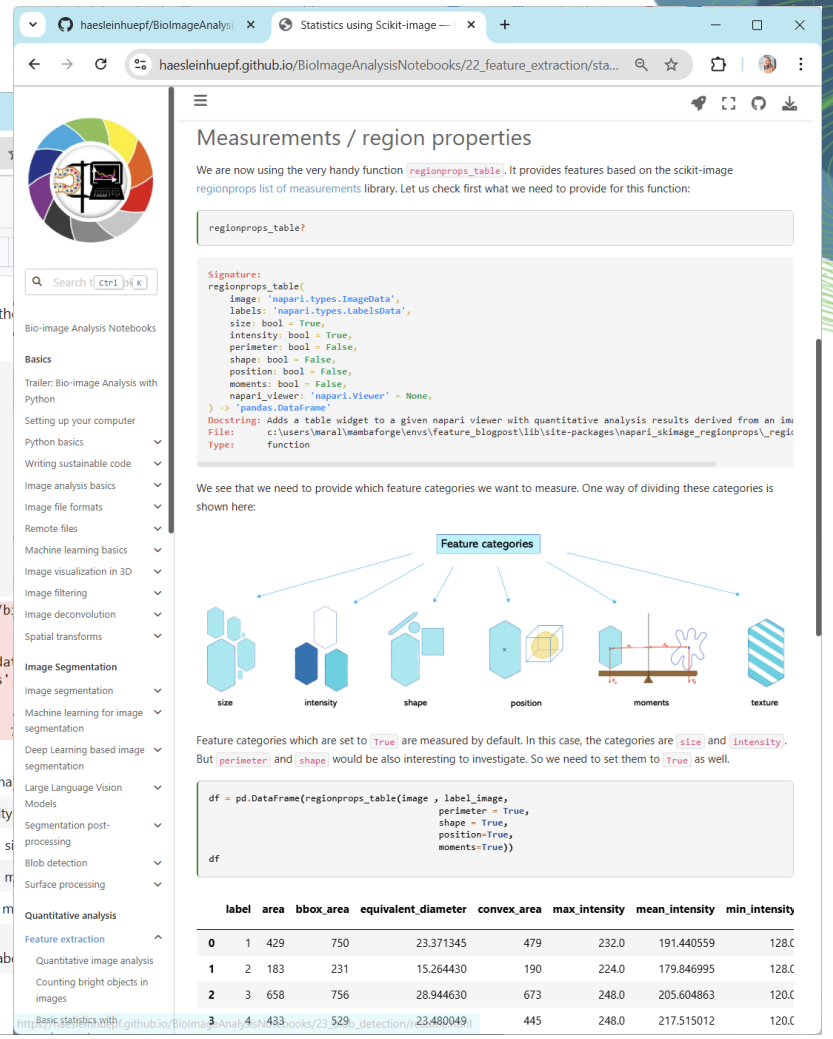
Reproducibility : Python + Jupyter



https://github.com/MouseLand/cellpose/blob/main/notebooks/run_Cellpose-SAM.ipynb

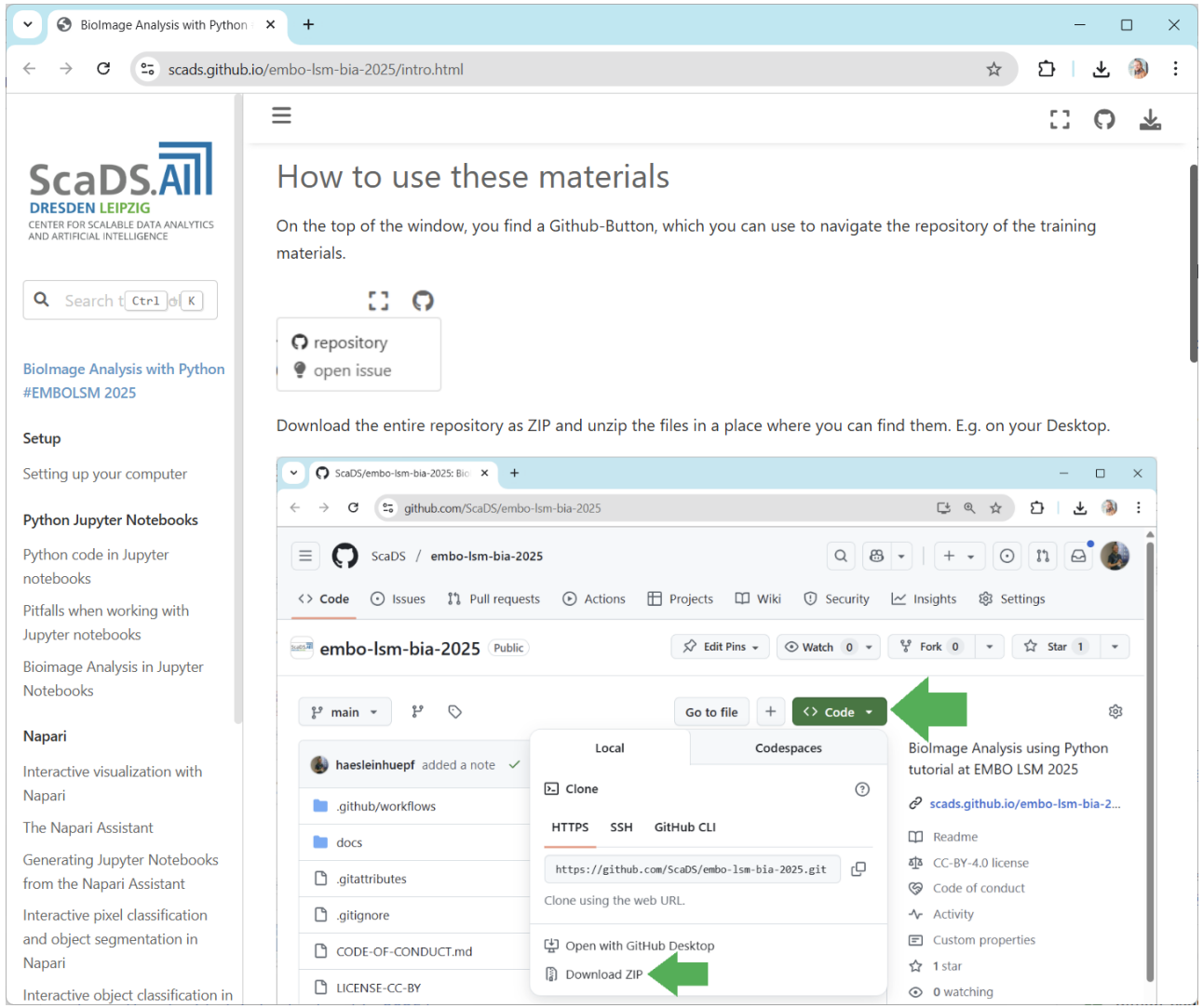


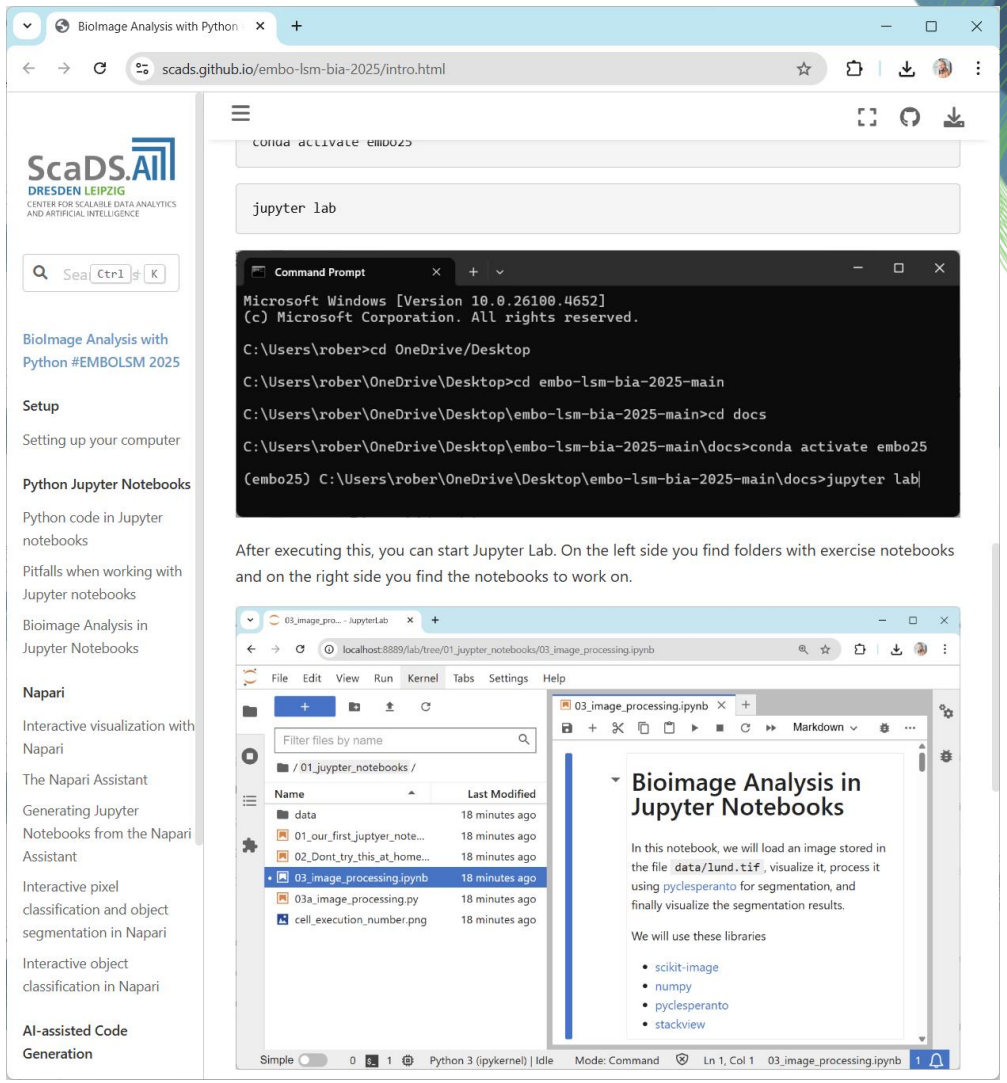
https://github.com/computational-cell-analytics/micro-sam/blob/master/notebooks/quick_start.ipynb



https://haesleinhuepf.github.io/BioImageAnalysisNotebooks/22_feature_extraction/statistics_with_scikit_image.html

Training materials for today





ScaDS.AI DRESDEN LEIPZIG

CENTER FOR SCALABLE DATA ANALYTICS AND ARTIFICIAL INTELLIGENCE

NFDI4 BIOIMAGE

NATIONAL RESEARCH DATA MANAGEMENT INFRASTRUCTURE FOR MICROSCOPY AND BIOIMAGE ANALYSIS

GloBIAS Global BioImage Analysts' Society

GLOBAL BIOIMAGE ANALYST'S SOCIETY

Reproducible Bio-Image Analysis using Python, Napari, Jupyter and AI

Robert Haase

Funded by Bundesministerium für Forschung, Technologie und Raumfahrt

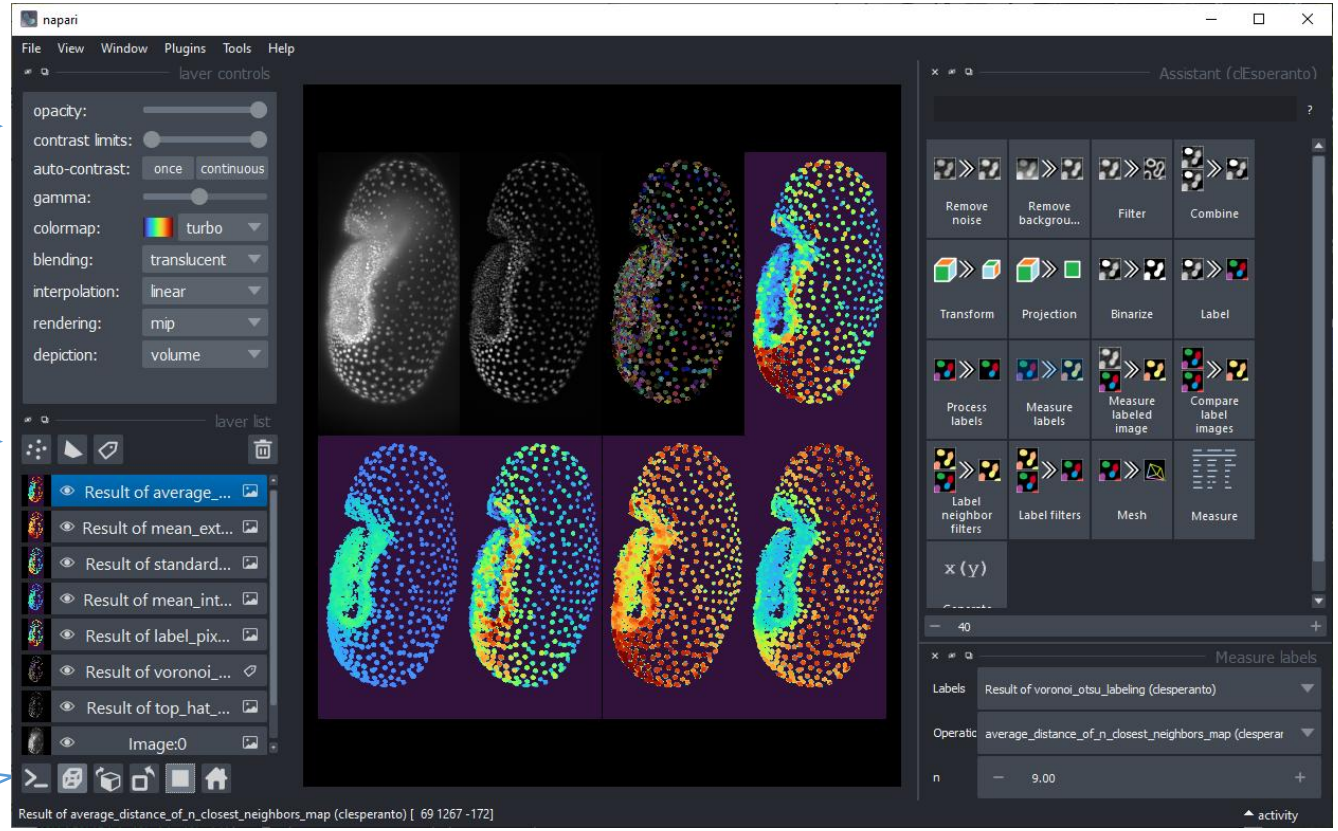
SACHSEN Diese Maßnahme wird gefördert durch die Bundesregierung aufgrund eines Beschlusses des Deutschen Bundestages. Diese Maßnahme wird mitfinanziert durch Steuermittel auf der Grundlage des von den Abgeordneten des Sächsischen Landtags beschlossenen Haushaltes.

Napari – Graphical User Interface

View configuration / tools

Layers

Viewer controls

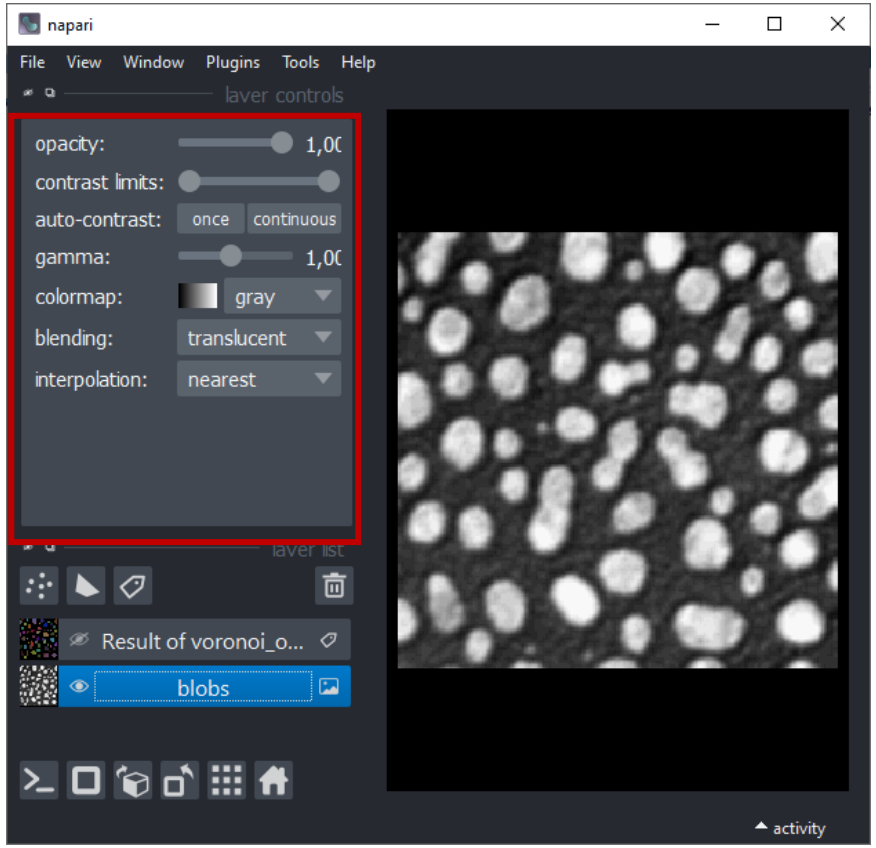


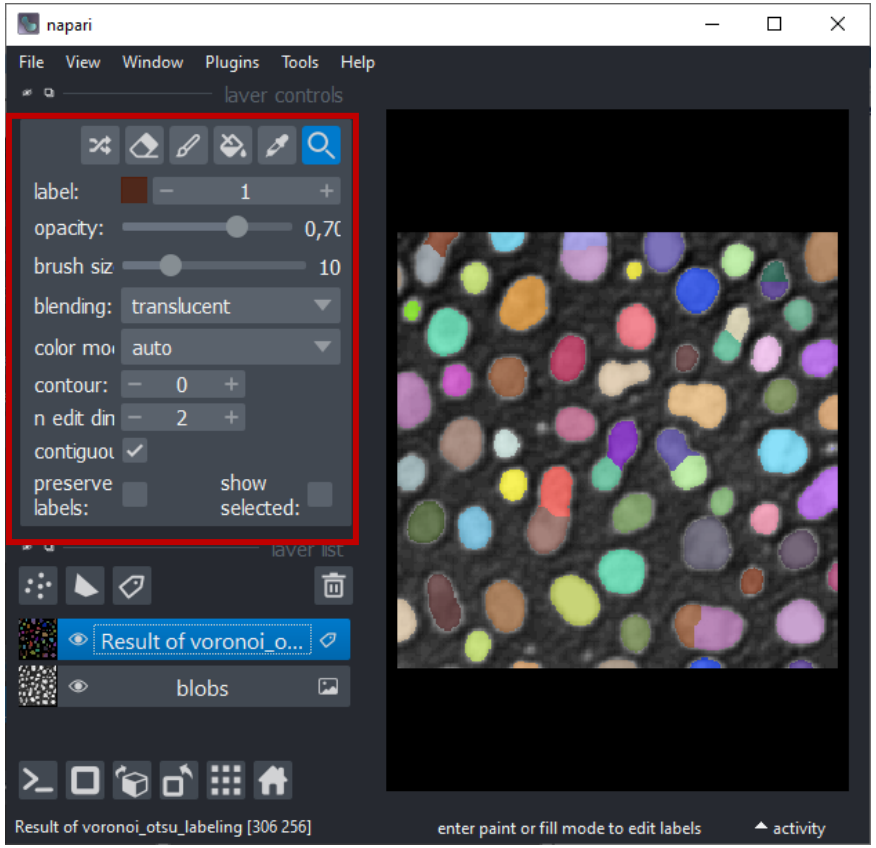
Dock widgets (custom plugins)

Function widgets (custom plugins)

Napari – Graphical User Interface

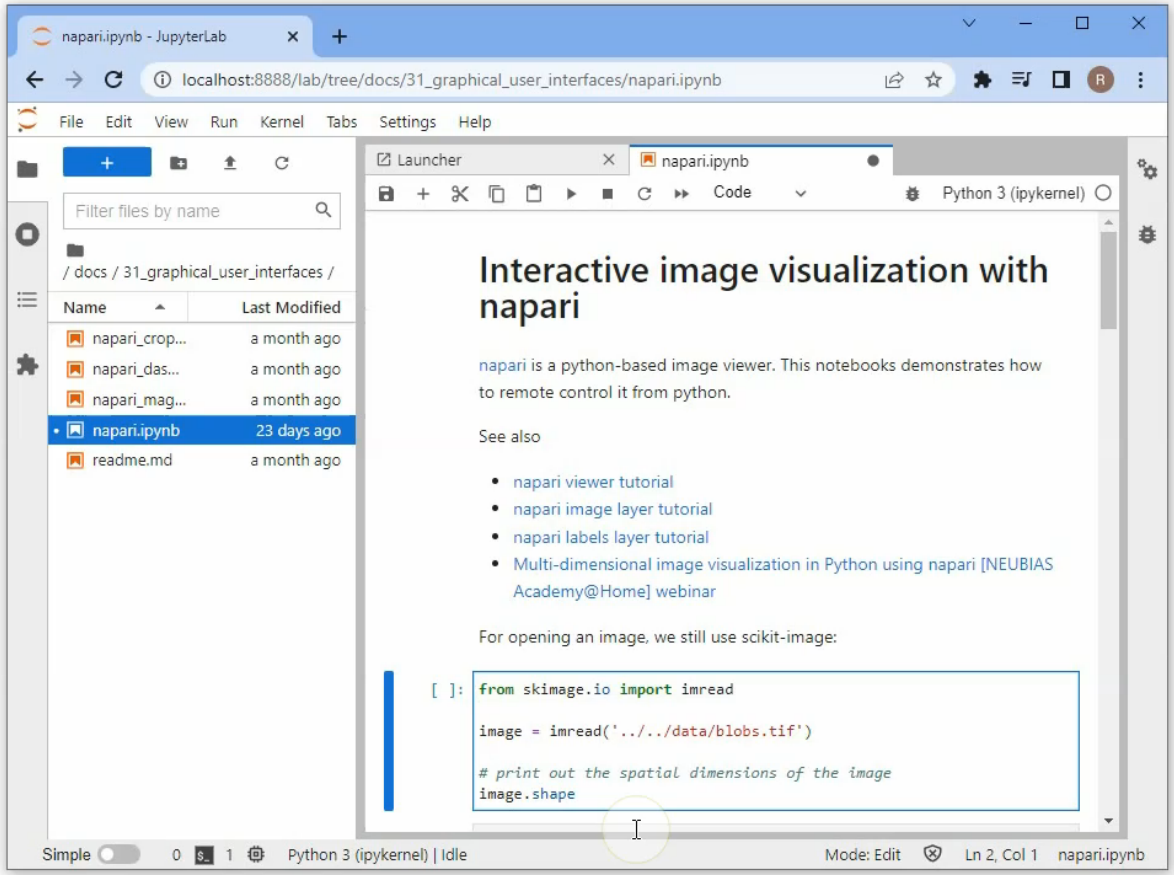
- Context / data type dependent tools





Napari – Python Scripting

- Mixing interactivity and reproducibility



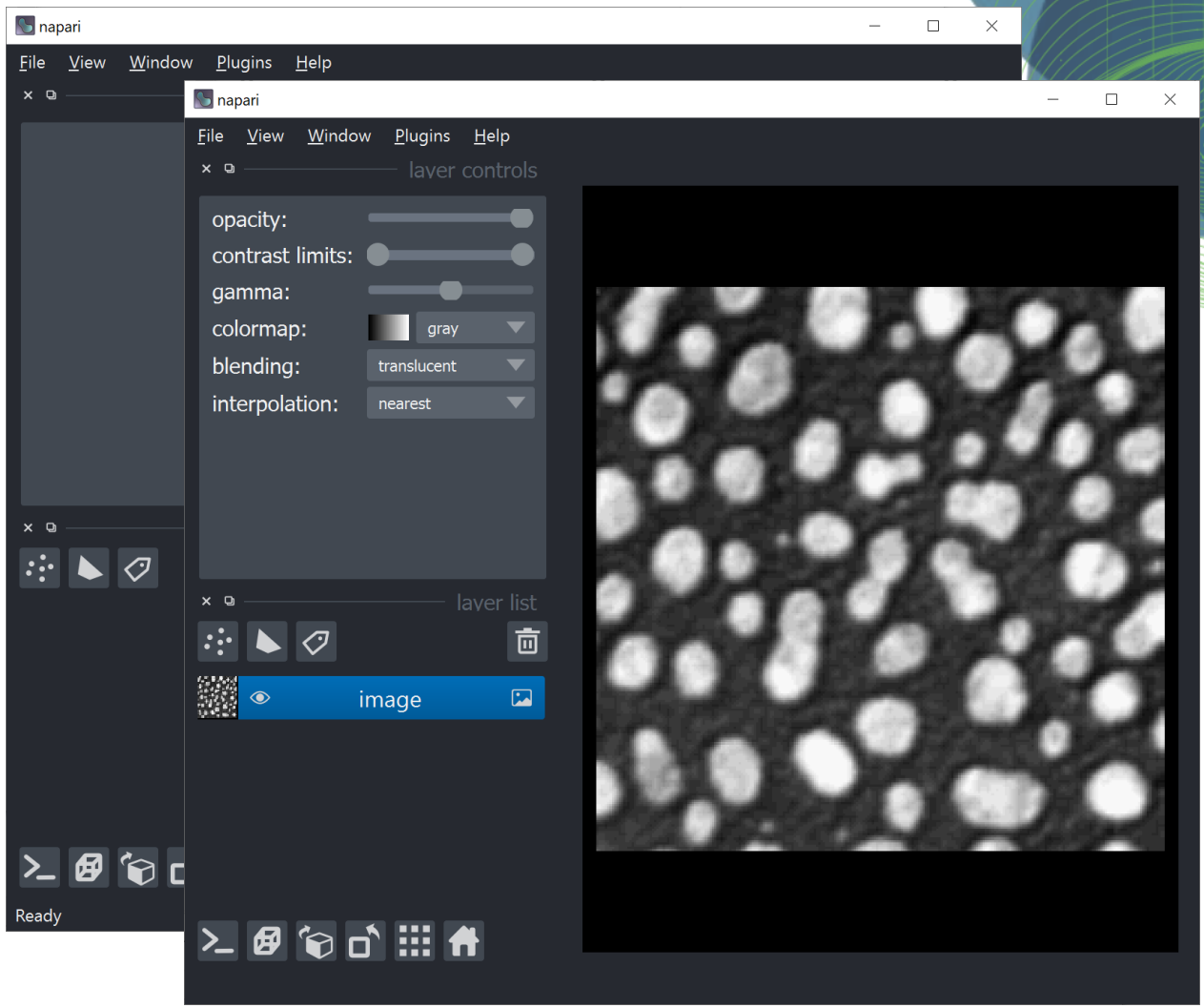
Napari – Python Scripting

- Initialization

```
import napari

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

- Adding images

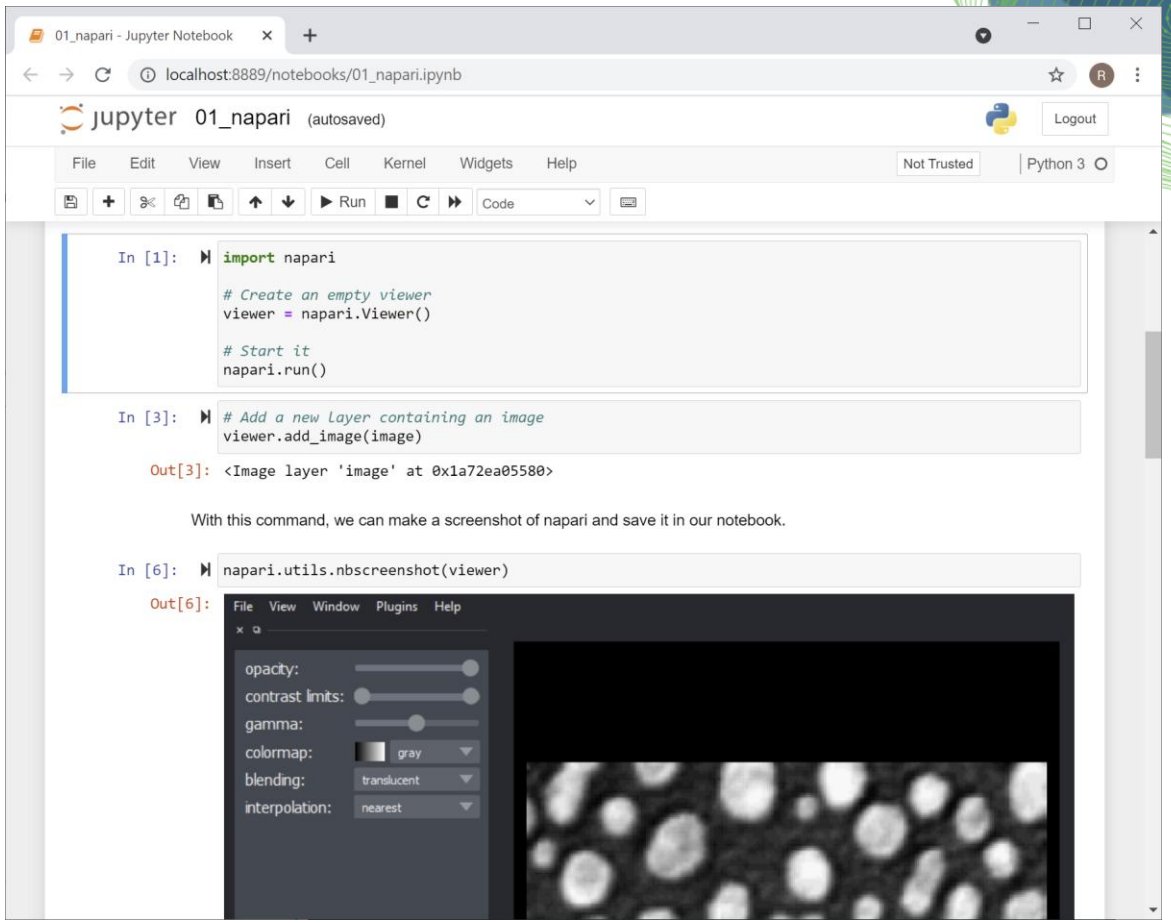
```
viewer.add_image(image)
```



Napari – Python Scripting

- Make screenshots from napari and put them in your jupyter notebook

```
napari.utils.nbscreenshot(viewer)
```



Good for reproducibility

Napari – Python Scripting

- Removing layers

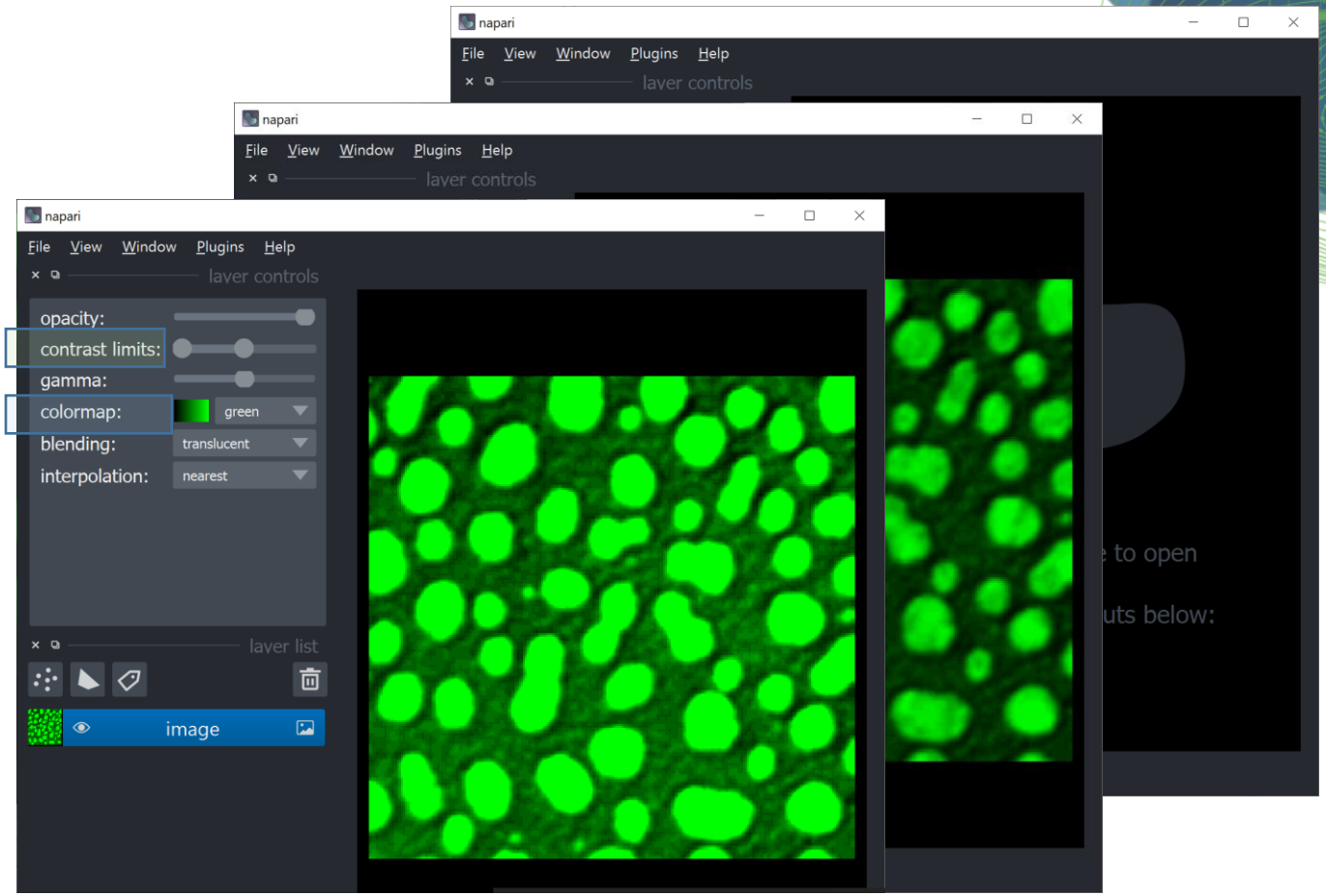
```
for l in viewer.layers:
    viewer.layers.remove(l)
```

- Modify visualization while adding layers

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

- Modify layers after adding

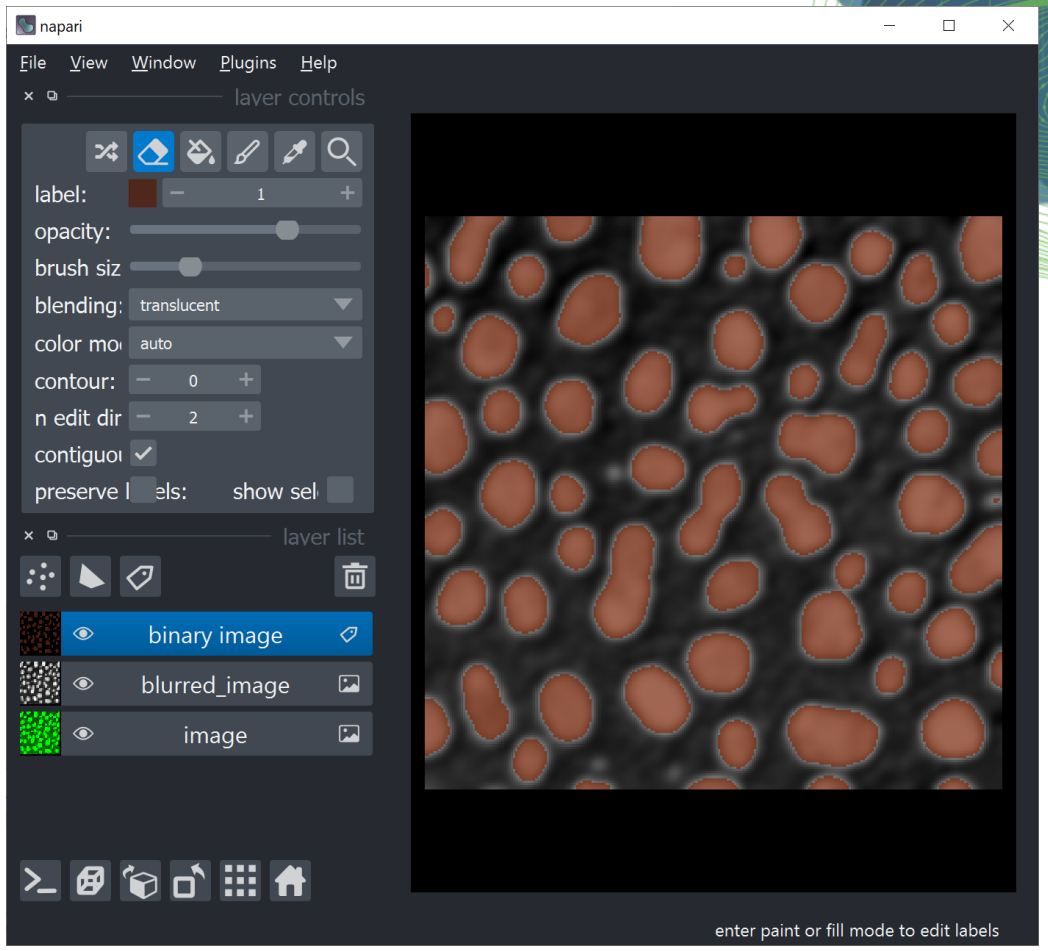
```
layer = viewer.add_image(image)
layer.colormap = 'green'
layer.contrast_limits = (0, 128)
```



Napari – Python Scripting

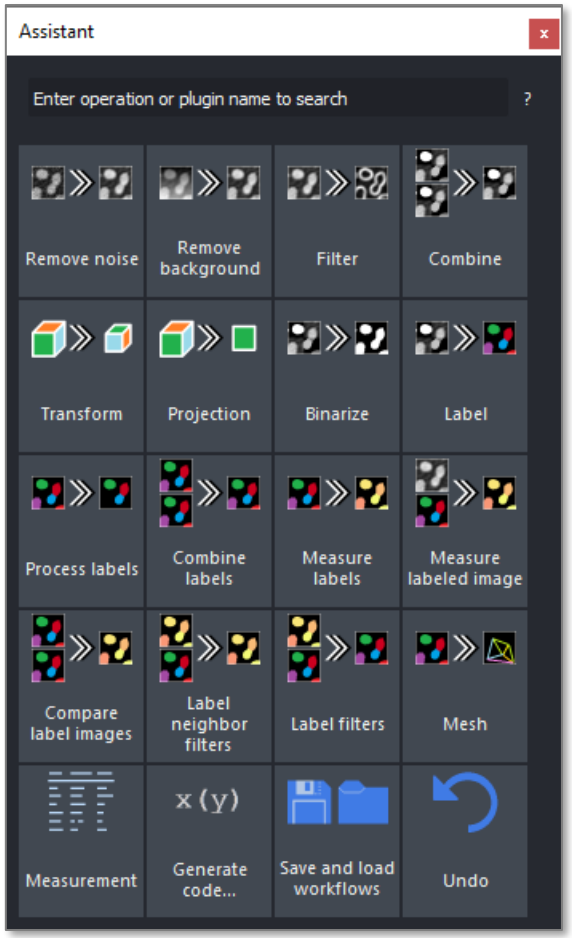
- Binary images and label images visualized as label layers

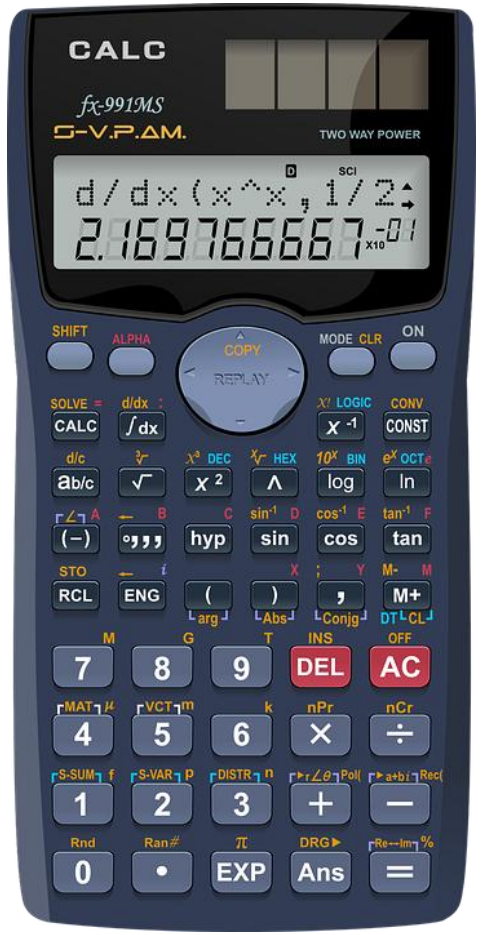
```
from skimage.filters import threshold_otsu
threshold = threshold_otsu(blurred_image)
binary_image = blurred_image > threshold

# 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

- Tools > Utilities > Assistant (na)

Viewer controls

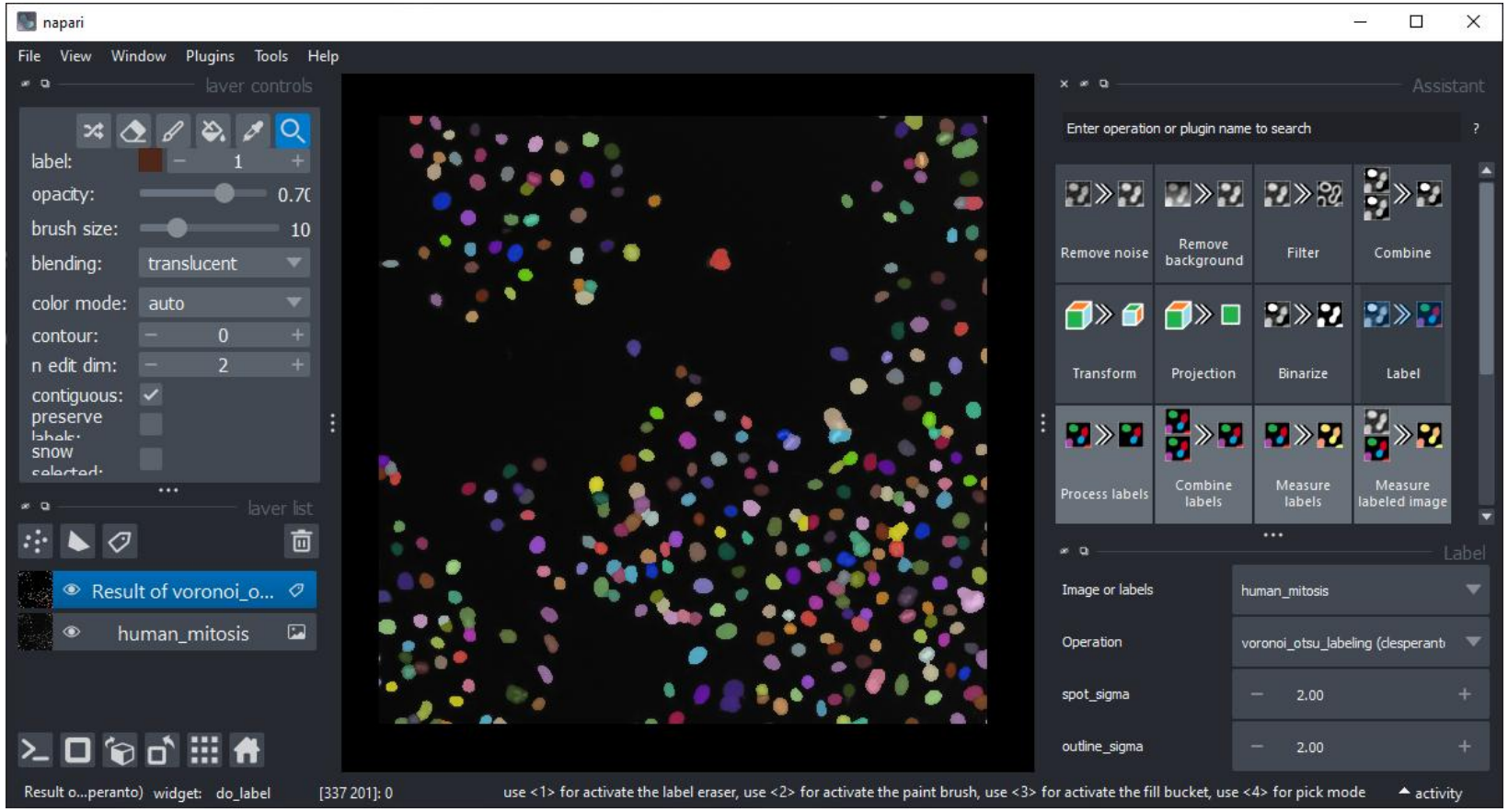
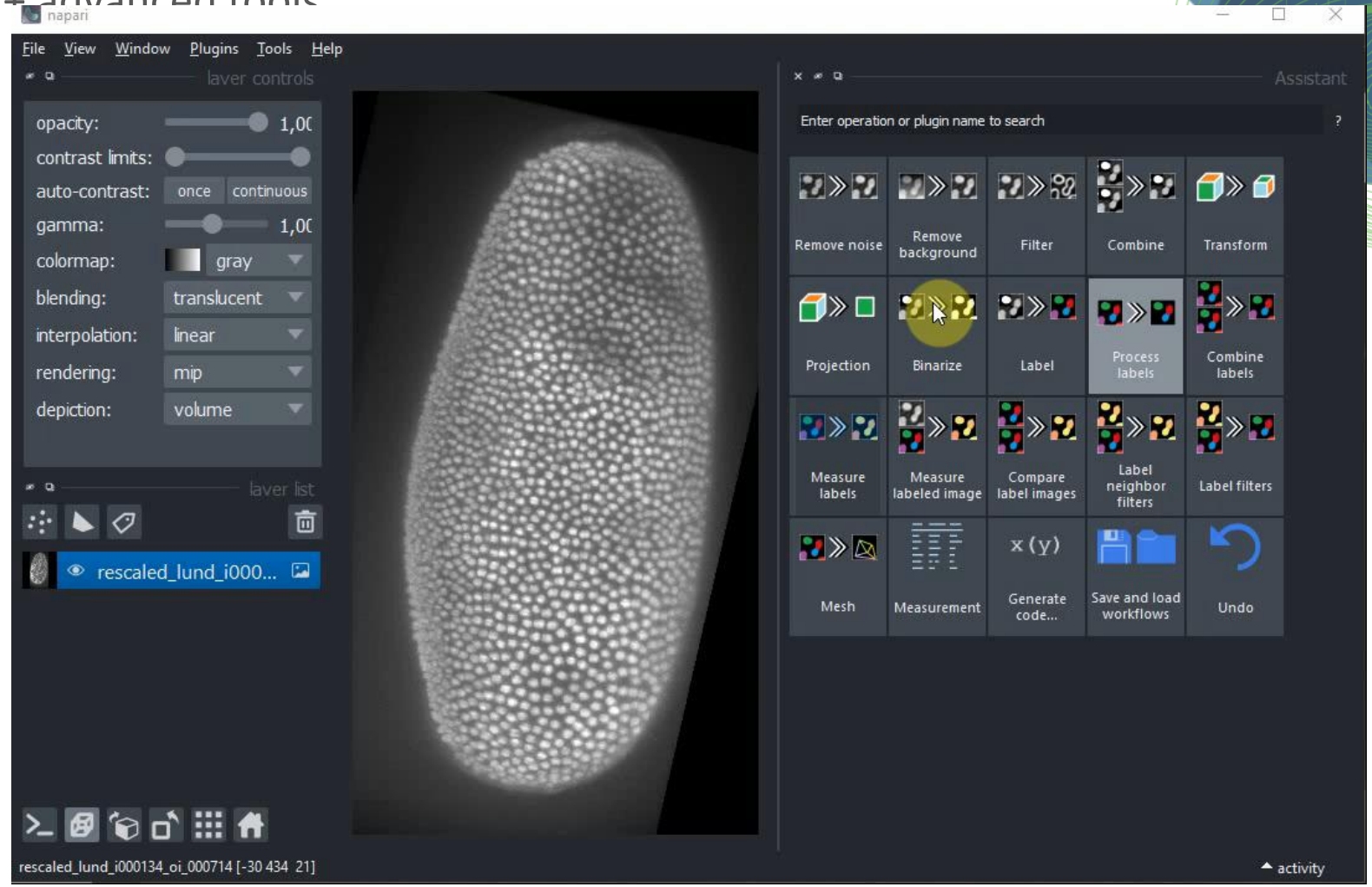


Image Processing

The Napari Assistant

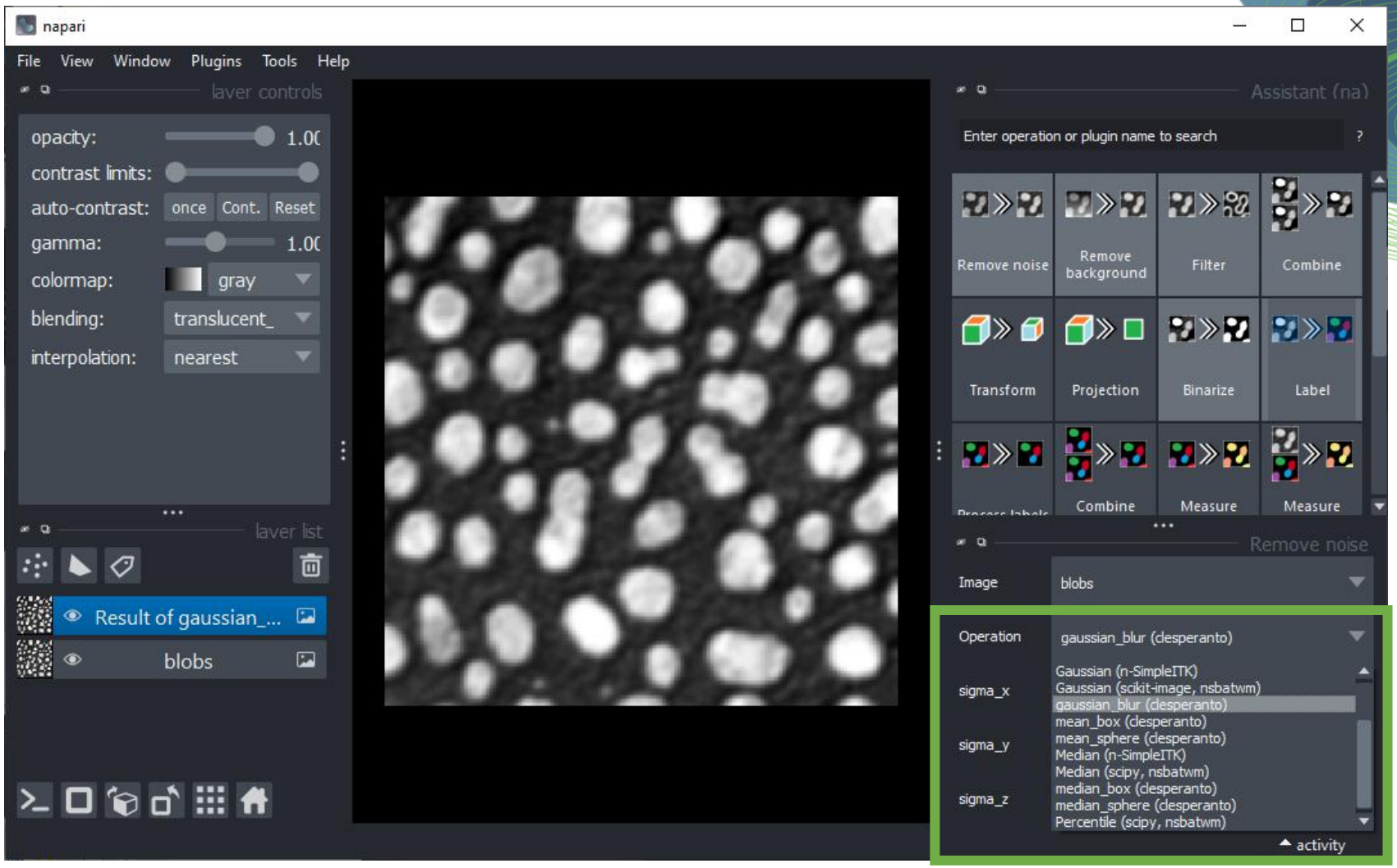
- Classical image processing operations + advanced tools
- Saving&loading supported
- Undo [redo]
- Hints for next steps
- …

Ryan Savill ()
@RyanSavill4



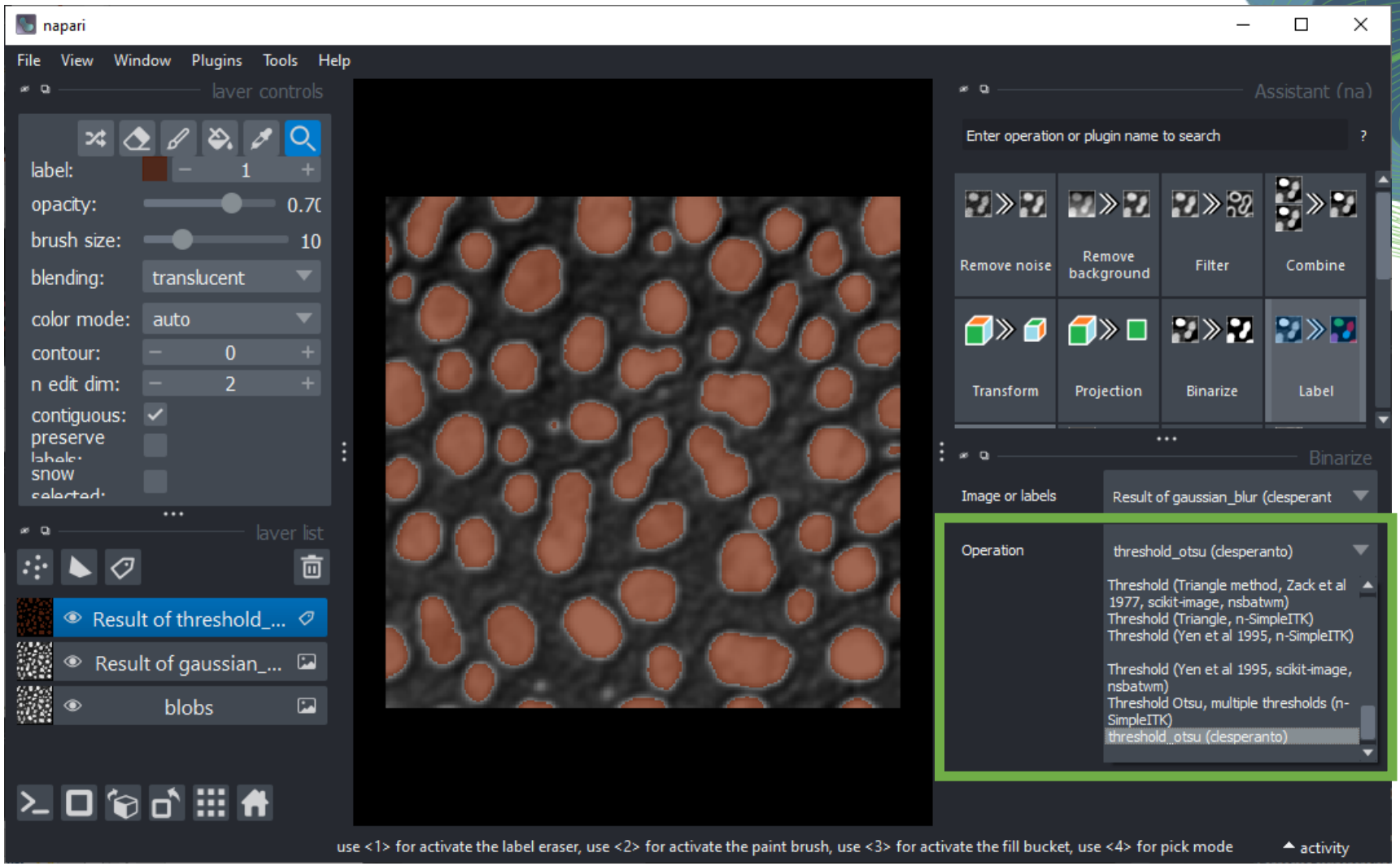
Workflow building

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



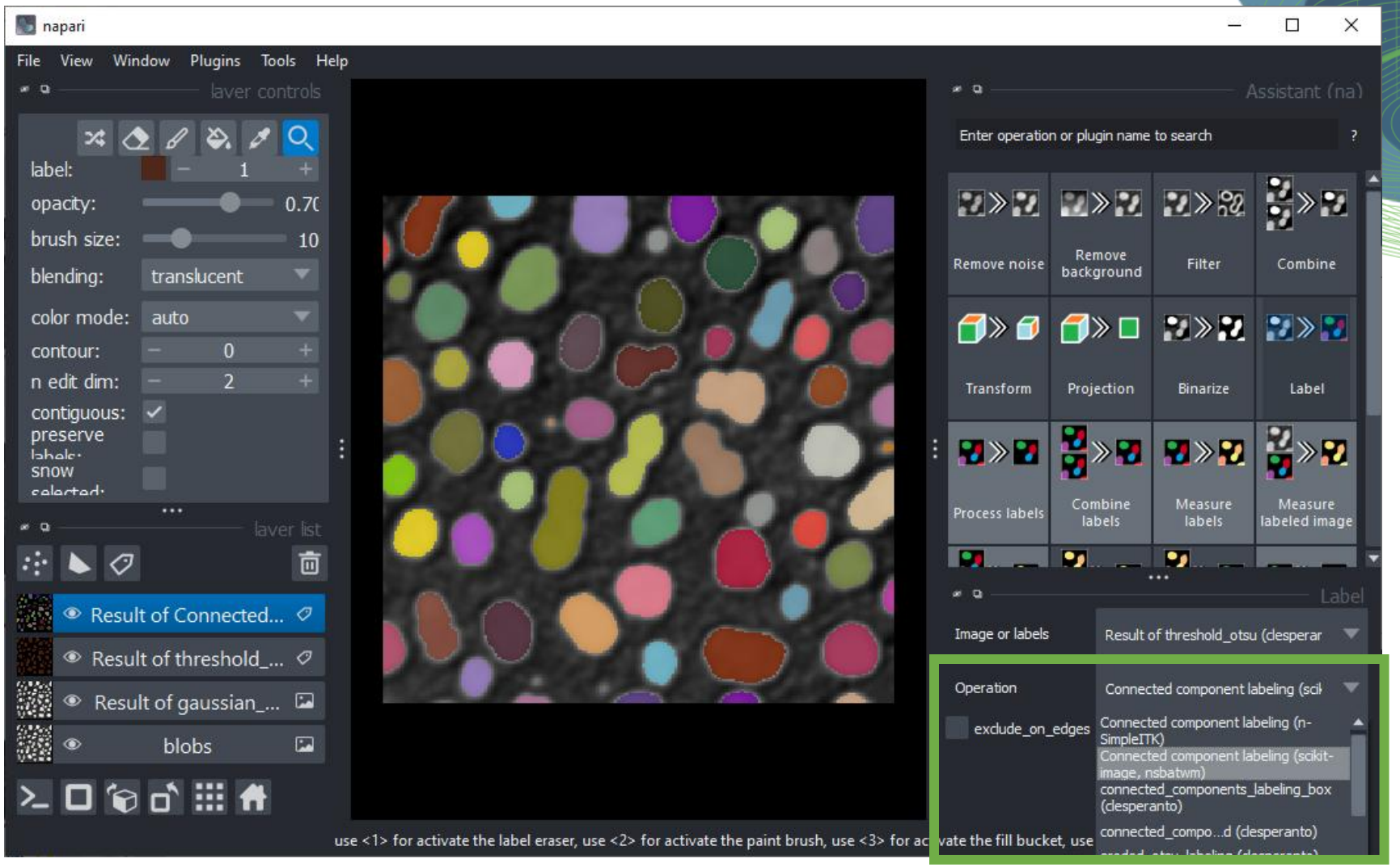
Workflow building

- Try different binarization algorithms

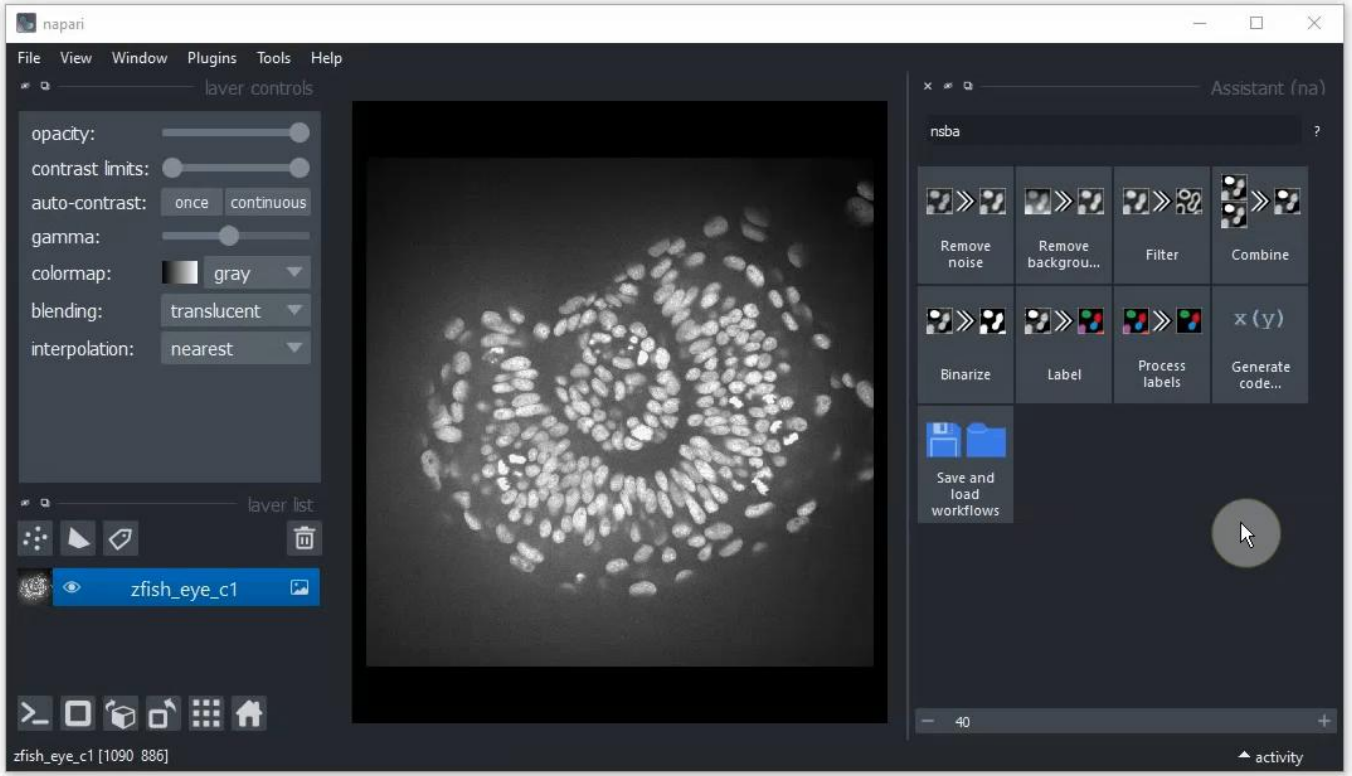


Workflow building

- Multiple labeling algorithms available



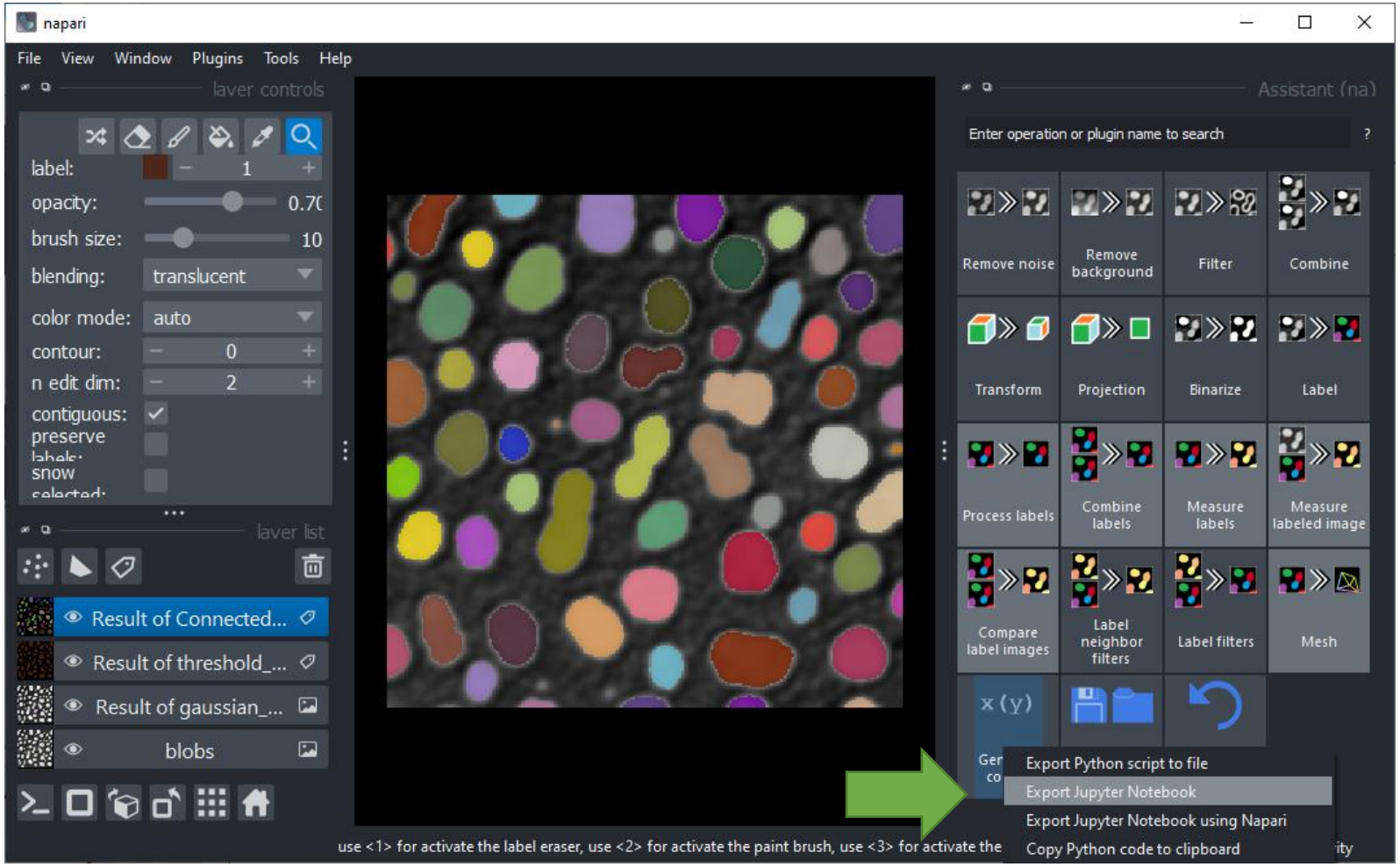
Export code to Jupyter Notebooks

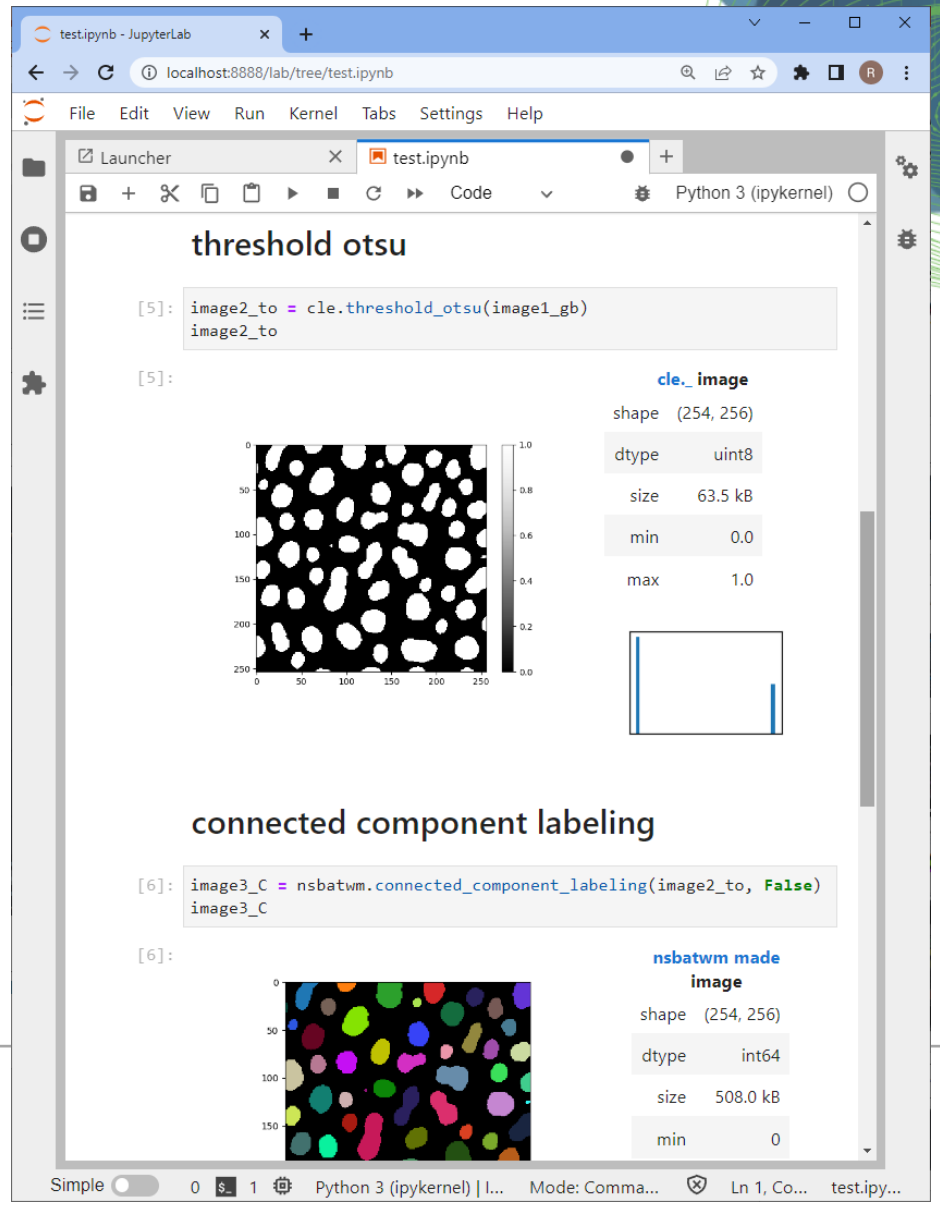


https://github.com/haesleinhuepf/napari-assistant

Image data source: Mauricio Rocha Martins, Norden lab, MPI CBG

Export code to Jupyter Notebooks





ScaDS.AI DRESDEN LEIPZIG

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

ScaDS.AI DRESDEN LEIPZIG

CENTER FOR SCALABLE DATA ANALYTICS AND ARTIFICIAL INTELLIGENCE

NFDI4 BIOIMAGE

NATIONAL RESEARCH DATA MANAGEMENT INFRASTRUCTURE FOR MICROSCOPY AND BIOIMAGE ANALYSIS

GloBIAS Global BioImage Analysts' Society

GLOBAL BIOIMAGE ANALYST'S SOCIETY

Reproducible Bio-Image Analysis using Python, Napari, Jupyter and AI

Robert Haase

Funded by Bundesministerium für Forschung, Technologie und Raumfahrt

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.

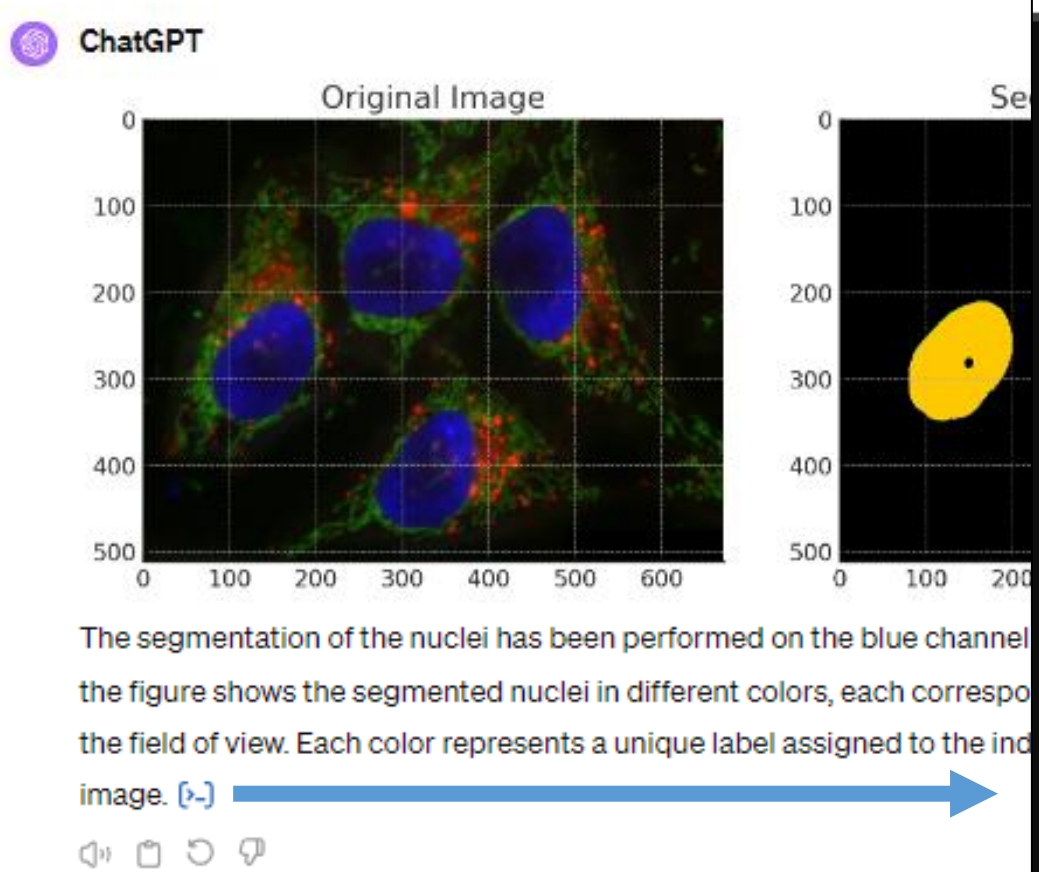
TECHNISCHE UNIVERSITÄT DRESDEN

UNIVERSITÄT LEIPZIG

Bio-image Analysis using Large-Language Models

- Analysing images in the browser, e.g. using ChatGPT

Prompt:
“Please segment the nuclei in the blue channel of the image”

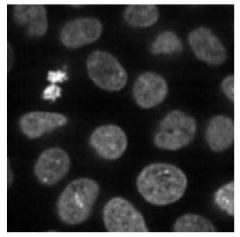


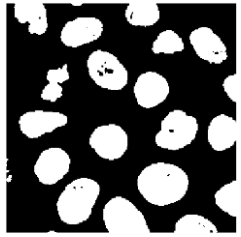
Analysis

```
from skimage import io, filters, measure, morphology
import matplotlib.pyplot as plt
import numpy as np

# Load the image
image = io.imread('/mnt/data/hela-cells-8bit.png')

# Since it's a color image, we'll just take the blue chann
blue_channel = image[:, :, 2]

# Threshold the image to get the binary image for nuclei
thresh = filters.threshold_otsu(blue_channel)
binary = blue_channel > thresh

# Remove small objects and perform morphological closing t
cleaned = morphology.remove_small_objects(binary, 50)
closed = morphology.closing(cleaned, morphology.disk(3))

# Label the image
label_image = measure.label(closed)

# Plot the original image and the binary image
fig, ax = plt.subplots(1, 2, figsize=(12, 6))
ax[0].imshow(image)
ax[0].set_title('Original Image')
ax[1].imshow(label_image, cmap='nipy_spectral')
ax[1].set_title('Segmented Nuclei')
plt.show()
```

How good are LLMs for Bio-image Analysis?

- Test case: segment the image and measure the average area of objects.





	area
0	955.0
1	31.0
2	815.0
3	1166.0
4	1135.0
⋮	

858.04

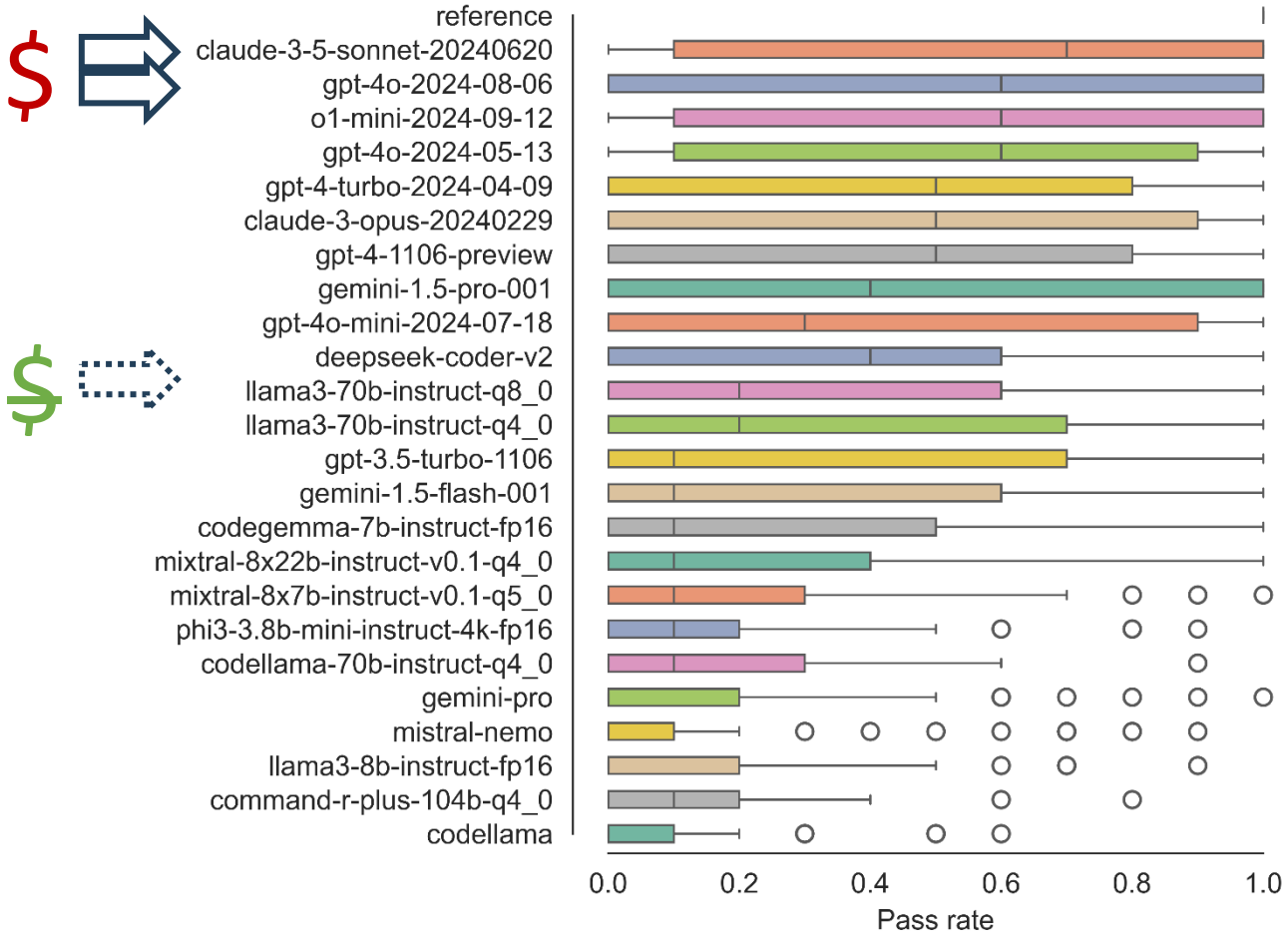
Unit-test pass-rate (n=10):

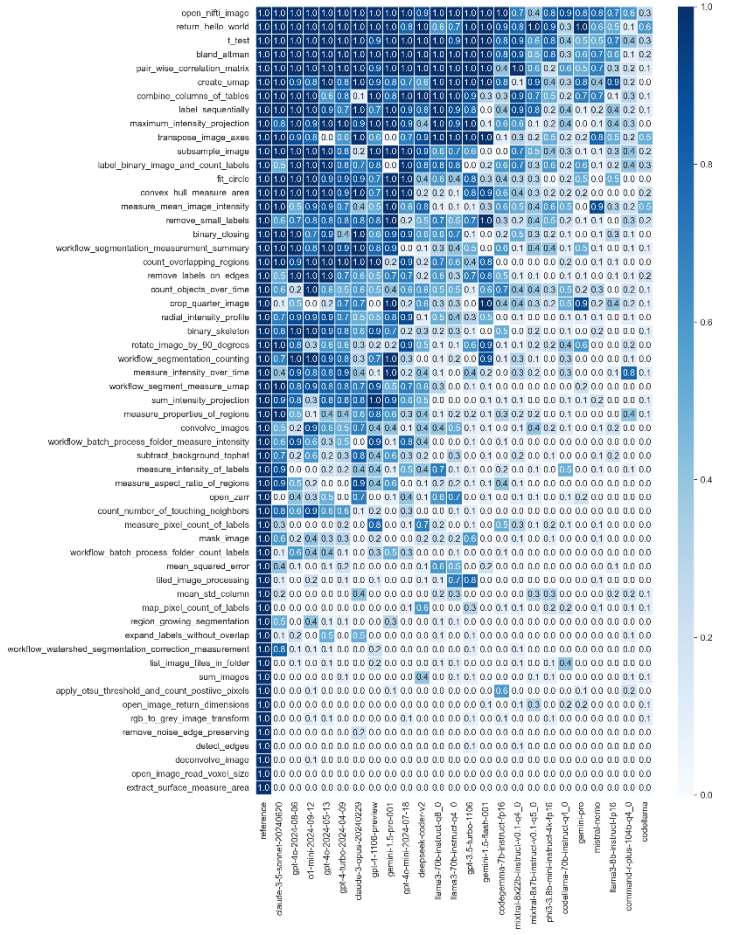
Large language models

workflow_segmentation_measurement_summary	reference	claude-3-5-sonnet-20240620	gpt-4o-2024-05-13	gpt-4-turbo-2024-04-09	claude-3-opus-20240229	gpt-4-1106-preview	gemini-1.5-pro-001	gpt-4o-mini-2024-07-18	llama3-70b-instruct-q8_0	llama3-70b-instruct-q4_0	gpt-3.5-turbo-1106	gemini-1.5-flash-001	codegemma-7b-instruct-fp16	mixtral-8x22b-instruct-v0.1-q4_0	mixtral-8x7b-instruct-v0.1-q5_0	phi3-3.8b-mini-instruct-4k-fp16	codellama-70b-instruct-q4_0	gemini-pro	llama3-8b-instruct-fp16	command-r-plus-104b-q4_0	codellama
	1.0	1.0	1.0	0.9	1.0	0.8	0.9	0.0	0.3	0.4	0.5	0.0	0.6	0.1	0.4	0.4	0.1	0.5	0.0	0.1	0.1

Benchmarking LLMs for Bio-image Analysis

- Summary: 57 use-cases (yet), 24 LLMs (yet), n=10





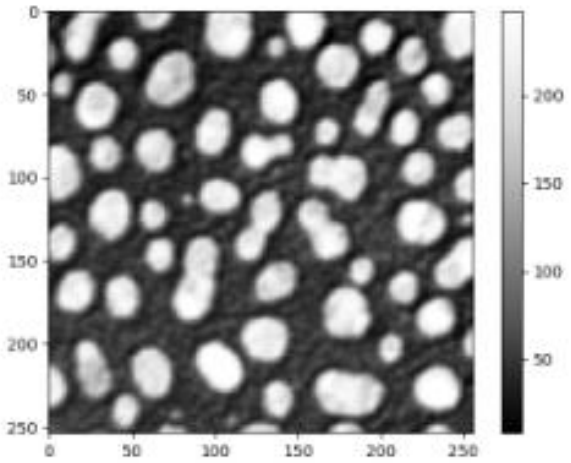
Scientific code writing: bia-bob

- First steps: Keep it simple

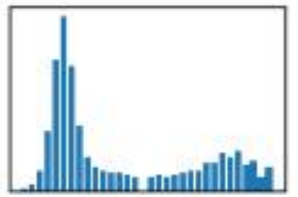
```
%bob Load the blobs.tif image file and show it.
```

I will load the `blobs.tif` image file and then display it using the appropriate visualization function.

```
from skimage.io import imread
import stackview

# Load the image file
filename = 'blobs.tif'
image = imread(filename)

# Display the image
stackview.insight(image)
```



shape	(254, 256)
dtype	uint8
size	63.5 kB
min	8
max	248



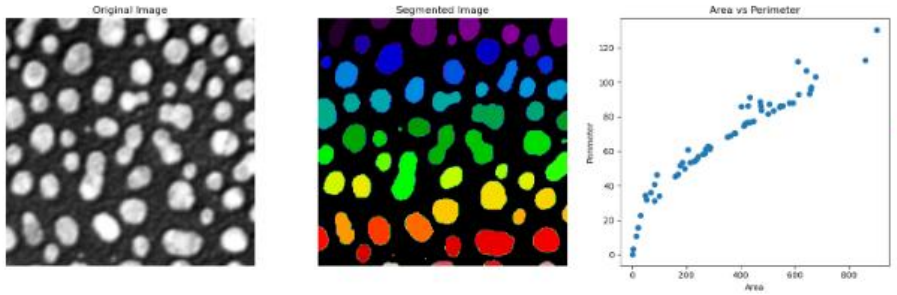
Scientific code writing: bia-bob

- After learning *how to prompt* for code: try more complex workflows

[1]:
```
import bia_bob
```

[2]:
```
%%bob
* Load blobs.tif from disk
* Segment the bright objects on the dark background
* Measure area and perimeter of the objects
* Show the original image, the segmented image and an
  area versus perimeter plot side-by-side using subfigures.
```

This notebook may contain text, code and images generated by artificial intelligence. Used model: claude-3-5-sonnet-20240620, vision model: claude-3-5-sonnet-20240620, endpoint: None, bia-bob version: 0.21.0.. Do not enter sensitive or private information and verify generated contents according to good scientific practice. Read more: https://github.com/haesleinhuepf/bia-bob#disclaimer

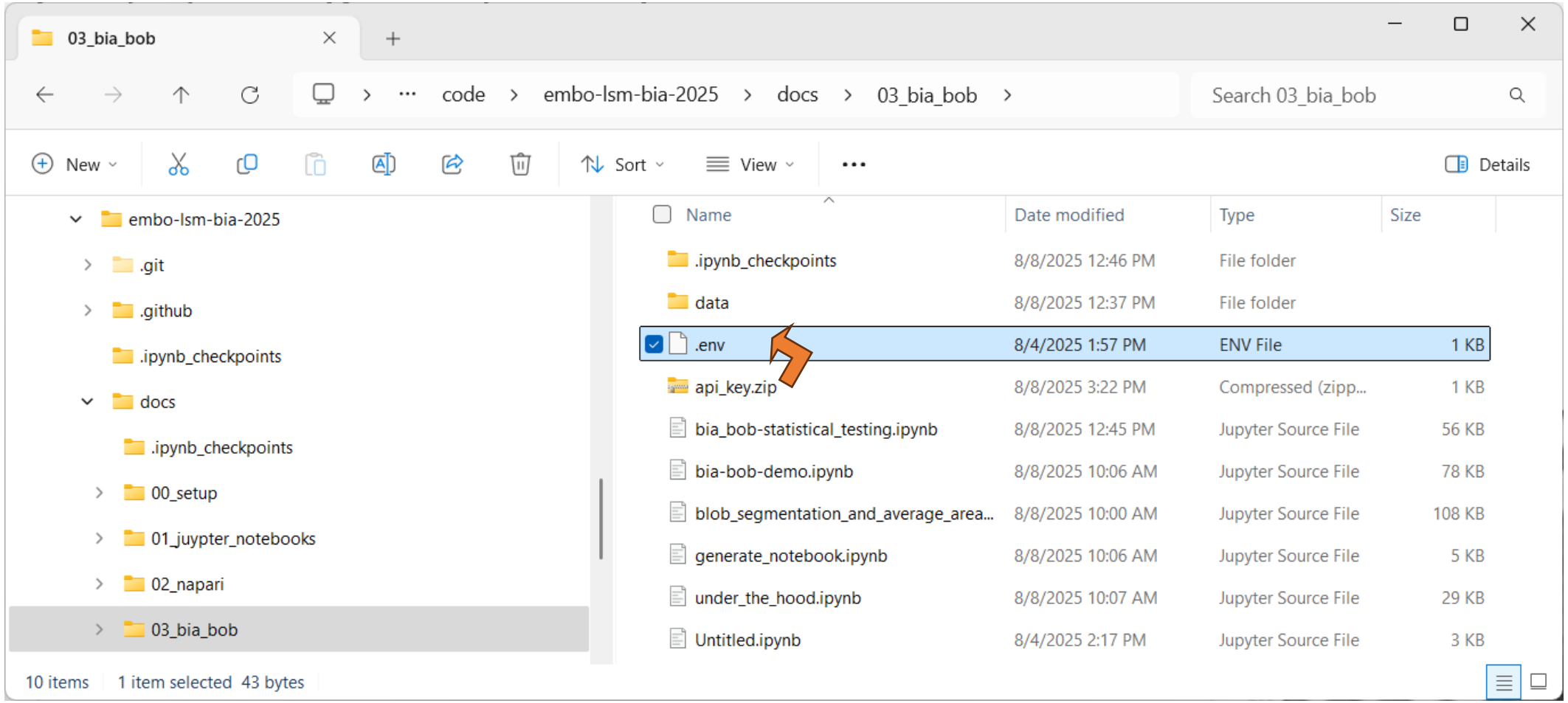


	label	area	perimeter
0	1	433.0	91.254834
1	2	185.0	53.556349
2	3	658.0	95.698485
3	4	434.0	76.870058
4	5	477.0	83.798990

```
import numpy as np
from skimage import io, filters, measure, segmentation
import matplotlib.pyplot as plt
import pandas as pd
import pyclesperanto_prototype as cle
import stackview

# Load the image
image = io.imread('blobs.tif')

# Segment the bright objects
thresh = filters.threshold_otsu(image)
binary = image > thresh

# Label the objects
labels = measure.label(binary)

# Measure properties
properties = ['label', 'area', 'perimeter']
df = pd.DataFrame(measure.regionprops_table(labels, properties=properties))

# Create the figure with subfigures
fig, (ax1, ax2, ax3) = plt.subplots(1, 3, figsize=(15, 5))

# Original image
ax1.imshow(image, cmap='gray')
ax1.set_title('Original Image')
ax1.axis('off')

# Segmented image
ax2.imshow(labels, cmap='nipy_spectral')
ax2.set_title('Segmented Image')
ax2.axis('off')

# Scatter plot
ax3.scatter(df['area'], df['perimeter'])
ax3.set_xlabel('Area')
ax3.set_ylabel('Perimeter')
ax3.set_title('Area vs Perimeter')

plt.tight_layout()
plt.show()

# Display the dataframe
print(df.head())
```

Setup: bia-bob

- Unzip the ".env" file out of "api_key.zip".

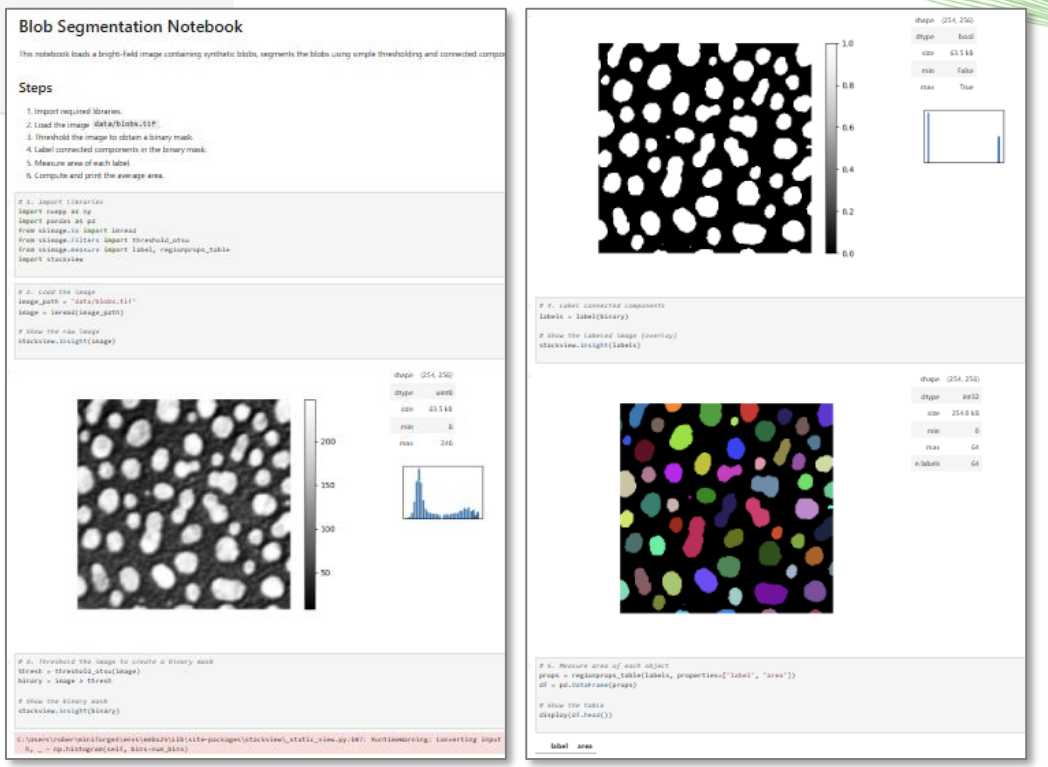


Generating Jupyter notebooks

- Ask %%bob to generate a Jupyter notebook

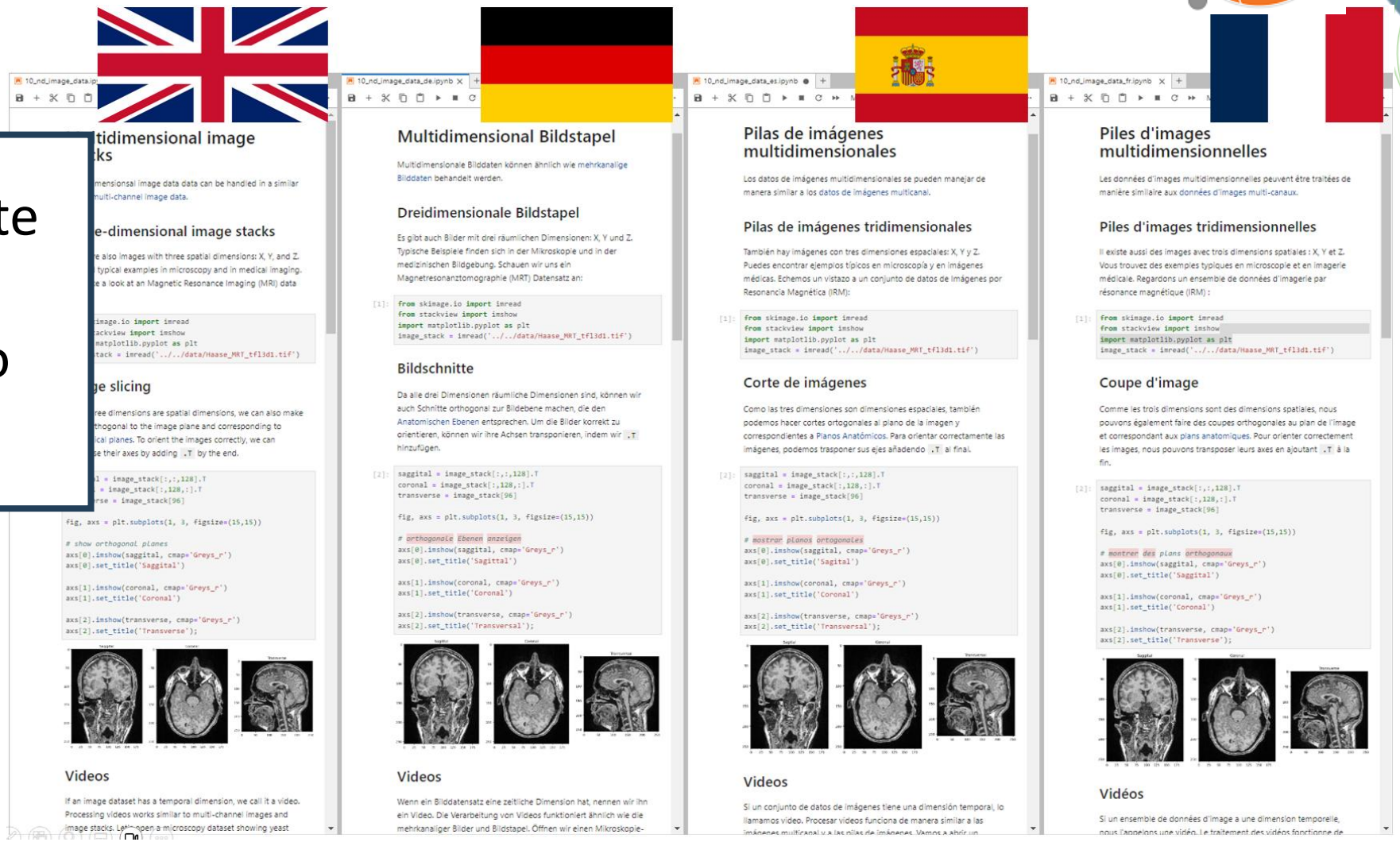
```
%%bob generate a jupyter notebook that
* loads data/blobs.tif
* segments the bright blobs using thresholding and connected component labeling
* measures the area of the objects
* prints out the average area
```

A notebook has been saved as blob_segmentation_and_average_area.ipynb.



Translating notebooks

"Please translate notebook <xyz.ipynb> to <language>."



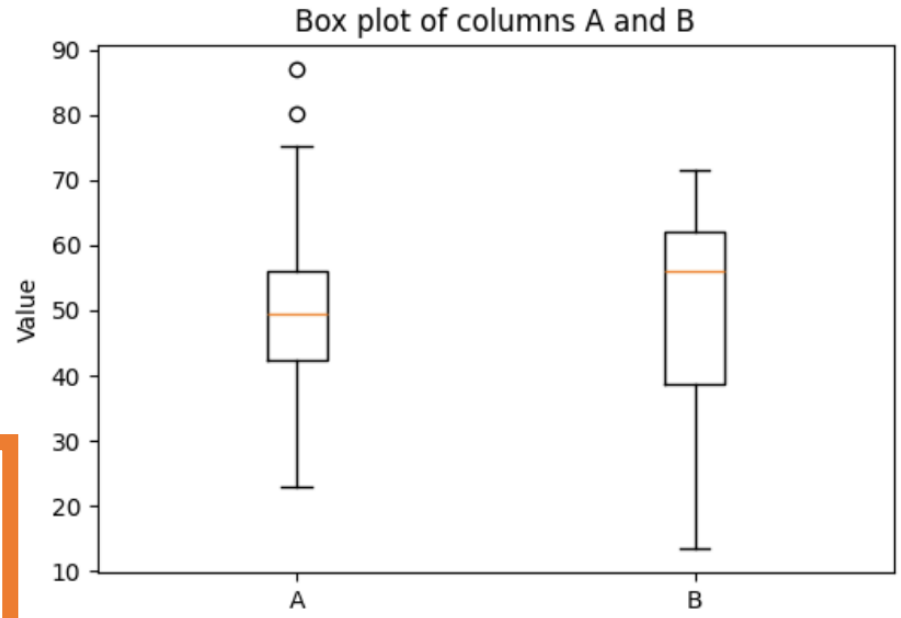
Code for statistical analysis

- Be careful when prompting for code and provide as many details as you can!

```
%%bob load data/data.csv and
determine if the columns A and B
are significantly different.
```

```python
import pandas as pd
import numpy as np
from scipy.stats import ttest_ind
import matplotlib.pyplot as plt

# 1. Load the CSV file
df = pd.read_csv('data/data.csv')

# 2. Keep only rows where both A and B are present
df = df[['A', 'B']].dropna()

# 3. Extract the two series
a = df['A'].values
b = df['B'].values

# 4. Perform independent two-sample t-test
t_stat, p_val = ttest_ind(a, b, equal_var=False)  # Welch's t-test

# 5. Output the results
print(f"t-statistic: {t_stat:.4f}")
print(f"p-value: {p_val:.4e}")
if p_val < 0.05:
    print("=> The difference between columns A and B is statistically significant (α=0.05).")
else:
    print("=> No statistically significant difference detected (α=0.05).")

# 6. Visual comparison: box-plot
plt.figure(figsize=(6, 4))
plt.boxplot([a, b], labels=['A', 'B'])
plt.title('Box plot of columns A and B')
plt.ylabel('Value')
plt.show()
```

```
t-statistic: -1.3287
p-value: 1.8428e-01
=> No statistically significant difference detected (α=0.05).
```



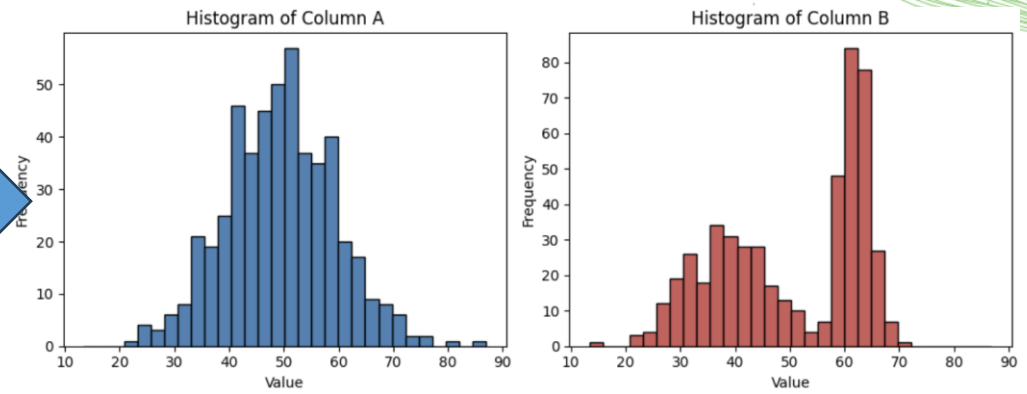
LLMs often provide “simple” solutions.

Code for statistical analysis

- Be careful when prompting for code and provide as many details as you can!

```
%%bob
Plot histograms for A and B
```

```python
# Assuming df, a, b are already defined from the previous step.
# If not, uncomment the following lines to load them again:
# import pandas as pd
# df = pd.read_csv('data/data.csv')[['A','B']].dropna()
# a = df['A'].values
# b = df['B'].values

# Determine a common bin range for both histograms
min_val = min(a.min(), b.min())
max_val = max(a.max(), b.max())
bins = 30  # you can adjust the number of bins

plt.figure(figsize=(10, 4))

# Histogram for column A
plt.subplot(1, 2, 1)
plt.hist(a, bins=bins, range=(min_val, max_val), color='steelblue', edgecolor='black')
plt.title('Histogram of Column A')
plt.xlabel('Value')
plt.ylabel('Frequency')

# Histogram for column B
plt.subplot(1, 2, 2)
plt.hist(b, bins=bins, range=(min_val, max_val), color='indianred', edgecolor='black')
plt.title('Histogram of Column B')
plt.xlabel('Value')
plt.ylabel('Frequency')

plt.tight_layout()
plt.show()
```



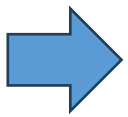
Look at your data!

Code for statistical analysis

- Be careful when prompting for code and provide as many details as you can!

```
%%bob
Please test again if columns A and B are significantly different.
Before applying a statistical test, ensure that the necessary
pre-conditions are met.
```

```
from scipy.stats import shapiro, levene, mannwhitneyu
# Note: pandas (pd), numpy (np), and ttest_ind are already available.

# 1. Load data (keep original df/a/b untouched)
data_path = 'data/data.csv'
df_raw = pd.read_csv(data_path)
# keep only rows where both A and B are present
df_clean = df_raw[['A', 'B']].dropna()
a_vals = df_clean['A'].values
b_vals = df_clean['B'].values

# 2. Normality tests (Shapiro-Wilk)
shapiro_a_stat, shapiro_a_p = shapiro(a_vals)
shapiro_b_stat, shapiro_b_p = shapiro(b_vals)

# 3. Equality of variances (Levene)
lev_stat, lev_p = levene(a_vals, b_vals)

# 4. Choose appropriate hypothesis test
if shapiro_a_p > 0.05 and shapiro_b_p > 0.05:
    # both distributions look normal
    if lev_p > 0.05:
        test_name = "Student's t-test (equal variances)"
        t_stat, p_val = ttest_ind(a_vals, b_vals, equal_var=True)
    else:
        test_name = "Welch's t-test (unequal variances)"
        t_stat, p_val = ttest_ind(a_vals, b_vals, equal_var=False)
else:
    # at least one non-normal → use non-parametric test
    test_name = "Mann-Whitney U test (non-parametric)"
    # two-sided test
    t_stat, p_val = mannwhitneyu(a_vals, b_vals, alternative='two-sided')

# 5. Print report
print("=== Assumption checks ===")
print(f"Shapiro-Wilk A:  statistic={shapiro_a_stat:.4f}, p={shapiro_a_p:.4e} "
      f"{'normal' if shapiro_a_p>0.05 else 'non-normal'}")
print(f"Shapiro-Wilk B:  statistic={shapiro_b_stat:.4f}, p={shapiro_b_p:.4e} "
      f"{'normal' if shapiro_b_p>0.05 else 'non-normal'}")
print(f"Levene variance test: statistic={lev_stat:.4f}, p={lev_p:.4e} "
      f"{'equal variances' if lev_p>0.05 else 'unequal variances'}")
print("\n=== Final hypothesis test ===")
print(f"Test used: {test_name}")
print(f"Statistic = {t_stat:.4f}")
print(f"P-value   = {p_val:.4e}")
if p_val < 0.05:
    print("=> Significant difference between co
else:
    print("=> No significant difference detected
```



```
=== Assumption checks ===
Shapiro-Wilk A:  statistic=0.9972, p=5.6236e-01 normal
Shapiro-Wilk B:  statistic=0.8937, p=4.0635e-18 non-normal
Levene variance test: statistic=80.6718, p=1.3016e-18 unequal variances

=== Final hypothesis test ===
Test used: Mann-Whitney U test (non-parametric)
Statistic = 113657.0000
P-value   = 1.3000e-02
=> Significant difference between columns A and B (α=0.05).
```

LLMs cannot be *brave scientists* yet, but you can!

Acknowledgements

Collaborators & contributors

Christian Tischer (EMBL Heidelberg), Jean-Karim Hériché (EMBL Heidelberg), Nico Scherf (MPI CBS Leipzig), Loïc A. Royer (CZ Biohub), Elena Nicolay (UFZ Leipzig), Kevin Yamauchi (ETH Zurich / Basel), Seth Hinz, Teun A.P.M. Huijben, Jordão Bragantini (CZ Biohub San Francisco), llan Theodoro (CZ Biohub and the State University of Campinas, Brazil), Ian C. (University of British Columbia), Pradeep Rajasekhar (WEHI Australia)

Communities & platforms

GloBIAS Global BioImage Analysts' Society

NFDI4 BIOIMAGE

image.sc

BIDS Team @ ScaDS.AI / UL

Lea Kabjesz, Lea Gihlein, Mara Lampert

https://scads.github.io/embo-lsm-bia-2025

Funding

Bundesministerium für Forschung, Technologie und Raumfahrt

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.

DFG Deutsche Forschungsgemeinschaft

RH acknowledges the financial support by the Federal Ministry of Education and Research of Germany and by Sächsische Staatsministerium für Wissenschaft, Kultur und Tourismus in the programme Center of Excellence for AI-research „Center for Scalable Data Analytics and Artificial Intelligence Dresden/Leipzig“, project identification number: ScaDS.AI