

Bio-image analysis, bio-statistics, programming and machine learning for computational biology

Anna Poetsch, Melissa Sanabria, Allyson Ryan, Robert Haase



Anna Poetsch, Melissa Sanabria, Allyson Ryan, Robert Haase

- We will focus on Python programming.
- Goal: **Allow you to do things automatically instead of suffering long time when doing it manually.**

Basics

```
b = 3  
c = a + b
```

```
print(c)
```

```
8
```

```
d = 6  
e = 7  
f = a * d  
g = f / e  
h = 1 + g
```

```
print(h)
```

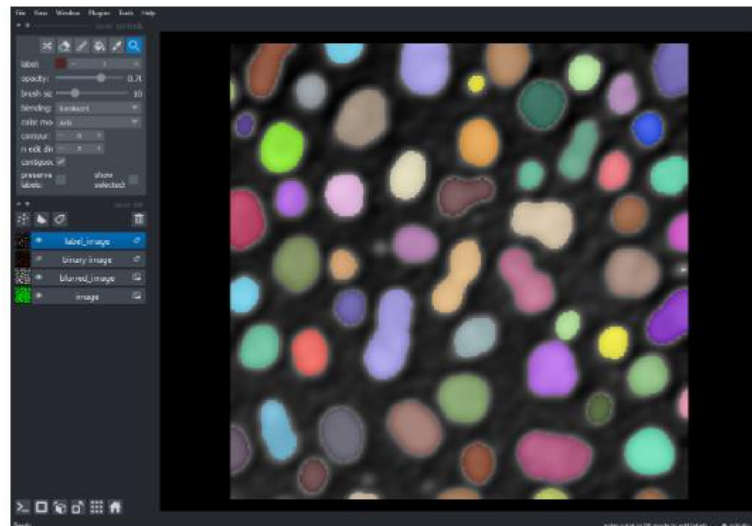
```
5.285714285714286
```

Image Analysis

```
from skimage.measure import label  
label_image = label(binary_image)  
  
# add labels to viewer  
label_layer = viewer.add_labels(label_image)
```

You can visualize labelled objects as overlay (per default)

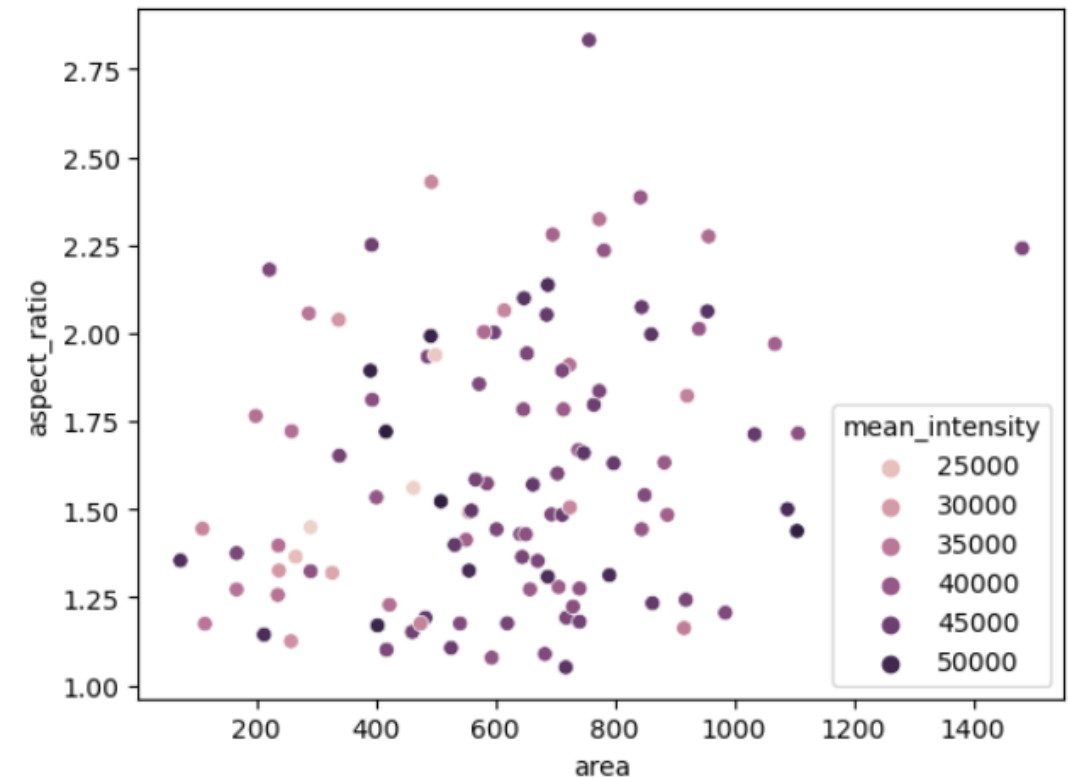
```
napari.utils.nbscreenshot(viewer)
```



Plotting / statistics

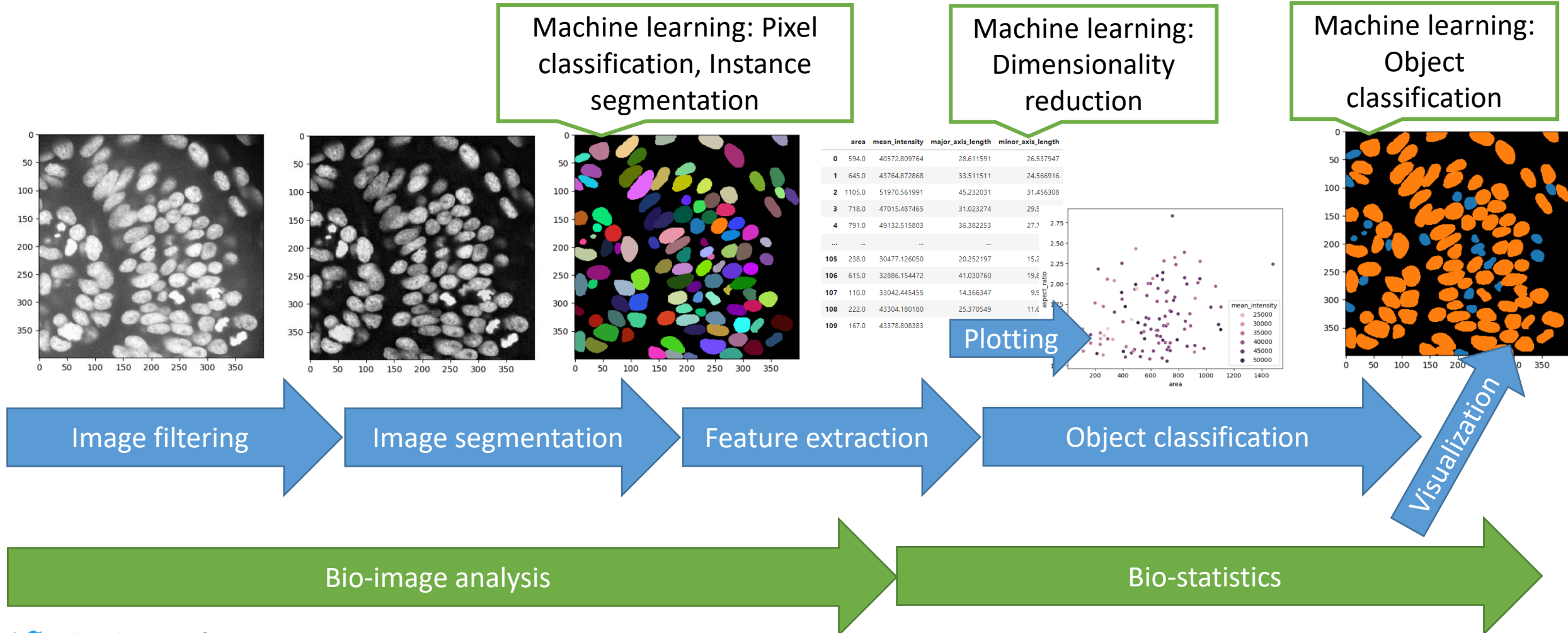
```
seaborn.scatterplot(dataframe, x='area', y='aspect_ratio', hue='mean_intensity')
```

```
<AxesSubplot: xlabel='area', ylabel='aspect_ratio'>
```

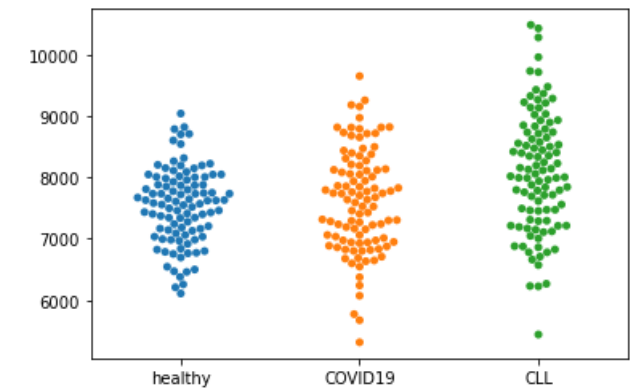
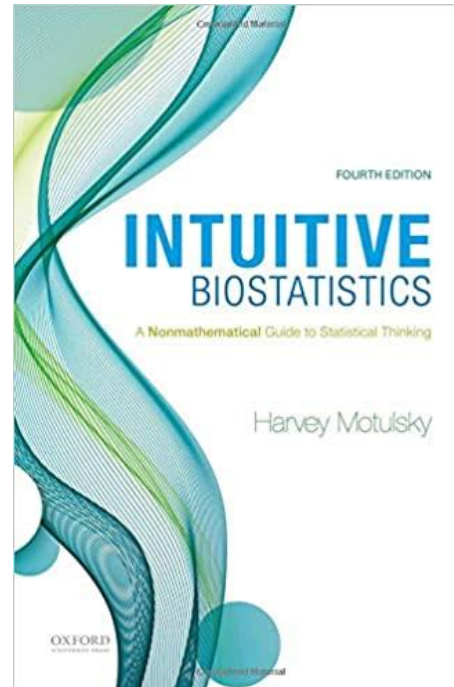
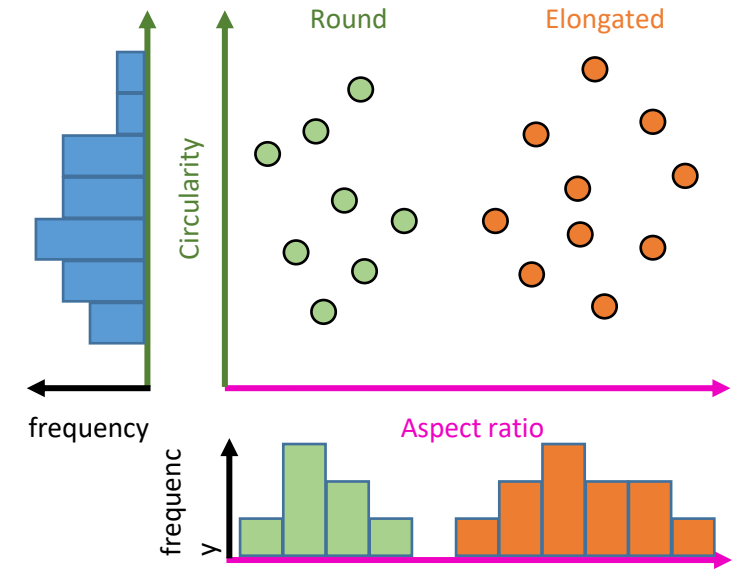
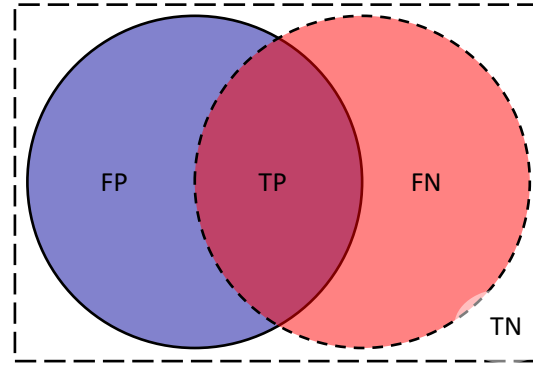


Lecture overview: Bio-image Analysis

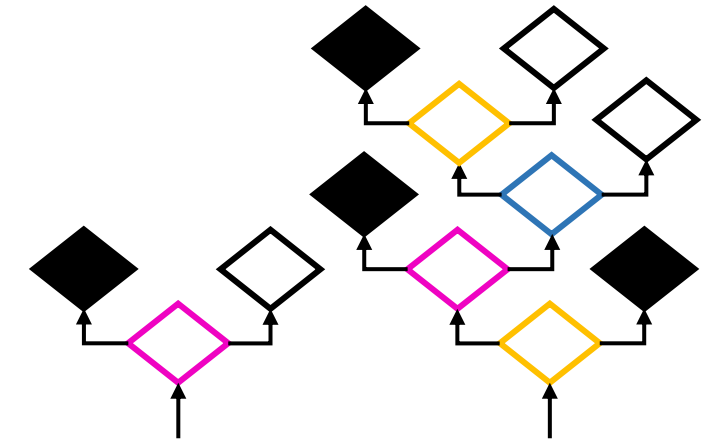
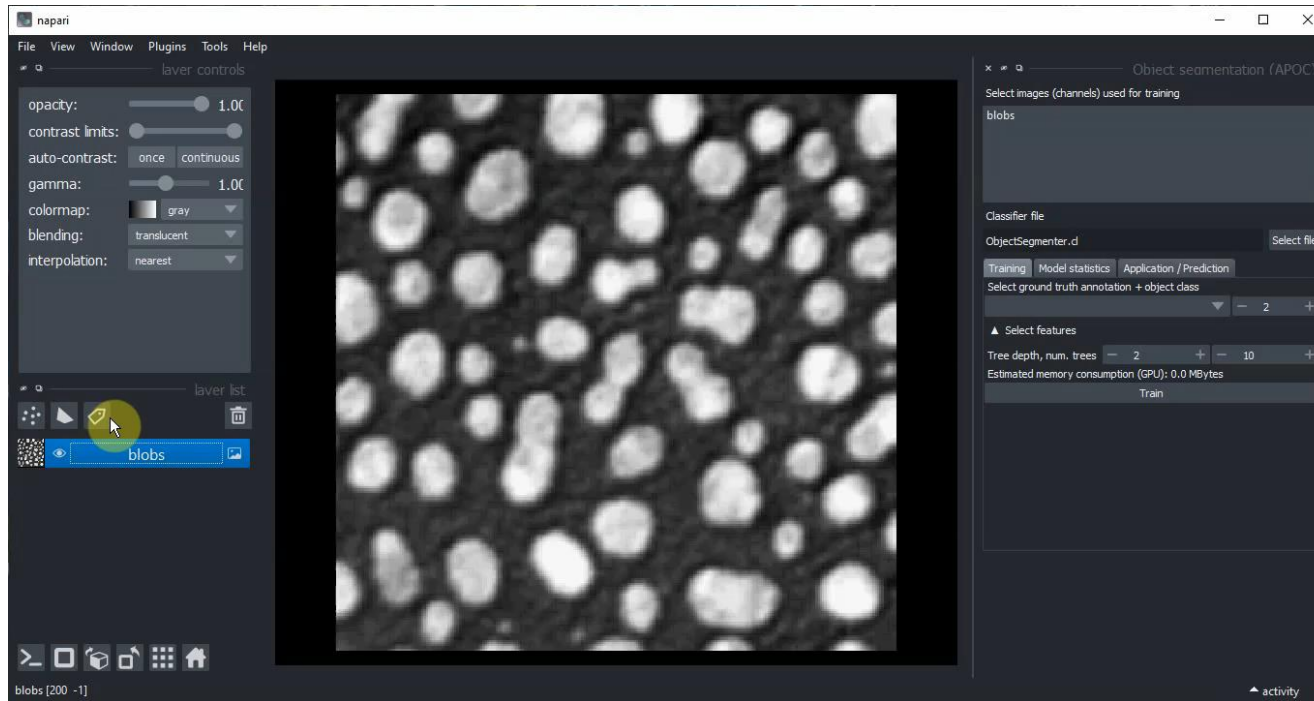
- Image Data Analysis workflows
- Goal: **Quantify observations, substantiate conclusions with numbers**



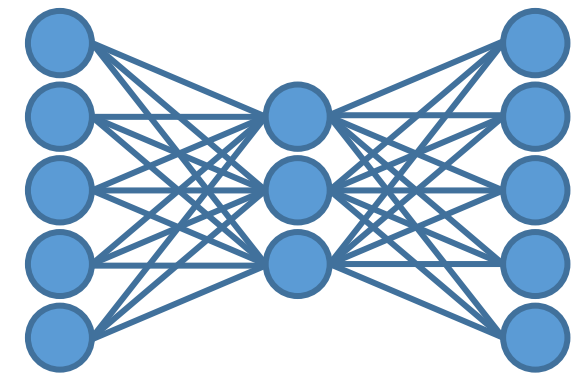
- Descriptive statistics
 - Distributions
 - Inferential statistics
 - Hypothesis testing
 - Multiple comparisons and correlations
 - Clustering, dimensionality reduction
-
- Goal: Allow you to draw conclusions from quantified observations.



- Machine learning
 - in the context of image analysis and genetics
- Computers can *learn* from data, potentially revealing relationships that are not obvious to a human
- Goal: **Give you an insight into state-of-the-art methods**



Random forest classifiers



Neural networks

- 4.4.2022 - General introduction, introduction to Python Programming I
- 11.4.2022 - Introduction to Python programming II
- 18.4.2022 - Image Processing
- 25.4.2022 - Image Segmentation + Quality Assurance
- 2.5.2022 - Feature extraction
- 9.5.2022 - Introduction to Biostatistics
- 16.5.2022 - Descriptive Statistics
- 23.5.2022 – Hypothesis Testing
- 6.6.2022 – Introduction to Machine Learning + Random Forest Classifiers
- 13.6.2022 – Unsupervised Machine Learning
- 20.6.2022 – Supervised Machine Learning / Deep Learning
- 27.6.2022 – Introduction to Genomic Data
- 4.7.2022 – Multimodal Machine Learning
- 11.7.2022 – Summary, exam preparation

- Every week will follow the same rough scheme
 - 13:00 : 90 min lecture
 - 14:20 : 90 min exercises
 - when you're done, enjoy the sun!
- Exam will cover the semester content accordingly
 - Theory of
 - image analysis,
 - statistics and
 - machine learning
 - Basics of programming
 - write simple < 10 line programs and
 - read code and describe what it does
 - “closed book exam”

In which category do you see yourself?

Molecular
Bioengineering
Bachelor /
Master student

Other
Bachelor /
Master
student

PhD
student

Other



Introduction to Bio-Image Analysis

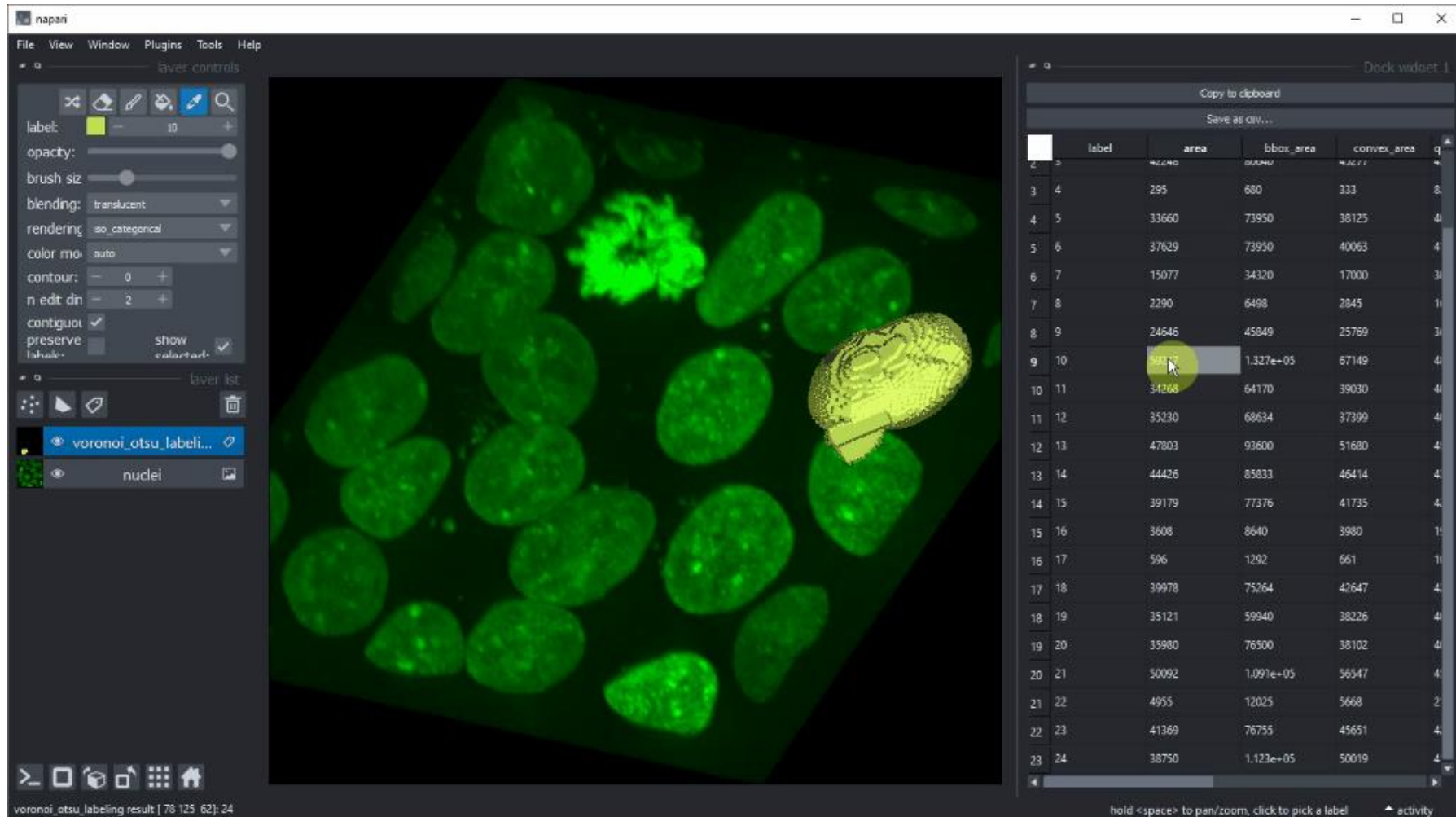
Robert Haase

- Deriving quantitative information from images of biological samples taken with microscopes

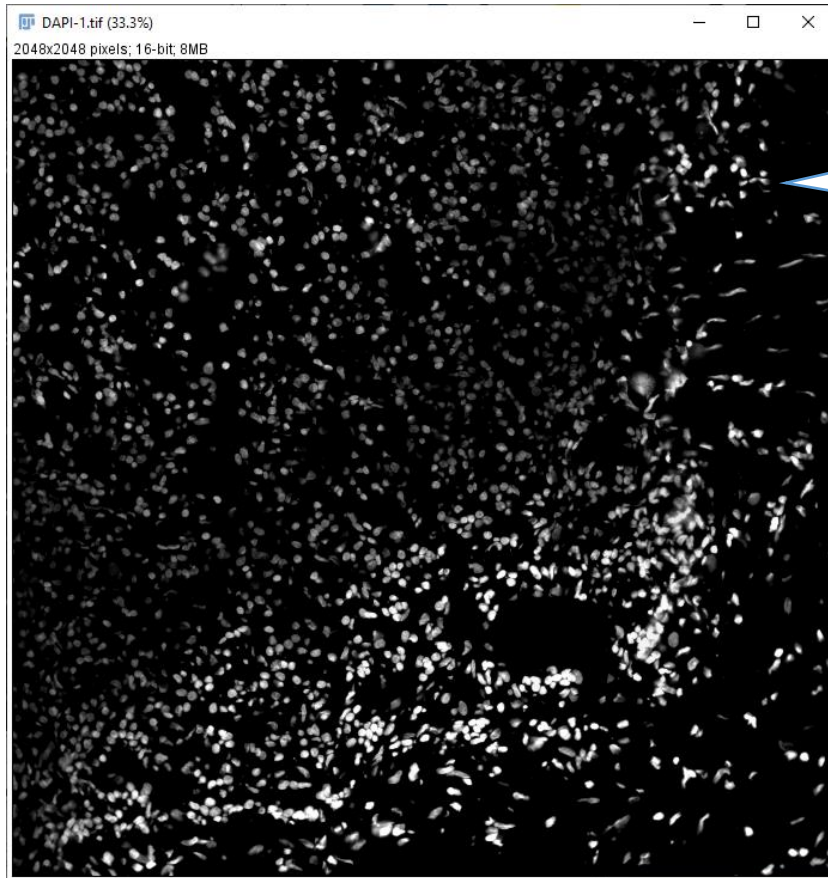


cat height = 1.5 x microscope height

- Deriving quantitative information from images of biological samples taken with microscopes + visualization



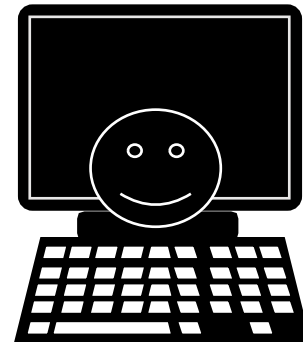
- Measurements should be objective, not influenced by human interpretation



Nuclei in this
image are ...

... more dense
than in this image.

Use automation for
less subjective
analysis.



April 2023

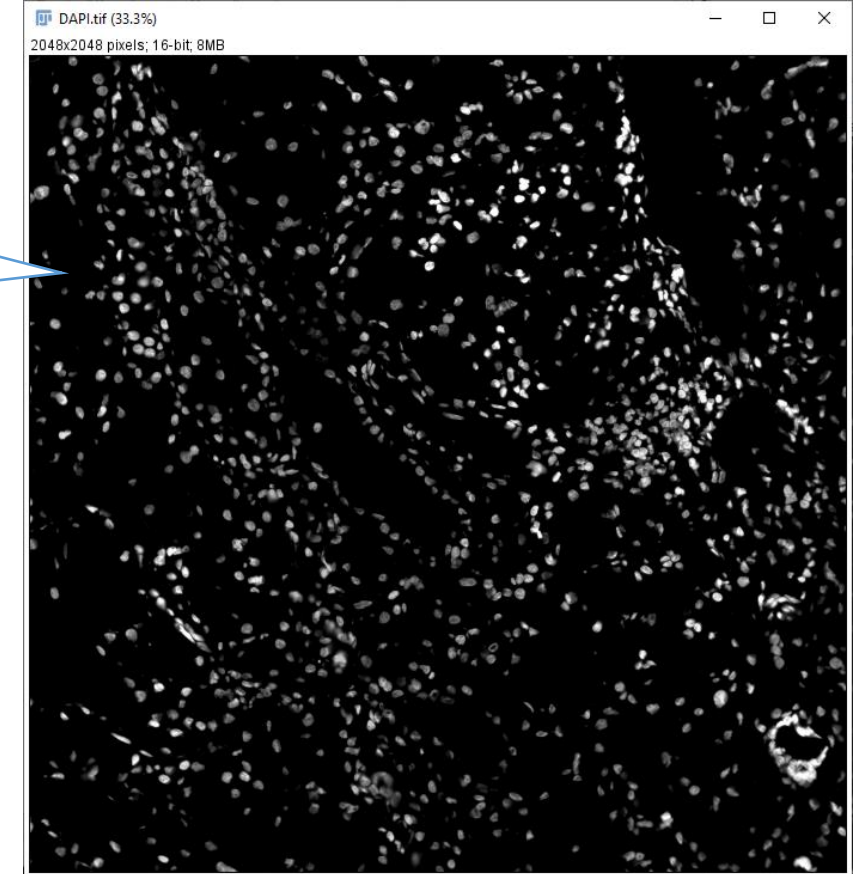
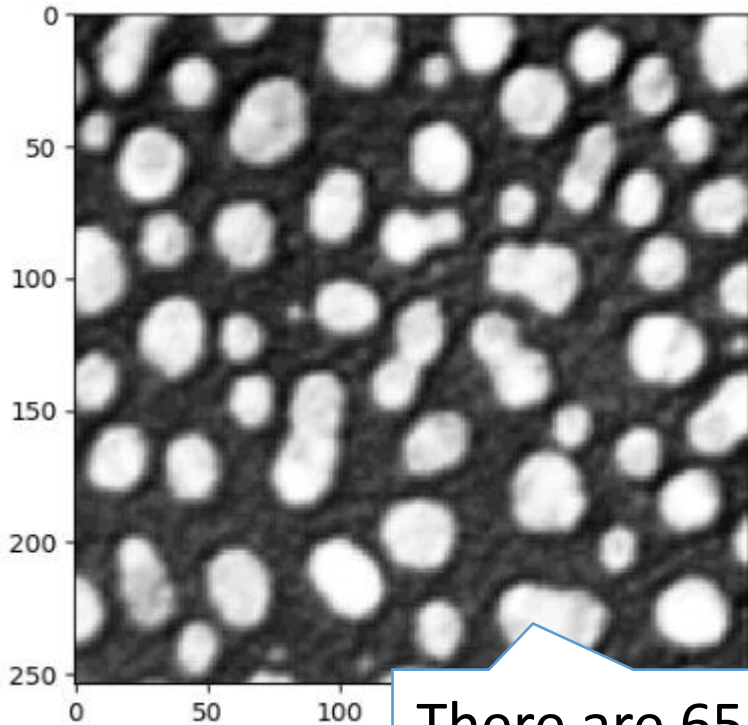


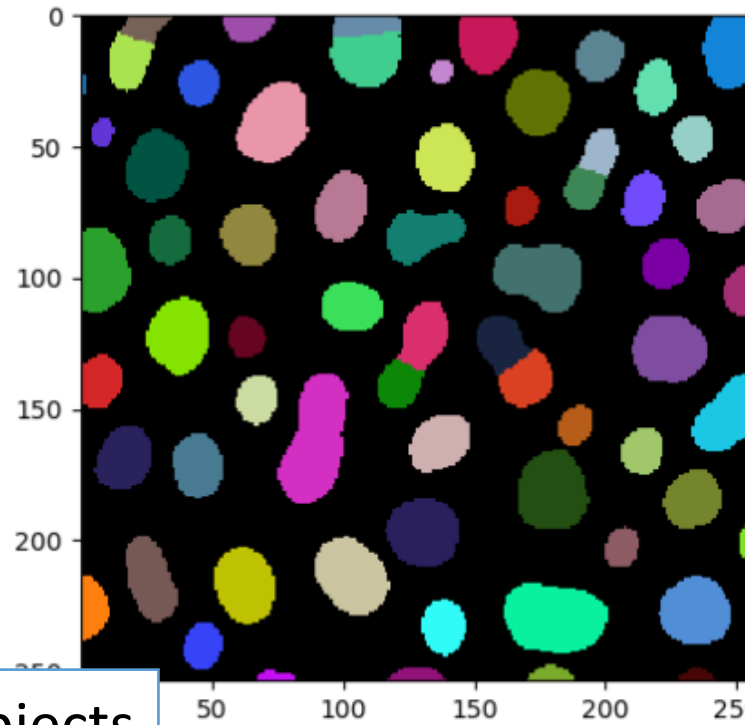
Image data source: Pascual-Reguant, Anna. (2021). Immunofluorescence staining of a human kidney (#2, peri-tumor area) obtained by MELC [Data set]. Zenodo. <http://doi.org/10.5281/zenodo.4434462> licensed [CC-BY 4.0](https://creativecommons.org/licenses/by/4.0/)

- Algorithms must be reliable (trustworthy).
- Visualization helps gaining trust in automated methods.

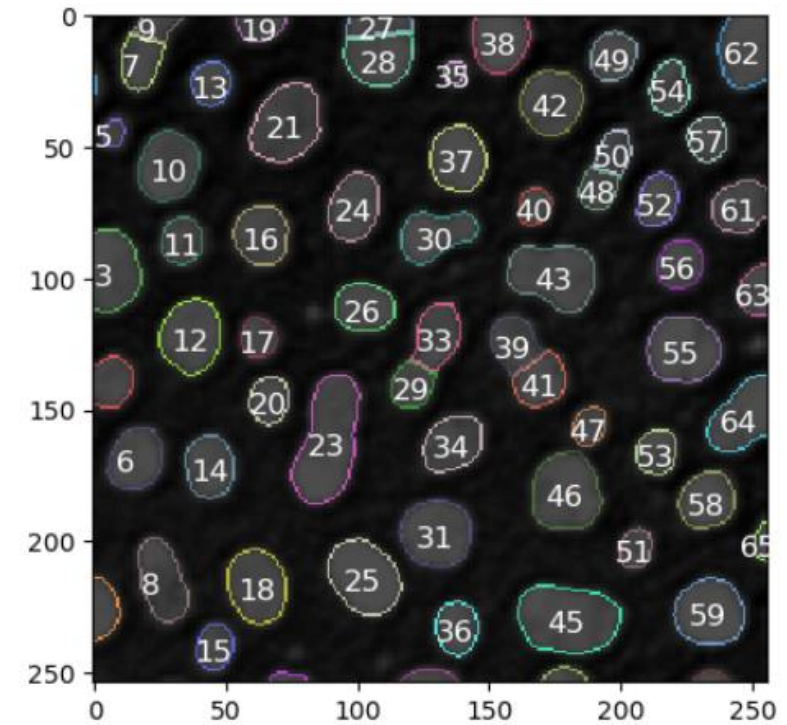
Original image



Label image



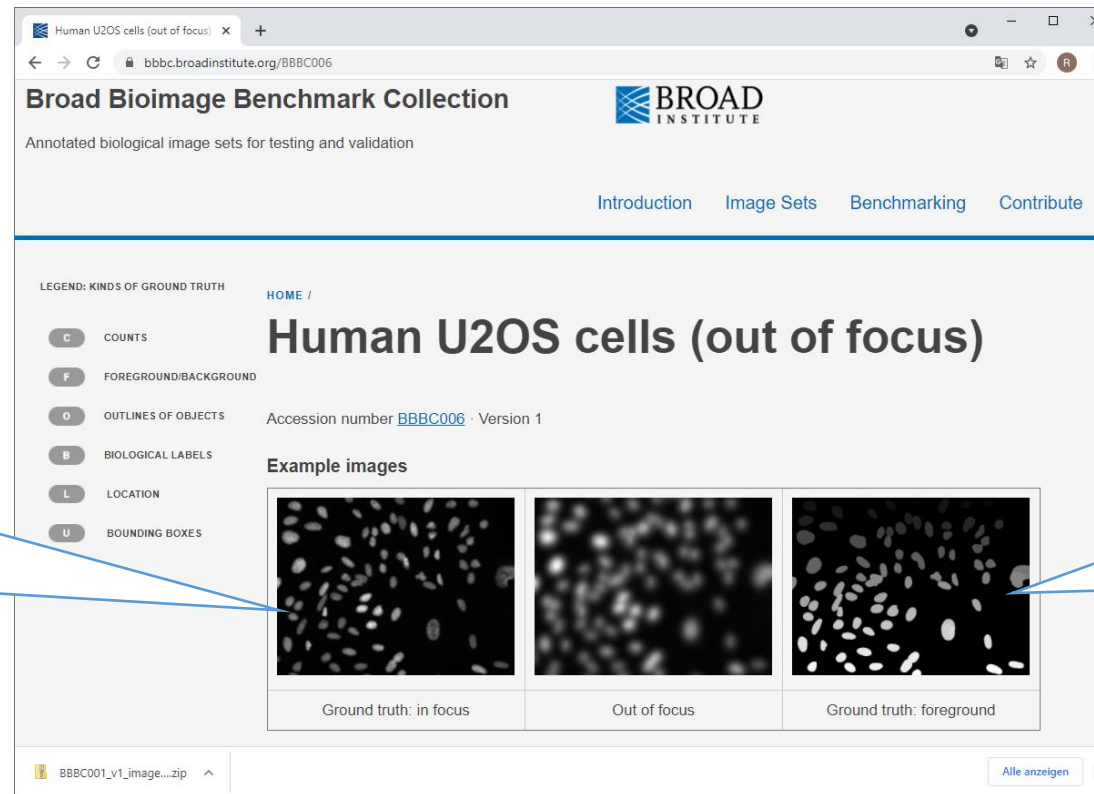
Overlay



There are 65 objects
in this image.

Source: M. Zoccoler & R. Haase licensed [CC-BY](https://creativecommons.org/licenses/by/4.0/)
https://haesleinhuepf.github.io/BioImageAnalysisNotebooks/60_data_visualization/overlay_text_on_image.html

- Algorithms must be reliable (validated methods).
- Publicly available benchmark data sets allow to compare algorithms on common data.



Original image
data

“Ground truth”
label images

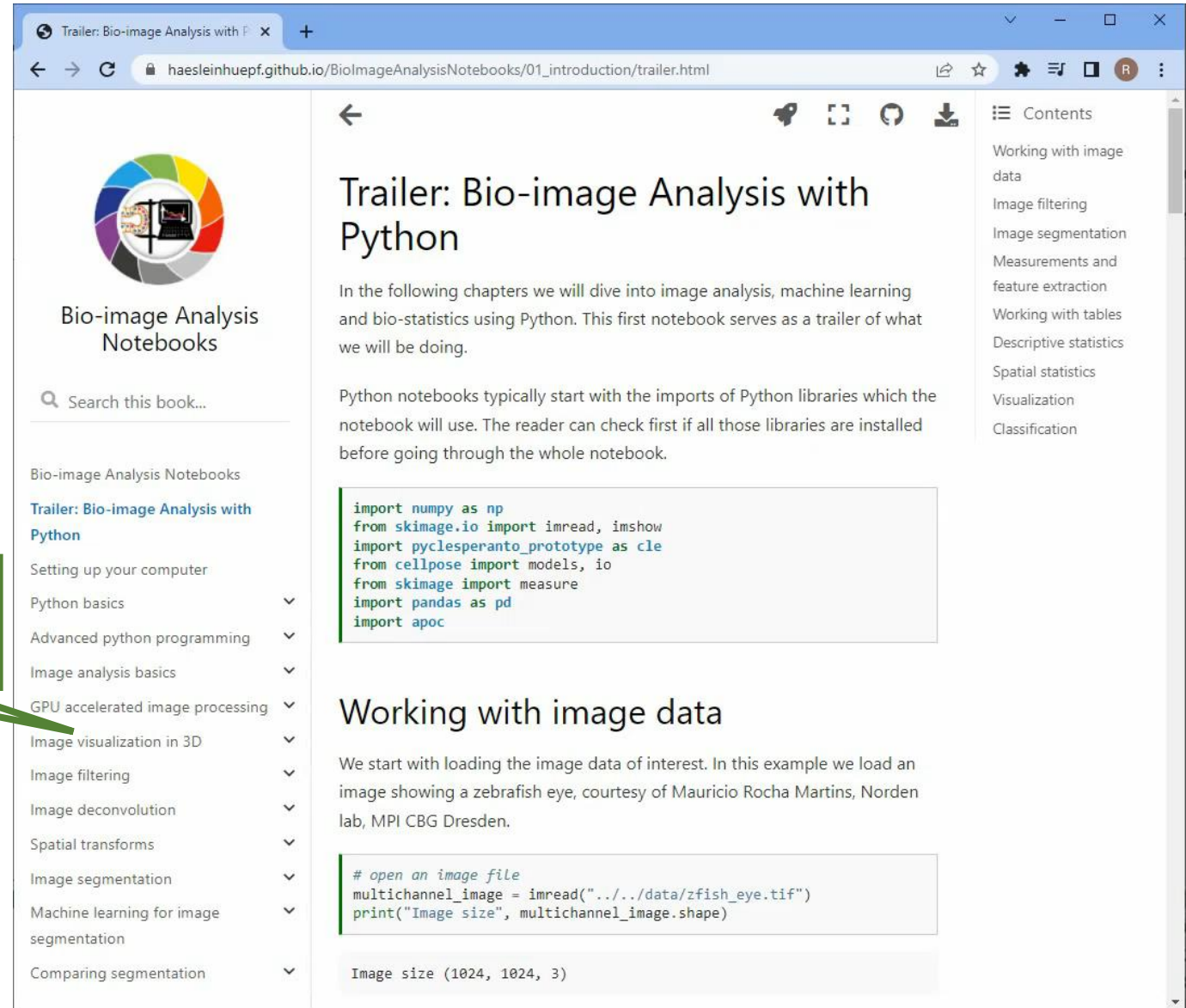
- Allowing others to do your experiment again.
- “The image data was analyzed with Python.”

Can you reproduce
what they did?

- Allowing others to do your experiment again.
- “The image data was analyzed with Python.”

Can you reproduce what they did?

Can you reproduce what they did?



The screenshot shows a web browser window with the URL haesleinhuepf.github.io/BioImageAnalysisNotebooks/01_introduction/trailer.html. The page features a sidebar with a search bar and a table of contents. The main content area displays the title 'Trailer: Bio-image Analysis with Python' and an introductory paragraph. Below the text, there are two code blocks: one showing Python imports for image analysis libraries, and another showing code to load and print the shape of an image file.

Bio-image Analysis Notebooks

Search this book...

- Bio-image Analysis Notebooks
 - Trailer: Bio-image Analysis with Python
 - Setting up your computer
 - Python basics
 - Advanced python programming
 - Image analysis basics
 - GPU accelerated image processing
 - Image visualization in 3D
 - Image filtering
 - Image deconvolution
 - Spatial transforms
 - Image segmentation
 - Machine learning for image segmentation
 - Comparing segmentation

Trailer: Bio-image Analysis with Python

In the following chapters we will dive into image analysis, machine learning and bio-statistics using Python. This first notebook serves as a trailer of what we will be doing.

Python notebooks typically start with the imports of Python libraries which the notebook will use. The reader can check first if all those libraries are installed before going through the whole notebook.

```
import numpy as np
from skimage.io import imread, imshow
import pyclesperanto_prototype as cle
from cellpose import models, io
from skimage import measure
import pandas as pd
import apoc
```

Working with image data

We start with loading the image data of interest. In this example we load an image showing a zebrafish eye, courtesy of Mauricio Rocha Martins, Norden lab, MPI CBG Dresden.

```
# open an image file
multichannel_image = imread("../data/zfish_eye.tif")
print("Image size", multichannel_image.shape)
```

Image size (1024, 1024, 3)

- Others run the same analysis on their data and have consistent results / same conclusions.
- Can only be achieved if data analysis protocol was documented reproducibly.
- See also: *Replication crisis*
 - In Psychology (surveys)
 - In Medicine (clinical trials)
 - In Computer Science (executable code)
 - ...

Open access, freely available online

Essay

Why Most Published Research Findings Are False

John P. A. Ioannidis

Summary

There is increasing concern that most current published research findings are false. The probability that a research claim is true may depend on study power and bias, the number of other studies on the same question, and, importantly, the ratio of true to no relationships among the relationships probed in each scientific field. In this framework, a research finding is less likely to be true when the studies conducted in a field are smaller; when effect sizes are smaller; when there is a greater number and lesser preselection of tested relationships; where there is greater flexibility in designs, definitions, outcomes, and analytical modes; when there is greater financial and other interest and prejudice; and when more teams are involved in a scientific field in chase of statistical significance. Simulations show that for most study designs and settings, it is more likely for

factors that influence this problem and some corollaries thereof.

Modeling the Framework for False Positive Findings

Several methodologists have pointed out [9–11] that the high rate of nonreplication (lack of confirmation) of research discoveries is a consequence of the convenient, yet ill-founded strategy of claiming conclusive research findings solely on the basis of a single study assessed by formal statistical significance, typically for a p -value less than 0.05. Research is not most appropriately represented and summarized by p -values, but, unfortunately, there is a widespread notion that medical research articles

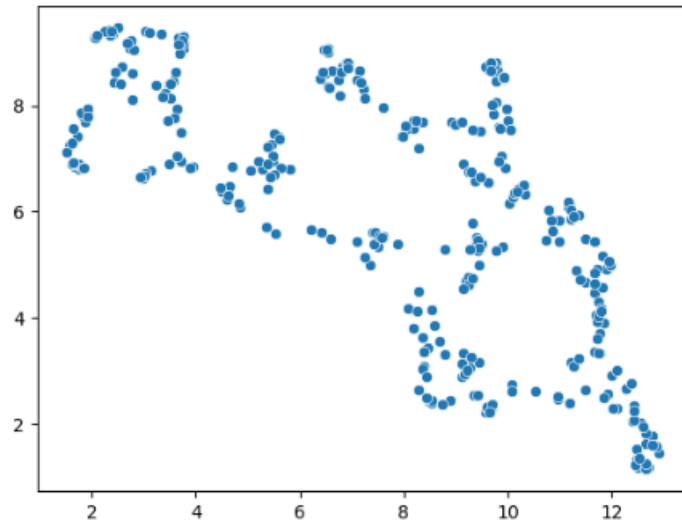
is characteristic of the field and can vary a lot depending on whether the field targets highly likely relationships or searches for only one or a few true relationships among thousands and millions of hypotheses that may be postulated. Let us also consider, for computational simplicity, circumscribed fields where either there is only one true relationship (among many that can be hypothesized) or the power is similar to find any of the several existing true relationships. The pre-study probability of a relationship being true is $R/(R + 1)$. The probability of a study finding a true relationship reflects the power $1 - \beta$ (one minus the Type II error rate). The probability of claiming a relationship when none truly exists reflects the Type I error rate, α . Assuming that c relationships are being probed in the field, the expected values of the 2×2 table are given in Table 1. After a research finding has been claimed based on

It can be proven that most claimed research findings are false.

- In wet-lab experiments, samples may get destroyed while executing the experiment.
- Repeatability is a property of the experiment / algorithm. You cannot improve repeatability by better documentation.

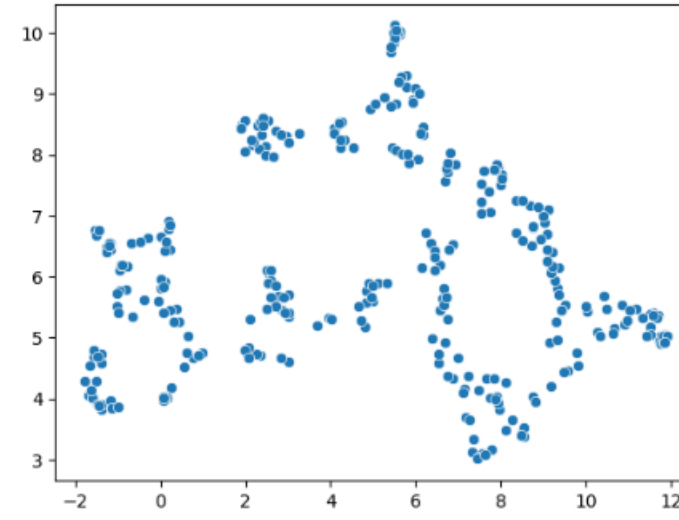
```
[11]: reducer = umap.UMAP()  
embedding2 = reducer.fit_transform(scaled_statistics)  
  
seaborn.scatterplot(x=embedding2[:, 0],  
                    y=embedding2[:, 1])
```

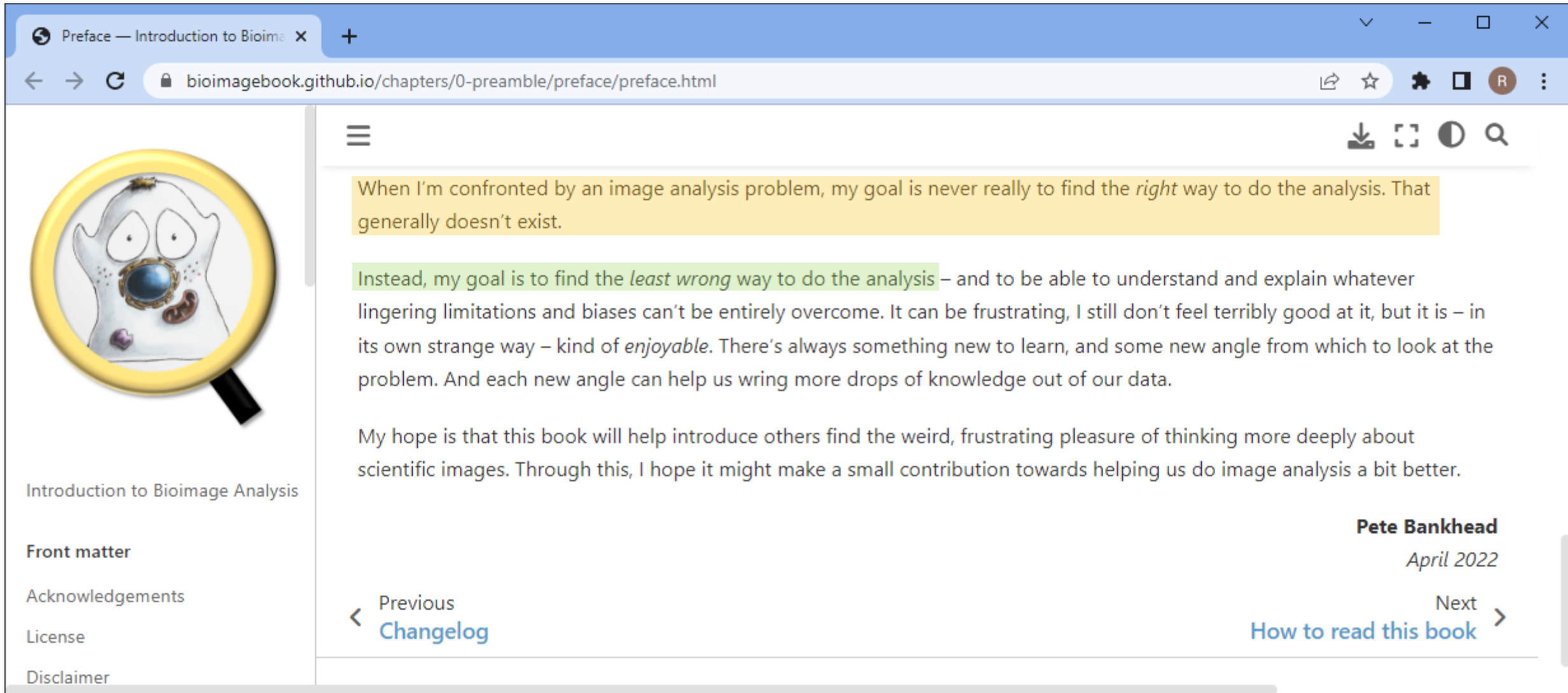
[11]: <AxesSubplot: >



```
[12]: reducer = umap.UMAP()  
embedding2 = reducer.fit_transform(scaled_statistics)  
  
seaborn.scatterplot(x=embedding2[:, 0],  
                    y=embedding2[:, 1])
```


[12]: <AxesSubplot: >





Preface — Introduction to Bioimage x +

bioimagebook.github.io/chapters/0-preamble/preface/preface.html



Introduction to Bioimage Analysis

- Front matter
- Acknowledgements
- License
- Disclaimer

When I'm confronted by an image analysis problem, my goal is never really to find the *right* way to do the analysis. That generally doesn't exist.

Instead, my goal is to find the *least wrong* way to do the analysis – and to be able to understand and explain whatever lingering limitations and biases can't be entirely overcome. It can be frustrating, I still don't feel terribly good at it, but it is – in its own strange way – kind of *enjoyable*. There's always something new to learn, and some new angle from which to look at the problem. And each new angle can help us wring more drops of knowledge out of our data.

My hope is that this book will help introduce others find the weird, frustrating pleasure of thinking more deeply about scientific images. Through this, I hope it might make a small contribution towards helping us do image analysis a bit better.

Pete Bankhead
April 2022

< Previous
[Changelog](#)

Next >
[How to read this book](#)

- Bio-image analysis is supposed to be
 - **Quantitative**
 - We derive numbers from images which describe physical properties of the observed sample.
 - **Objective**
 - The derived measurement does not depend on who did the measurement. The measurement is free of interpretation.
 - **Reliable (trustworthy / validated)**
 - We are confident that the measurement is describing what it is supposed to describe.
 - **Reproducible**
 - Enabling others to re-do the experiment. For this, documentation is crucial!
 - **Replicability**
 - Others *do* execute the same analysis, potentially on other data, and see consistent results.
 - **Repeatable**
 - We can do the same experiment twice under the *same conditions* and get the same measurements.

Demonstration of the exponential decay law using beer froth

A Leike

Ludwig-Maximilians-Universität München

E-mail: leike@th.phy.uni-muenchen.de

Received 22 July 2001

Published 17 August 2002

Online at stack.iop.org

Abstract

The volume of beer froth to demonstrate the exponential decay law depends on the type of beer. The author commonly used data, parameters and

In the following, the demonstration is described in detail. In our experiment, a cylindrical beer mug with a diameter of 7.2 cm was filled with beer immediately after opening the bottle. The temperature of the beer was 19 °C.

The froth appears while filling the mug with the beer. The froth reaches its maximum height within a few seconds. This indicates that the typical time scale of the expansion of the froth is a few seconds. On the other hand, the froth lasts for a few minutes (see table 1). Therefore, the time scale for the decay is a few minutes. The two time scales are very different. We therefore assume that a few seconds after the time where the froth reaches its maximum height only the decay plays a significant role.

We began with the measurement at the time where the froth dropped to a certain initial height $h^{\text{exp}}(0) = h(0)$. The error $\Delta h^{\text{exp}}(0)$ in the measurement of $h(0)$ was estimated to be 2 mm.

We investigated three different beers. With every beer, the experiment was repeated several times. We performed seven experiments with Erdinger Weissbier (the author's favourite!), four experiments with Augustinerbräu München and four experiments with Budweiser Budvar. Our data are shown in table 1. The entries for $h^{\text{exp}}(t_i)$ are obtained by averaging over all individual measurements at time t_i . To obtain the errors $\Delta h^{\text{exp}}(t_i)$ of the measurements $h^{\text{exp}}(t_i)$, we first

Is this experiment reproducible?

Yes

No

Demonstration of the exponential decay law using beer froth

A Leike

Ludwig-Maximilians-Universität München

E-mail: leike@th.phy.uni-muenchen.de

Received 22 July 2001

Published 17 August 2002

Online at stack.iop.org

Abstract

The volume of beer froth to demonstrate the exponential decay law depends on the type of beer. The author commonly used data, parameters and

In the following, the demonstration is described in detail. In our experiment, a cylindrical beer mug with a diameter of 7.2 cm was filled with beer immediately after opening the bottle. The temperature of the beer was 19 °C.

The froth appears while filling the mug with the beer. The froth reaches its maximum height within a few seconds. This indicates that the typical time scale of the expansion of the froth is a few seconds. On the other hand, the froth lasts for a few minutes (see table 1). Therefore, the time scale for the decay is a few minutes. The two time scales are very different. We therefore assume that a few seconds after the time where the froth reaches its maximum height only the decay plays a significant role.

We began with the measurement at the time where the froth dropped to a certain initial height $h^{\text{exp}}(0) = h(0)$. The error $\Delta h^{\text{exp}}(0)$ in the measurement of $h(0)$ was estimated to be 2 mm.

We investigated three different beers. With every beer, the experiment was repeated several times. We performed seven experiments with Erdinger Weissbier (the author's favourite!), four experiments with Augustinerbräu München and four experiments with Budweiser Budvar. Our data are shown in table 1. The entries for $h^{\text{exp}}(t_i)$ are obtained by averaging over all individual measurements at time t_i . To obtain the errors $\Delta h^{\text{exp}}(t_i)$ of the measurements $h^{\text{exp}}(t_i)$, we first

Is this experiment repeatable?

Yes

No

Demonstration of the exponential decay law using beer froth

A Leike

Ludwig-Maximilians-Universität München

E-mail: leike@th.phy.uni-muenchen.de

Received 22 July 2001

Published 17 August 2002

Online at stack.iop.org

Abstract

The volume of beer froth to demonstrate the exponential decay law depends on the type of beer. The analysis of commonly used data, parameters and the

In the following, the demonstration is described in detail. In our experiment, a cylindrical beer mug with a diameter of 7.2 cm was filled with beer immediately after opening the bottle. The temperature of the beer was 19 °C.

The froth appears while filling the mug with the beer. The froth reaches its maximum height within a few seconds. This indicates that the typical time scale of the expansion of the froth is a few seconds. On the other hand, the froth lasts for a few minutes (see table 1). Therefore, the time scale for the decay is a few minutes. The two time scales are very different. We therefore assume that a few seconds after the time where the froth reaches its maximum height only the decay plays a significant role.

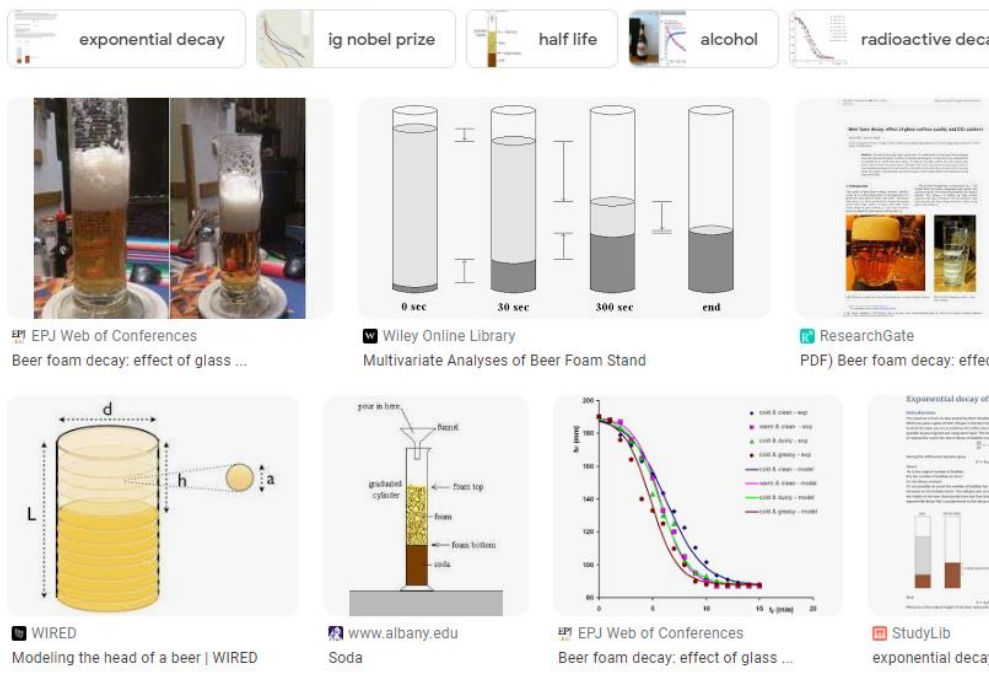
We began with the measurement at the time where the froth dropped to a certain initial height $h^{\text{exp}}(0) = h(0)$. The error $\Delta h^{\text{exp}}(0)$ in the measurement of $h(0)$ was estimated to be 2 mm.

We investigated three different beers. With every beer, the experiment was repeated several times. We performed seven experiments with Erdinger Weissbier (the author's favourite!), four experiments with Augustinerbräu München and four experiments with Budweiser Budvar. Our data are shown in table 1. The entries for $h^{\text{exp}}(t_i)$ are obtained by averaging over all individual measurements at time t_i . To obtain the errors $\Delta h^{\text{exp}}(t_i)$ of the measurements $h^{\text{exp}}(t_i)$, we first

Do you think this experiment was replicated?

Yes

No



A grid of search results for "beer foam decay experiment". The results include:

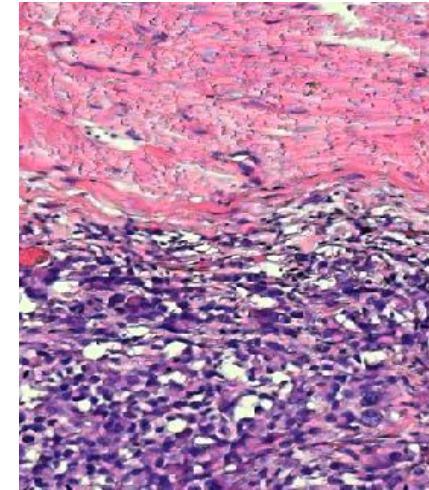
- exponential decay**: A diagram showing a glass of beer with foam at different stages of decay (0 sec, 30 sec, 300 sec, end).
- ig nobel prize**: A photograph of a glass of beer with foam.
- half life**: A diagram showing a glass of beer with foam at different stages of decay.
- alcohol**: A diagram showing a glass of beer with foam.
- radioactive decay**: A diagram showing a glass of beer with foam.
- EPJ Web of Conferences**: Beer foam decay: effect of glass ...
- Wiley Online Library**: Multivariate Analyses of Beer Foam Stand
- ResearchGate**: PDF) Beer foam decay: effect of glass ...
- WIRED**: Modeling the head of a beer | WIRED
- www.albany.edu**: Soda
- EPJ Web of Conferences**: Beer foam decay: effect of glass ...
- StudyLib**: exponential decay



A screenshot of the physicsworld website. The page features a purple header with the text "soft matter and liquids". Below the header, there is a navigation bar with "IOP Publishing" and social media icons. The main content area displays the article "Beer paper wins Ig Nobel physics prize" by Arnd Leike, dated 07 Oct 2002. The article text states: "A German physicist who used beer to teach students about the analysis of experimental data has received the 2002 Ig Nobel prize in physics. Arnd Leike of the Ludwig Maximilians University receives one of the awards - which are given for research that cannot or should not be repeated - for demonstrating that beer froth obeys the mathematical law of exponential decay. Leike's work was published in the *European Journal of Physics* and later highlighted in *Physics World*."

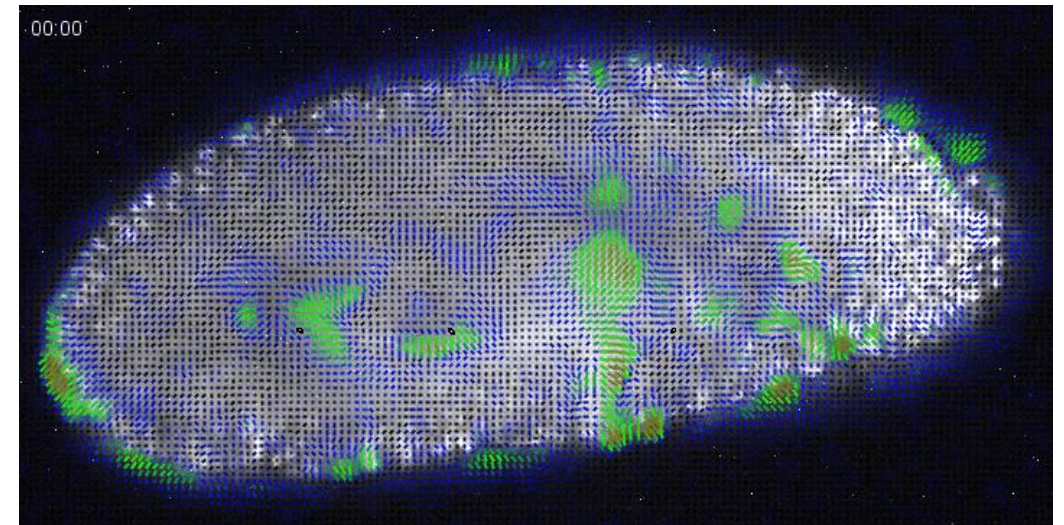
- Typical questions bio-image analysts deal with
 - Is signal intensity different under varying conditions?
 - How many cells are in my image?
 - How high is cell density?
 - Bio-statistics / medicine / disease staging
 - How are different tissues characterized?
 - Machine learning

- Typical questions bio-image analysts struggle with
 - What force drives the observed processes?
 - What is the lineage tree of one particular cell?
 - Are observation A and observation B related?
 - Are structures observed in different color channels colocalized?



muscle, normal tissue

squamous-cell carcinoma



Hypothesis-driven quantitative biology

- Hypothesis: Cell shape can be influenced by modifying X.
- Null-Hypothesis: Circularity of modified cells is similar to cells in the control group.

Should we use a different segmentation algorithm?

- Sample preparation
 - Imaging
- Cell segmentation
- Circularity measurement
- Statistics

Shall we use a different microscope?

Is circularity the right parameter to measure?

• ~~Hypothesis: Cell shape can be influenced by modifying X.~~

• Question: Which image-derived parameter is influenced when modifying X?

- Sample preparation

- Imaging

Which segmentation algorithms allow measurements that show a relationship with X?

- Cell segmentation algorithm A, algorithm B, algorithm C

Why?

- Measurement of circularity, solidity, elongation, extend, texture, intensity, topology ...

- Statistics

Which parameter shows any relationship with X?



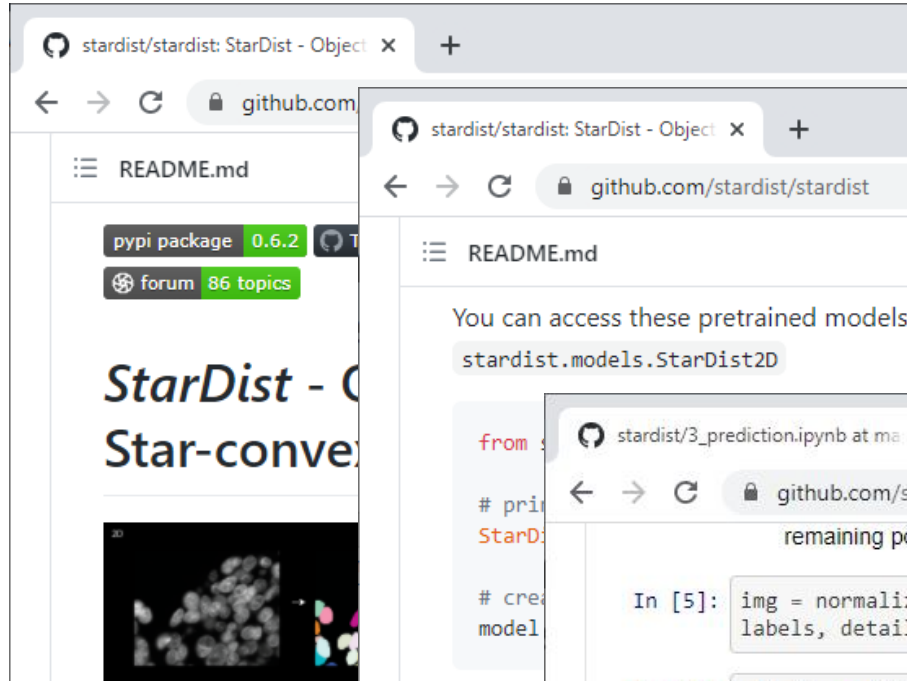
Python Programming

Robert Haase

April 2023

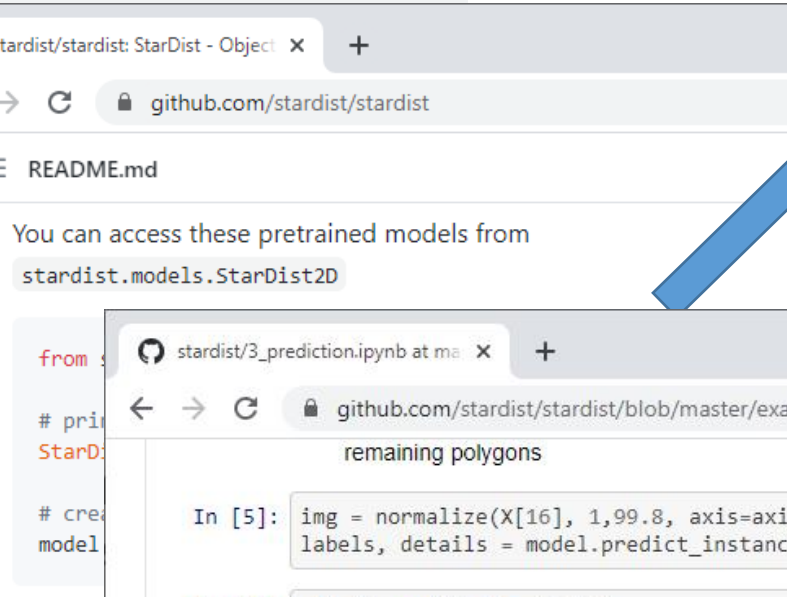
- Why Python?

Because copy&paste works so great.



This repository contains object detection for 2D

- Uwe Schmidt, Malin Broaddus, and George M. B. Riegler. *Cell Detection with Deep Learning*. International Conference on Medical Image Computing and Intervention (MICCAI), September 2018.



Annot

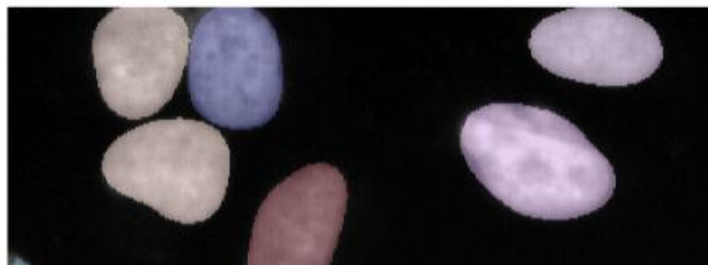
To train a model, you need to provide annotated images (corresponding labels with 0). Among the options, you can choose having a

```
from stardist.models import StarDist2D
# pretrained model
model = StarDist2D.from_pretrained('2D_versatile_fluo')

# create image
img = ...

# predict
labels, details = model.predict_instances(img)

# show
plt.figure(figsize=(8,8))
plt.imshow(img if img.ndim==2 else img[...,0], clim=(0,1), cmap=...
plt.imshow(labels, cmap=lbl_cmap, alpha=0.5)
plt.axis('off');
```



```
In [3]: # normalize image
from csbdeep.utils import normalize
normalized_image = normalize(image, 1,99.8, axis=(0,1))

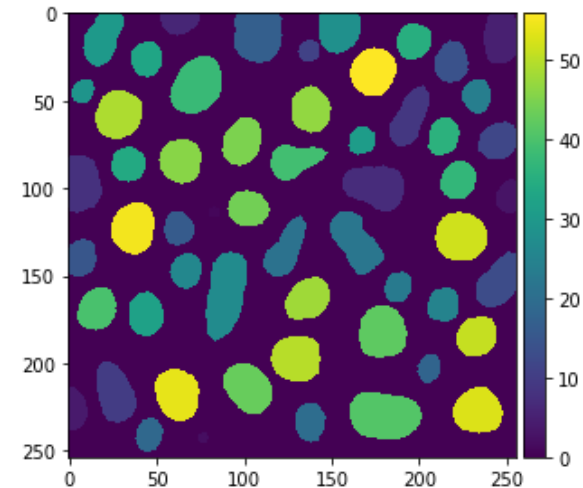
# Load pretrained deep-Learning model
from stardist.models import StarDist2D
model = StarDist2D.from_pretrained('2D_versatile_fluo')

# predict labels
label_image, details = model.predict_instances(normalized_image)
imshow(label_image)
```

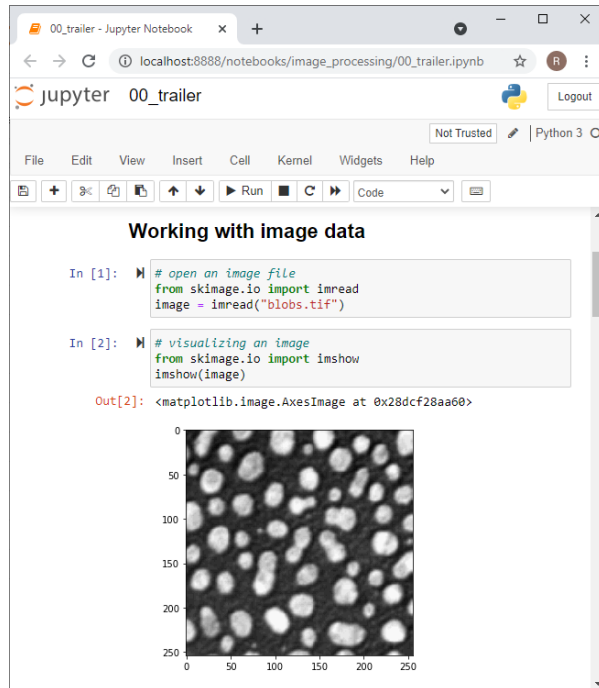
Found model '2D_versatile_fluo' for 'StarDist2D'.
Loading network weights from 'weights_best.h5'.
Loading thresholds from 'thresholds.json'.
Using default values: prob_thresh=0.479071, nms_thresh=0.3.

matplotlib_plugin.py (150): Low image data range; displaying image

<matplotlib.image.AxesImage at 0x28dd9f991c0>



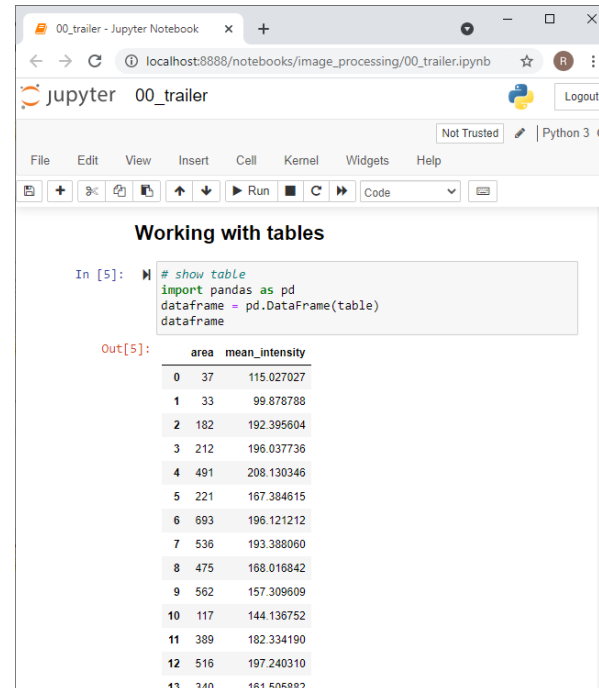
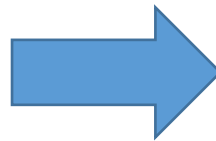
- Major goals of [image] data analysis via scripting:
 - reproducible workflows for processing images (raw data) into quantitative information and visualizing biological properties.
 - automation
 - Sharing code, knowledge
 - Prevent reinventing the wheel



```
In [1]: # open an image file
from skimage.io import imread
image = imread("blobs.tif")

In [2]: # visualizing an image
from skimage.io import imshow
imshow(image)

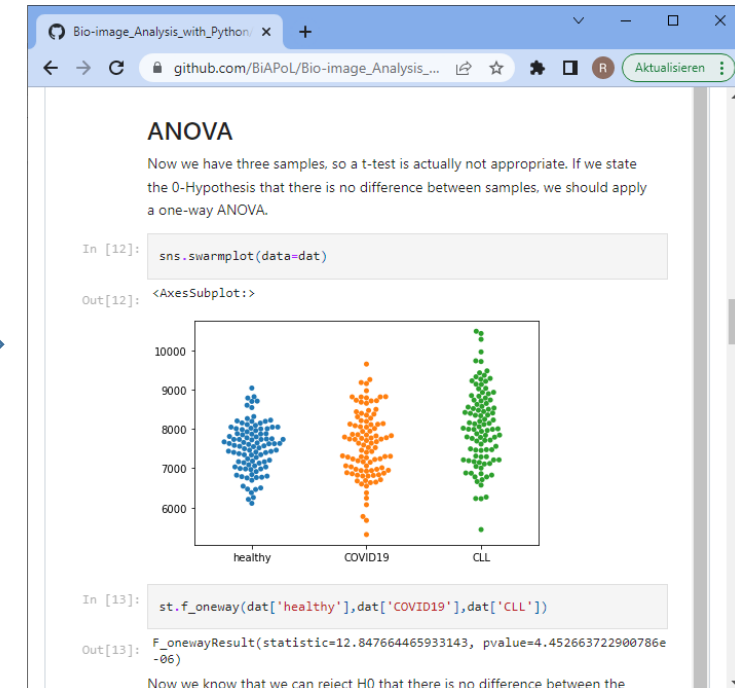
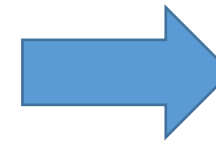
Out[2]: <matplotlib.image.AxesImage at 0x28dcf28aa60>
```



```
In [5]: # show table
import pandas as pd
dataframe = pd.DataFrame(table)
dataframe

Out[5]:
```

	area	mean_intensity
0	37	115.027027
1	33	99.878788
2	182	192.395604
3	212	196.037736
4	491	208.130346
5	221	167.384615
6	693	196.121212
7	536	193.380060
8	475	168.016842
9	562	157.309609
10	117	144.136752
11	389	182.334190
12	516	197.240310
13	340	161.505882

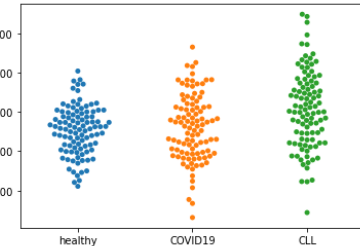


ANOVA

Now we have three samples, so a t-test is actually not appropriate. If we state the 0-Hypothesis that there is no difference between samples, we should apply a one-way ANOVA.

```
In [12]: sns.swarmplot(data=dat)

Out[12]: <AxesSubplot:>
```

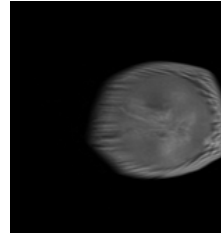


```
In [13]: st.f_oneway(dat['healthy'], dat['COVID19'], dat['CLL'])

Out[13]: F_onewayResult(statistic=12.847664465933143, pvalue=4.452663722900786e-06)
```

Now we know that we can reject H0 that there is no difference between the

banana0008.tif
banana0009.tif
banana0010.tif
banana0011.tif
banana0012.tif



- Remove shell
- Repeat until nothing left:
 - Take a bite
 - Chew
 - Swallow
- Digest

- Access folder
- Repeat for all images:
 - Open an image file
 - Segment the banana slice
 - Analyse it
- Save measurements

```
slice_areas = []
for root, dirs, files in os.walk(data_folder):
    for file in files:
        if file.endswith('.tif'):

            # load data
            from skimage.io import imread
            image = imread(root + file)

            # segment it
            from skimage.filters import threshold_otsu
            binary_image = image > threshold_otsu(image)

            from skimage.measure import label
            labels = label(binary_image)

            # measure radius
            from skimage.measure import regionprops
            statistics = regionprops(labels)
            areas = [s.area for s in statistics]

            # store result in array
            import numpy as np
            slice_areas.append(np.max(areas))
```




Python

Data structures

Robert Haase

April 2023

- Variables can hold numeric values and you can do math with them

```
▶ # initialize program  
a = 5  
b = 3  
  
# run algorithm on given parameters  
sum = a + b  
  
# print out result  
print (sum)
```

8

- Also strings as values for variables are supported

Single and double quotes
allowed

```
▶ firstname = "Robert"  
  lastname = 'Haase'  
  
print("Hello " + firstname + " " + lastname)
```

Hello Robert Haase

- Math commands supplement operators to be able to implement any form of calculations

- Power

```
▶ pow(3, 2)
```

```
] : 9
```

- Absolute

```
▶ abs(-8)
```

```
] : 8
```

- Rounding

```
▶ round(4.6)
```

```
] : 5
```

```
▶ round(4.5)
```

```
] : 4
```

Be careful with
some of them!

https://en.wikipedia.org/wiki/Rounding#Round_half_to_even

- Arrays are variables, where you can store multiple values

Give me a "0", five times!

```
array = [0] * 5
```

Computer memory

array

1	0	5	0	Rab bit
---	---	---	---	------------

```
▶ # Arrays  
numbers = [0, 1, 2, 3, 4, 5, 6, 7, 8, 9]  
print(numbers)
```

[0, 1, 2, 3, 4, 5, 6, 7, 8, 9]

- Creating subsets of arrays

Start

End

```
▶ subset = numbers[2:4]  
print(subset)
```

[2, 3]

Step

```
▶ subset_with_gaps = arr[1:8:2]  
print(subset_with_gaps)
```

[1, 3, 5, 7]

data [start : stop : step]

- “Indexing” is addressing certain elements in arrays. The first element is “0” away from the start.

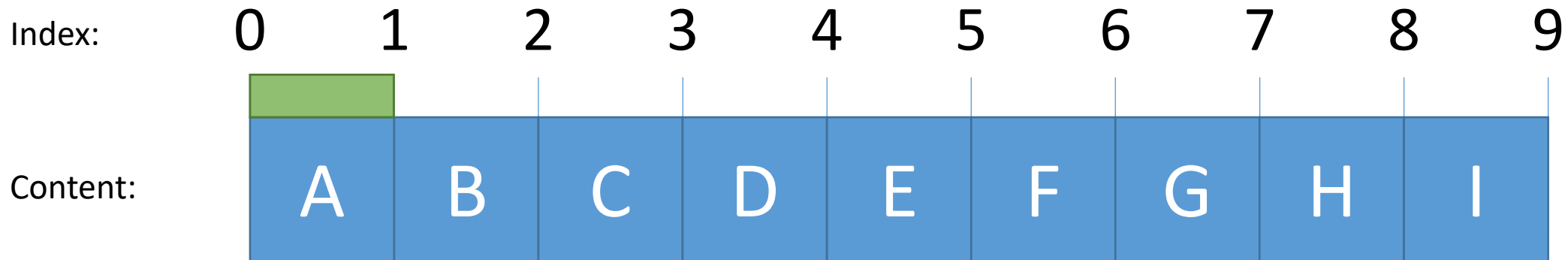
```
data = ['A', 'B', 'C', 'D', 'E', 'F', 'G', 'H', 'I']
```

Index:	0	1	2	3	4	5	6	7	8	9
Content:	A	B	C	D	E	F	G	H	I	

Indexing, cropping, subsets

- “Indexing” is addressing certain elements in arrays. The first element is “0” away from the start.

```
data = ['A', 'B', 'C', 'D', 'E', 'F', 'G', 'H', 'I']
```

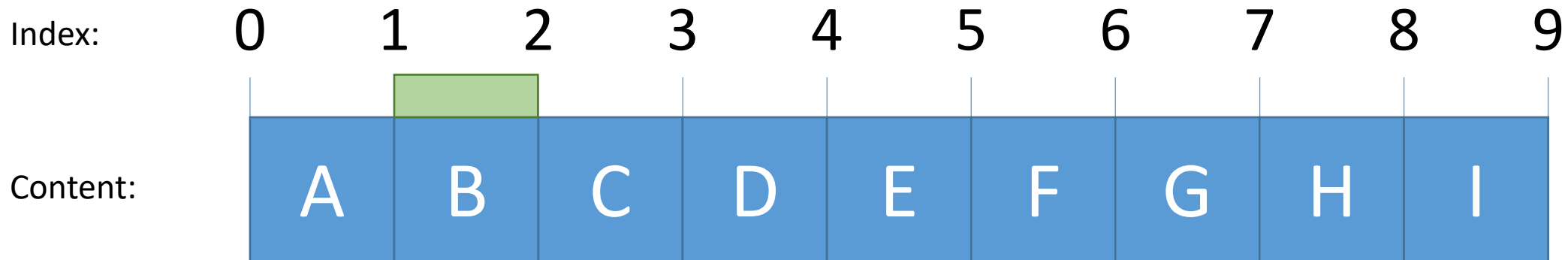


```
data[0]
```

'A'

- “Indexing” is addressing certain elements in arrays. The first element is “0” away from the start.

```
data = ['A', 'B', 'C', 'D', 'E', 'F', 'G', 'H', 'I']
```



```
data[0]
```

'A'

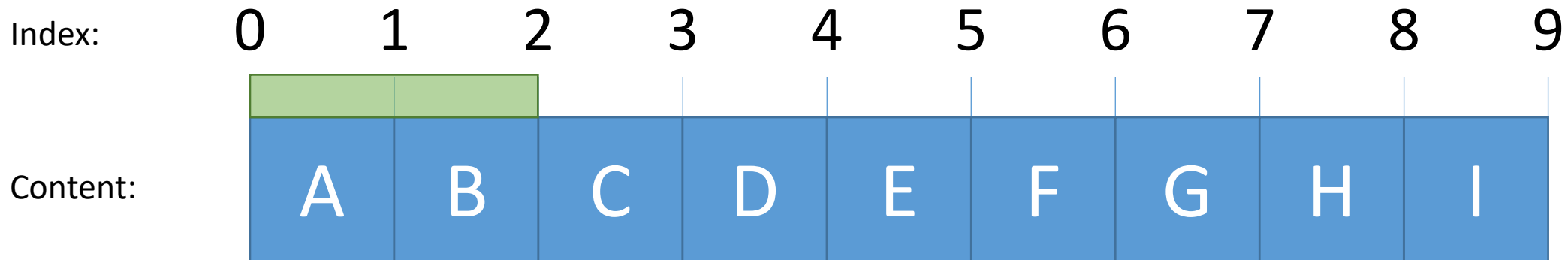
```
data[1]
```

'B'

Indexing, cropping, subsets

- “Indexing” is addressing certain elements in arrays. The first element is “0” away from the start.

```
data = ['A', 'B', 'C', 'D', 'E', 'F', 'G', 'H', 'I']
```



```
data[0]
```

'A'

```
data[1]
```

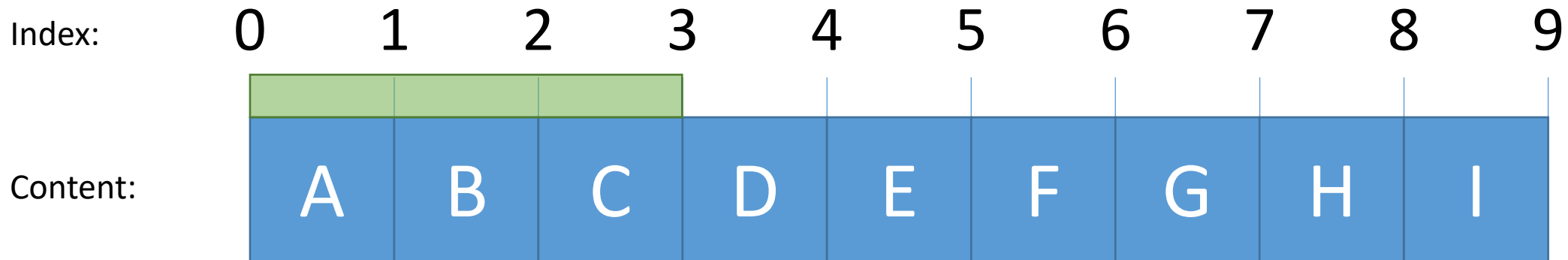
'B'

```
data[0:2]
```

['A', 'B']

- “Indexing” is addressing certain elements in arrays. The first element is “0” away from the start.

```
data = ['A', 'B', 'C', 'D', 'E', 'F', 'G', 'H', 'I']
```



```
data[0]
```

```
'A'
```

```
data[1]
```

```
'B'
```

```
data[0:2]
```

```
['A', 'B']
```

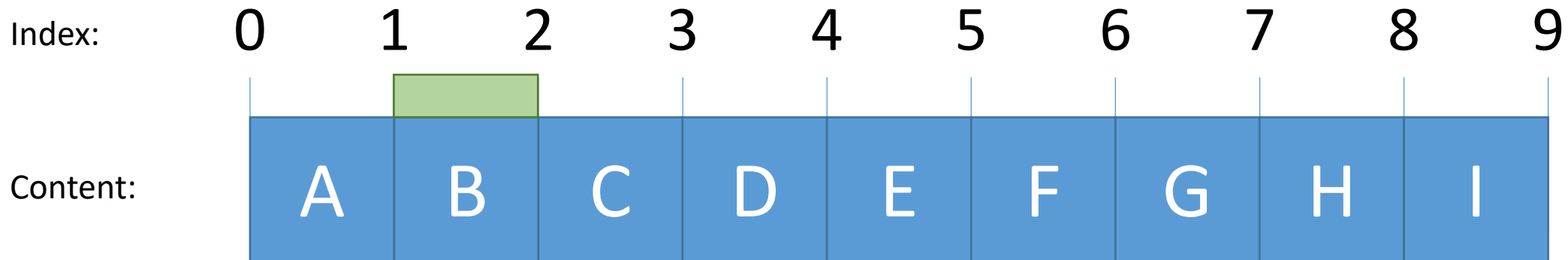
```
data[0:3]
```

```
['A', 'B', 'C']
```

Indexing, cropping, subsets

- “Indexing” is addressing certain elements in arrays. The first element is “0” away from the start.

```
data = ['A', 'B', 'C', 'D', 'E', 'F', 'G', 'H', 'I']
```



```
data[0]
```

'A'

```
data[1]
```

'B'

```
data[0:2]
```

['A', 'B']

```
data[0:3]
```

['A', 'B', 'C']

```
data[1:2]
```

['B']

Indexing, cropping, subsets

- “Indexing” is addressing certain elements in arrays. The first element is “0” away from the start.

```
data = ['A', 'B', 'C', 'D', 'E', 'F', 'G', 'H', 'I']
```

Index:	0	1	2	3	4	5	6	7	8	9
Content:	A	B	C	D	E	F	G	H	I	

```
data[0]
```

'A'

```
data[1]
```

'B'

```
data[0:2]
```

['A', 'B']

```
data[0:3]
```

['A', 'B', 'C']

```
data[1:2]
```

['B']

```
len(data)
```

9

Indexing, cropping, subsets

- You can leave start and end out when specifying index ranges

```
data = ['A', 'B', 'C', 'D', 'E', 'F', 'G', 'H', 'I']
```

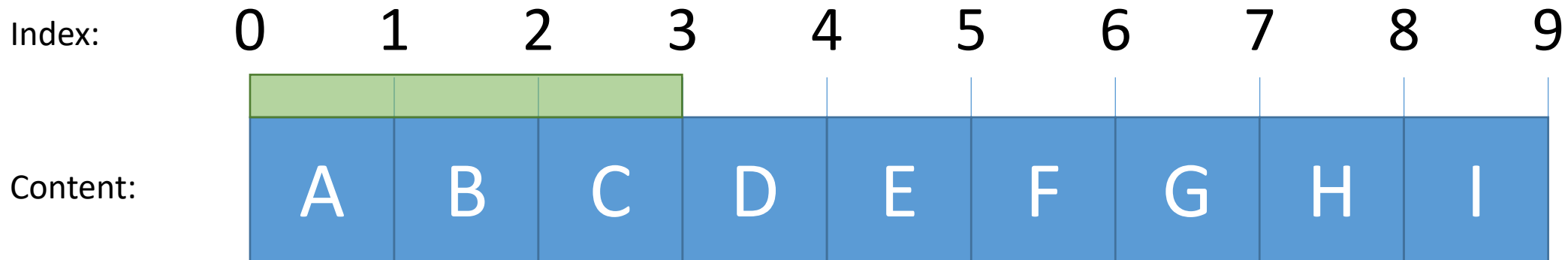
Index:	0	1	2	3	4	5	6	7	8	9
Content:	A	B	C	D	E	F	G	H	I	

```
data[:2]
```

```
['A', 'B']
```

- You can leave start and end out when specifying index ranges

```
data = ['A', 'B', 'C', 'D', 'E', 'F', 'G', 'H', 'I']
```



```
data[:2]
```

```
['A', 'B']
```

```
data[:3]
```

```
['A', 'B', 'C']
```

- You can leave start and end out when specifying index ranges

```
data = ['A', 'B', 'C', 'D', 'E', 'F', 'G', 'H', 'I']
```

Index:	0	1	2	3	4	5	6	7	8	9
Content:	A	B	C	D	E	F	G	H	I	

```
data[:2]
```

```
['A', 'B']
```

```
data[:3]
```

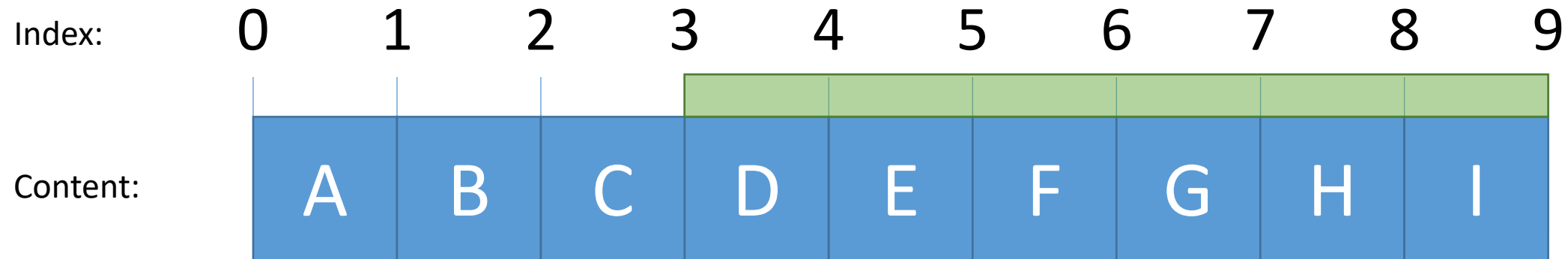
```
['A', 'B', 'C']
```

```
data[2:]
```

```
['C', 'D', 'E', 'F', 'G', 'H', 'I']
```



```
data = ['A', 'B', 'C', 'D', 'E', 'F', 'G', 'H', 'I']
```



What's the output of

```
data[3:]
```

?

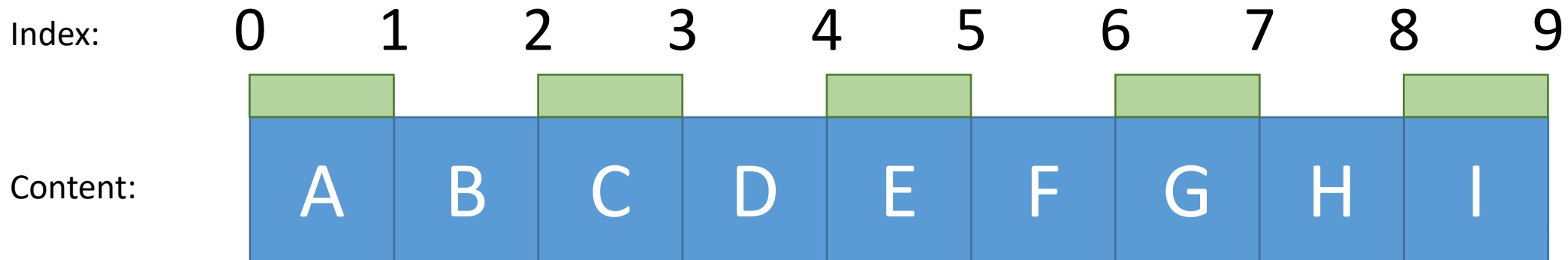
ABC

CDEFGHI

DEFGHI

- The step-size allows skipping elements

```
data = ['A', 'B', 'C', 'D', 'E', 'F', 'G', 'H', 'I']
```

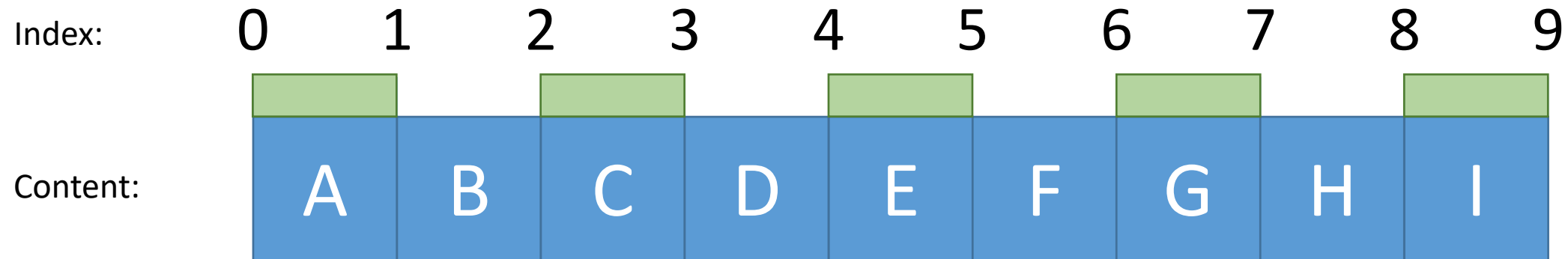


```
data[0:10:2]
```

```
['A', 'C', 'E', 'G', 'I']
```

- The step-size allows skipping elements

```
data = ['A', 'B', 'C', 'D', 'E', 'F', 'G', 'H', 'I']
```



```
data[0:10:2]
```

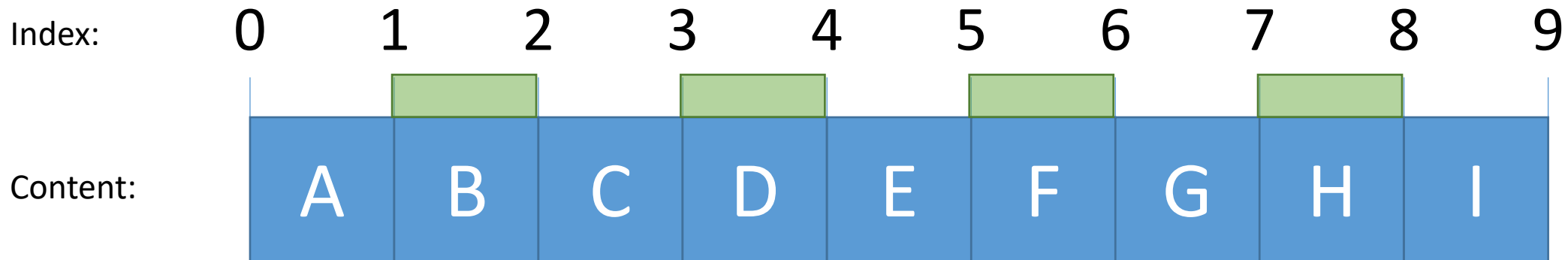
```
data[::2]
```

```
['A', 'C', 'E', 'G', 'I']
```

```
['A', 'C', 'E', 'G', 'I']
```

- The step-size allows skipping elements

```
data = ['A', 'B', 'C', 'D', 'E', 'F', 'G', 'H', 'I']
```



```
data[0:10:2]
```

```
data[::2]
```

```
data[1::2]
```

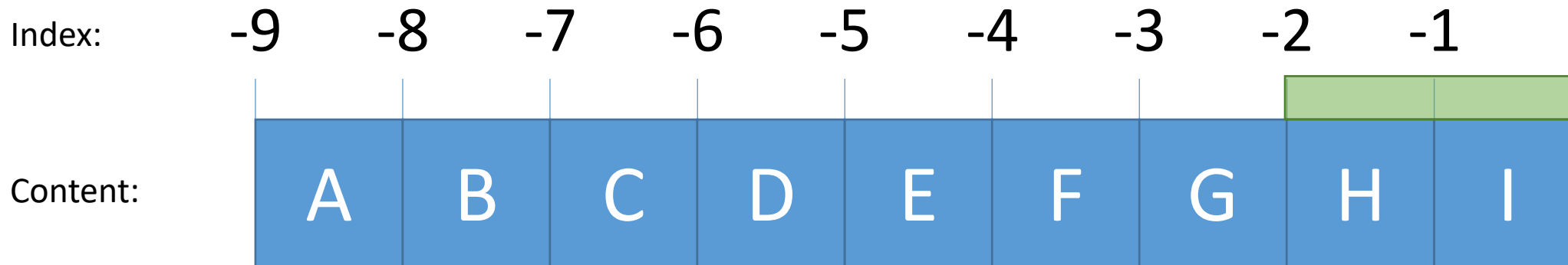
```
['A', 'C', 'E', 'G', 'I']
```

```
['A', 'C', 'E', 'G', 'I']
```

```
['B', 'D', 'F', 'H']
```

- Indexing also works with negative indices

```
data = ['A', 'B', 'C', 'D', 'E', 'F', 'G', 'H', 'I']
```



```
data[-2:]
```

```
['H', 'I']
```

- Indexing also works with negative indices

```
data = ['A', 'B', 'C', 'D', 'E', 'F', 'G', 'H', 'I']
```

Index:	-9	-8	-7	-6	-5	-4	-3	-2	-1
Content:	A	B	C	D	E	F	G	H	I

```
data[-2:]
```

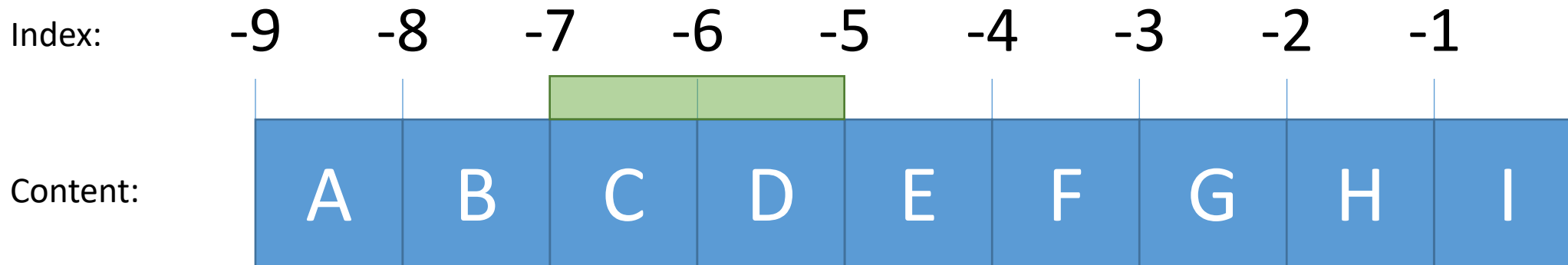
```
['H', 'I']
```

```
data[:-2]
```

```
['A', 'B', 'C', 'D', 'E', 'F', 'G']
```

- Indexing also works with negative indices

```
data = ['A', 'B', 'C', 'D', 'E', 'F', 'G', 'H', 'I']
```



```
data[-2:]
```

```
['H', 'I']
```

```
data[:-2]
```

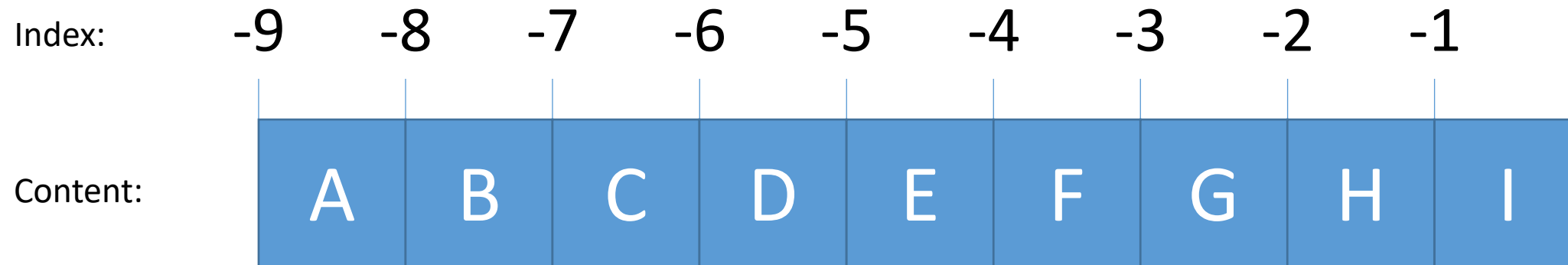
```
['A', 'B', 'C', 'D', 'E', 'F', 'G']
```

```
data[-7:-5]
```

```
['C', 'D']
```

- Indexing also works with negative indices

```
data = ['A', 'B', 'C', 'D', 'E', 'F', 'G', 'H', 'I']
```



```
data[-2:]
```

```
['H', 'I']
```

```
data[:-2]
```

```
['A', 'B', 'C', 'D', 'E', 'F', 'G']
```

```
data[-7:-5]
```

```
['C', 'D']
```

```
data[-5:-7]
```

```
[]
```


- Negative stepping also works

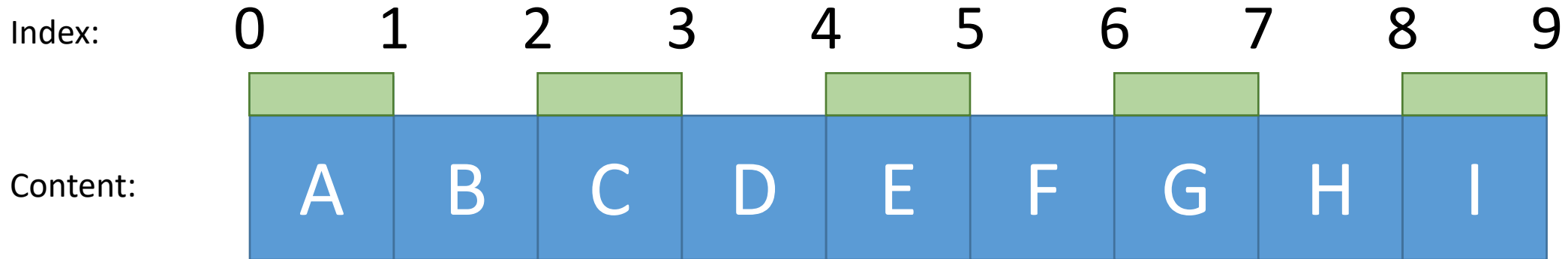
```
data = ['A', 'B', 'C', 'D', 'E', 'F', 'G', 'H', 'I']
```

Index:	0	1	2	3	4	5	6	7	8	9
Content:	A	B	C	D	E	F	G	H	I	

```
data[::-1]
```

```
['I', 'H', 'G', 'F', 'E', 'D', 'C', 'B', 'A']
```

```
data = ['A', 'B', 'C', 'D', 'E', 'F', 'G', 'H', 'I']
```



What's the output of

```
data[::-2]
```

?

IGECA

HGFEDCBA

GFEDCBA

- Modifying array elements

```
▶ numbers = [0, 1, 2, 3, 4]
# write in one array element
numbers[1] = 5
print(numbers)
[0, 5, 2, 3, 4]
```

Note: The first element has index 0!

- Creating arrays of defined size

What?

How many?

```
▶ zeros = [0] * 10
print(zeros)
[0, 0, 0, 0, 0, 0, 0, 0, 0, 0]
```

- Concatenating arrays

```
▶ ones = [1, 1, 1]
twos = [2, 2, 2, 2]
# concatenate arrays
numbers = ones + twos
print(numbers)
[1, 1, 1, 2, 2, 2, 2]
```

+ means appending

- Lists can be modified

```
▶ measurements = [5.5, 6.3, 7.2, 8.0, 8.8]
```

```
▶ measurements[1] = 25
```

```
▶ measurements.append(10.2)
```

```
▶ measurements
```

```
] : [5.5, 25, 7.2, 8.0, 8.8, 10.2]
```

- Tuples not

```
▶ immutable = (4, 3, 7.8)
```

Note: round brackets

```
▶ immutable[1] = 5
```

TypeError

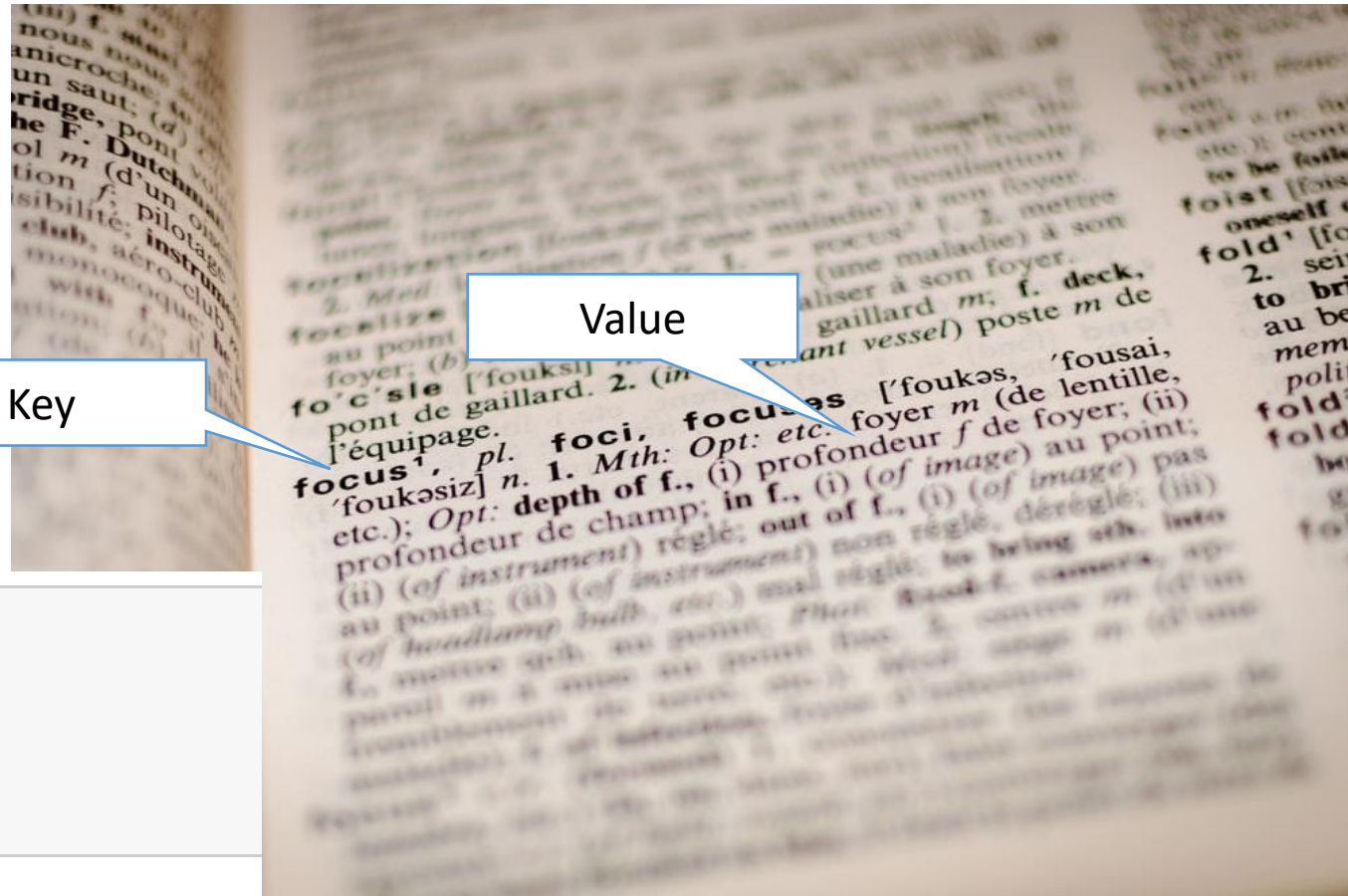
Traceback (most recent call last)

<ipython-input-49-a01b13633c23> in <module>

----> 1 immutable[1] = 5

TypeError: 'tuple' object does not support item assignment

- Dictionary: a list of key-value pairs



Key

Value

Key

Value

```
▶ german_english_dictionary = {  
    'Vorlesung': 'Lecture',  
    'Gleichung': 'Equation'  
}
```

```
▶ german_english_dictionary
```

```
]: {'Vorlesung': 'Lecture', 'Gleichung': 'Equation'}
```

- Dictionary: a list of key-value pairs

```
▶ german_english_dictionary = {  
    'Vorlesung': 'Lecture',  
    'Gleichung': 'Equation'  
}
```

- Look up something in the dictionary: it's an array with named entries!

```
▶ german_english_dictionary['Vorlesung']  
]: 'Lecture'
```

- Tables can be dictionaries with arrays as values

```
▶ measurements_week = {  
    'Monday': [2.3, 3.1, 5.6],  
    'Tuesday': [1.8, 7.0, 4.3],  
    'Wednesday': [4.5, 1.5, 3.2],  
    'Thursday': [1.9, 2.0, 6.4],  
    'Friday': [4.4, 2.3, 5.4]  
}
```

```
▶ measurements_week
```

```
]: {'Monday': [2.3, 3.1, 5.6],  
    'Tuesday': [1.8, 7.0, 4.3],  
    'Wednesday': [4.5, 1.5, 3.2],  
    'Thursday': [1.9, 2.0, 6.4],  
    'Friday': [4.4, 2.3, 5.4]}
```

- Retrieve a column

```
▶ measurements_week['monday']
```

```
]: [2.3, 3.1, 5.6]
```

If your program throws error messages:

- Don't panic.
- *"There are two ways to write error-free programs; only the third one works."*

Alan J. Perlis, Yale University

- Read where the error happened.
 - You may see your fault immediately, when looking at the right point.
- Read what appears to be wrong.
 - If you know, what's missing, you may see it, even if it's missing in a slightly different place.
 - Sometimes, something related is missing

```
▶ print(round(4.5))
```

```
File "<ipython-input-15-09a9be4a90c5>", line 1
```

```
print(round(4.5))
```

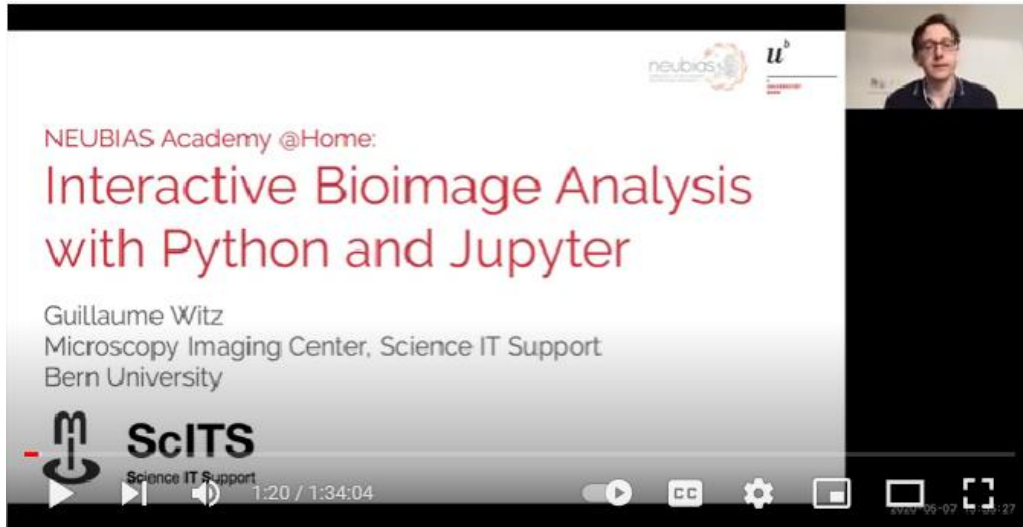
```
^
```

```
SyntaxError: unexpected EOF while parsing
```


Comments should contain additional information such as

- User documentation
 - What does the program do?
 - How can this program be used?
- Your name / institute in case a reader has a question
- Comment why things are done.
- Do not comment what is written in the code already!

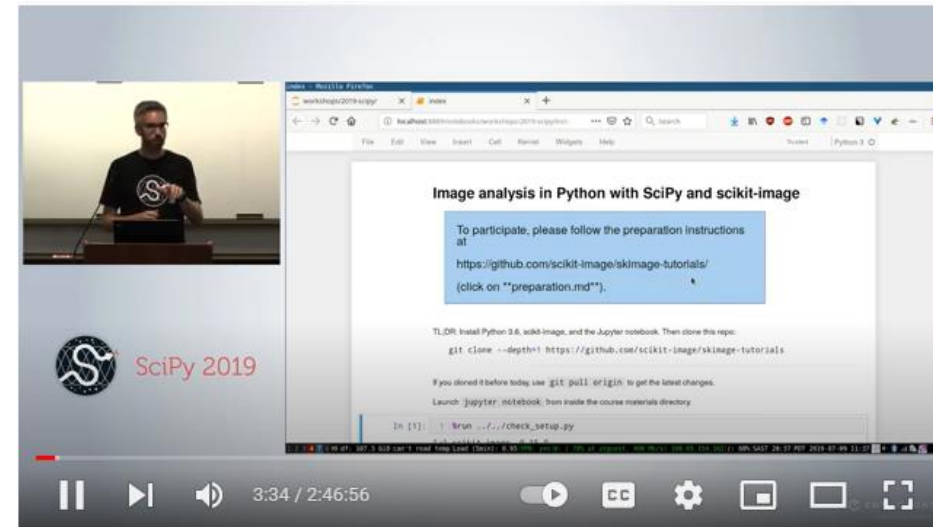
```
#  
# This program sums up two numbers.  
#  
# Usage:  
# * Run it in Python 3.8  
#  
# Author: Robert Haase, PoL TUD  
#         Robert.haase@tu-dresden.de  
# April 2021  
  
# initialise program  
a = 1  
b = 2.5  
  
# run complicated algorithm  
final_result = a + b  
  
# print the final result  
print( final_result )
```



Guillaume Witz, NEUBIAS Academy 2020

Watch more:

- <https://www.youtube.com/watch?v=2KF8vBrp3Zw>
- <https://www.youtube.com/watch?v=d1CIV9irQAY>
- https://www.youtube.com/watch?v=X_pCiVQ4c4E



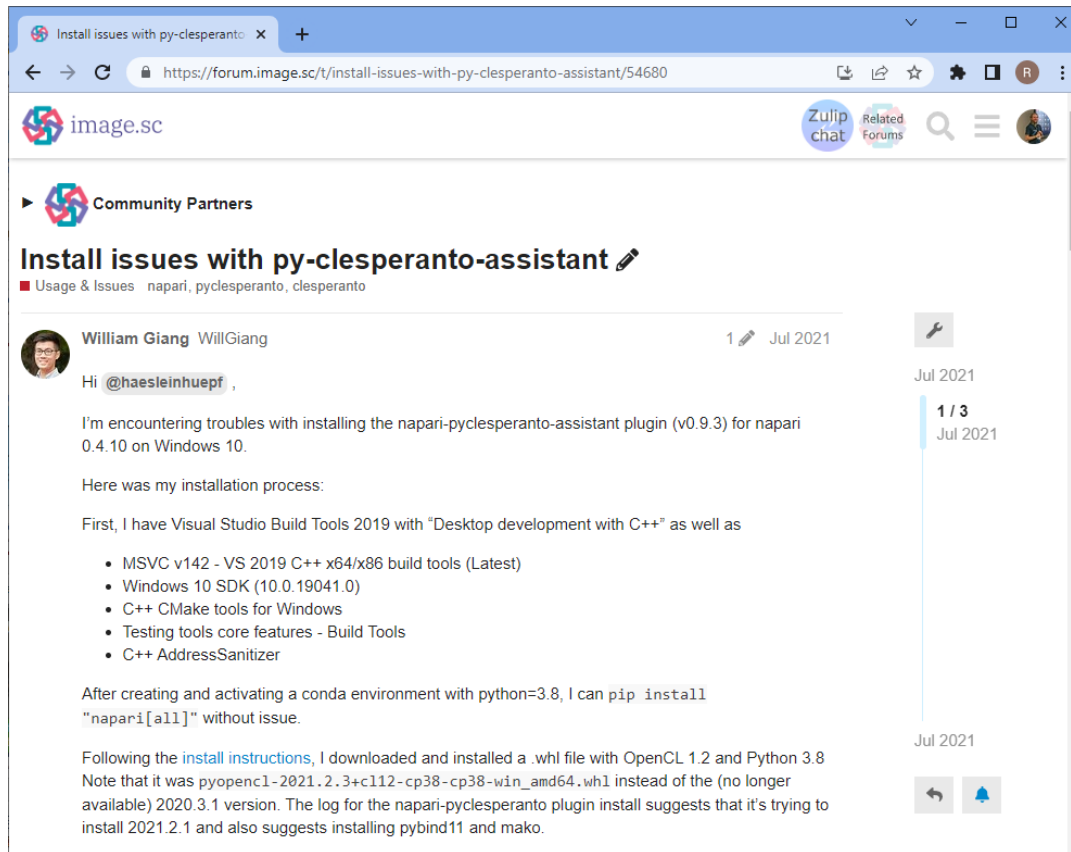
Stéfan van der Walt, Juan Nunez-Iglesias, SciPy 2019



Sreenivas Bhattiprolu, Python for Microscopists @Youtube 2019-...
April 2023

The Image Science Community

- Ask your question online and an expert will likely reply the same day 😊



Install issues with py-clesperanto

image.sc

Community Partners

Install issues with py-clesperanto-assistant

Usage & Issues napari, pyclesperanto, clesperanto

William Giang WillGiang Jul 2021

Hi @haesleinhuepf ,

I'm encountering troubles with installing the napari-pyclesperanto-assistant plugin (v0.9.3) for napari 0.4.10 on Windows 10.

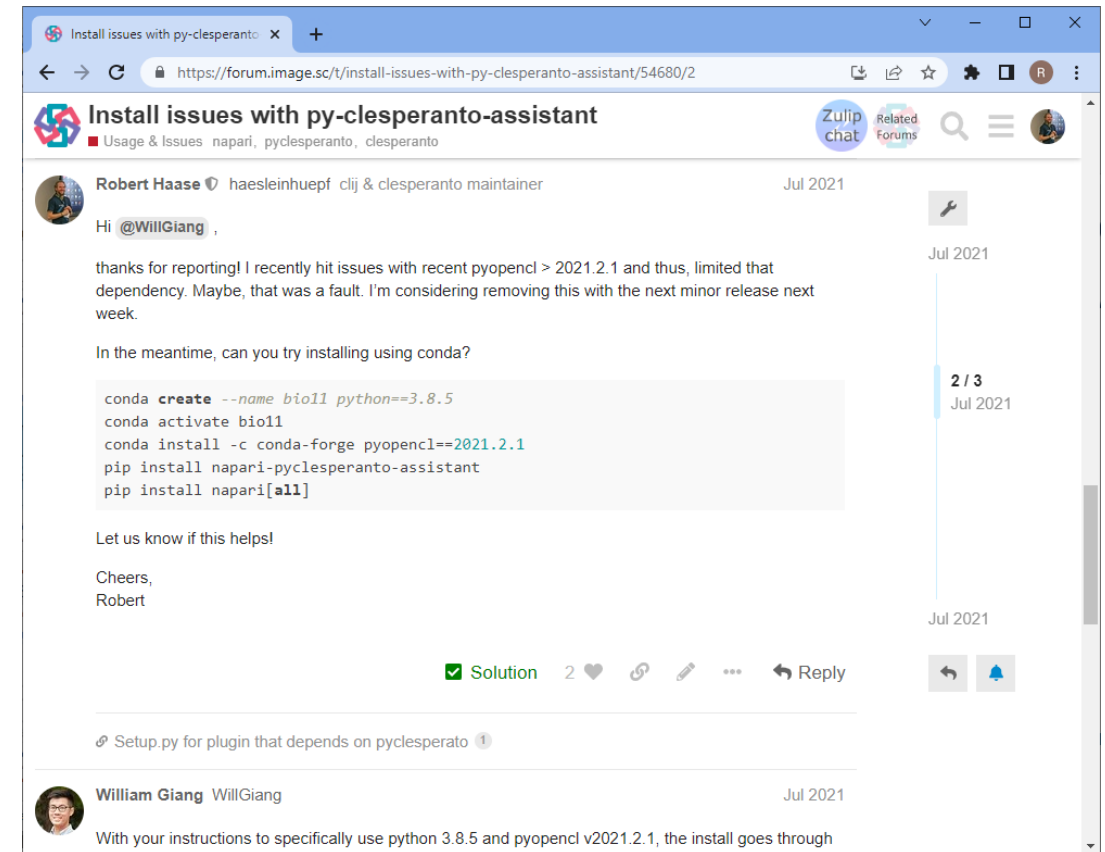
Here was my installation process:

First, I have Visual Studio Build Tools 2019 with "Desktop development with C++" as well as

- MSVC v142 - VS 2019 C++ x64/x86 build tools (Latest)
- Windows 10 SDK (10.0.19041.0)
- C++ CMake tools for Windows
- Testing tools core features - Build Tools
- C++ AddressSanitizer

After creating and activating a conda environment with python=3.8, I can pip install "napari[all]" without issue.

Following the [install instructions](#), I downloaded and installed a .whl file with OpenCL 1.2 and Python 3.8. Note that it was pyopenc1-2021.2.3+c112-cp38-cp38-win_amd64.whl instead of the (no longer available) 2020.3.1 version. The log for the napari-pyclesperanto plugin install suggests that it's trying to install 2021.2.1 and also suggests installing pybind11 and mako.



Install issues with py-clesperanto

Usage & Issues napari, pyclesperanto, clesperanto

Robert Haase haesleinhuepf clij & clesperanto maintainer Jul 2021

Hi @WillGiang ,

thanks for reporting! I recently hit issues with recent pyopenc1 > 2021.2.1 and thus, limited that dependency. Maybe, that was a fault. I'm considering removing this with the next minor release next week.

In the meantime, can you try installing using conda?

```
conda create --name bio11 python==3.8.5
conda activate bio11
conda install -c conda-forge pyopenc1==2021.2.1
pip install napari-pyclesperanto-assistant
pip install napari[all]
```

Let us know if this helps!

Cheers,
Robert

Solution 2 ❤️ 🔗 ✎️ ⋮ ↩️ Reply

Setup.py for plugin that depends on pyclesperanto 1

William Giang WillGiang Jul 2021

With your instructions to specifically use python 3.8.5 and pyopenc1 v2021.2.1, the install goes through

Today, you learned

- *Bio-image analysis*
 - Quantitative
 - Objective
 - Reproducible
 - Repeatable
 - Reliable
- The command line
- Jupyter Lab
- Python programming
 - Variables
 - Arrays (lists, tuples)
 - Dictionaries

`data[start:stop:step]`

Coming up next

- Loops
- Conditions
- Functions
- Libraries

```
# going through arrays pair-wise
measurement_1 = [1, 9, 7, 1, 2, 8, 9, 2, 1, 7, 8]
measurement_2 = [4, 5, 5, 7, 4, 5, 4, 6, 6, 5, 4]

for m_1, m_2 in zip(measurement_1, measurement_2):
    print("Paired measurements: " + str(m_1) + " and " + str(m_2))
```

```
Paired measurements: 1 and 4
Paired measurements: 9 and 5
Paired measurements: 7 and 5
Paired measurements: 1 and 7
Paired measurements: 2 and 4
Paired measurements: 8 and 5
Paired measurements: 9 and 4
Paired measurements: 2 and 6
Paired measurements: 1 and 6
Paired measurements: 7 and 5
Paired measurements: 8 and 4
```

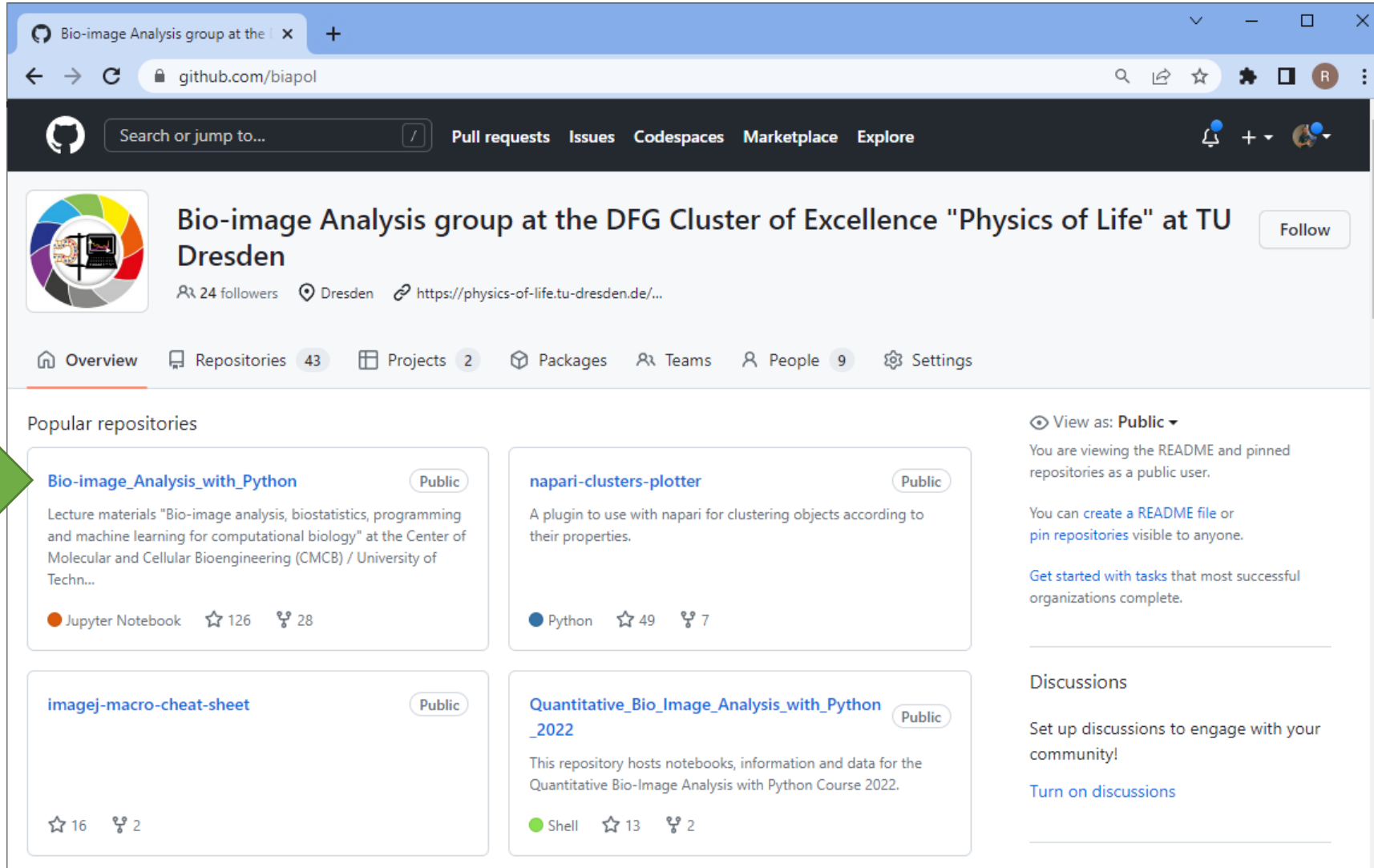
Exercises

Robert Haase

April 2023

Exercise: Explore our GitHub repository

- <https://github.com/biapol>



Bio-image Analysis group at the DFG Cluster of Excellence "Physics of Life" at TU Dresden

24 followers · Dresden · <https://physics-of-life.tu-dresden.de/...>

Overview · Repositories 43 · Projects 2 · Packages · Teams · People 9 · Settings

Popular repositories

- Bio-image_Analysis_with_Python** (Public)
Lecture materials "Bio-image analysis, biostatistics, programming and machine learning for computational biology" at the Center of Molecular and Cellular Bioengineering (CMCB) / University of Techn...
Jupyter Notebook · 126 stars · 28 forks
- napari-clusters-plotter** (Public)
A plugin to use with napari for clustering objects according to their properties.
Python · 49 stars · 7 forks
- imagej-macro-cheat-sheet** (Public)
16 stars · 2 forks
- Quantitative_Bio_Image_Analysis_with_Python_2022** (Public)
This repository hosts notebooks, information and data for the Quantitative Bio-Image Analysis with Python Course 2022.
Shell · 13 stars · 2 forks

View as: Public

You are viewing the README and pinned repositories as a public user.

You can [create a README file](#) or [pin repositories](#) visible to anyone.

[Get started with tasks](#) that most successful organizations complete.

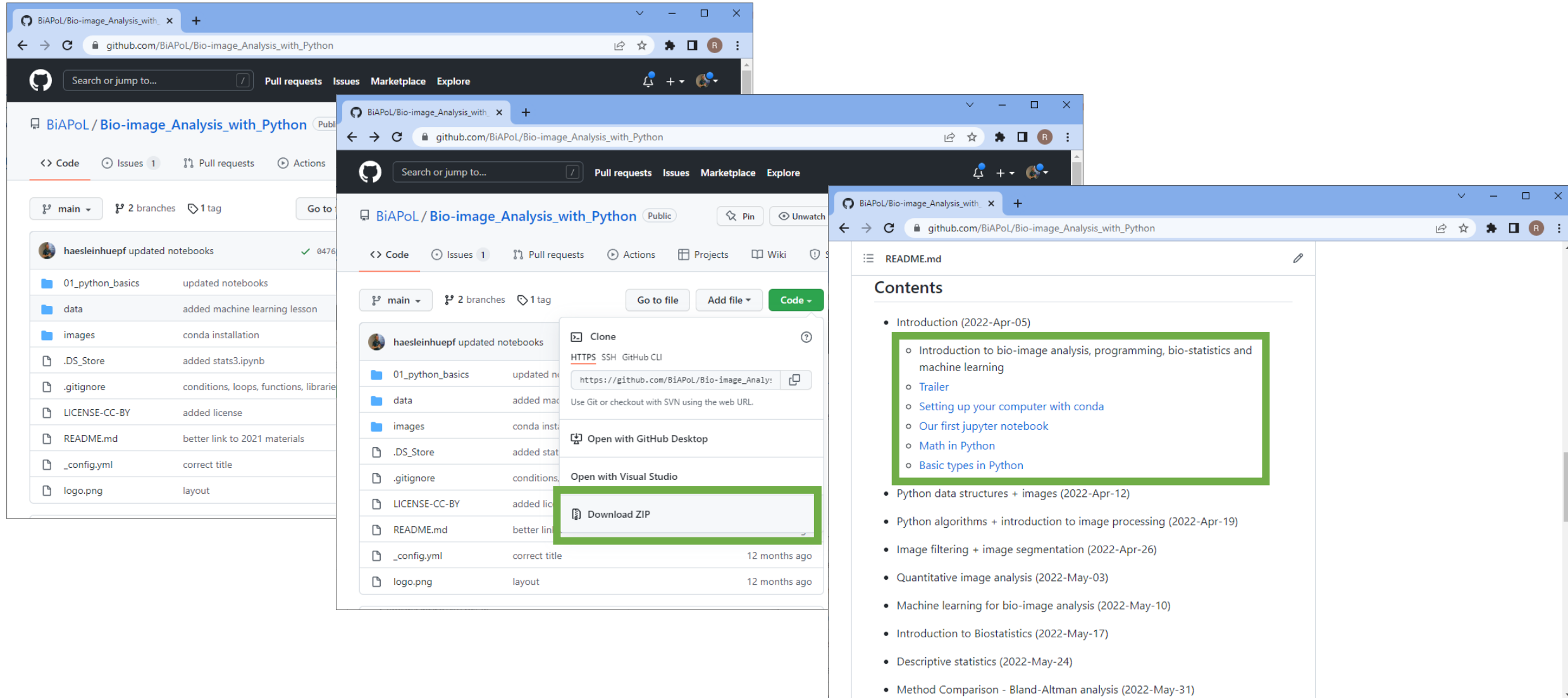
Discussions

Set up discussions to engage with your community!

[Turn on discussions](#)

Exercise: Explore our GitHub repository

- https://github.com/BiAPoL/Bio-image_Analysis_with_Python



The image shows three overlapping screenshots of a GitHub repository page for 'BiAPoL/Bio-image_Analysis_with_Python'. The top-left screenshot shows the repository's file structure, including folders like '01_python_basics', 'data', and 'images', and files like '.DS_Store', '.gitignore', 'LICENSE-CC-BY', 'README.md', and 'logo.png'. The middle screenshot shows the 'Code' button highlighted, with a dropdown menu open showing options: 'Clone' (with sub-options for HTTPS, SSH, and GitHub CLI), 'Open with GitHub Desktop', 'Open with Visual Studio', and 'Download ZIP' (highlighted with a green box). The bottom-right screenshot shows the 'README.md' file, which contains a 'Contents' section with a list of topics. A green box highlights the first five items in the list: 'Introduction (2022-Apr-05)', 'Introduction to bio-image analysis, programming, bio-statistics and machine learning', 'Trailer', 'Setting up your computer with conda', 'Our first jupyter notebook', 'Math in Python', and 'Basic types in Python'.

BiAPoL / Bio-image_Analysis_with_Python

Code Issues 1 Pull requests Actions

main 2 branches 1 tag

haesleinhuepf updated notebooks ✓ 0476

- 01_python_basics updated notebooks
- data added machine learning lesson
- images conda installation
- .DS_Store added stats3.ipynb
- .gitignore conditions, loops, functions, libraries
- LICENSE-CC-BY added license
- README.md better link to 2021 materials
- _config.yml correct title
- logo.png layout

BiAPoL / Bio-image_Analysis_with_Python

Code Issues 1 Pull requests Actions Projects Wiki

main 2 branches 1 tag

haesleinhuepf updated notebooks

- 01_python_basics updated notebooks
- data added machine learning lesson
- images conda installation
- .DS_Store added stats3.ipynb
- .gitignore conditions, loops, functions, libraries
- LICENSE-CC-BY added license
- README.md better link to 2021 materials
- _config.yml correct title
- logo.png layout

Clone

- HTTPS SSH GitHub CLI
- https://github.com/BiAPoL/Bio-image_Analysis_with_Python
- Use Git or checkout with SVN using the web URL.

Open with GitHub Desktop

Open with Visual Studio

Download ZIP

README.md

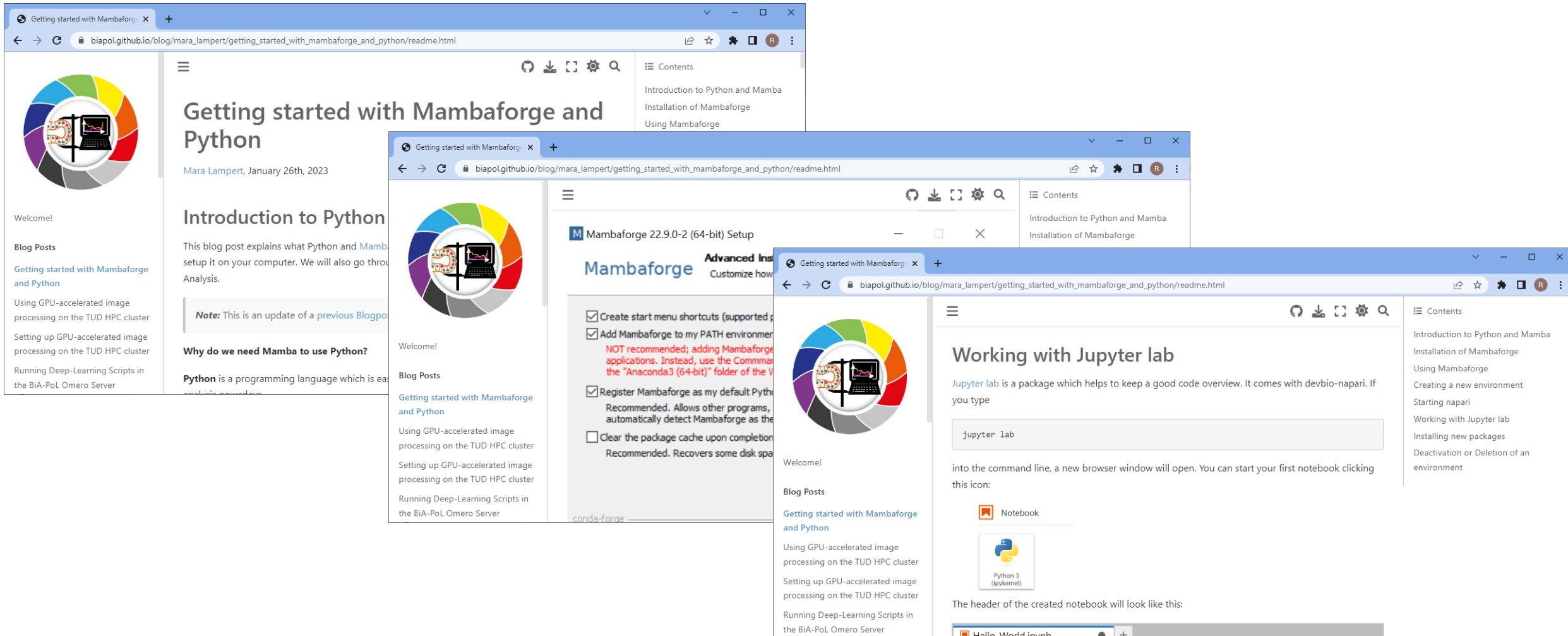
Contents

- Introduction (2022-Apr-05)
 - Introduction to bio-image analysis, programming, bio-statistics and machine learning
 - Trailer
 - Setting up your computer with conda
 - Our first jupyter notebook
 - Math in Python
 - Basic types in Python
- Python data structures + images (2022-Apr-12)
- Python algorithms + introduction to image processing (2022-Apr-19)
- Image filtering + image segmentation (2022-Apr-26)
- Quantitative image analysis (2022-May-03)
- Machine learning for bio-image analysis (2022-May-10)
- Introduction to Biostatistics (2022-May-17)
- Descriptive statistics (2022-May-24)
- Method Comparison - Bland-Altman analysis (2022-May-31)

Exercise: Install mambaforge and test Python

Detailed instructions:

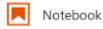

- https://biapol.github.io/blog/mara_lampert/getting_started_with_mambaforge_and_python/readme.html



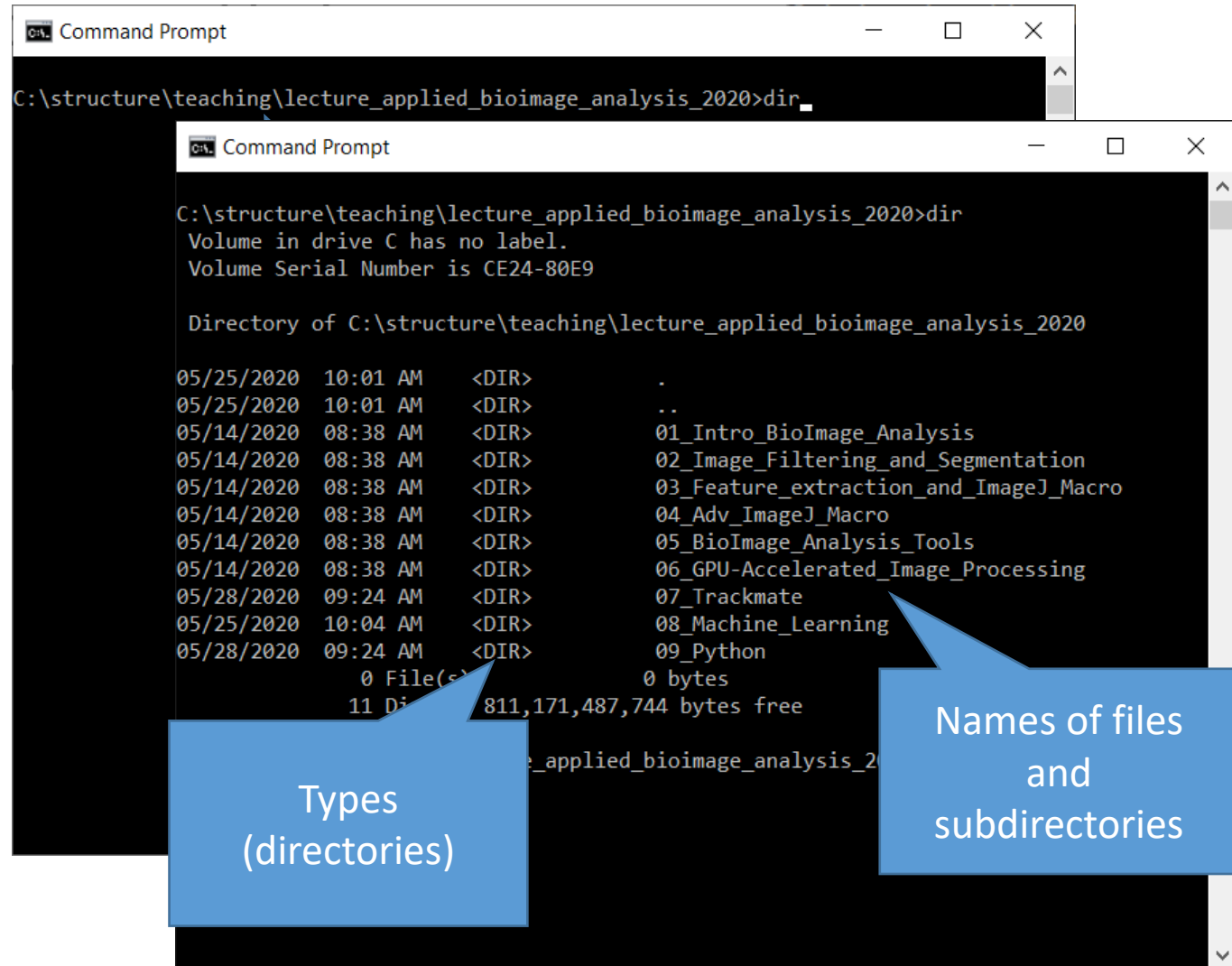
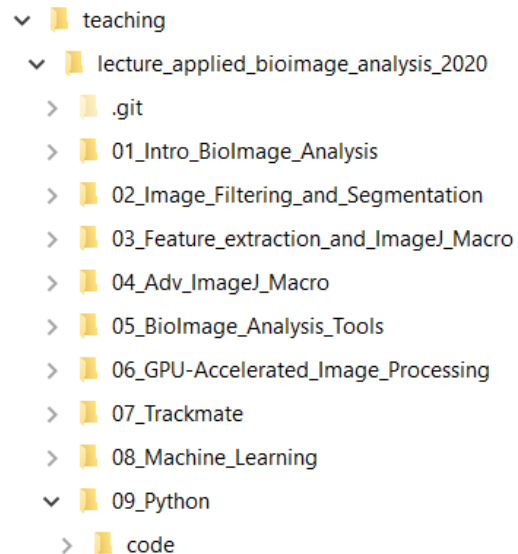
The image displays three overlapping screenshots from a web browser, illustrating the installation and testing of Mambaforge and Python.

The top-left screenshot shows the article "Getting started with Mambaforge and Python" by Mara Lampert, dated January 26th, 2023. The article includes an introduction to Python and Mambaforge, and a section titled "Why do we need Mamba to use Python?".

The top-right screenshot shows the "Mambaforge 22.9.0-2 (64-bit) Setup" window. The window displays the Mambaforge logo and the text "Advanced Installation". The "Create start menu shortcuts" and "Add Mambaforge to my PATH environment" options are checked. A note states: "NOT recommended; adding Mambaforge applications. Instead, use the Command Prompt to run the 'Anaconda3 (64-bit)' folder of the V...".

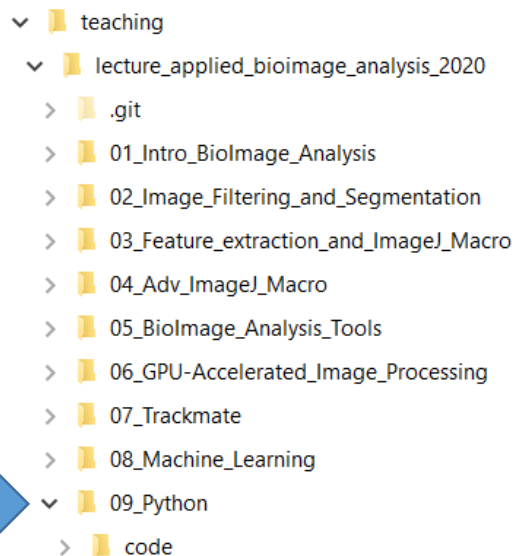
The bottom screenshot shows the "Working with Jupyter lab" section of the article. It explains that Jupyter lab is a package which helps to keep a good code overview. It comes with devbio-napari. If you type `jupyter lab` into the command line, a new browser window will open. You can start your first notebook clicking this icon: . The header of the created notebook will look like this: . The screenshot also shows a terminal window with the command `Hello World.ipynb`.

- A.k.a. the Terminal or Eingabeaufforderung: Welcome to the 20th century!
- The `dir` command tells you what's in the current directory
- On Mac and Linux the command is called `ls -l`



The command line

- A.k.a. the Terminal or Eingabeaufforderung: Welcome to the 20th century!
- The `cd` command let's you move between different directories.
- With `cd <pathname>` you go into a sub-directory



```
Command Prompt
C:\structure\teaching\lecture_applied_bioimage_analysis_2020>cd 09_Python

Command Prompt
C:\structure\teaching\lecture_applied_bioimage_analysis_2020>cd 09_Python
C:\structure\teaching\lecture_applied_bioimage_analysis_2020\09_Python>dir

Command Prompt
C:\structure\teaching\lecture_applied_bioimage_analysis_2020>cd 09_Python
C:\structure\teaching\lecture_applied_bioimage_analysis_2020\09_Python>dir
Volume in drive C has no label.
Volume Serial Number is CE24-80E9

Directory of C:\structure\teaching\lecture_applied_bioimage_analysis_2020\09_Python

05/28/2020  09:24 AM    <DIR>          .
05/28/2020  09:24 AM    <DIR>          ..
05/28/2020  09:24 AM                26,985,803  09_Switching_from_ImageJ_to_Python.pptx
05/28/2020  09:42 AM    <DIR>          code
05/14/2020  11:52 AM                9,169,920  python_installation.pptx
                2 File(s)    36,155,723 bytes
                3 Dir(s)   811,170,570,240 bytes free

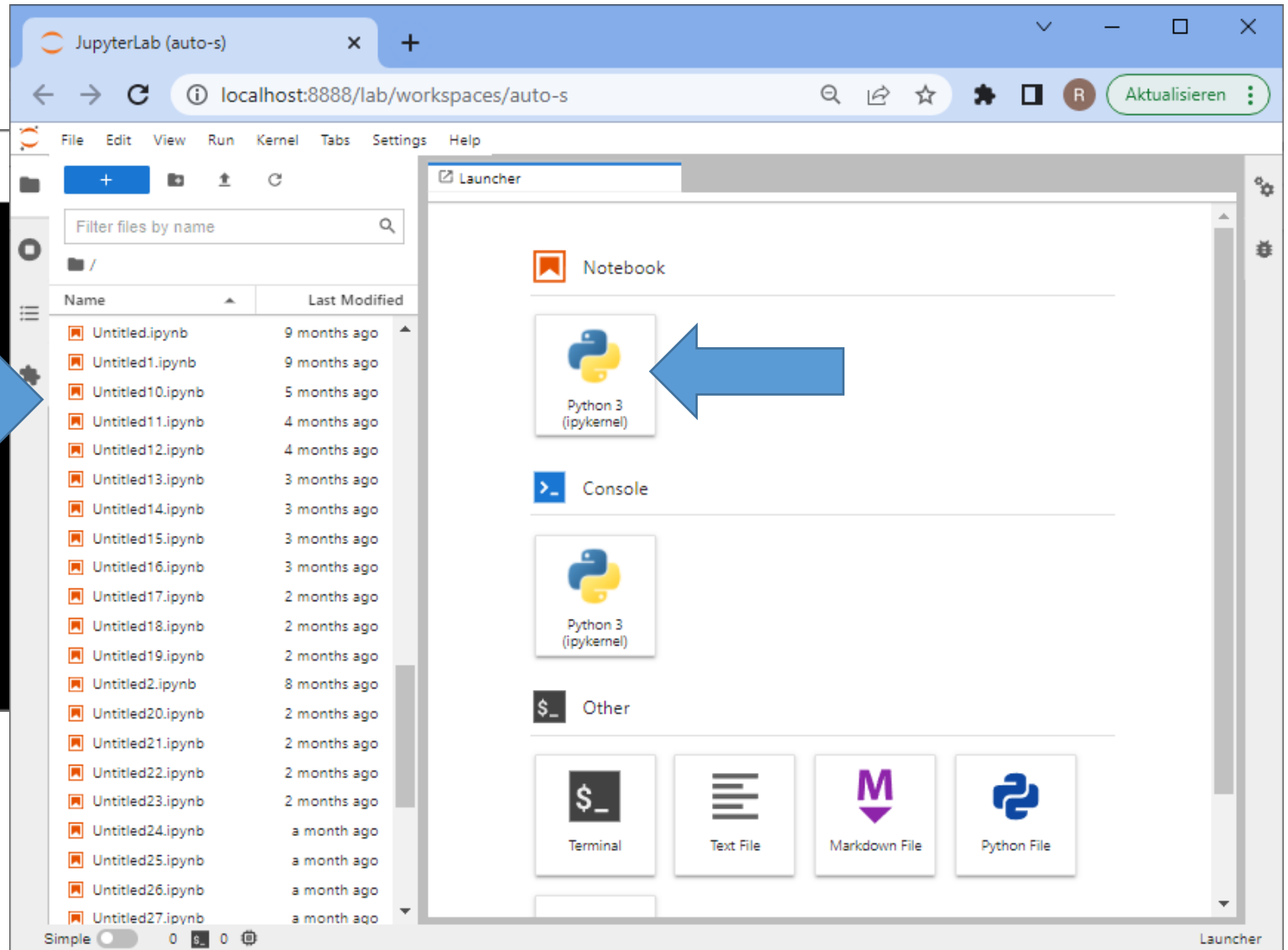
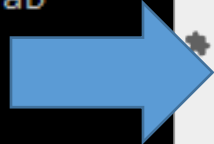
C:\structure\teaching\lecture_applied_bioimage_analysis_2020\09_Python>
```

File size in bytes

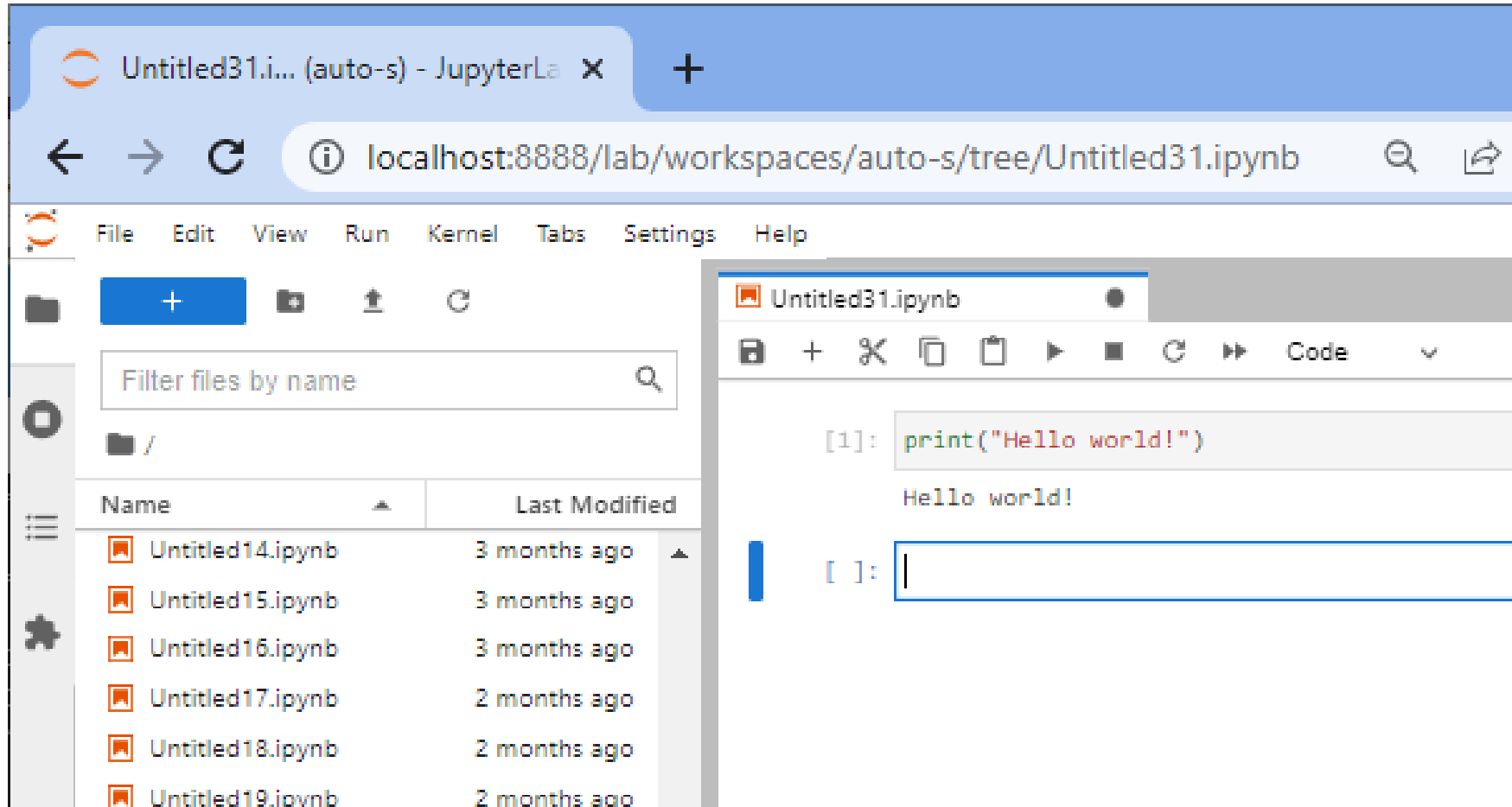
File names

- Start Jupyter lab from the folder you want to work in

```
Command Prompt - conda deactivate - cond...
c:\Users\rober>conda activate bio_39
(bio_39) c:\Users\rober>jupyter lab
```



- Execute code cell-by-cell and see results instantaneously

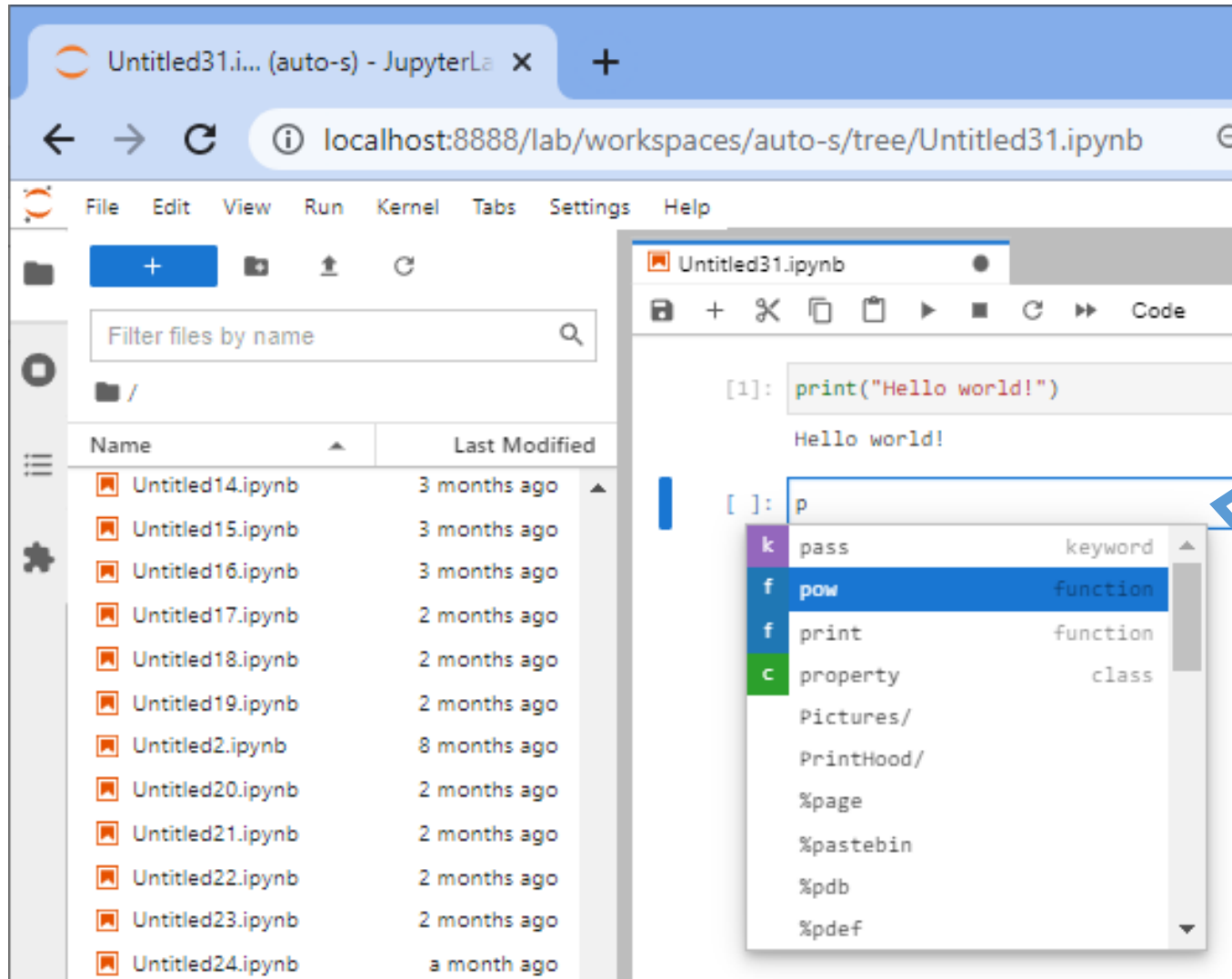


The screenshot shows the JupyterLab interface in a browser window. The address bar displays `localhost:8888/lab/workspaces/auto-s/tree/Untitled31.ipynb`. The interface includes a menu bar (File, Edit, View, Run, Kernel, Tabs, Settings, Help) and a file browser on the left. The main area shows a code cell for `Untitled31.ipynb` with the following content:

```
[1]: print("Hello world!")  
Hello world!  
[ ]: |
```

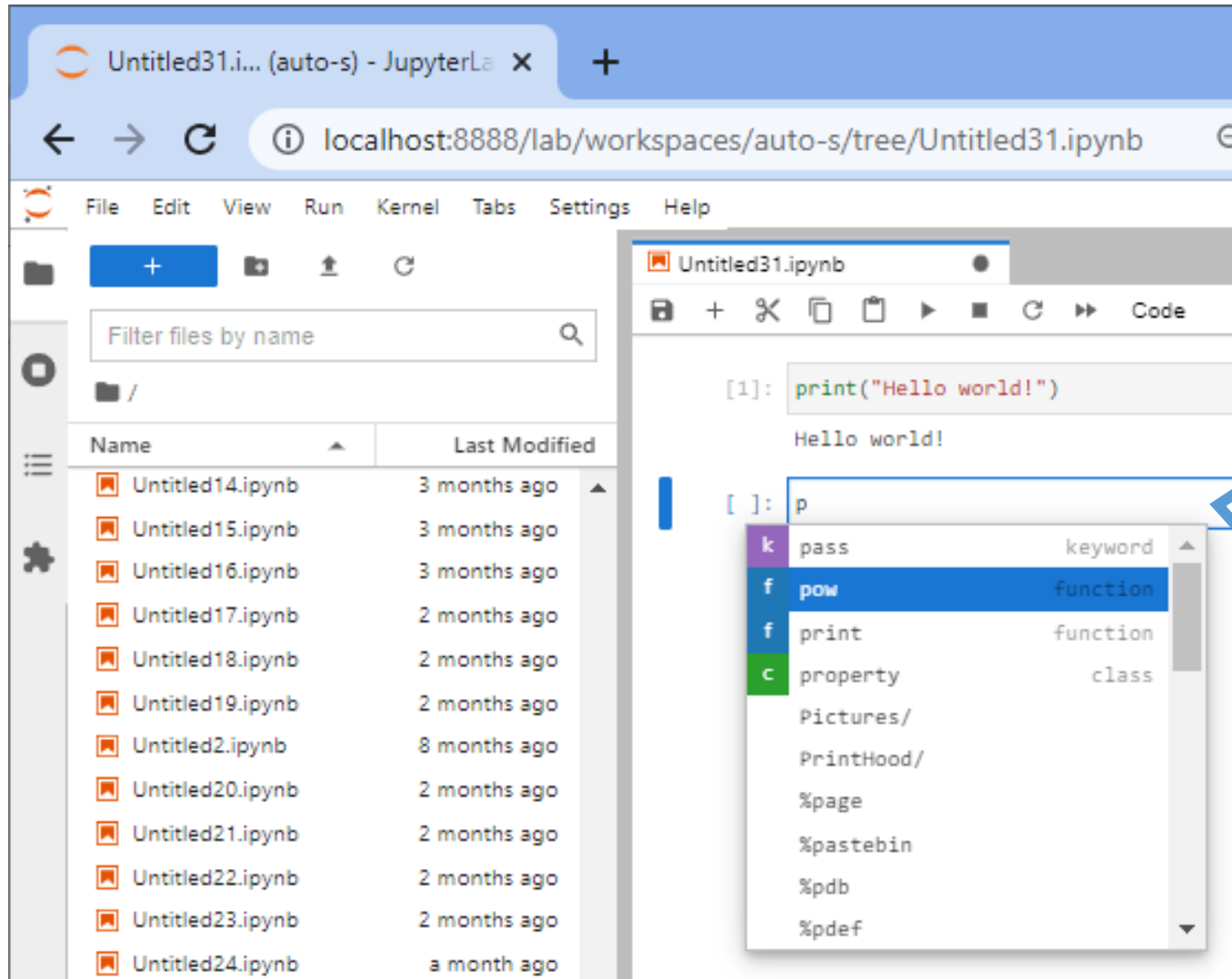
SHIFT + ENTER
to execute a
code cell

- Context-specific help, auto-completion



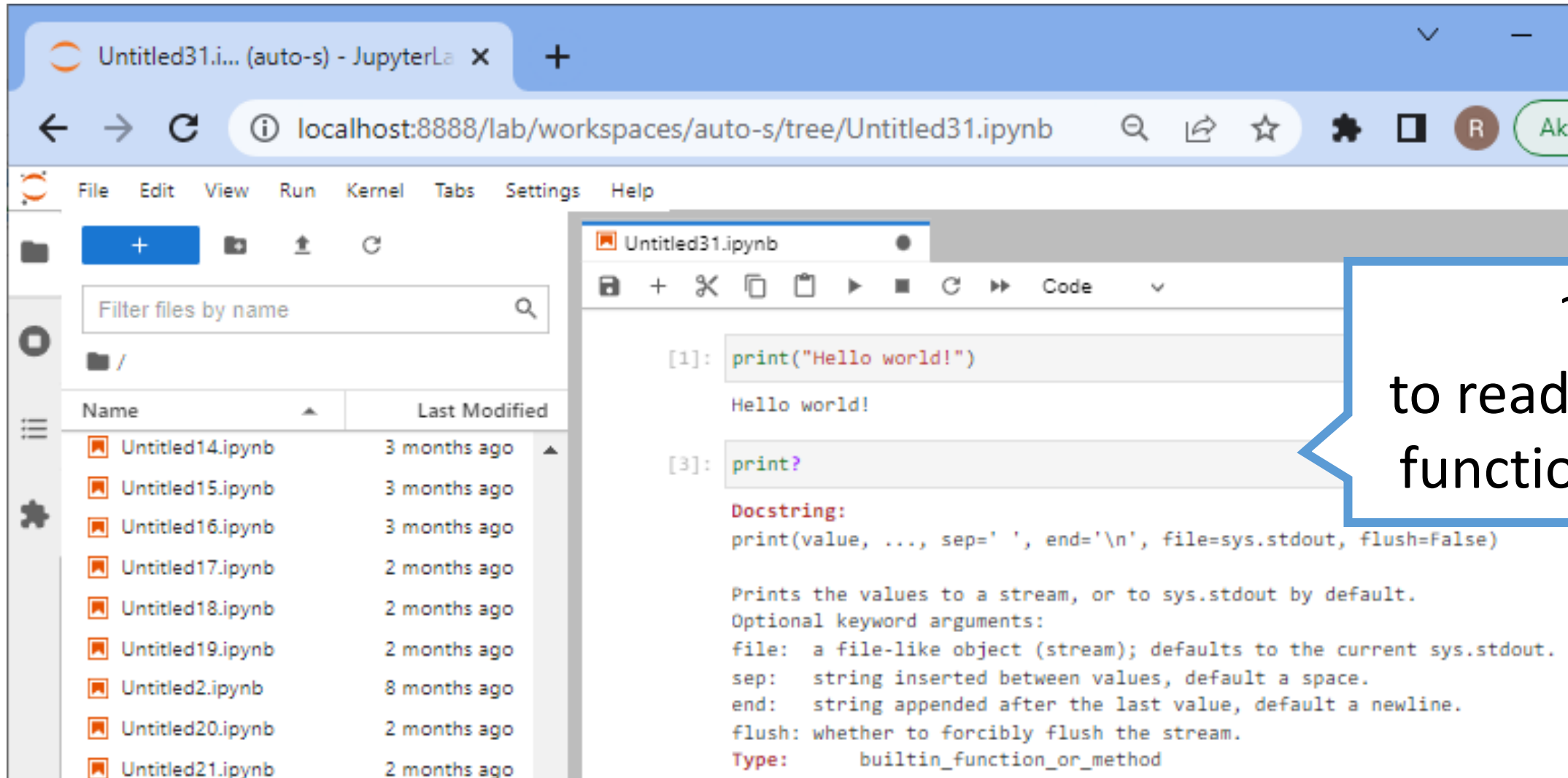
TAB
to open auto-
completion

- Context-specific help, auto-completion



TAB
to open auto-
completion

- Help / “docstrings”



The screenshot shows the JupyterLab interface. On the left is a file browser with a search bar and a list of files. On the right is a code editor for 'Untitled31.ipynb'. The code editor shows two code cells. The first cell contains `print("Hello world!")` and has executed, showing the output 'Hello world!'. The second cell contains `print?` and has also executed, showing the docstring for the `print` function.

Name	Last Modified
Untitled14.ipynb	3 months ago
Untitled15.ipynb	3 months ago
Untitled16.ipynb	3 months ago
Untitled17.ipynb	2 months ago
Untitled18.ipynb	2 months ago
Untitled19.ipynb	2 months ago
Untitled2.ipynb	8 months ago
Untitled20.ipynb	2 months ago
Untitled21.ipynb	2 months ago

```
[1]: print("Hello world!")
Hello world!

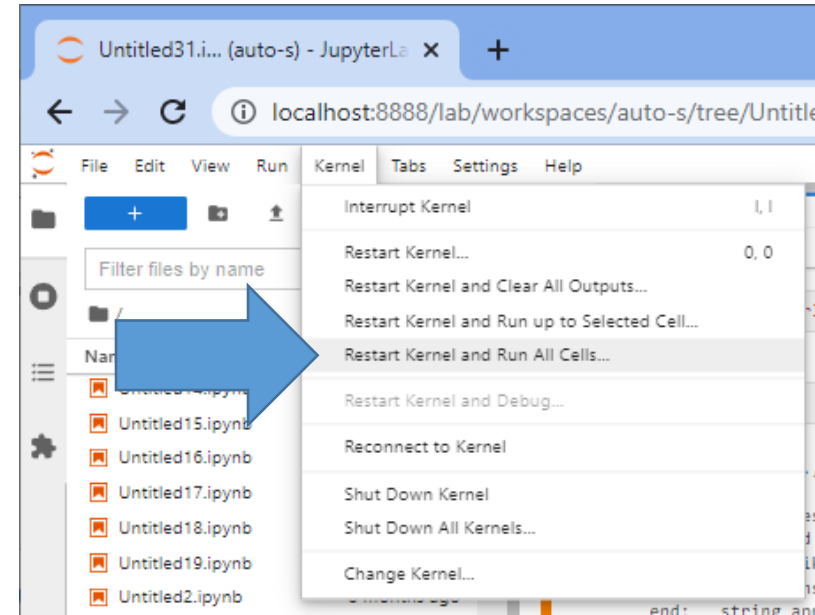
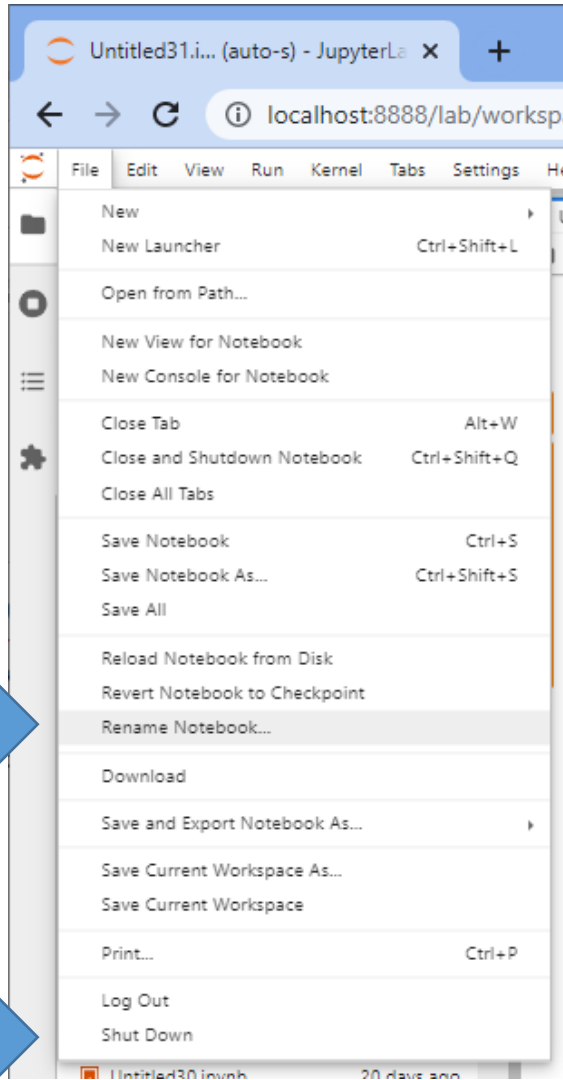
[3]: print?

Docstring:
print(value, ..., sep=' ', end='\n', file=sys.stdout, flush=False)

Prints the values to a stream, or to sys.stdout by default.
Optional keyword arguments:
file: a file-like object (stream); defaults to the current sys.stdout.
sep:  string inserted between values, default a space.
end:  string appended after the last value, default a newline.
flush: whether to forcibly flush the stream.
Type: builtin_function_or_method
```

?
to read what a
function does

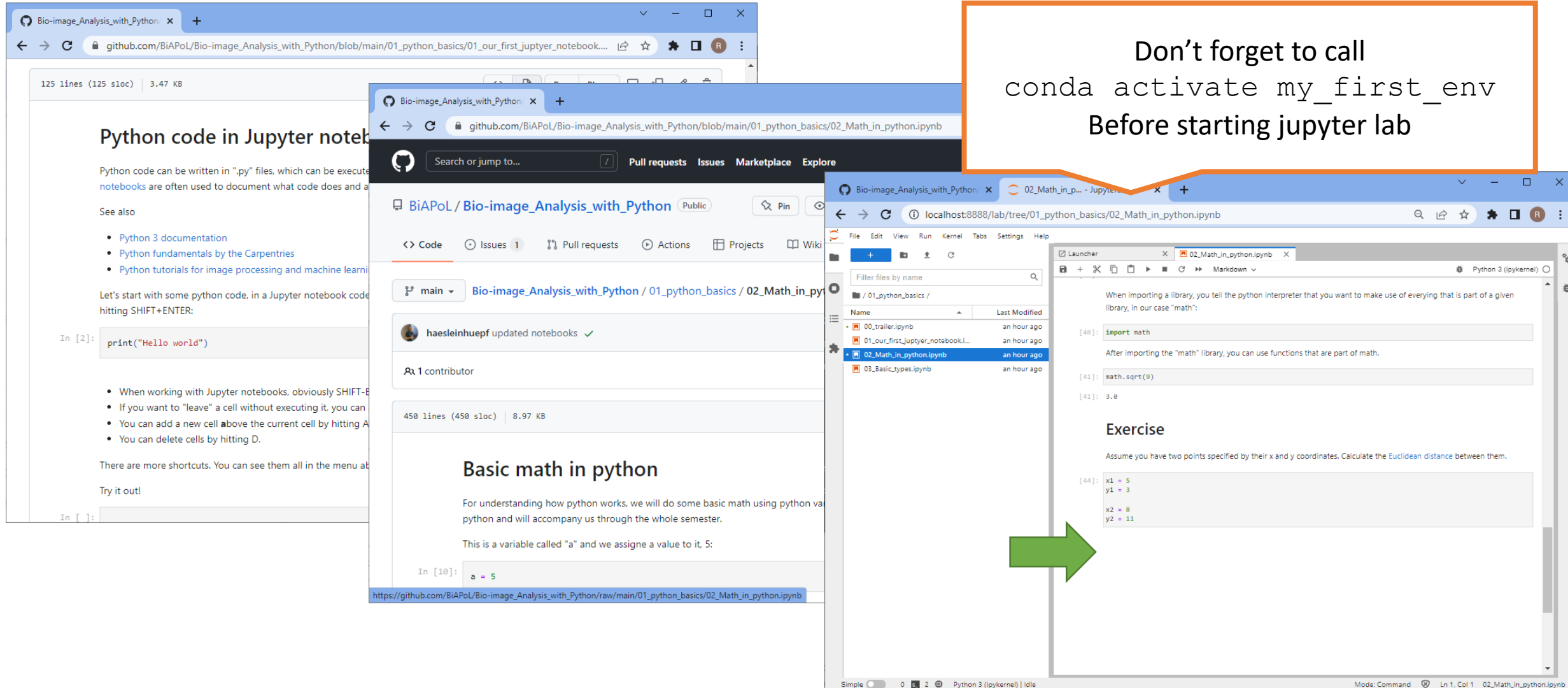
- Saving / renaming / closing



Enforcing a “clean” execution state is important for ensuring reproducibility and repeatability

Exercise: Basic python

- [https://github.com/BiAPoL/Bio-image Analysis with Python](https://github.com/BiAPoL/Bio-image_Analysis_with_Python)



Don't forget to call
conda activate my_first_env
Before starting jupyter lab

Python code in Jupyter notebook

Python code can be written in ".py" files, which can be executed in Jupyter notebooks. Notebooks are often used to document what code does and how it works.

See also

- Python 3 documentation
- Python fundamentals by the Carpentries
- Python tutorials for image processing and machine learning

Let's start with some python code, in a Jupyter notebook code cell. To execute the code, hit SHIFT+ENTER:

```
In [2]: print("Hello world")
```

- When working with Jupyter notebooks, obviously SHIFT+ENTER is not always the best option.
- If you want to "leave" a cell without executing it, you can hit ESC.
- You can add a new cell above the current cell by hitting A.
- You can delete cells by hitting D.

There are more shortcuts. You can see them all in the menu at the top.

Try it out!

```
In [ ]:
```

Basic math in python

For understanding how python works, we will do some basic math using python variables. This will accompany us through the whole semester.

This is a variable called "a" and we assign a value to it. 5:

```
In [10]: a = 5
```

02_Math_in_python.ipynb

02_Math_in_python.ipynb

When importing a library, you tell the python interpreter that you want to make use of everything that is part of a given library, in our case "math":

```
[40]: import math
```

After importing the "math" library, you can use functions that are part of math.

```
[41]: math.sqrt(9)
```

```
[41]: 3.0
```

Exercise

Assume you have two points specified by their x and y coordinates. Calculate the Euclidean distance between them.

```
[44]: x1 = 5  
y1 = 3  
x2 = 8  
y2 = 11
```