Bioimage Analyses and Extended Phenotyping - Exercises

Christian Kappel

October 20, 2023

Contents

- Exercises are subject to modification based on progress and feedback.
- Updated exercise sheets are put on the Moodle course page.
- Active participation is expected from everybody (condition for exam admission).
- You may show active participation by showing up from time to time and by handing in 2 exercises. One exercise from part 2 (data analysis) by the end of November. One exercise from part 3.2 on by the end of the lecture period (aim for the most difficult one).
- For those who aren't able show up from time to time, you have to hand in at least 3 exercises from the image processing part.
- The dates when the exercises are discussed in the course session or on the Moodle forum are given at section, subsection or exercise level.
- You are encouraged to prepare the exercises prior to discussion session.
- There is no use to work through all exercises if you're already familiar with it. Just do the minimum requirements, hand in some exercises. Or negociate.

Part 1

Introduction to Python

1.1 Getting started (2023-10-27)

Exercise 1.1.1 Python setup

1. Please download and install the Anaconda Python distribution with Python version 3.9, https://www.anaconda.com. We particularly need the following modules: scikit-image, pandas and matplotlib.
2. Start the Python interpreter and print "Hello World!" on the screen.
3. Check that all the necessary modules are installed by entering the following commands:

```
import skimage
import pandas
import matplotlib
```

4. Close the program.

Exercise 1.1.2 Python as a calculator

1. Solve the following using Python: $1 + 1$.
2. An now a more difficult one: 2^{10}.
3. Calculate the cubic root of 8: $\sqrt[3]{8}$.
4. Calculate the following: $"Hello" + "World"$.

Exercise 1.1.3 Numbers

1. What's the difference between 2 and 2.0?
2. Calculate the following and comment the output (make sure you're using Python 3):

```
7 / 3
7 // 3
7 % 3
4 / 2
```

Exercise 1.1.4 Variables

Variables start with a letter or an underscore (_) followed by the same or digits. That's the same for all Python identifiers.

1. Assign the string "Hello world"! to the variable a".
2. Print the content of the variable in your console.
3. Append the variable by " My name is X.".
4. Assign 2 and 3.5 to the variables b and c.
5. Assign the sum of b and c to d.

Exercise 1.1.5 Modules

1. Import everything from the math module.
2. Get π from the math module and assigns it to the variable pi.
3. Can you only import pi from the math module?
4. Calculate the area of a circle with radius 5 cm.

Exercise 1.1.6 Getting help

Where can you get help if you have some problems?

1.2 Data types (2023-10-27)

Python has 5 standard data types: String, number, list, tuple, dictionary. Additional data types and structures are added by modules, like the NumPy array numpy.ndarray you got when loading the Mimulus flower image.

Exercise 1.2.1 Special string characters

1. Print ”Hello world!” on the screen, including the double quotes.
2. Print it again using single quotes now (’Hello world!’).
3. Now print ”Hello world, it’s me!”, including the double quotes.

Exercise 1.2.2 Lists

Lists are a collection of items, their values can change.

1. Put the numbers 1 to 5 in a list a.
2. Print the type of a.
3. Extract the first item from the list and assign it to a variable a1.
4. Change the third item to the number 99.
5. Extend a by adding the numbers 6 and 7 to the end of the list.

Exercise 1.2.3 Tuples

Tuples are similar to lists. Yet values are fixed after creation.

1. Put the numbers 1 to 5 in a tuble b.
2. Print the type of b.
3. Try to modify the last item in the tuple. Comment.
4. Add the numbers 6 to 10 to b.
5. Get the length of b.
6. Get the sum of all items in b.
7. Calculate the mean of all items in b.

Exercise 1.2.4 Mixed lists

1. Put the following items in a list d: 1, 2, 3, 4, 5, ”Hello world!”, 4.0.
2. Print the type of the second, third and last item.

Exercise 1.2.5 Slices

1. Get the first 3 items of list d.
2. Get items 2 and 3.
3. Get all but the first item.
4. Extract the third, fourth and the first item and assign it to the variable e.

Exercise 1.2.6 Character lists

1. Assign the string ”abcdefghijklmnapqrstuvwxyz” to the variable abc.
2. Print out the type of abc.
3. Get the length of the string in abc. Compare to the length of a.
4. Extract the first 7 letters from abc.
5. Extract the last 7 letters from abc.
6. Print every second character in abc.
7. Invert abc and print it out.

Exercise 1.2.7 Dictionaries

A dictonary allows you to map a key to one or multiple values. You can create a simple dictonary as follows:

```
>>> e = { 'a' : 1, 'b' : 2, 'c' : 3 }
```

1. Print the type of e.
2. Get the value with the key b.
3. Add the value 4 with the key d.
4. Create another dictionary f containing the age and year of birth of Fritz (44, 1970) and Jean (43, 1970).
5. Print out the age of Fritz.

Exercise 1.2.8 Multidimensional list

You can create a multidimensional list in the following way.

```
table = []
table.append([1,2,3,4,5])
table.append([6,7,8,9,10])
table.append([11,12,13,14,15])
```

1. Print out the content of table.
2. Print out the second row ([6,7,8,9,10]).

3. Print out the first element of the third row (11).
4. Calculate the mean of the first row.

1.3 Control structures (2023-11-03)

Exercise 1.3.1 if

Have a look at the following example. Python enforces indentation, there is a tabulation before the print statement.

```
if 1 == 1:
  print("The numbers are the same")
```

1. Run the example on your computer.
2. Try without a tabulation before the print statement.
3. Can you use spaces instead of a tabulation?
4. Can you mix spaces and tabulations?

Exercise 1.3.2 if...else

```
x = 11
```

1. Test if x is 0. If yes, than print "The number is 0", otherwise print "The number is above 0.".
2. Is the above control correct? Does it handle all possible outcomes?
3. Write a control to check if the number is above or below 0.

Exercise 1.3.3 while

```
x = 0
while x < 10:
  print(x)
  x = x + 1
```

1. Run on your computer.
2. Use this structure to print out the content of the list containing the whole alphabet.

Exercise 1.3.4 for

```
x = [1, 2, 3, 4, 5]
```

1. Print out all numbers in x one by one.
2. Print out all numbers from 1 to 100.
3. Print out all odd numbers from 1 to 100.

1.4 Working environment (2023-11-03)

Exercise 1.4.1 Python IDE

1. Identify some Python IDE for your used operating system.
2. Try out one or more of these IDE.
3. Present the basic functionalities of a selected IDE to your colleagues.

Exercise 1.4.2 Running Python scripts

There are multiple ways to run a Python script, sometimes depending on your operating system.

1. Create a new file and name it script01.py.
2. Make it to print out ”script01.py is running!”.
3. Run the script from your operating system.
4. Add a comment at the top of the file to describe it’s purpose.

Part 2

Data analysis

2.1 Python Data Analysis Library: pandas (2023-11-10)

”pandas is an open source, BSD-licensed library providing high-performance, easy-to-use data structures and data analysis tools for the Python programming language.” See http://pandas.pydata.org for more information.

Exercise 2.1.1 Reading data from a file

1. Name and describe pandas data structures.
2. Load the data from the iris.tsv file into a pandas DataFrame.
3. Show the first 5 rows of the data. Then show the last five rows of the data.

Exercise 2.1.2 DataFrame and data extraction

1. Print out the first 10 entries of the first column of the DataFrame.
2. Extract the Sepal.Length column and save in a variable. What’s the type of the data in the new variable?
3. Print out the first 5 entries of the first two columns.
4. Print out the last 5 rows for Petal.Width and Species.
5. Extract data for the species virginica only.

Exercise 2.1.3 DataFrame and data summarization

1. Get the minimum, maximum, mean and median petal width.
2. Get the number of entries per species.
3. Plot species count as barplot.
4. Get the mean petal width by species.

Exercise 2.1.4 Filling-up a DataFrame

1. Create a DataFrame with three columns species, height and weight and add an entry to the end.
2. Create a DataFrame with the same columns and 50 empty entries. Fill-up the data frame row by row with 50 random entries.
3. Update every second entry, set the species name to *M.musculus*.
4. Save the DataFrame to your harddisk as a csv file. Include the header but not the row indexes.
5. Save the DataFrame as a tsv file (tab separated values).

Part 3

Image processing

We are going to use scikit-image (van der Walt et al., 2014). You may also have a look at http://scikit-image.org/ to get more information. Documentation on the scikit-image Website is a good starting point to get help for solving the exercises, as is you favorite search engine.

3.1 Loading, viewing and writing image data (2023-11-17)

Import skimage input/output (i/o) and data modules.

```
import skimage.io
import skimage.data
```

skimage.io is needed to load and write images, skimage.data provides some examples.

Exercise 3.1.1 Images

1. Get the Mimulus-77.jpg image from the Moodle2 page.
2. Load the image into the variable im:

   ```
   import skimage.io as io
   im = io.imread("data/Mimulus-77.jpg")
   ```

3. Plot the image:

   ```
   io.imshow(im)
   io.show() % not necessary in Spyder
   ```

4. Save the file in bitmap format.
5. Is there a size difference between the initial JPG and the bitmap file? Why?
6. Which one is better in terms of image quality?

Exercise 3.1.2 Visualize a simple image

1. Load the coins image provided by skimage.data and store it in a variable.

   ```
   img1 = skimage.data.coins()
   ```

2. Visualize the image on your screen. First you need to send the image to plotting (via skimage.io.imshow) and then you need to show it on the screen (via skimage.io.show).

   ```
   skimage.io.imshow(img1)
   skimage.io.show()
   ```

   You can continue working in Python when the plotting window is closed.

Exercise 3.1.3 Image structure

1. How is the image from the previous exercise stored/represented in Python? What's its data type?
2. What are the dimensions of the image?

3. What are the possible values of each entry?

Exercise 3.1.4 Some notes on importing Python modules

Python modules can be imported in different way. You can give modules defined (short) names to avoid writing the whole module name, or you can import single functions or module parts.

1. Close python and open it again, to have a blank python console
2. Import only the coins function from skimage.data
3. Import the data module from skimage
4. There are now two ways to address the coins functions! How?
5. A very efficient way is to give a module a short name

Exercise 3.1.5 Loading and saving images

1. Get the coins.jpg image from Moodle.
2. Open a new Python session and import the 2 skimage modules (skimage.io and skimage.data) and give it short names, e.g.:

```
import skimage.io as io
import skimage.data as data
```

3. Load the coins.jpg image via the imread function from the skimage.io module. Note the you need to now the location of you image (or you put it directly in the folder where you opened python). Note that the file name has to be given in quotation marks.
4. Look at the image on you screen (as you did in the first exercise of this chapter)
5. Save the image on the hard drive under a different name using the imsave funtion from the skimage.io module. Note: you have to give two parameters to the function, first write the file name in quotation marks followed by the variable holding the image (separate the parameters via a comma as usual).

Exercise 3.1.6 Processing multiple images

1. Download the Capsella-leaves.zip archive from moodle and extract it in a separated directory.
2. Get all the filenames into a list using the glob module.
3. Loop through the list and save all files in the bitmap format.

Exercise 3.1.7 Extracting image parts

1. Create a new variable/image with by cutting the area from 100 to 200 in x-direction and from 150 to 250 in y-direction from the coins image.
2. Visualize on the screen.
3. Get the lena.jpg image from Moodle and load it into your Python session.
4. Get the dimensions of the image. What's different compared to the coins image?

5. Get the area 100:200 in x- and y-direction but only for the green channel.
6. Visualize the extracted part and save it to your drive.

3.2 Intensity transformation and spatial filtering (2023-11-24)

You can do things in different ways. Feel free to implement transformations by yoursef or use existing scikit-image functions.

Exercise 3.2.1 Intensity transformations

1. Load the image Gonzalez2009-Fig3.4.tif (Gonzalez and Woods, 2009, Figure 3.4), get its negative (invert intensity values) and save it to a png file.
2. Load the image Gonzalez2009-Fig3.5.tif (Gonzalez and Woods, 2009, Figure 3.5), apply a log-transformation as described in the lecture and visualize both the original image and the transformed image one after another. What does the log transformatino do here?
3. Load the image Mimulus-CRW5320.jpg and apply gamma corrections with gamma values above and below 1. Visualize and comment your output images.
4. Feel free to test other intensy transformations shown in the lecture.

Exercise 3.2.2 Global and local histogram equalization

1. Load the image Gonzalez2009-Fig3.26.tif (Gonzalez and Woods, 2009, Figure 3.26).
2. Apply a global and local histogram equalization and compare the two output images.
3. Find a simple real world example where local histogram equalization gives superior results to global histogram equalization.

Exercise 3.2.3 Spatial transformations

1. Load the images Gonzalez2009-Fig3.33.tif and Gonzalez2009-Fig3.35.tif (Gonzalez and Woods, 2009, Figures 3.33 and 3.35).
2. Apply an average filter to both images, test at least three different filter sizes. Save the transformed images.
3. Apply an median filter to both images and save the transformed images.
4. Sharpen both images and save the transformed images.
5. Can you revert the sharpened images bach to their initial state?

3.3 Segmentation (2023-12-01)

Exercise 3.3.1 Simple thresholding

1. Load the image A.alpina-flower-with-key.jpg.
2. Choose an appropriate intensity threshold to segment flower and key.
3. Get an approximation of the total flower area.

Exercise 3.3.2 Global vs adaptive thresholding

Load the page image from skimage.data:

```
from skimage import data
im1 = data.page()
```

1. Do a global thresholding using Otsu's method.
2. Do an adaptive thresholding.
3. Put all three images into the same plot. Which thresholding method gives the better result?
4. Save your combinded plot as a jpg and tif image. What are the differences between the two formats?
5. Can you get a similarly good result using local histogram equalization? Try it out!

Exercise 3.3.3 Edge detection

1. Load the rectangle.png image.
2. Do Prewitt, Sobel and Canny edge detection using skimage functions. Save the outputs to separate image files.
3. Investigate the outputs of Laplacian and horizontal and vertical Prewitt filtering. You may use scipy.ndimage.filters.convolve and the kernels as given in the lecture. (optional).

3.4 Morphological operations (2023-12-08)

Exercise 3.4.1 Filtering out small particles

1. How can you filter out small (background) particles.
2. Go back to the simple thresholding exercise and determine the exact flower area after filtering out small particles.

Exercise 3.4.2 Outline extraction

1. Load the Mimulus-77.jpg image and segment the flower. Make sure to filter background particles and to fill internal holes. Save the resulting binary image.
2. Extract the outline of the image.
3. Visualize the outline in blue within the original image, save it under a different name.

Exercise 3.4.3 Connected components

1. Load the Barley_seeds_1.jpg image and segment the seeds.
2. Identify and label all seeds (connected components).
3. How many of them are there?
4. Print out the size for each one of them? Aim for the simplest way.

3.5 Image descriptors (2023-12-15, 2024-01-12, 2024-01-19)

Exercise 3.5.1 Counting objects (2023-12-15)

1. Load the seed images Barley_seeds_1.jpg and Barley_seeds_2.jpg representing two different genotypes. Segment the seeds, make sure to filter out all background noise.
2. Extract the area for each seed.
3. Is there a significant difference in seed size between those in _1 and _2?

Exercise 3.5.2 Extracting object properties (2023-12-15)

1. Extract all necessary parameters for the seeds in both Barley images from the previous exercise that are necessary to calculate extent, solidity and roundness. Save everything in a single Pandas DataFrame.
2. Calculate extent, solidity and roundness for each seed. Is there a significant difference for one of those between the two genotypes.

Exercise 3.5.3 Size difference (2024-01-12)

1. Review exercise 3.5.1 using the full photos (Barley_seeds_1_with_key.jpg, Barley_seeds_2_with_key.jpg). Is there a significant size difference between the seeds in the two pictures? Please use the easiest and fastest approach you can think of.
2. Please give the sizes in mm^2. Barcode and key were printed on a standard DIN A4 paper sheet (portrait mode).

Exercise 3.5.4 Comparing leaf sizes and shapes (2024-01-12)

1. Get the *Capsella* leaf images from Moodle (Capsella-leaves.zip) and save them in a subdirectory.
2. Get all leaf image filenames in a variable (at best including the relative or absolute path).
3. Segment all images and extract area and perimeter in cm^2 and cm respectively. All images were scanned at 300 dpi.
4. Is there a difference in area between images from species A and B. See filenames to differentiate A and B.
5. Is there a difference in dissection index (inverse roundness) between A and B.
6. Save filename, species, area, perimeter and dissection index in a csv-file.

Exercise 3.5.5 Comparing flower intensity distributions (2024-01-19)

1. Get the *Mimulus* flower images from Moodle (Mimulus-flowers.zip) and save them in a sub-directory.
2. Extract some basic intensity statistics (mean, median, sd) for each flower. Choose freely how to handle the three color channels here?
3. Calculate and plot simple grayscale co-occurrence matrices considering the pixel on the right of each pixel. You may choose to simplify your image.
4. Can you separate the flowers in two groups using either simple intensity statistics or statistics based on the co-occurrence matrices?

Exercise 3.5.6 Studying red and blue flowers (2024-01-19)

1. Get and load the flowers-on-the-lawn.png picture. Let the red and blue circles be flowers on the green lawn.
2. Transform to HSV color space and count the number of red and blue flowers. Use the following approximation to start: 1 connected component = one flower.
3. Are there a statistically significant size and shape differences between the two flower types? Save the size and selected shape parameters to a file.
4. What proportion of the lawn is covored by red and blue flowers?
5. Propose as simple way how to count overlapping flowers of the same type.
6. Get and load the red-flower-fields-during-daytime-3.jpg. What proportion of the field are covered by red and blue flowers. Use the simple pixel count as a first approximation.
7. Is it possible to count the number of red and blue flowers? What are the major issues with that picture? How would you suggest to take pictures for that purpose?

Exercise 3.5.7 Flower field (2024-01-19)

1. Get and load the 1081-1276181898ETWp.jpg picture.
2. How many flowers are there for each color?
3. Can you think about a way how to automatically identify the different colors?

3.6 Object recognition (2024-01-26)

Exercise 3.6.1 Simple template matching

1. Get and load the triangles-and-squere.png and square.template.png images.
2. Get the position of the square object in the image using a template matching approach.

Exercise 3.6.2 Where's Waldo? Où est Charlie?

1. Get the Waldo_puzzle_small.jpg image and identify Waldo on it. How long did it take you to find him?
2. Use a template matching approach to find Waldo. How long does it take your program to find him?
3. Download more Waldo images from the Internet and test your template matching approach. (optional)

3.7 Convolutional neural networks (2024-02-09)

Exercise 3.7.1 Handwritten digit classification, MNIST

This exercise is based on the ”Simple MNIST convnet” example from the Keras Website, https://keras.io/examples/vision/mnist_convnet/.

1. Start by setting up keras in Anaconda. Run conda install -c conda-forge tensorflow. Test in python whether you can import the module (import tensorflow).
2. What is the MNIST dataset? How can you load it into Python.
3. Prepare tha data as described on the Website. Describe the following model. Build and train it as escribed on the Website.

```
import numpy as np
from tensorflow import keras
from tensorflow.keras import layers

num_classes = 10
input_shape = (28, 28, 1)

model = keras.Sequential([
  keras.Input(shape=input_shape),
  layers.Conv2D(filters=8, kernel_size=(3,3)),
  layers.MaxPooling2D(pool_size=(2,2)),
  layers.Flatten(),
  layers.Dense(num_classes, activation='softmax')
])

model.compile(loss="categorical_crossentropy", optimizer="adam", metrics=["accuracy"])
model.fit(x_train, y_train, epochs=3, validation_split=0.1)

score = model.evaluate(x_test, y_test, verbose=0)
print("Test loss:", score[0])
print("Test accuracy:", score[1])
```

4. Try to extend and improve your model by adding more layers for example. Compare it to the model on the Keras Website. You may find a commented tutorial on doing a similar digit classification by Victor Zhou on the following Website https://victorzhou.com/blog/keras-cnn-tutorial/. The suggested approach there gives a very high accuracy.
5. Write down some digits on a paper. Digitize them using your smartphone or a scanner for example. Extract individual digits and resize them to 28x28 pixel. Test if your model(s) can properly classify them using model.predict.

Part 4

Specials

4.1 Christmas segmentation work (2023-12-22)

Exercise 4.1.1 Christmas trees

1. Get the christmas-trees.zip archive from Moodle and decompress it in a separate directory.
2. Find a way to segment and count the number of trees for an individual image.
3. Segment and count the number of trees for all images. Save it in a table.

Exercise 4.1.2 Christmas trees and snowmans

1. Get the christmas-trees+snowmans.zip archive from Moodle and decompress it in a separate directory.
2. Find a way to segment and count the number of trees and snowmans for an individual image.
3. Segment and count the number of trees and snowmans for all images. Save it in a table.

Exercise 4.1.3 Making nicer images

1. Any ideas how to make aesthetically nicer images? Without them boing too big.

4.2 Networks (2024-02-02)

Exercise 4.2.1 Petal cell network

Get the skel-1504-1_1.png image from Moodle. It contains already segmented cells from a petal imprint image (with remaining errors).

1. How can you generate a network where neighbouring cells are linked out of this image? Discuss with your colleagues.
2. Implement your approach. You may work in a group.
3. Get some basic network statistics. How to do this? Start with the average number of neighbouring cells maybe.

Bibliography

Rafael C. Gonzalez and Richard E. Woods. *Digital Image Processing, 3/e(New Edition)*. Pearson Education, 2009. ISBN 9788131726952.

Stéfan van der Walt, Johannes L Schönberger, Juan Nunez-Iglesias, François Boulogne, Joshua D Warner, Neil Yager, Emmanuelle Gouillart, and Tony Yu. scikit-image: image processing in Python. *PeerJ*, 2:e453, jun 2014. ISSN 2167-8359. doi: 10.7717/peerj.453. URL `http://www.ncbi.nlm.nih.gov/pmc/articles/PMC4081273/https://peerj.com/articles/453`.