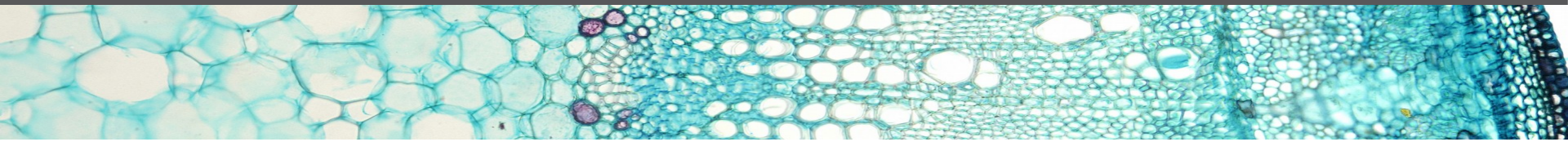
Programming bioimage analysis workflows in python and R using Jupyter Notebook

BCM



MRI
Montpellier Ressources Imagerie

cnrs Inserm

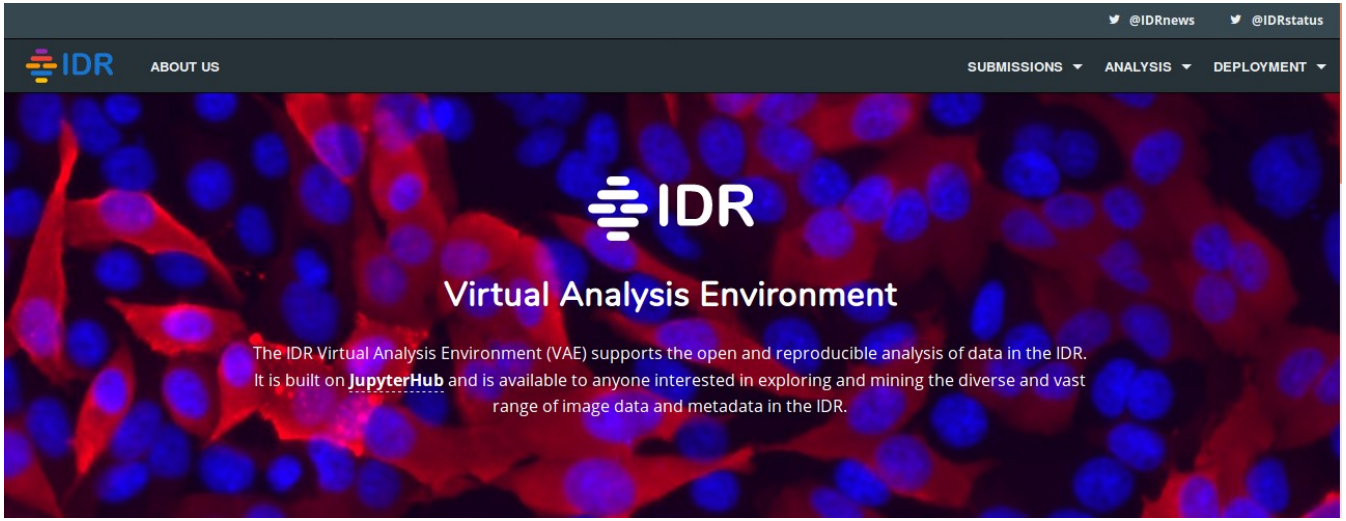
Outline

1) Motivation
2) Example IDR
3) Jupyter Notebook
4) OMERO
5) Python client for the OMERO Blitz API
6) Python in image analysis
7) R in statistical analysis

Interest of using Jupyter

- Reproducible Research
- Education
- Keep track of data analysis work
- Prototyping
- Share notebooks
- Open-source project
- Multi-language
- Interactive Output

IDR

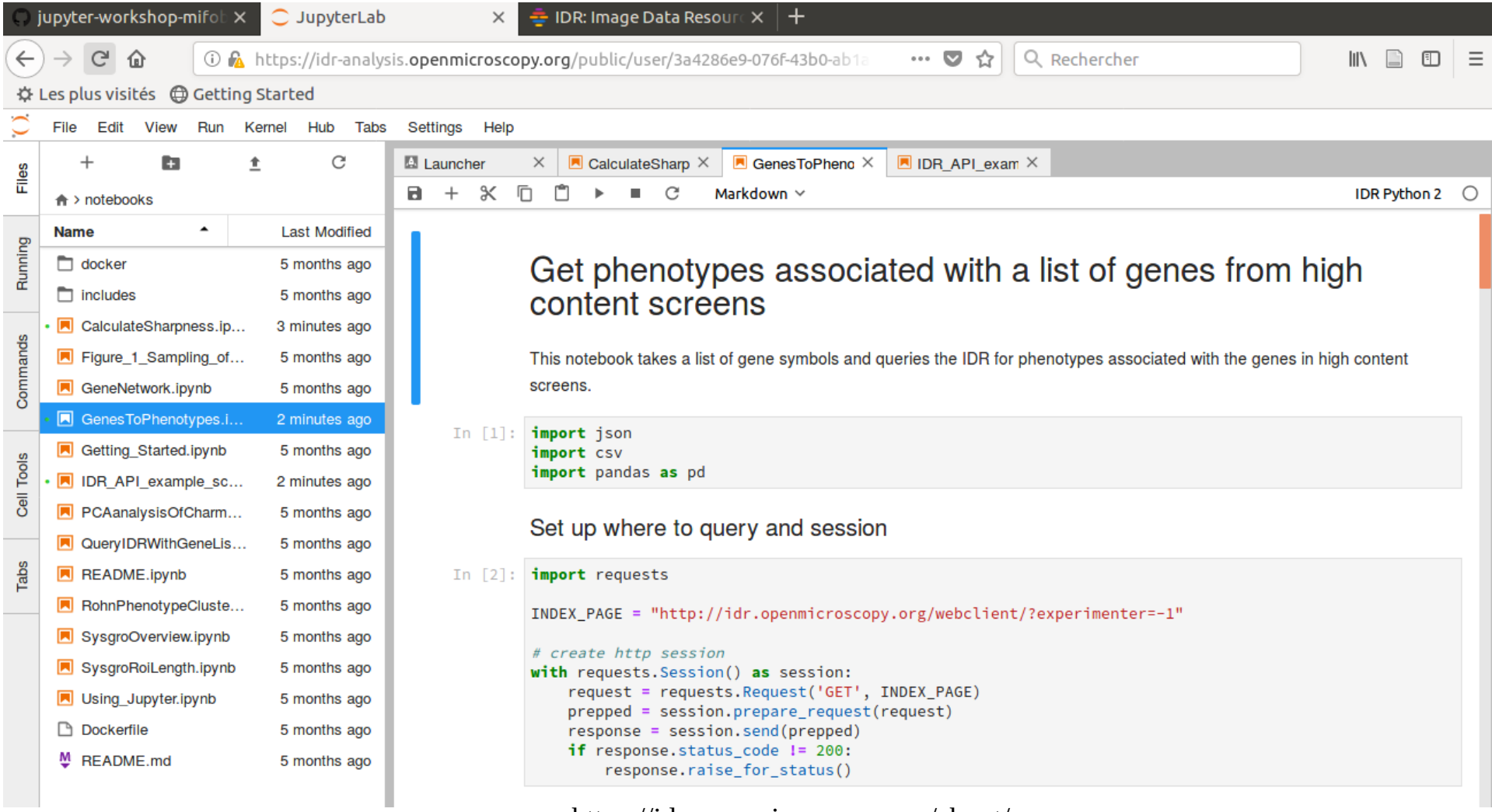


- ✗ Public data repository
  - ✔ Data from published scientific studies
- ✔ Image Database - OMERO
  - ✔ Data that is frequently accessed and cited
  - ✔ Links to public genetic or chemical database
  - ✔ Links to cell and tissue phenotype
- ✗ Analyze data – Jupyter
  - ✔ Enable re-analysis
  - ✔ Analysis of gene networks

Eleanor Williams, Josh Moore, Simon W Li, Gabriella Rustici, Aleksandra Tarkowska, Anatole Chessel, Simone Leo, Bálint Antal, Richard K Ferguson, Ugis Sarkans, Alvis Brazma, Rafael E Carazo Salas, Jason R Swedlow. Image Data Resource: a bioimage data integration and publication platform. Nature Methods, 2017; DOI: 10.1038/nmeth.4326

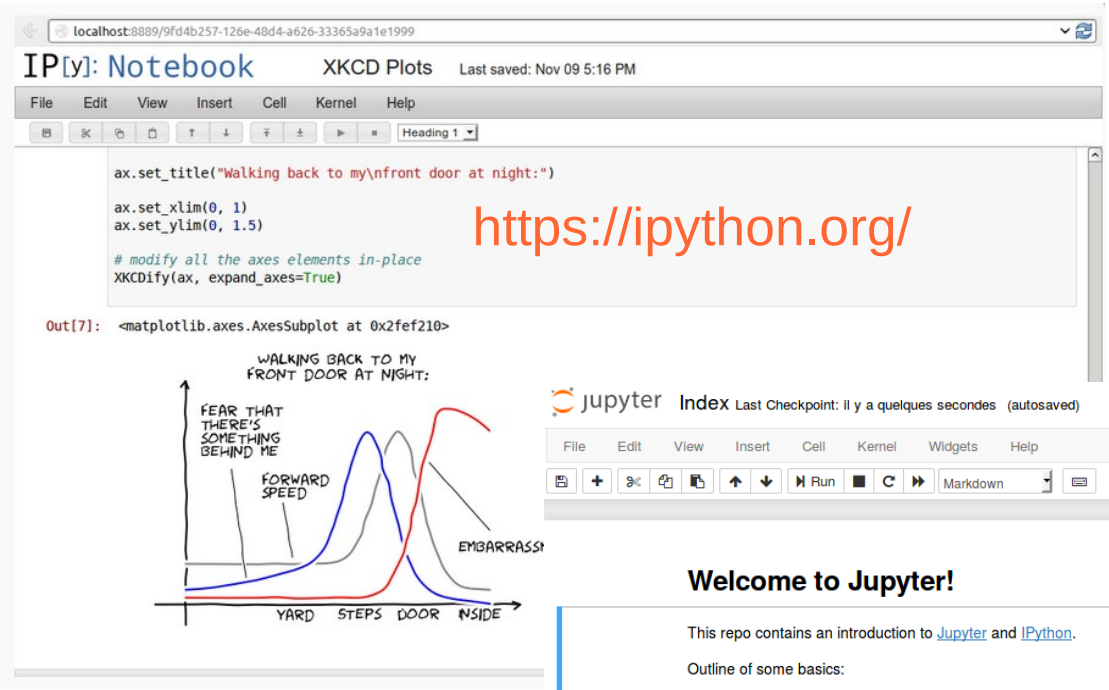
source: https://idr.openmicroscopy.org/about/

IDR

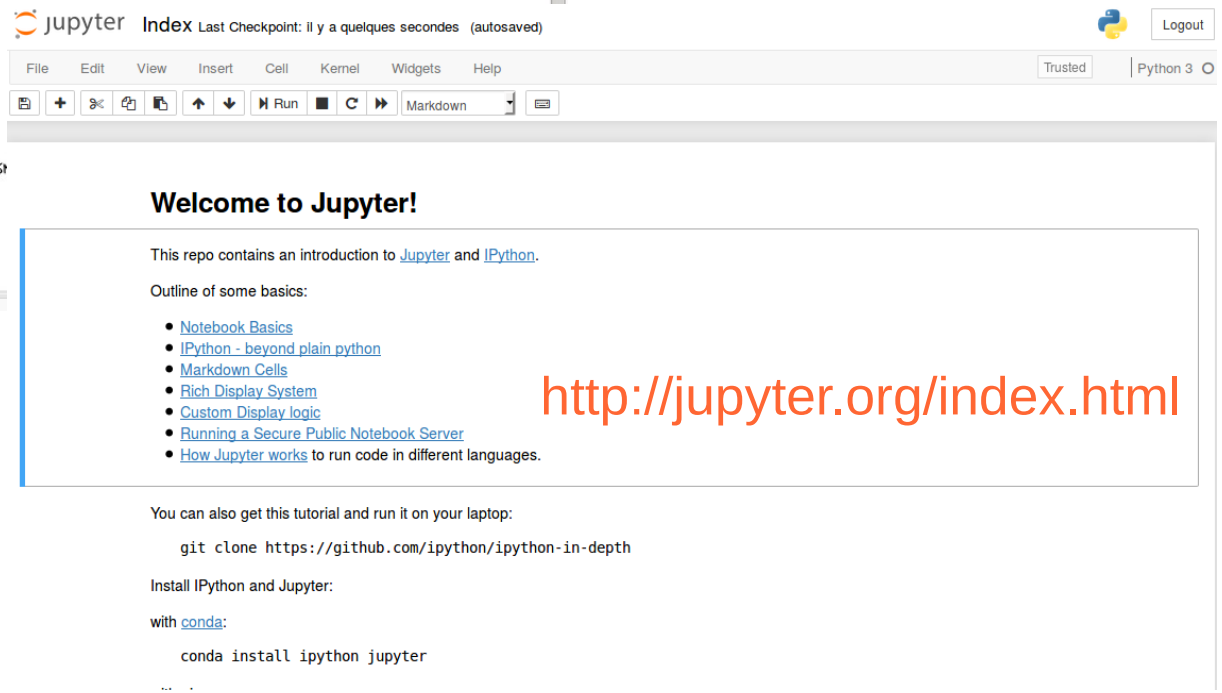


source: https://idr.openmicroscopy.org/about/

Jupyter Notebook



https://ipython.org/



http://jupyter.org/index.html

Jupyter Notebook

localhost:8889/notebooks/image_analysis.ipynb

Rechercher

Les plus visités Getting Started

jupyter image_analysis Last Checkpoint: 19/09/2018 (autosaved)

Logout

File Edit View Insert Cell Kernel Widgets Help

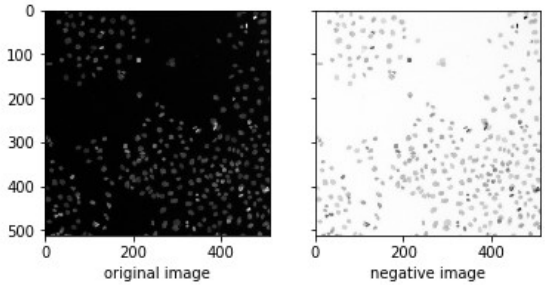
Trusted | Python 2

Run | Code

Manipulating images

Being numeric arrays, images can be manipulated by any of python's arithmetic operators. For example, we can produce a **negative image** by simply subtracting the image from its maximum value.

In [6]:
```
nuc_neg = nuc.max() - nuc
fig, axes = plt.subplots(ncols=2)
ax0, ax1 = axes.flatten()
ax0.imshow(nuc, cmap='gray')
ax0.set_xlabel("original image")
ax1.imshow(nuc_neg, cmap='gray')
ax1.set_xlabel("negative image")
plt.setp(ax1.get_yticklabels(), visible=False)
plt.show()
```



Jupyter Notebook
What is it ?

localhost:8889/notebooks/image_analysis.ipynb

Rechercher

Les plus visités Getting Started

jupyter image_analysis Last Checkpoint: 19/09/2018 (autosaved)

Logout

File Edit View Insert Cell Kernel Widgets Help

Trusted | Python 2

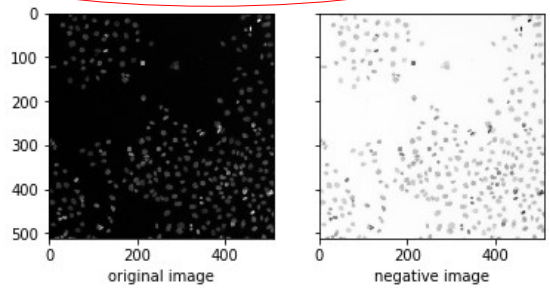
Run | Code

Manipulating images

Being numeric arrays, images can be manipulated by any of python's arithmetic operators. For example, we can produce a **negative image** by simply subtracting the image from its maximum value.

In [6]:
```
nuc_neg = nuc.max() - nuc
fig, axes = plt.subplots(ncols=2)
ax0, ax1 = axes.flatten()
ax0.imshow(nuc, cmap='gray')
ax0.set_xlabel("original image")
ax1.imshow(nuc_neg, cmap='gray')
ax1.set_xlabel("negative image")
plt.setp(ax1.get_yticklabels(), visible=False)
plt.show()
```

Live Code



Jupyter Notebook
What is it ?

localhost:8889/notebooks/image_analysis.ipynb

Les plus visités Getting Started

jupyter image_analysis Last Checkpoint: 19/09/2018 (autosaved)

Logout

File Edit View Insert Cell Kernel Widgets Help

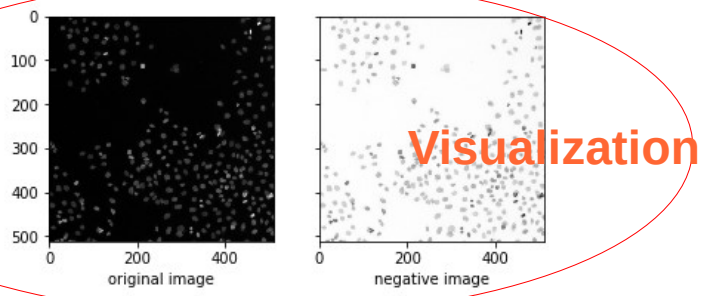
Trusted | Python 2

Run | Code

Manipulating images

Being numeric arrays, images can be manipulated by any of python's arithmetic operators. For example, we can produce a **negative image** by simply subtracting the image from its maximum value.

In [6]:
```
nuc_neg = nuc.max() - nuc
fig, axes = plt.subplots(ncols=2)
ax0, ax1 = axes.flatten()
ax0.imshow(nuc, cmap='gray')
ax0.set_xlabel("original image")
ax1.imshow(nuc_neg, cmap='gray')
ax1.set_xlabel("negative image")
plt.setp(ax1.get_yticklabels(), visible=False)
plt.show()
```



Visualization

Jupyter Notebook
What is it ?

localhost:8889/notebooks/image_analysis.ipynb

Les plus visités Getting Started

jupyter **image_analysis** Last Checkpoint: 19/09/2018 (autosaved)

Logout

File Edit View Insert Cell Kernel Widgets Help

Trusted | Python 2

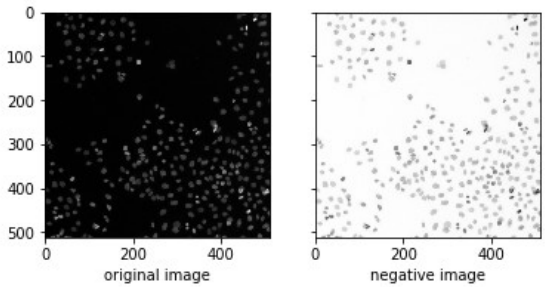
Run | Code

Manipulating images

Rich text for explanation

Being numeric arrays, images can be manipulated by any of python's arithmetic operators. For example, we can produce a **negative image** by simply subtracting the image from its maximum value.

In [6]:
```
nuc_neg = nuc.max() - nuc
fig, axes = plt.subplots(ncols=2)
ax0, ax1 = axes.flatten()
ax0.imshow(nuc, cmap='gray')
ax0.set_xlabel("original image")
ax1.imshow(nuc_neg, cmap='gray')
ax1.set_xlabel("negative image")
plt.setp(ax1.get_yticklabels(), visible=False)
plt.show()
```



Jupyter Notebook

jupyter Introduction (autosaved)

Logout

File Edit View Insert Cell Kernel Widgets Help

Not Trusted | Python 3

Run | Markdown

Edit Metadata

Equations

LaTeX

Equations can be used inline, so you can have short equations like $E = mc^2$ within the text. Longer equations should be separated from the text:

$$f(x) = \sum_{n=0}^{\infty} A_n \cos\left(\frac{n\pi x}{L}\right) + \sum_{n=1}^{\infty} B_n \sin\left(\frac{n\pi x}{L}\right)$$

Jupyter Notebook

jupyter tuto_image_analysis_pipeline (autosaved)

Logout

File Edit View Insert Cell Kernel Widgets Help

Not Trusted | Python 3

New Notebook
Open...
Make a Copy...
Rename...
Save and Checkpoint
Revert to Checkpoint
Print Preview
Download as
Trust Notebook
Close and Halt

Notebook (.ipynb)
Python (.py)
HTML (.html)
Reveal.js slides (.html)
Markdown (.md)
reST (.rst)
LaTeX (.tex)
PDF via LaTeX (.pdf)

Run Markdown

mport modules & packages

able to use declarations and definitions of an existing module, you should use the *import* clause.

```
ort module [as alias]
mple
```

/2018/mifobio/workshop/exercices

Use t ve to import additional modules or packages for the future use in this pipeline.

In [2]:
```
#the plotlib.pyplot as plt
import matplotlib.pyplot as plt

#the numpy package for scientific computing is also a great tool for manipulating arrays
import numpy

#the image processing package skimage and ndimage from scipy
import skimage
import scipy.ndimage
```

Jupyter Notebook

Jupyter notebook is a web application that allows you to run live code, embed visualization and explanatory text all in one place

EXERCICES

01-Introduction.ipynb

OMERO

https://www.openmicroscopy.org

Rechercher

Les plus visités Getting Started

OMERO

OMERO is client-server software for managing, visualizing and analyzing microscopy images and associated metadata.

OMERO for

Scientists

Your microscopy images are securely stored but shareable and available from anywhere you have internet access.

OMERO for

Developers

Join the OME community and extend OMERO's functionality to suit your individual needs.

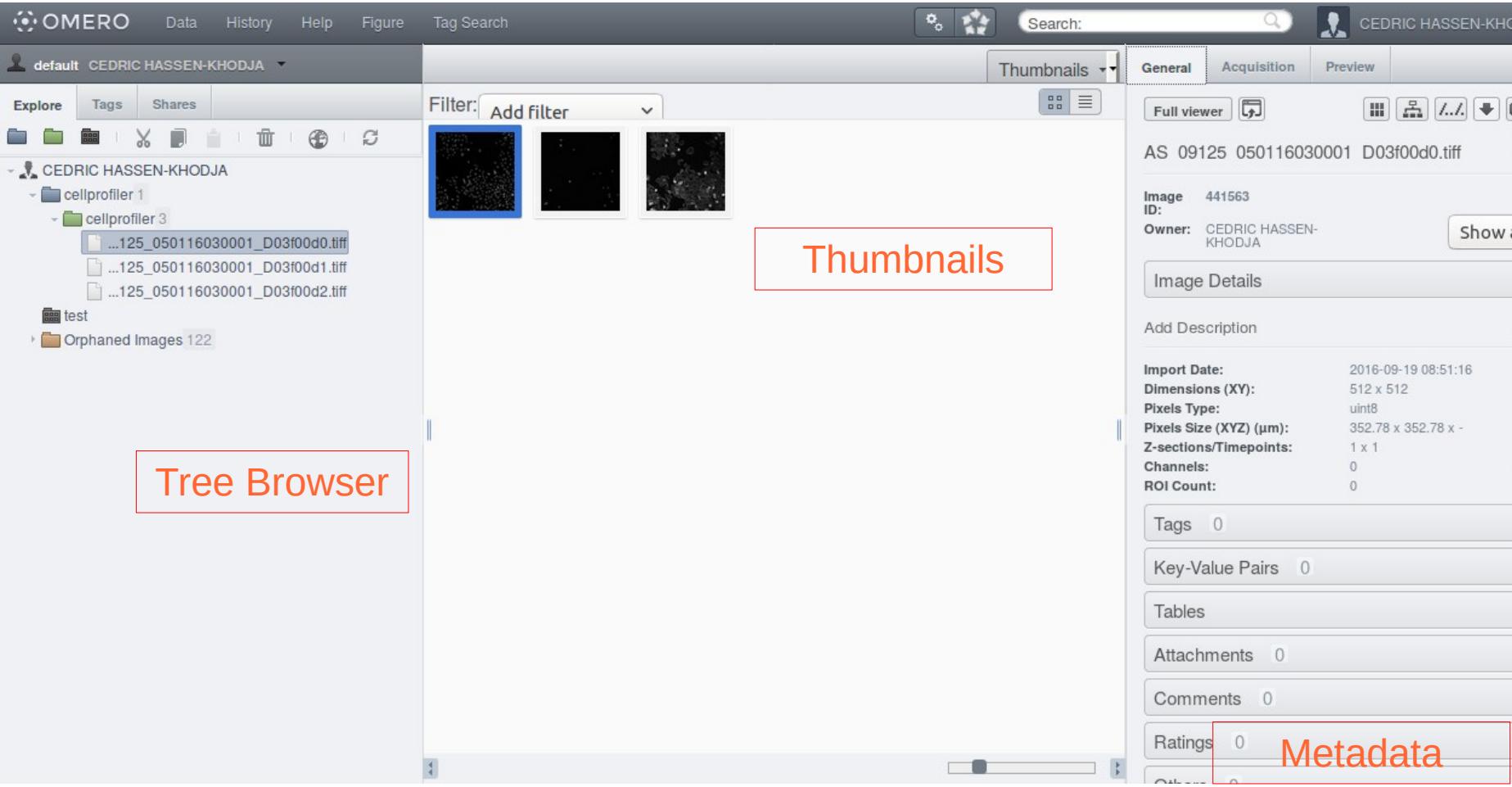
OMERO for

Your Institution

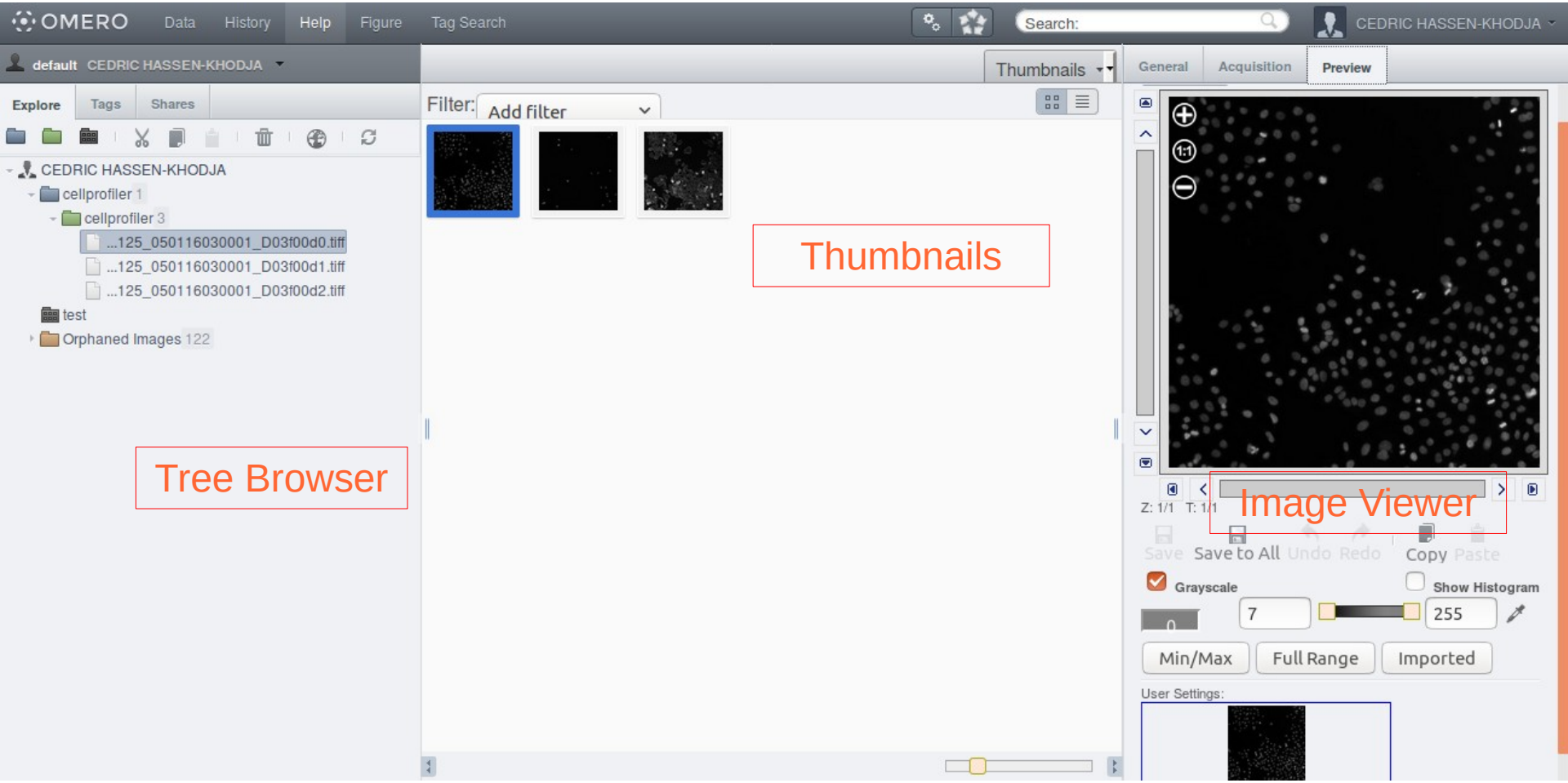
OMERO securely stores image data and enables all of your users to manage the data from the same platform.

source: https://www.openmicroscopy.org/

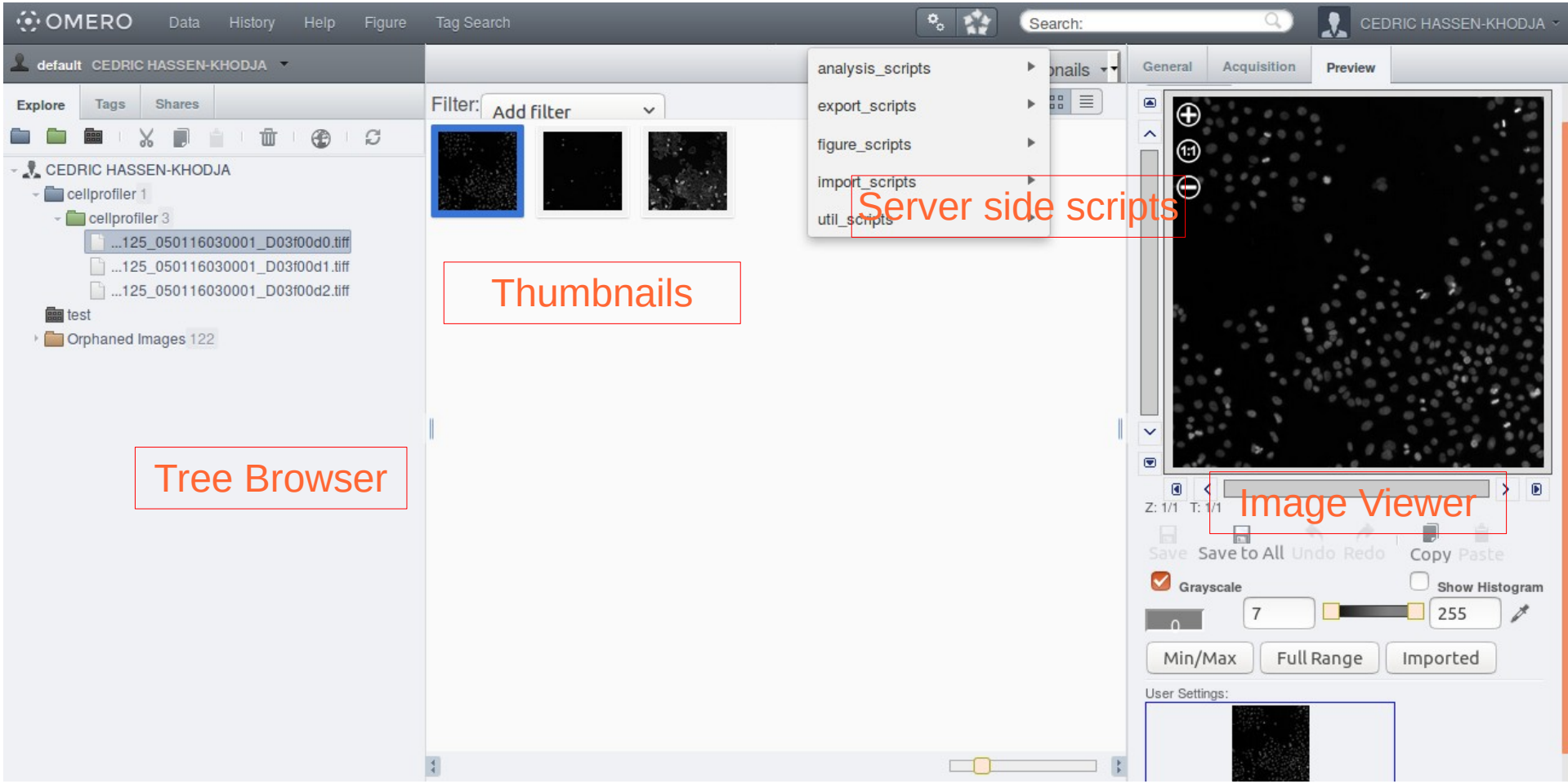
OMERO
What is it ?



OMERO
What is it ?



OMERO
What is it ?



Python Blitz API

https://docs.openmicroscopy.org/omero/5.4.8/developers/Python.html

source: https://www.openmicroscopy.org/omero/features/analyze/

Python Blitz API

Connect to omero server:

- import omero.gateway package
- Use the wrapper BlitzGateway:

```
class omero.gateway._BlitzGateway(username=None, passwd=None,
                                  client_obj=None, group=None,
                                  clone=False, try_super=False,
                                  host=None, port=None,
                                  extra_config=None, secure=False,
                                  anonymous=True, useragent=None,
                                  userip=None)
```

Get user information:

- Use the getUser() function to return current experimenter
- Use getName() function to return the experimenter name
- Use getFullName() function to return the full name of this experimenter
- getGroupsMemberOf() function return current users groups
- getGroupFromContext() function return your current group

source: https://www.openmicroscopy.org/omero/features/analyze/

Python Blitz API

Get user information:

- Use the getUser() function to return current experimenter
- Use getName() function to return the experimenter name
- Use getFullName() function to return the full name of this experimenter
- getGroupsMemberOf() function return current users groups
- getGroupFromContext() function return your current group

Get Project / Dataset information:

- Use getObjects() function to retrieve Objects by type, e.g. “Image” returns generator of appropriate BlitzObjectWrapper type, e.g. ImageWrapper

```
getObjects(obj_type, ids=None, params=None, attributes=None,
           respect_order=False, opts=None)
```

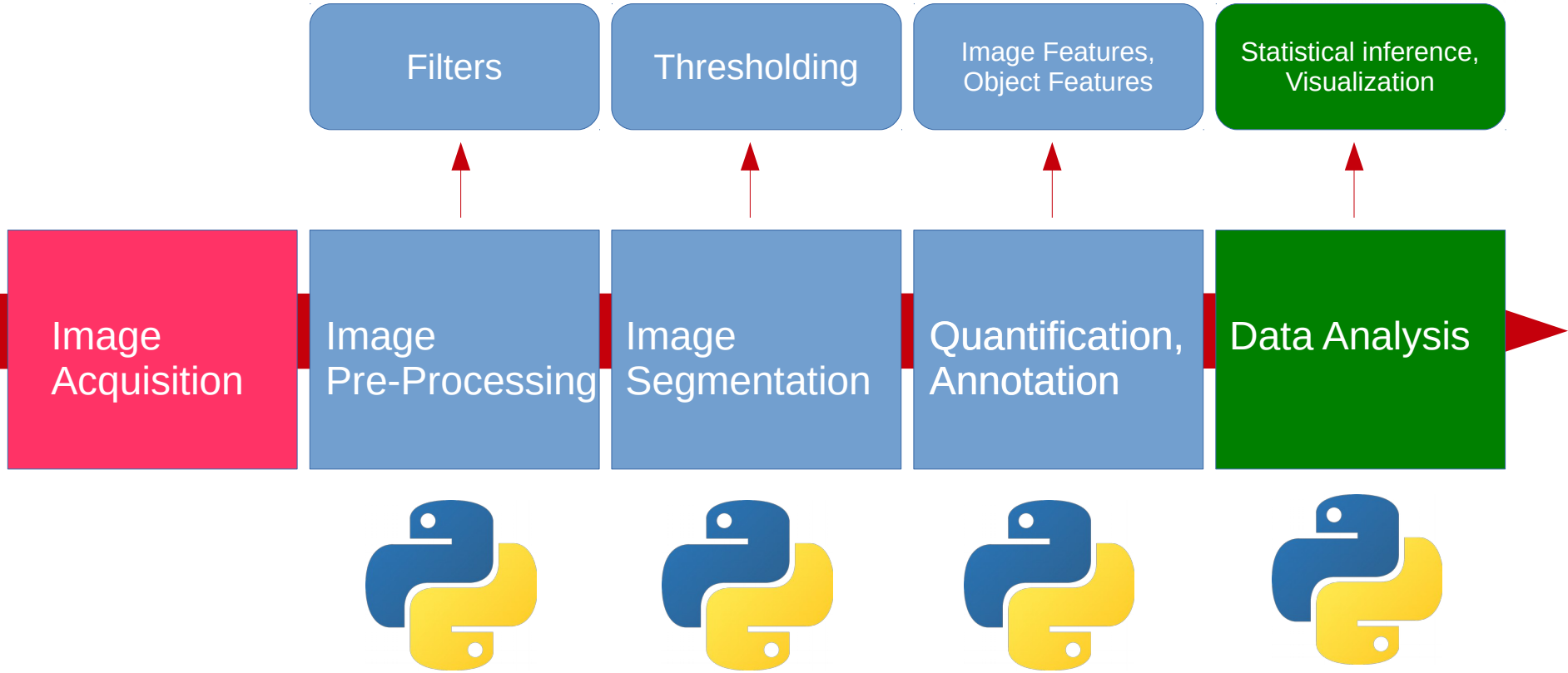
- listChildren() lists available child objects

```
listChildren(ns=None, val=None, params=None)
```

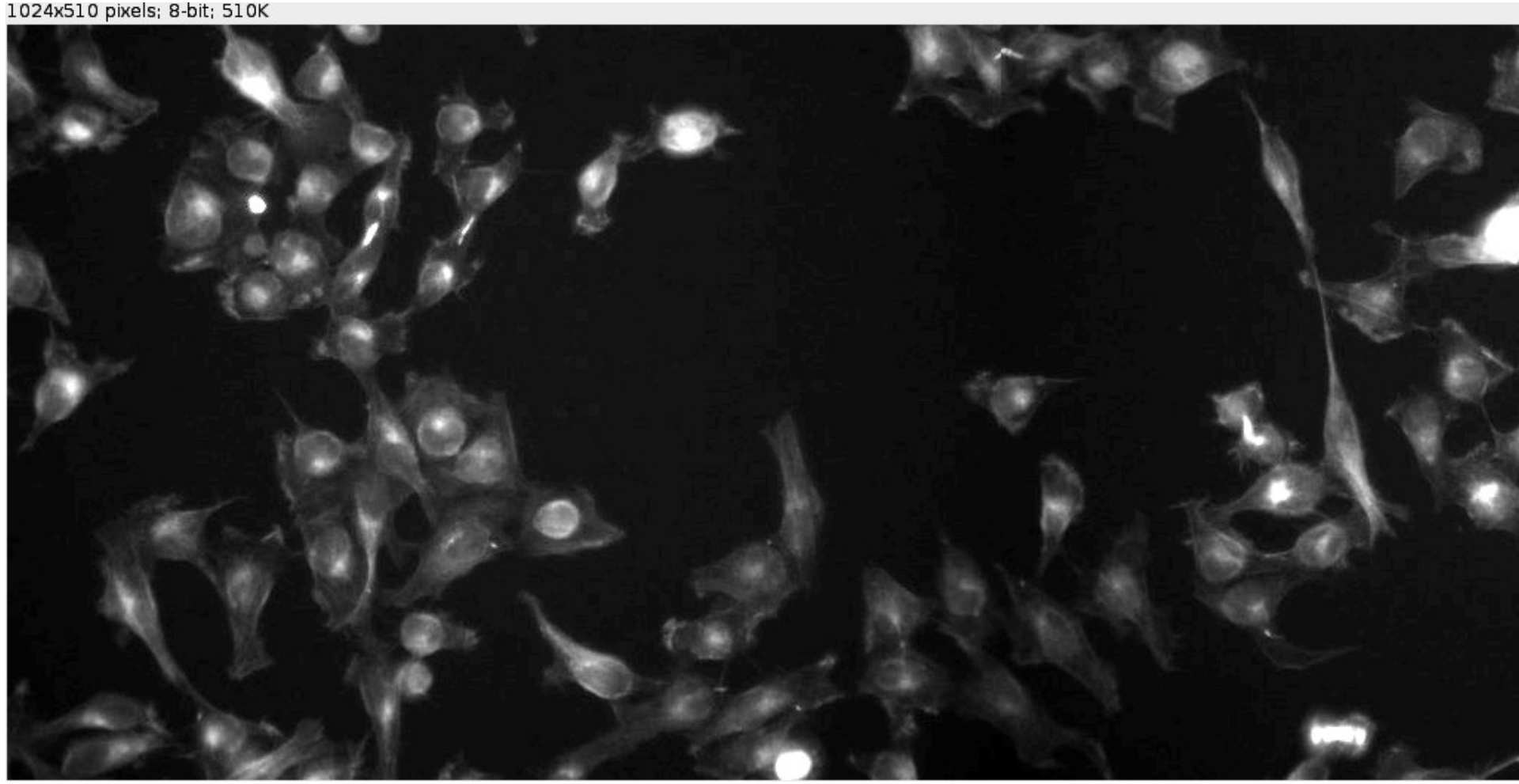
source: https://www.openmicroscopy.org/omero/features/analyze/

02-Introduction_Omero_PartI.ipynb

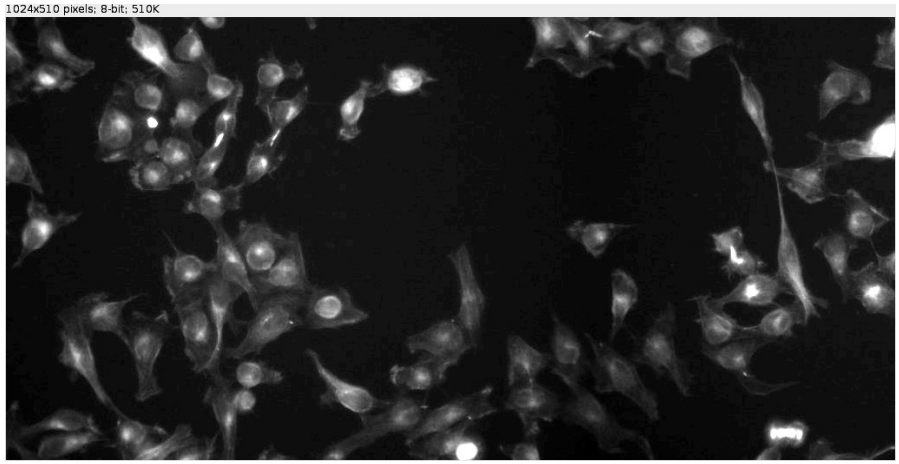
Python for image analysis



Digital Images



Digital Images

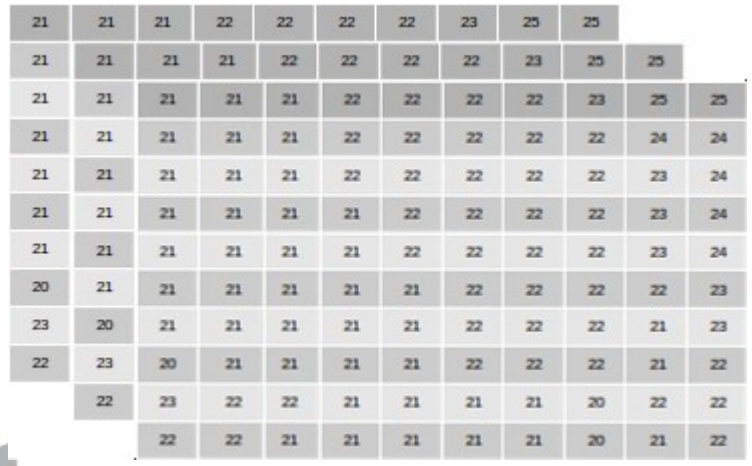


21	21	21	22	22	22	22	23	25	25
21	21	21	22	22	22	22	22	24	24
21	21	21	22	22	22	22	22	23	24
21	21	21	21	22	22	22	22	23	24
21	21	21	21	22	22	22	22	23	24
21	21	21	21	21	22	22	22	22	23
21	21	21	21	21	22	22	22	21	23
20	21	21	21	21	22	22	22	21	22
23	22	22	21	21	21	21	20	22	22
22	22	21	21	21	21	21	20	21	22

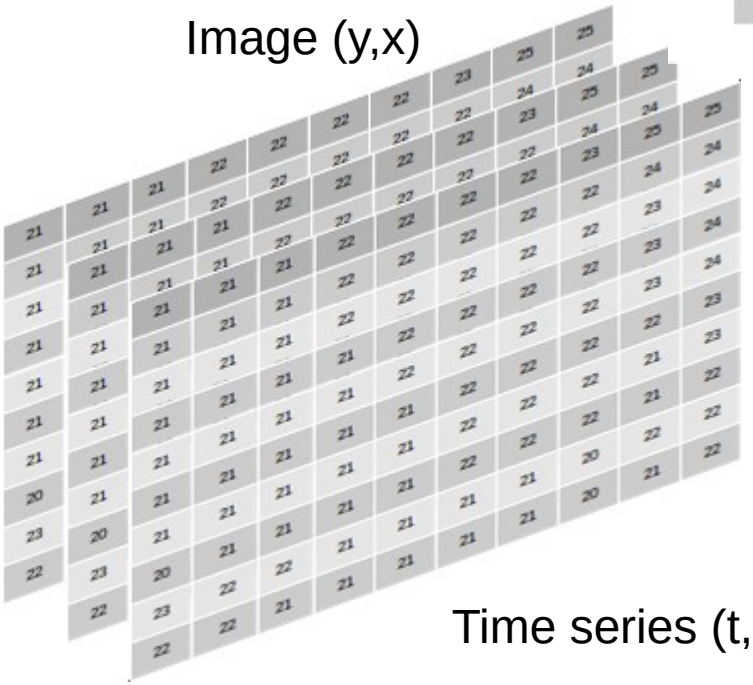
Dimensions

21	21	21	22	22	22	22	23	25	25
21	21	21	22	22	22	22	22	24	24
21	21	21	22	22	22	22	22	23	24
21	21	21	21	22	22	22	22	23	24
21	21	21	21	22	22	22	22	23	24
21	21	21	21	21	22	22	22	22	23
21	21	21	21	21	22	22	22	21	23
20	21	21	21	21	22	22	22	21	22
23	22	22	21	21	21	21	20	22	22
22	22	21	21	21	21	21	20	21	22

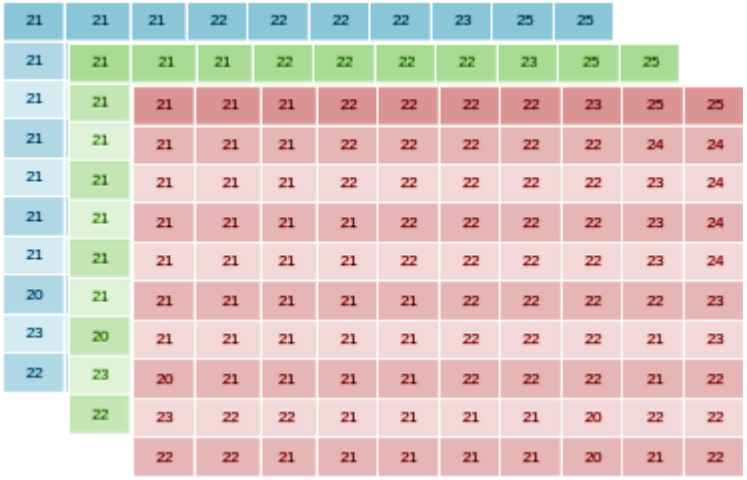
Image (y,x)



Stack (z,y,x)



Time series (t,y,x)



Channel (c,y,x)

Data types & unexpected output

uint8	Unsigned integer (0 to 255)
uint16	Unsigned integer (0 to 65535)
float32	Single precision float: sign bit, 8 bits exponent, 23 bits mantissa

in python 2

20 / 3 = 6

```
from __future__ import division
```

20 / 3 = 6.666666666666667

20 // 3 = 6

source: https://docs.scipy.org/doc/numpy-1.13.0/user/basics.types.html

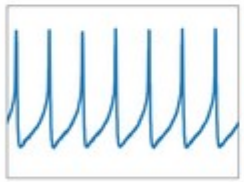
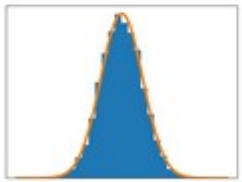
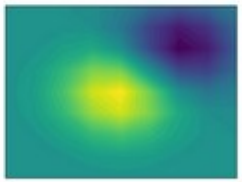
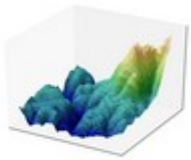
Visualization

matplotlib

Version 3.0.0

home | examples | tutorials | API | docs »

Matplotlib is a Python 2D plotting library which produces publication quality figures in a variety of hardcopy formats and interactive environments across platforms. Matplotlib can be used in Python scripts, the Python and IPython shells, the Jupyter notebook, web application servers, and four graphical user interface toolkits.

Matplotlib tries to make easy things easy and hard things possible. You can generate plots, histograms, power spectra, bar charts, errorcharts, scatterplots, etc., with just a few lines of code. For examples, see the sample plots and thumbnail gallery.

For simple plotting the `pyplot` module provides a MATLAB-like interface, particularly when combined with IPython. For the power user, you have full control of line styles, font properties, axes properties, etc, via an object oriented interface or via a set of functions familiar to MATLAB users.

source: https://matplotlib.org/

Visualization

matplotlib.pyplot ¶

`matplotlib.pyplot` is a state-based interface to matplotlib. It provides a MATLAB-like way of plotting.

pyplot is mainly intended for interactive plots and simple cases of programmatic plot generation:

`figure`([num, figsize, dpi, facecolor, ...])	Create a new figure.
`imshow`(X[, cmap, norm, aspect, ...])	Display an image, i.e.
`show`(*args, **kw)	Display a figure.

source: https://matplotlib.org/

Omero Blitz Gateway

Get images metadata

- image = conn.getObject("Image", imageId)
  - image.getSizeX()
  - image.getSizeY()
  - image.getSizeZ()
  - image.getSizeC()
  - Image.getSizeT()

Display images

- rendered_image = image.renderImage(z, t, compression=0.9)
  - get a compressed and rendered version of the slice z and frame t of the image
- rendered_image.show()

source: https://www.openmicroscopy.org/omero/features/analyze/

EXERCICES

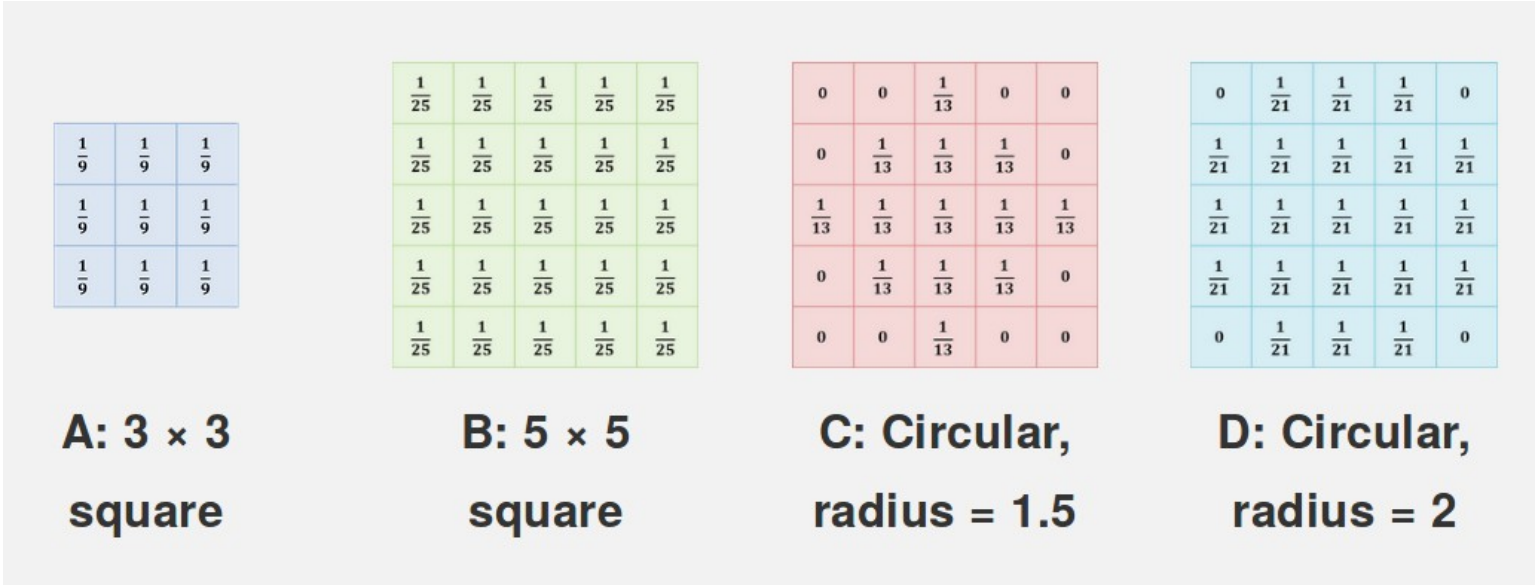
Import_Handle_Local_Image_Data.ipynb
Introduction_Omero_PartII.ipynb

Preprocessing - Filters

Image Filtering is used to:

- Remove noise
- Sharpen contrast
- Highlight contours
- Detect edges

Preprocessing - Filters



A kernel resembles another (usually small and rectangular) image in which each pixel is known as a filter coefficient and these correspond to the the weights used for scaling

Preprocessing - Filters

skimage.morphology.square (width[, dtype])	Generates a flat, square-shaped structuring element.
skimage.morphology.rectangle (width, height)	Generates a flat, rectangular-shaped structuring element.
skimage.morphology.diamond (radius[, dtype])	Generates a flat, diamond-shaped structuring element.
skimage.morphology.disk (radius[, dtype])	Generates a flat, disk-shaped structuring element.
skimage.morphology.cube (width[, dtype])	Generates a cube-shaped structuring element.
skimage.morphology.octahedron (radius[, dtype])	Generates a octahedron-shaped structuring element.
skimage.morphology.ball (radius[, dtype])	Generates a ball-shaped structuring element.
skimage.morphology.octagon (m, n[, dtype])	Generates an octagon shaped structuring element.
skimage.morphology.star (a[, dtype])	Generates a star shaped structuring element.

You can create your own filter of convolution

source: http://scikit-image.org/docs/dev/api/skimage.morphology.html

Preprocessing - Filters

generic_laplace(input, derivative2[, ...])	N-dimensional Laplace filter using a provided second derivative function :Parameters: **input** : array_like Input array to filter.
laplace(input[, output, mode, cval])	N-dimensional Laplace filter based on approximate second derivatives.
maximum_filter(input[, size, footprint, ...])	Calculates a multi-dimensional maximum filter.
maximum_filter1d(input, size[, axis, ...])	Calculate a one-dimensional maximum filter along the given axis.
median_filter(input[, size, footprint, ...])	Calculates a multidimensional median filter.
minimum_filter(input[, size, footprint, ...])	Calculates a multi-dimensional minimum filter.
minimum_filter1d(input, size[, axis, ...])	Calculate a one-dimensional minimum filter along the given axis.
percentile_filter(input, percentile[, size, ...])	Calculates a multi-dimensional percentile filter.
prewitt(input[, axis, output, mode, cval])	Calculate a Prewitt filter.
rank_filter(input, rank[, size, footprint, ...])	Calculates a multi-dimensional rank filter.
sobel(input[, axis, output, mode, cval])	Calculate a Sobel filter.
uniform_filter(input[, size, output, mode, ...])	Multi-dimensional uniform filter.
uniform_filter1d(input, size[, axis, ...])	Calculate a one-dimensional uniform filter along the given axis.

Segmentation - Thresholding

Thresholding is used to:

- Identify interesting objects
- Binarizing an image

How ?:

- Global vs Adaptive

Segmentation - Thresholding

Otsu's Thresholding Method (1979)

- Based on a very simple idea: Find the threshold that *minimizes the weighted within-class variance.*
- This turns out to be the same as *maximizing the between-class variance.*
- Operates directly on the gray level histogram [*e.g.* 256 numbers, P(i)], so it's fast (once the histogram is computed).
- I've used it with considerable success in "murky" situations.

Nobuyuki Otsu (1979). "A threshold selection method from ----- histograms". IEEE Trans. Sys., Man., Cyber. 9 (1): 62–66.

Segmentation - Thresholding

threshold_otsu

skimage.filters.**threshold_otsu**(*image*, *nbins=256*) [source]

Return threshold value based on Otsu's method.

Parameters:

image : (N, M) ndarray

Grayscale input image.

nbins : int, optional

Number of bins used to calculate histogram. This value is ignored for integer arrays.

Returns:

threshold : float

Upper threshold value. All pixels with an intensity higher than this value are assumed to be foreground.

Raises:

ValueError

If image only contains a single grayscale value.

source: http://scikit-image.org/docs/dev/api/skimage.filters.html#skimage.filters.threshold_otsu

PostProcessing – Morphological operation

Erosion will make objects smaller

Dilation will make objects bigger

Goal: clean up binary image

Opening consists of an erosion followed by a dilation

Closing is the opposite of opening, i.e. a dilation followed by an erosion

Original | Erosion | Dilation | Opening | Closing

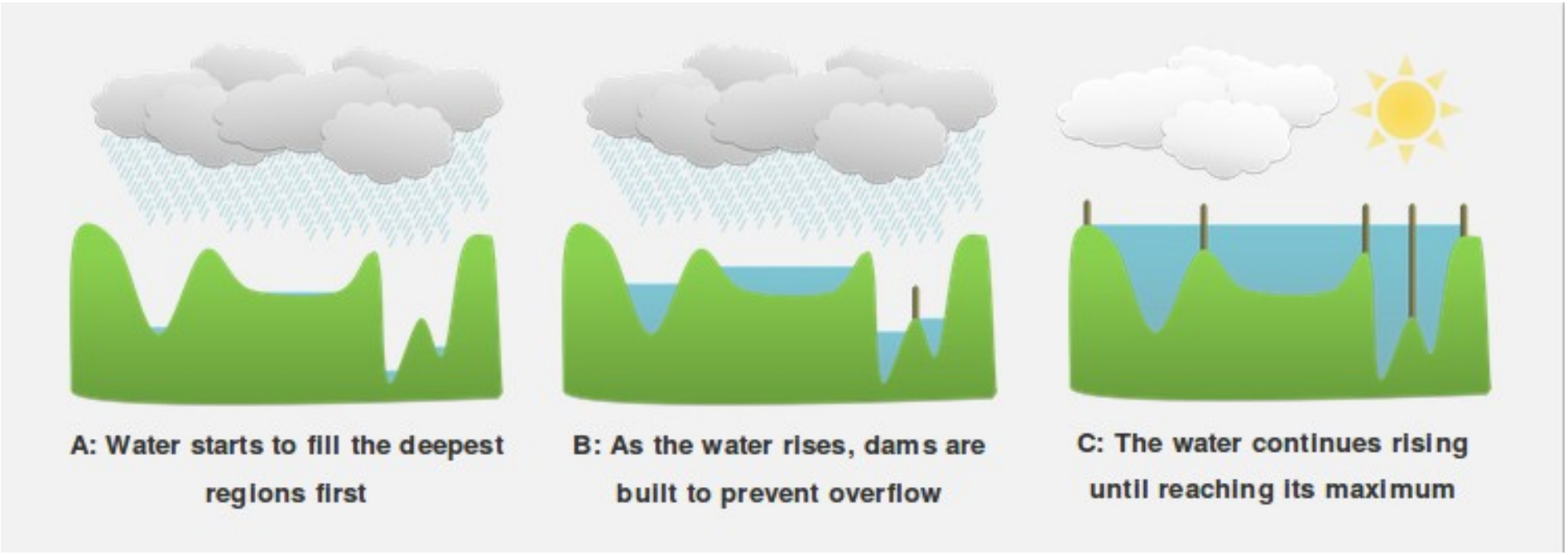
skimage.morphology.erosion (image[, selem, ...])	Return greyscale morphological erosion of an image.
skimage.morphology.dilation (image[, selem, ...])	Return greyscale morphological dilation of an image.
skimage.morphology.opening (image[, selem, out])	Return greyscale morphological opening of an image.
skimage.morphology.closing (image[, selem, out])	Return greyscale morphological closing of an image.

source: http://scikit-image.org/docs/dev/api/skimage.morphology.html

EXERCICES

nuclei_analysis.ipynb

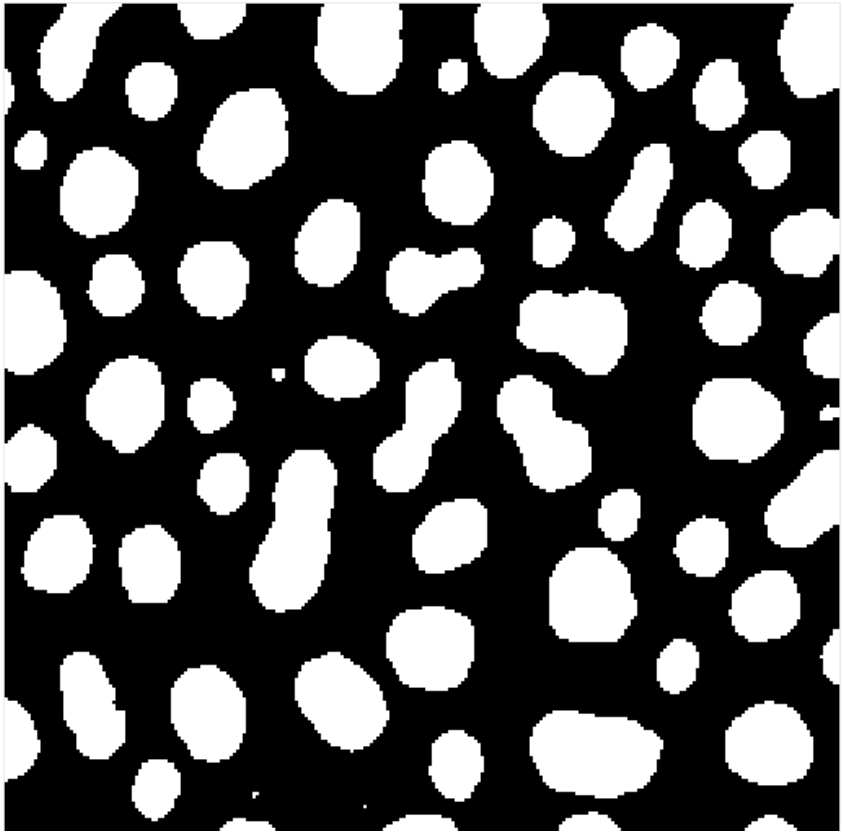
Segmentation – Watershed



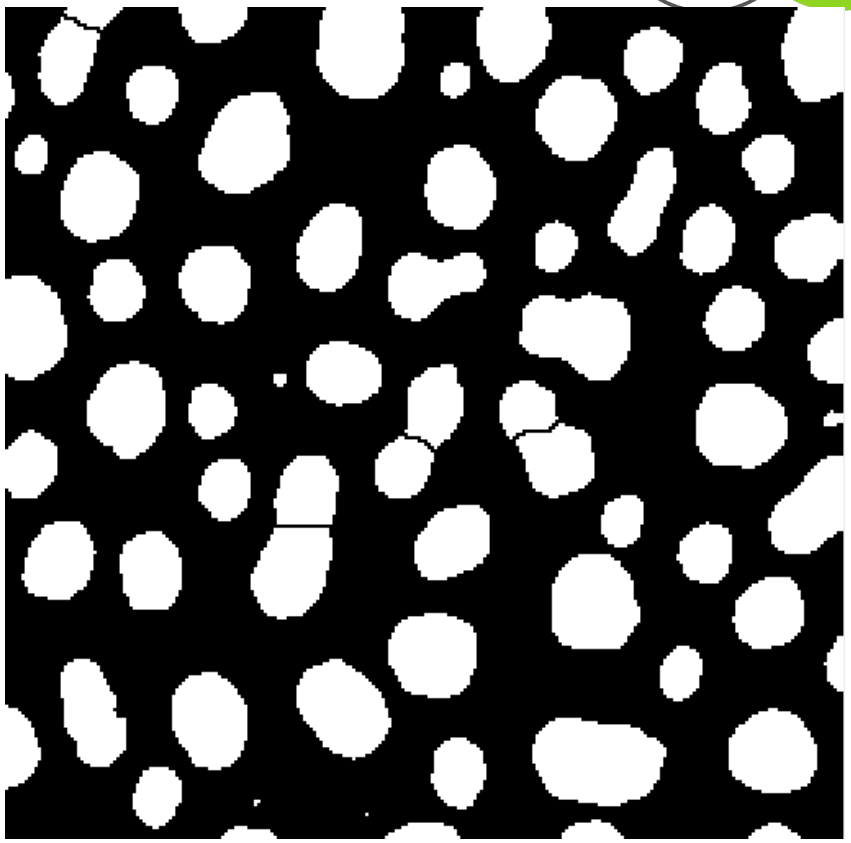
source: Analyzing fluorescence microscopy images with ImageJ. / Bankhead, Peter

Segmentation – Watershed

`skimage.morphology.watershed` (image, markers) | Find watershed basins in image flooded from given markers.

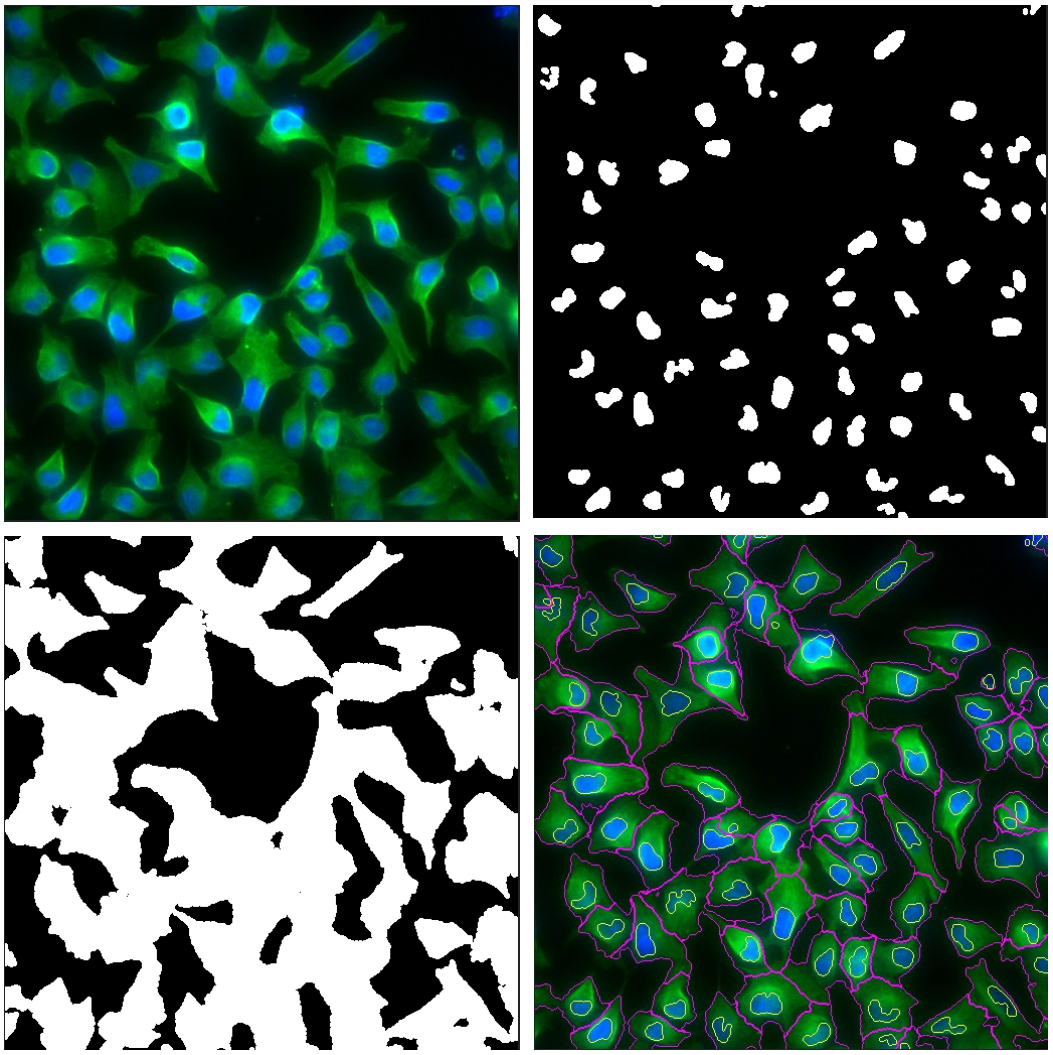


Original



Watershed segmentation
on original image

Segmentation – Voronoi based segmentation on image manifolds



T. Jones, A. Carpenter and P. Golland, "Voronoi-Based Segmentation of Cells on Image Manifolds", CVBIA05 (535-543), 2005

EXERCICES

image_analysis.ipynb

Quantification

- Extract features on interested objects
  - area, perimeter, circularity, intensities...
- Statistics
  - Descriptive
  - Regression models
  - Classification

EXERCICES

image_analysis.ipynb

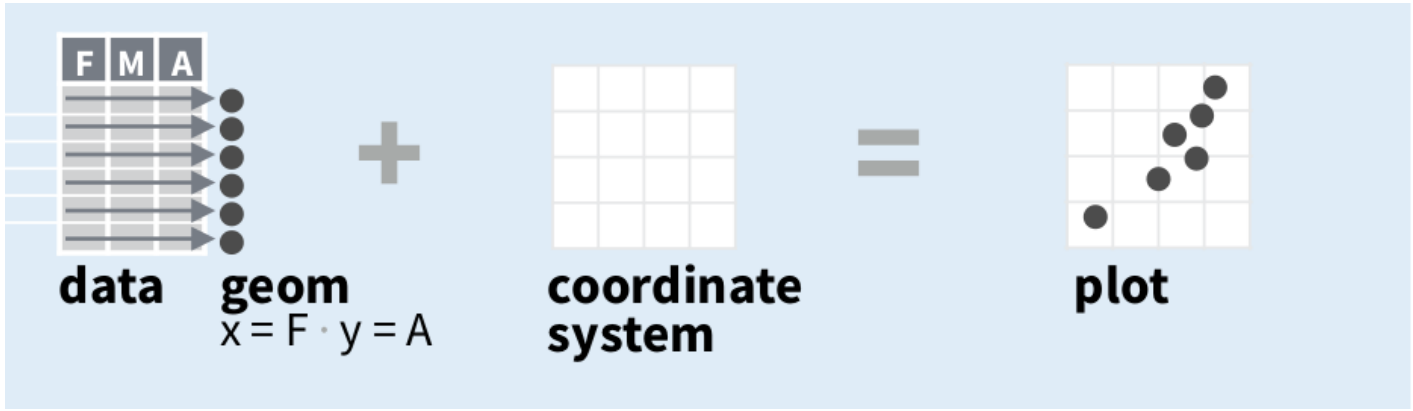
(Exercises 6, 7 & 8)

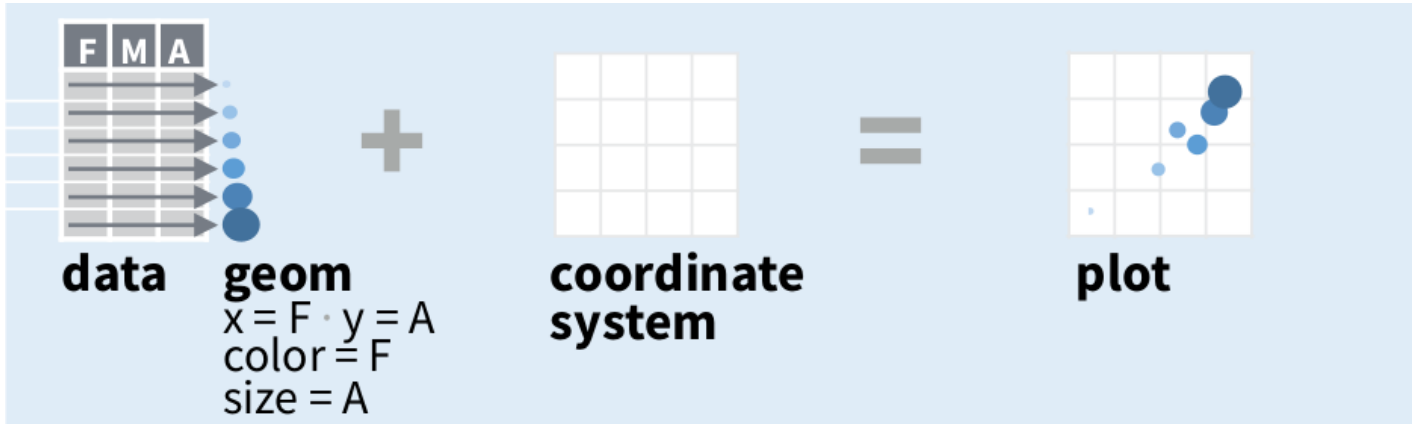
Data analysis – Descriptive statistics in R

- Descriptive statistics offers:
  - Graphics
    - Qualitative var → bar chart / pie chart
    - Quantitative var → bar chart / histogram
  - Indicators
    - Mean
    - Median
    - Quantiles
    - Standard deviation & variance

R code: summary(result)
sd(result$var)
var(result$var)

Data analysis – Visualization in R





R code: ggplot(mpg, aes(hwy, cty)) +
geom_point(aes(color=cyl)) +
geom_smooth(method ="lm") +coord_cartesian() +
scale_color_gradient() +theme_bw()

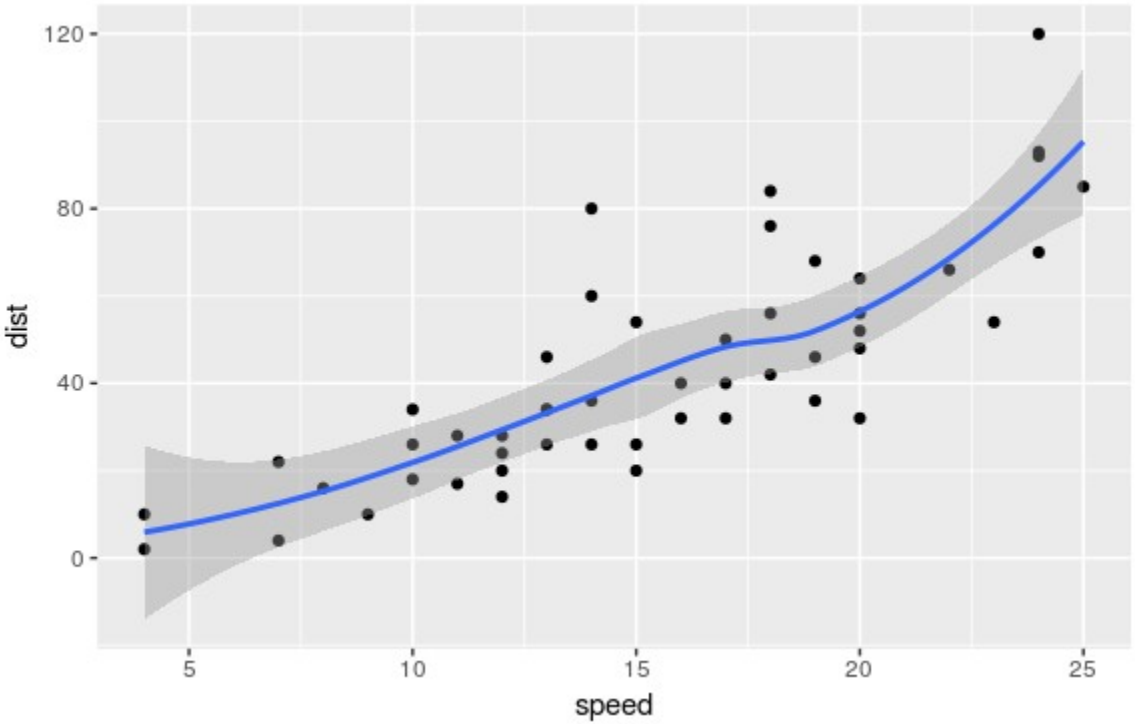
source: https://ggplot2.tidyverse.org/

Data analysis – Regression models in R

$$y = b0 + b1*x + e$$

- b0 -> intercept of the regression line
- b1 -> slope of the regression line
- e -> residuals errors

Data analysis – Regression models in R




```
dist = b0 + b1 * speed
```

```
model <- lm(dist ~ speed, data = cars)
```

```
Call:
lm(formula = dist ~ speed, data = cars)
Coefficients:
(Intercept)        speed
    -17.579        3.932
```

Interpretation

- dist = −17.579 + 3.932∗speed
- for a speed equal to 20 mph, we can expect an increase of 78.64 ft.
- That is, dist = −17.579 + 3.932∗20 = 61.061 ft.

Before using this formula to predict future dist, you should make sure that this model is statistically significant, that is:

- there is a statistically significant relationship between the predictor and the outcome variables
- the model that we built fits very well the data in our hand.

Data analysis – Regression models in R

```
summary(model)

Call:
lm(formula = dist ~ speed, data = cars)
Residuals:
    Min      1Q  Median      3Q     Max
-29.069  -9.525  -2.272   9.215  43.201
Coefficients:
            Estimate Std. Error t value Pr(>|t|)
(Intercept) -17.5791     6.7584  -2.601   0.0123 *
speed         3.9324     0.4155   9.464 1.49e-12 ***
---
Signif. codes:
0 *** 0.001 ** 0.01 * 0.05 . 0.1 1

Residual standard error: 15.38 on 48 degrees of freedom
Multiple R-squared:  0.6511,    Adjusted R-squared:  0.6438
F-statistic: 89.57 on 1 and 48 DF,  p-value: 1.49e-12
```

Interpretation

A statistically significant coefficient indicates that there is an association between the predictor (x) and the outcome (y) variable. In our example, both the p-values for the intercept and the predictor variable are significant, so we can reject the null hypothesis and accept the alternative hypothesis, which means that there is a significant association between the predictor and the outcome variables.

Data analysis – Regression models in R

```
Call:
lm(formula = dist ~ speed, data = cars)
Residuals:
    Min      1Q  Median      3Q     Max
-29.069  -9.525  -2.272   9.215  43.201
Coefficients:
            Estimate Std. Error t value Pr(>|t|)
(Intercept) -17.5791     6.7584  -2.601   0.0123 *
speed         3.9324     0.4155   9.464 1.49e-12 ***
---
Signif. codes:
0 *** 0.001 ** 0.01 * 0.05 . 0.1 1
```

```
Residual standard error: 15.38 on 48 degrees of freedom
Multiple R-squared:  0.6511,    Adjusted R-squared:  0.6438
F-statistic: 89.57 on 1 and 48 DF,  p-value: 1.49e-12
```

Interpretation

- In our example, **RSE = 15.38**, meaning that the observed dist values deviate from the true regression line by approximately 15.38 units in average.
- we can calculate the percentage error = 35.78312%
- The R2 measures, how well the model fits the data. For a simple linear regression, R2 is the square of the Pearson correlation coefficient.

EXERCICES

statistical_analysis.ipynb