



# Data Visualization & More Tools Exercises

June 28, 2019

Licence: CC-by-sa

Before you start:

- create a suitable directory for this exercise
- download the lecture material from [http://www.physik.uzh.ch/~python/python/lecture\\_visual/](http://www.physik.uzh.ch/~python/python/lecture_visual/) for reference.

## Exercise 1: Replicating a plot

Your predecessor has created the graph shown below for some publication. Since then, some things have changed and you need to reproduce the plot with new data. However, the code that produced it was lost (it only ever existed in an interactive `ipython` session, the computer on which it was run has long been replaced and nobody bothered with using `git`). In addition, the person who created the plot is no longer in academia and cannot be reached.

From looking at the plot you notice multiple features:

- There are two subplots (`plt.subplot` or `plt.subplots?`). The lower one is smaller (`gridspec_kw?`) and the two subplots share an  $x$ -axis.
- The top subplot shows both the PDF and a normalized histogram of  $N = 1000$  randomly generated values of (presumably) a normal distribution (`scipy.stats.norm`). It has a meaningful title.
- The top plot also contains the corresponding CDF on a separate  $y$ -axis (`ax.twinx`). It is a different color than the other plots.
- The second plot contains the residual between the PDF and the histogram, using `plt.step` in order to match the binning of the histogram.
- The overall plot style is not the default one. Hopefully a preset style was used. The top plot contains a grid matching the right  $y$ -axis and the bottom plot has a grid matching both the  $x$ - and  $y$ -axis.
- In addition, the number of events is added to the plot (`plt.text`) as well as a legend in the upper left corner, which even has a title. It only contains the label for the histogram and the PDF.
- The axis are all properly labeled. The  $x$ -axis even has a fancy  $\text{\LaTeX}$  label.

*Try to replicate the plot as closely as possible.*

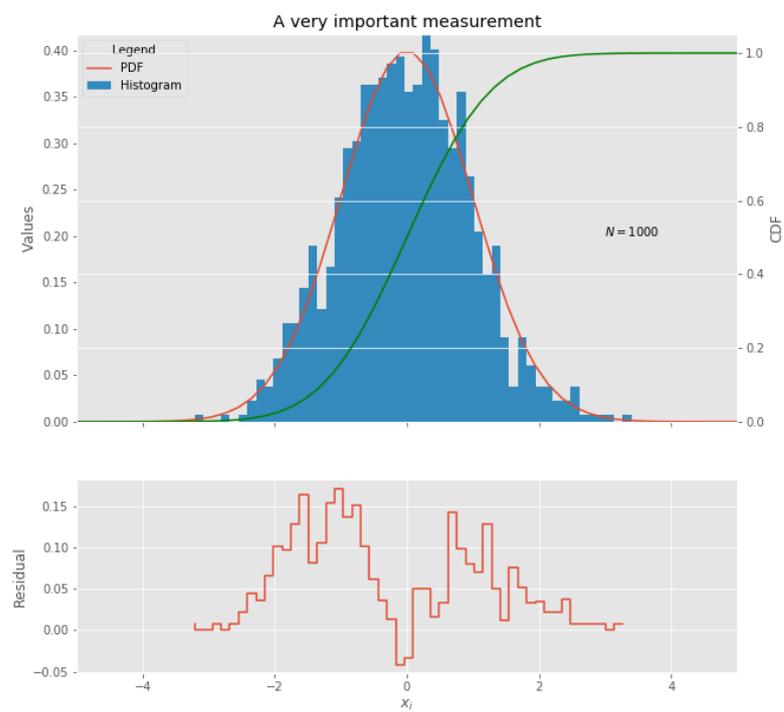


Figure 1: Plot to be replicated in Exercise 1.

## Exercise 2: Plotting geospatial data

Use `geopandas` to create an interesting visualization of data on a map.

- Since this package is not installed on the machines, you will have to install it first. Activate the virtual environment with `source ~/school_venv/bin/activate`. Install the package via `pip install geopandas`. If you want to use it in the notebook you need to recreate the kernel with `python3 -m ipykernel install --user --name=school_venv`.
- You can find a shapefile for the boroughs of London at the link <https://data.london.gov.uk/dataset/statistical-gis-boundary-files-london>. Download the file `statistical-gis-boundaries-london.zip` and load the file `statistical-gis-boundaries-london/ESRI/London_Borough_Excluding_MHW.shp`.
- For a lot of statistical data per borough, visit <https://data.london.gov.uk/dataset/london-borough-profiles> and download the `london-borough-profiles.csv` file.
- Think about how to best combine the two dataframes.
- Use the `plot` method of the `geopandas.DataFrame` to plot some statistics per borough. An example of what this can look like can be found in Fig. 2.

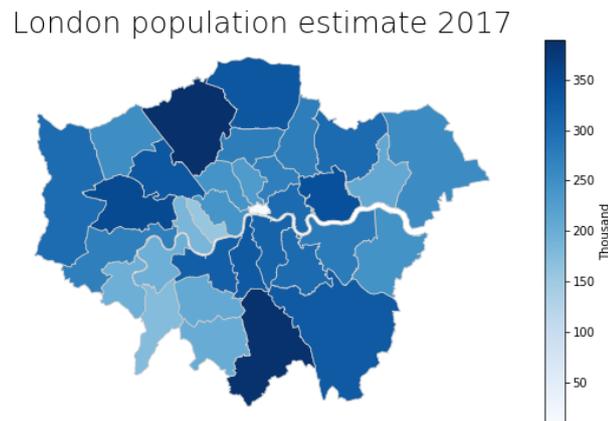


Figure 2: Example plot for Exercise 3.

## Exercise 3: Web scraping

Use the `requests` library and `bs4.BeautifulSoup` to parse our homepage for the material of each lecture.

- You can use the developer tools of your browser to figure out the names and attributes of elements.
- CSS selectors can be used with `soup.select`, or you can directly operate on the tags with `soup.find/soup.find_all`.
- Write a `download` function that automatically downloads the material to a specified directory. For this use `response.content`, instead of `response.text`, and `open(file_name, "wb")` in order to directly write the binary content to a file. Make sure to create the directory if it does not exist and that you deal with file names which would be illegal (for example file names containing `"/"`).