University of Sulaimani
College of Science
Computer Department
4th Stage

Data Science Management

Visualization and Matplotlib in Python

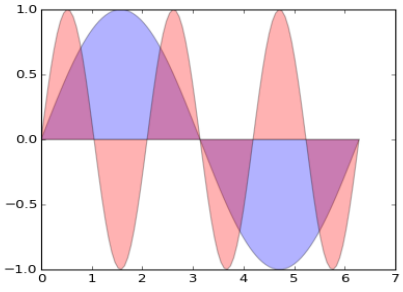
Class 6

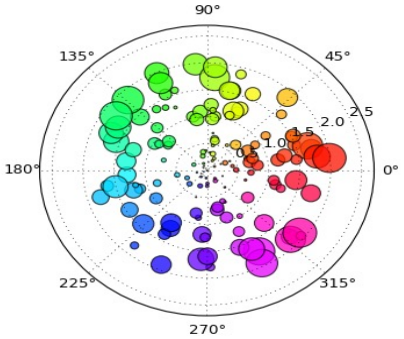
Theoretical and practical lectures

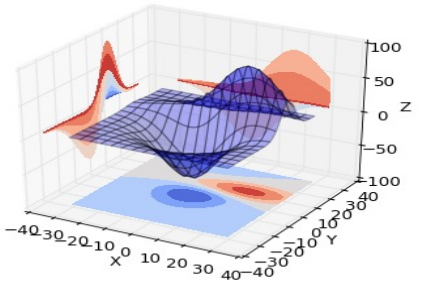
Assist. Prof. Dr. Miran Taha Abdullah
2025-2026

What is data visualization?

- Data visualization is the graphical representation of information and data.
  - Can be achieved using visual elements like figures, charts, graphs, maps, and more.
- Data visualization tools provide a way to present these figures and graphs.
- Often, it is essential to analyze massive amounts of information and make data-driven decisions.
  - converting complex data into an easy to understand representation.







Matplotlib

- **Matplotlib** is one of the most powerful tools for data visualization in Python.
- **Matplotlib** is an incredibly powerful (and beautiful!) 2-D plotting library.
  - It is easy to use and provides a huge number of examples for tackling unique problems

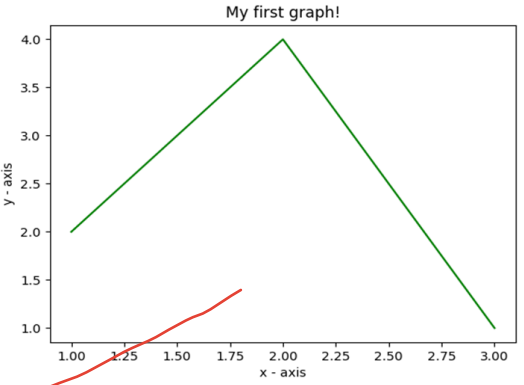
Matplotlib

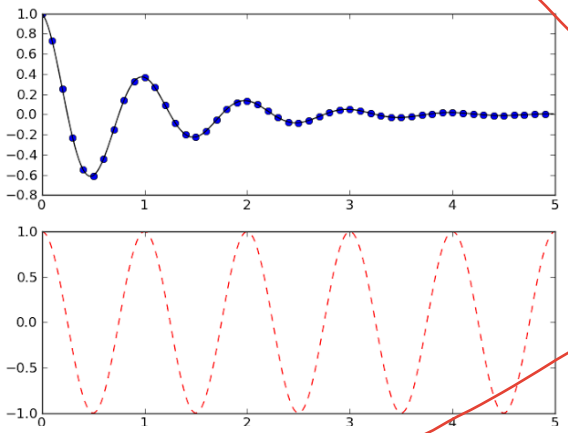
- Strives to emulate MATLAB
  - `matplotlib.pyplot` is a collection of command style functions that make `matplotlib` work like **MATLAB**.
- Each `pyplot` function makes some change to the figure:
  - e.g.,
    - creates a figure,
    - creates a plotting area in the figure,
    - plots some lines in the plotting area,
    - decorates the plot with labels, etc.
- **Note** that **various states** are preserved **across function calls**

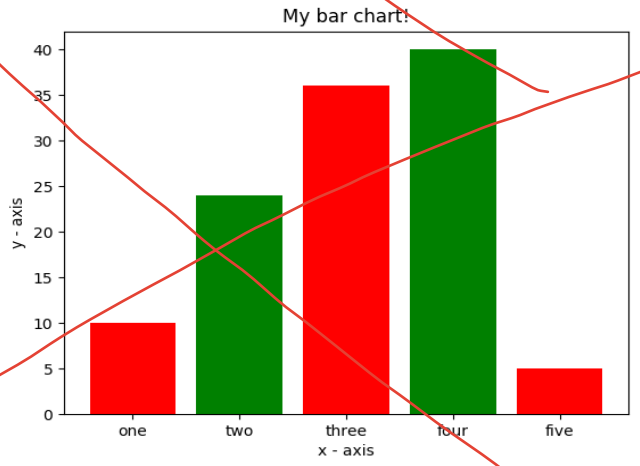
- Whenever you plot with `matplotlib`, the two main code lines should be considered:
  - Type of graph
    - this is where you define a bar chart, line chart, etc.
  - Show the graph
    - this is to display the graph

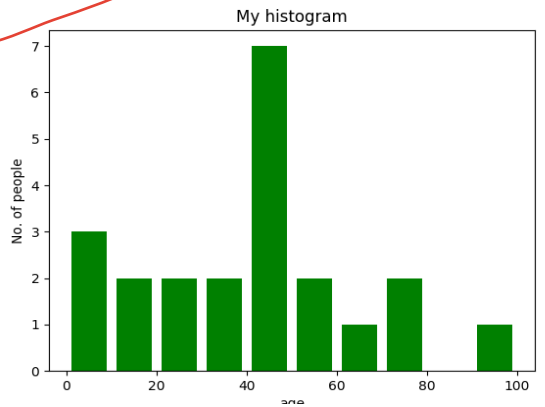
E.g. Matplotlib

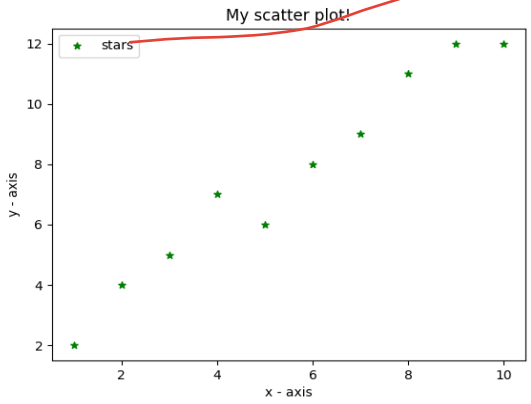
- **`Matplotlib`** allows you to make easy things
- You can generate **plots**, **histograms**, **power spectra**, **bar charts**, **errorcharts**, **scatterplots**, etc., with just a few lines of code.

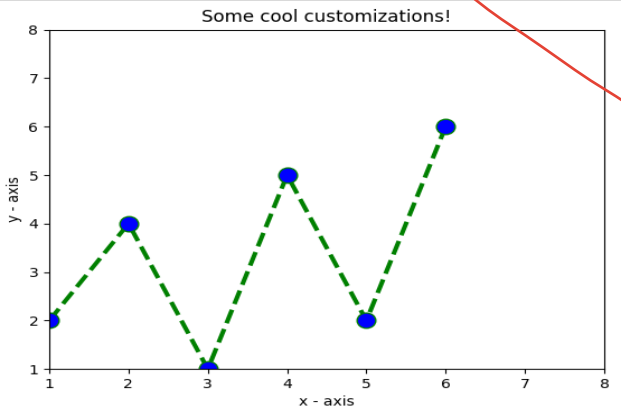


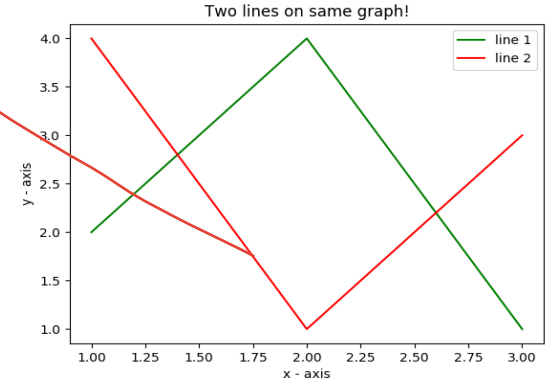


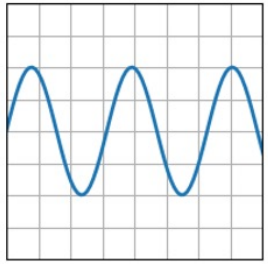




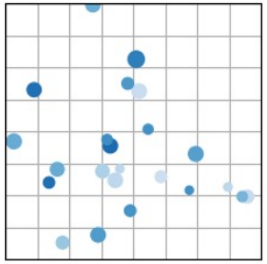




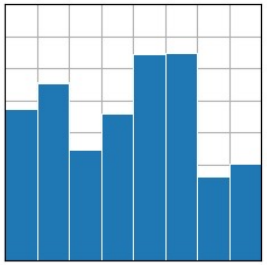




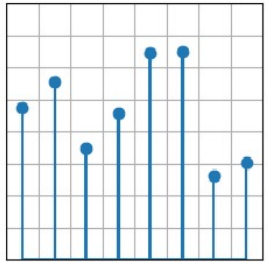
plot(x, y)



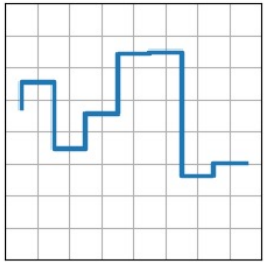
scatter(x, y)



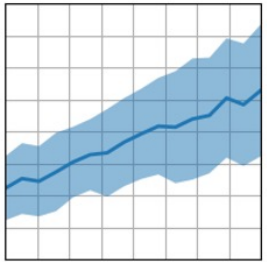
bar(x, height)



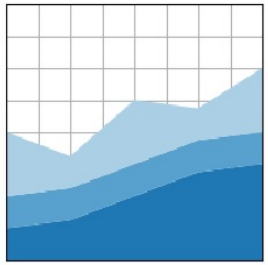
stem(x, y)



step(x, y)

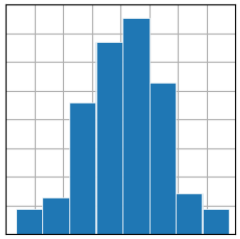


fill_between(x, y1, y2)

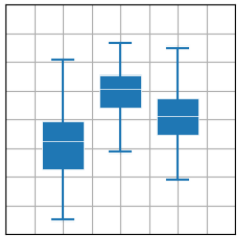


stackplot(x, y)

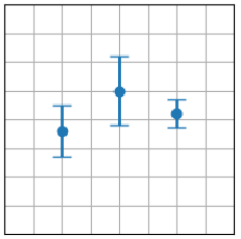
Basic Plots



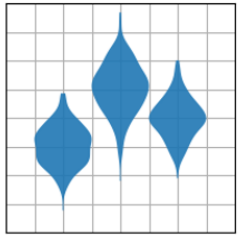
hist(x)



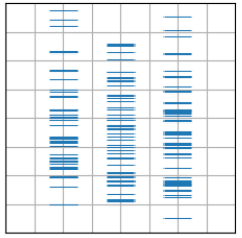
boxplot(X)



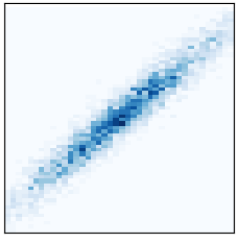
errorbar(x, y, yerr, xerr)



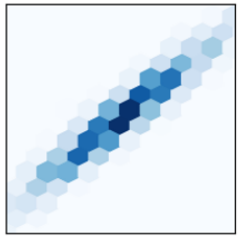
violinplot(D)



eventplot(D)



hist2d(x, y)



hexbin(x, y, C)

pie(x)

Statistics plots

Seaborn is a high-level **data visualization library** built on top of Matplotlib.

It provides **beautiful, modern, and statistical visualizations** with very few lines of code.

Comparison of Seaborn and Matplotlib for Data Visualization in Python

Feature / Criteria	Seaborn	Matplotlib
Level of Abstraction	High-level (simpler, fewer lines of code)	Low-level (more control, more coding)
Ease of Use	Very easy, beginner-friendly	Requires more coding and setup
Default Appearance / Style	Modern, attractive by default	Basic and it needs customization
Integration with Pandas	Excellent, it works directly with DataFrames	Good, but manual setup needed
Statistical Plotting	Built-in (heatmaps, boxplots, pairplots)	Limited, it requires manual coding
Customization / Flexibility	Medium	Very high
Performance	Good for medium datasets	Good, it supports large datasets
Complex Layouts / Subplots	Limited	Excellent
Best Use Cases	Data analysis, machine learning, quick plots	Scientific figures, research, full control

Types of Plots

Function	Description
Bar	Make a bar plot.
Barh	Make a horizontal bar plot.
Boxplot	Make a box and whisker plot.
Hist	Plot a histogram.
hist2d	Make a 2D histogram plot.
Pie	Plot a pie chart.
Plot	Plot lines and/or markers to the Axes.
Scatter	Make a scatter plot of x vs y.
Polar	Make a polar plot.
Stackplot	Draws a stacked area plot.
Stem	Create a stem plot
Step	Make a step plot.
Quiver	Plot a 2-D field of arrows.

Axis Functions

Function	Description
Axes	Add axes to the figure.
Text	Add text to the axes.
Title	Set a title of the current axes.
Xlabel	Set the x axis label of the current axis.
Xlim	Get or set the x limits of the current axes.
Xscale	Set the scaling of the x-axis.
Xticks	Get or set the x-limits of the current tick locations and labels.
Ylabel	Set the y axis label of the current axis.
Ylim	Get or set the y-limits of the current axes.
Yscale	Set the scaling of the y-axis.
Yticks	Get or set the y-limits of the current tick locations and labels.
Axes	Add axes to the figure.
Text	Add text to the axes.

Figure Functions

The **figure() function** in pyplot module of matplotlib library is used to create a new figure.

Syntax: *matplotlib.pyplot.figure(num=None, figsize=None, dpi=None, facecolor=None, edgecolor=None, frameon=True, FigureClass=, clear=False, **kwargs)*

Function	Description
Figtext	Add text to figure.
Figure	Creates a new figure.
Show	Display a figure.
Savefig	Save the current figure.
Close	Close a figure window.

Plotting commands: List of some commands which corresponding to `Matplotlib`

`acorr`	Plot the autocorrelation of *x*.
`angle_spectrum`	Plot the angle spectrum.
`annotate`	Annotate the point *xy* with text *text*.
`arrow`	Add an arrow to the Axes.
`autoscale`	Autoscale the axis view to the data (toggle).
`axes`	Add an Axes to the current figure and make it the current Axes.
`axhline`	Add a horizontal line across the Axes.
`axhspan`	Add a horizontal span (rectangle) across the Axes.
`axis`	Convenience method to get or set some axis properties.
`axline`	Add an infinitely long straight line.
`axvline`	Add a vertical line across the Axes.
`axvspan`	Add a vertical span (rectangle) across the Axes.

`bar`	Make a bar plot.
`bar_label`	Label a bar plot.
`barbs`	Plot a 2D field of barbs.
`barh`	Make a horizontal bar plot.
`box`	Turn the axes box on or off on the current axes.
`boxplot`	Draw a box and whisker plot.
`broken_barh`	Plot a horizontal sequence of rectangles.

`cla`	Clear the current axes.
`clabel`	Label a contour plot.
`clf`	Clear the current figure.
`clim`	Set the color limits of the current image.
`close`	Close a figure window.

`xlabel`	Set the label for the x-axis.
`xlim`	Get or set the x limits of the current axes.
`xscale`	Set the xaxis' scale.
`xticks`	Get or set the current tick locations and labels of the x-axis.
`ylabel`	Set the label for the y-axis.
`ylim`	Get or set the y-limits of the current axes.
`yscale`	Set the yaxis' scale.
`yticks`	Get or set the current tick locations and labels of the y-axis.

`grid`	Configure the grid lines.
`hexbin`	Make a 2D hexagonal binning plot of points *x*, *y*.
`hist`	Compute and plot a histogram.
`hist2d`	Make a 2D histogram plot.
`hlines`	Plot horizontal lines at each *y* from *xmin* to *xmax*.
`imread`	Read an image from a file into an array.
`imsave`	Colormap and save an array as an image file.
`imshow`	Display data as an image, i.e., on a 2D regular raster.

pyplot

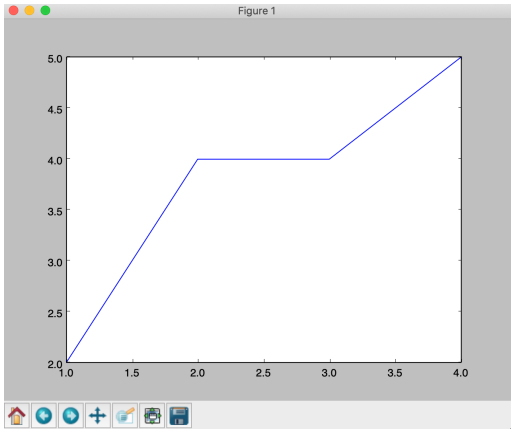
- `text()` : adds text in an arbitrary location
- `xlabel()` : adds text to the x-axis
- `ylabel()` : adds text to the y-axis
- `title()` : adds title to the plot
- `clear()` : removes all plots from the axes.
- `savefig()` : saves your figure to a file
- `legend()` : shows a legend on the plot

How To Clear A Plot In Python

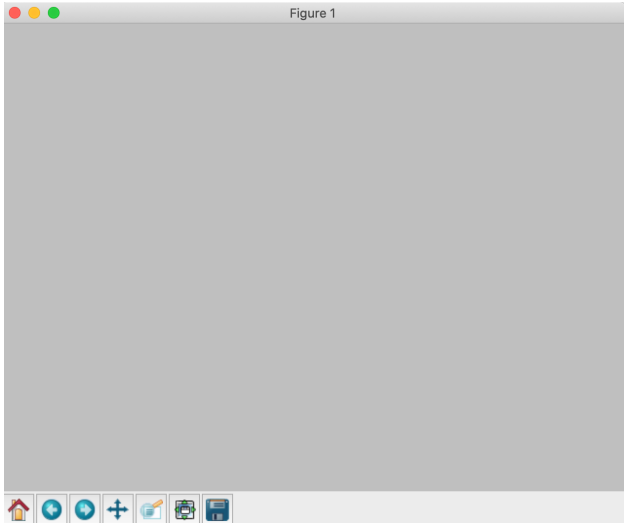
clf()* | *class: matplotlib.pyplot.clf(). Used to clear the current Figure's state without closing it.

cla()* | *class: matplotlib.pyplot.cla(). Used to clear the current Axes state without closing it.

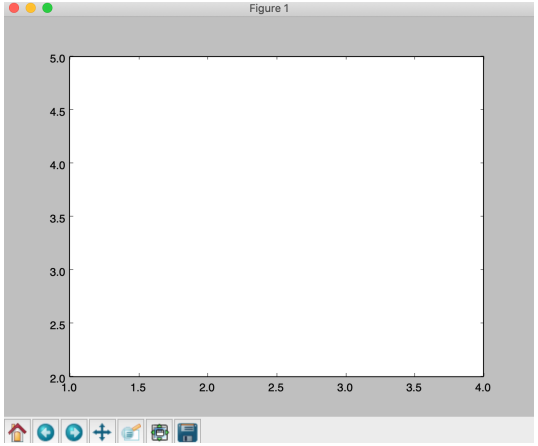
```
import matplotlib.pyplot as plt
f1 = plt.figure()
x = [1,2,3,4]
y = [2,4,4,5]
plt.plot(x,y)
plt.show()
```




```
import matplotlib.pyplot as plt
f1 = plt.figure()
x = [1,2,3,4]
y = [2,4,4,5]
plt.plot(x,y)
plt.clf()
plt.show()
```




```
import matplotlib.pyplot as plt
f1 = plt.figure()
x = [1,2,3,4]
y = [2,4,4,5]
plt.plot(x,y)
plt.cla()
plt.show()
```



End of Class 6