**IT8701 Introduction to Programming for Data Science**

**Lab 03 – Data Visualization using Matplotlib**

**What you will learn / do in this lab**

1. How to use the Matplotlib package to create various data visualizations such as line charts, bar charts, pie charts, scatter plots, histograms and box-plots
2. How to introduce simple interactivity into your charts

**Table of Contents**

[1. Overview 2](#_Toc529650527)

[A. What you will do for this lab 2](#_Toc529650528)

[B. Intro to Matplotlib 2](#_Toc529650529)

[2. Create line graphs 3](#_Toc529650530)

[3. Working with multiple figures and axes 6](#_Toc529650531)

[4. Working with text 8](#_Toc529650532)

[5. Bar Charts 10](#_Toc529650533)

[6. Pie Charts 11](#_Toc529650534)

[7. Scatter Plots 12](#_Toc529650535)

[8. Histograms 13](#_Toc529650536)

[9. Box Plots 14](#_Toc529650537)

[10. Interactive Charts 15](#_Toc529650538)

# Overview

## What you will do for this lab

Python has a variety of visualization libraries, including [seaborn](https://seaborn.pydata.org/), [bokeh](https://bokeh.pydata.org/), [networkx](https://networkx.github.io/), and [vispy](http://vispy.org/). Most Python visualization libraries are based wholly or partially on [matplotlib](https://matplotlib.org/), which often makes it the first resort for making simple plots, and the last resort for making plots too complex to create in other libraries.

In this matplotlib lab, you will learn the basics of using using the library.

## Intro to Matplotlib

*A picture is worth a thousand words*

When presenting a data analysis to our stakeholders, our goal is to provide as much information as possible in a user-friendly and intuitive manner.

If the users are unable to obtain the information they needs from your analysis, your work becomes useless to them. However, if you try to present a lot of information, but make the process of digesting the information too complex, then the information's signal gets lost in the noise of the complexity.

A graphical presentation often strikes the right balance. The Matplotlib library can help you present your data as graphs in your data analysis. Matplotlib not only allows you to generate a wide array of graphs from your data, it also enables you to create interactive applications that the user can exert control over, such as the way the graphs are drawn for instance.

|  |  |
| --- | --- |
| **Matplotlib can help you draw simple graphs like this** | **Or more complex ones like this** |
|  |  |

# Create line graphs

Task 1: Create a basic line graph with y values only

|  |  |  |
| --- | --- | --- |
|  | Plot a basic line graph with the array below.  Your output should resemble that of the figure on the right.  Notice that the x values have been auto generated for you. Since there are 6 values in the y array, six values are also created for x correspondingly.  **y = [0,2,4,6,8,10]** |  |

Task 2: Create a basic line graph with x and y values

|  |  |  |
| --- | --- | --- |
|  | Plot a basic line graph with the arrays below.  Your output should resemble that of the figure on the right.  **import numpy as np**  **x = np.arange(0,20,2) y = np.arange(0,40,4)** | A close up of a logo  Description generated with high confidence |

Task 3: Create a basic line graph with specified x-axis and y-axis

|  |  |  |
| --- | --- | --- |
|  | Plot a basic line graph with the arrays below.  Your x-axis should range from 0 to 200 while your y-axis should range from 0 to 600.  **import numpy as np**  **x = np.linspace(10,180,20)**  **y = np.linspace(10,550,20)**  Your output should resemble that of the figure on the right. | A close up of a logo  Description generated with high confidence |

Task 4: Create three basic line graphs with specified colors, line-style, line-width

|  |  |
| --- | --- |
|  | Plot three basic line graphs with the arrays below.   * Your x-axis should range from 0 to 150 while your y-axis should range from 0 to 200. * The first graph should be drawn with a red dotted line using arrays x1,y1 * The second graph should be drawn with a green solid line with line width of 10 using arrays x2, y2 * The third graph should be be drawn with blue triangle using arrays x3,y3   Refer to the link below to find out what symbol you should use for various markers and colors   * https://matplotlib.org/api/markers\_api.html   **import numpy as np**  **x1 = np.array([10,20,30,40,50,60,70,80,90,100])**  **y1 = x1\*3**  **x2 = np.array([20,40,60,80,100])**  **y2 = x2+5**  **x3 = np.random.randint(50,100,10)**  **y3 = x3/2**    Your output should resemble that of the figure below.  A picture containing object  Description generated with high confidence |

# Working with multiple figures and axes

Task 1: Create a multi-plot

|  |  |  |
| --- | --- | --- |
|  | Use the ***subplot*** method to produce a multi-plot that contains four line charts laid out in 4 rows, with following properties | |
|  | **Chart 1**   |  |  | | --- | --- | | **y-axis values** | np.arange(1,10,0.5) | | **color** | blue | | **marker type** | star |   **Chart 2**   |  |  | | --- | --- | | **y-axis values** | np.arange(1,50,5) | | **color** | #ff69b4 | | **marker type** | square marker |   **Chart 3**   |  |  | | --- | --- | | **y-axis values** | np.arange(100,200,10) | | **color** | Green | | **marker type** | diamond |   **Chart 4**   |  |  | | --- | --- | | **y-axis values** | np.arange(1000,2000,50) | | **color** | rgb (1.0,0.5, 0.5) | | **marker type** | dashed line | |  |
|  | Redraw the figure such that the plots are now displayed as a 2x2 layout instead. |  |

Task 2: Create a multi-figure

|  |  |  |  |
| --- | --- | --- | --- |
|  | Modify your code in Task 1 such that Charts 1 and 2 are drawn in one figure, and Charts 3 and 4 are drawn in a second figure. | | |
|  | Chart 1 and Chart 2 should be drawn on two separate rows | | Chart 3 and 4 are in 1 row with 2 columns.  A close up of a logo  Description generated with very high confidence |
|  |  |  | |

# Working with text

Task 1: Add text and “text-in-box”

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  | Create the following annotated graph using the given x and y values generated using the Numpy library.   |  | | --- | | import numpy as np | | x = np.arange(0.0, 5.0, 0.01) | | y = np.cos(2\*np.pi\*x) | |  |

Task 2: Add annotation (arrow)

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  | Create the following annotated graph using the given x and y values generated using the Numpy library.   |  | | --- | | import numpy as np | | x = np.arange(0.0, 5.0, 0.01) | | y = np.cos(2\*np.pi\*x) | |  |

Task 3: Add axis labels, subtitles and legend

<https://matplotlib.org/users/legend_guide.html>

|  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | Complete the Python code below to produce the figure as shown on the right.   |  | | --- | | import numpy as np | | x = np.linspace(0, 2\*np.pi, 128) | | y1 = np.cos(x) # y values for c1 | | y2 = np.sin(x) # y values for c2 | |  | | fig = plt.figure(figsize=(3, 3)) | | ax = fig.add\_subplot(111) | | ax.plot(x, y1, label = "c1") | | ax.plot(x, y2, label = "c2") | |  |

# Bar Charts

Task 1 – Analyse live births

|  |
| --- |
| For this task, you are required to generate a bar chart that shows the birth rates from the years **2006 to 2016** using the dataset below  Dataset: **live-births.csv**  Your output should look similar to that below. |
|  |

# Pie Charts

Task 1 – Analyse gender composition in years 1960 vs 2016

|  |
| --- |
| For this task, you are required to generate appropriate pie-charts to analyse the composition of total males and total females in the year 1960 compared to 2016 using the dataset below  Dataset: **singapore-residents-by-ethnic-group-and-sex-end-june-annual.csv**  Your output should look similar to that below. |
| A close up of a logo  Description generated with high confidence |

# Scatter Plots

Task 1 – Analyse relation between rainfall and temperature with scatterplot

|  |
| --- |
| For this task, you are required to generate a scatter plot graph that shows the relationship between the mean monthly rainfall and temperature in Singapore since Jan 1980.  Datasets to use:   * **surface-air-temperature-monthly-mean.csv** * **rainfall-monthly-number-of-rain-days.csv**   Your output should look similar to that below. |
| A close up of a map  Description generated with very high confidence |

# Histograms

Task 1 – Analyse trends in rainfall with histogram

|  |
| --- |
| For this task, you are required to generate a histogram graph to analyse the trends in the mean number of days of rainfall each month since 1980.  Dataset to use:   * **rainfall-monthly-number-of-rain-days.csv**   Your output should look similar to that below. |
| A screenshot of a cell phone  Description generated with very high confidence |

# Box Plots

Task 1 – Analyse median rentals in different areas using box plot

|  |
| --- |
| For this task, you are required to generate a box plot graph that compares the different rental rates in Orchard, Central Area and outside Central Area from 2012 to 2016.  Datasets to use:   * **median-rentals-and-vacancy-of-retail-space-by-locality-quarterly.csv**   Your output should look similar to that below. |
| A screenshot of a cell phone  Description generated with very high confidence |

# Interactive Charts

Task 1 – Create interactive chart showing x,y values as textbox

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  | Create the following interactive chart using the given x and y values generated using the Numpy library.  Your interactive chart should show a textbox showing the the x and y values at which the user clicks on the chart.   |  | | --- | | import numpy as np | | x = np.arange(0.0, 5.0, 0.01) | | y = np.cos(2\*np.pi\*x) | |  |

**-- End of Lab --**