

---

---

MEMORANDUM  
PROJECT 1 - EXPLORE WEATHER TRENDS

---

TO: UDACITY REVIEWER  
FROM: Jhonatan Nagasako  
SUBJECT: Project 1 – Explore Weather Trends  
DATE: 08-DEC-2020

---

## PURPOSE:

Using CSV files from UDACITY, compare local city and global data temperature (in degrees C) trends. Tools to be used is undefined and project scope is open ended.

## SUMMARY:

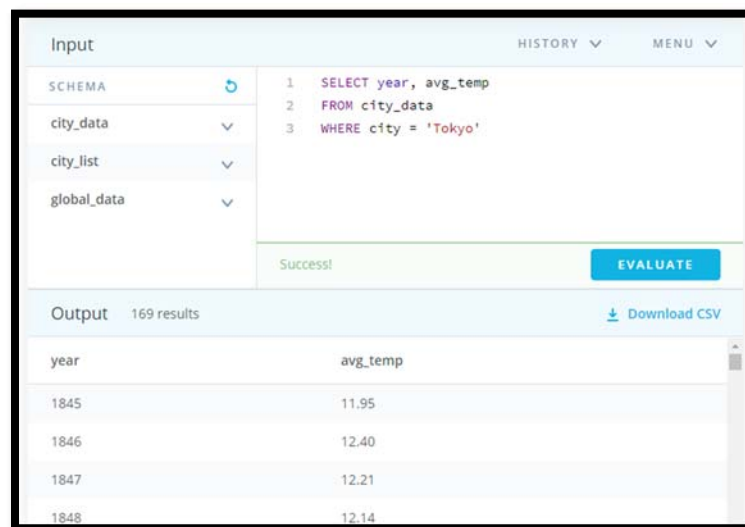
What tools did you use for each step? (Python, SQL, Excel, etc)

Primary tool used was Python because of prior experience programming (in other languages) and currently building on skills in Python language. Although it would have been easier to complete project in Excel—the project was more challenging completing it in Python! Acknowledgements and notes that helped me overcome the Python learning curve are shown in code and Table 1—all pieces of code utilized/referenced are understood of its intricacies and purposes. As practice, data was pulled from UDACITY database, then uploaded to personal Github account, and then pulled from for data analysis. There was data cleaning of the .csv prior to Python manipulation.

## SQL UDACITY TOOL

Two queries were made.

1. Query city\_data table for 'Tokyo' information, Figure 1.



The screenshot shows the Udacity SQL tool interface. On the left, under 'Input', there is a 'SCHEMA' section with a refresh icon and a list of tables: 'city\_data', 'city\_list', and 'global\_data', each with a dropdown arrow. The 'city\_data' table is selected. In the center, the SQL query is displayed: 

```
1 SELECT year, avg_temp
2 FROM city_data
3 WHERE city = 'Tokyo'
```

 Below the query, there is a green 'Success!' message and a blue 'EVALUATE' button. On the right, there are 'HISTORY' and 'MENU' dropdowns. Below the query editor, the 'Output' section shows '169 results' and a 'Download CSV' link. The output is a table with two columns: 'year' and 'avg\_temp'. The first four rows are visible: 1845 (11.95), 1846 (12.40), 1847 (12.21), and 1848 (12.14).

year	avg_temp
1845	11.95
1846	12.40
1847	12.21
1848	12.14

Figure 1: Query 1 for Tokyo, Japan data

2. Query global\_data table for avg\_temp pertaining to the same span of year selected in Query 1, Figure 2.

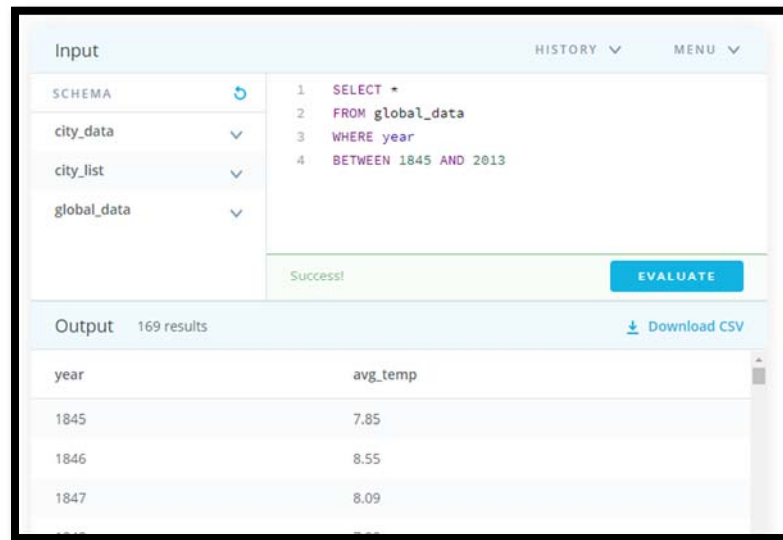


Figure 2: Query 2 for global data that spans date range from Query 1

## DATA CLEANING

Data analysis followed the common data analysis pipeline process for Tokyo, Japan. Which are the following steps:

1. Data import
2. Data clean
3. Data process/manipulation (in this case the averaging functions of global temps)
4. Evaluation

Table 1: Code help block – acknowledgements and credits to sources that were a key component with helping me understand Python coding – Entire code seen end of memo

```
'''
Jhonatan Nagasako
Udacity Data Analysis
Project 1 - Analyzing Global Temps
Purpose:
    Using code and CSV files provide to determine
    moving averages and line graph comparisons

OPPORTUNITY FOR IMPROVEMENT
1. Test scripts
2. Data cleaning (pre-cleaned the docs using excel)

HELP WITH CODING PROJECT IN PYTHON
# project help in PYTHON --> https://towardsdatascience.com/moving-averages-
in-python-16170e20f6c
# doing the data analysis in EXCEL would have been EASY -- it was more of a
challenge programming in PYTHON!

HELP WITH PLOTTING
# plotting help --> https://swcarpentry.github.io/python-novice-
```

```
gapminder/09-plotting/
# subplot help --> https://bertvandenbroucke.netlify.app/2019/07/10/the-
many-ways-to-combine-plots-in-python/
# HELP WITH PLOT AXIS CONTROL -->
https://www.kite.com/python/docs/matplotlib.pyplot.xlim

HELP WITH SQL
# BETWEEN function --> https://www.w3schools.com/sql/sql_between.asp
# WHERE function --> https://www.w3schools.com/sql/sql_where.asp
'''
```

How did you calculate the moving average?

A simple moving average (SMA) plot was utilized to smooth the data (over 10 year period), Equation 1 [1]. Python tool *pandas.Series.rolling* method was deployed for the analysis of the data.

*Equation 1: Cumulative Moving Average function*

$$SMA_t = \frac{x_t + x_{t-1} + x_{t-2} + \dots + x_{M-(t-1)}}{M}$$

What were your key considerations when deciding how to visualize the trends?

Key considerations used was imposing the raw data and SMA for both the local city and global data. These two plots were compared side by side using a sub plot function to observe any obvious differences or similarities. The years were concatenated to make this comparison easier, Figure 3 and Figure 4 (Figure 5 shows both plots on one figure).

At least four observations about the similarities and/or differences in the trends

1. Both the local city and global temperature data shows **similar** upward trend
2. Tokyo, Japan data is **different** from the global data such that there is an outlier in original dataset that may significantly increase the range and standard deviation of the Tokyo, Japan SMA temperature dataset.
3. Tokyo, Japan data is **different** from the global data such that the overall average temperate is higher.
4. Global data is **different** from the local city data such that the average temperature increase rate is greater in the last 50 years (specifically years 1950 to 2000)

## DISCUSSION – QUESTIONS

Is your city hotter or cooler on average compared to the global average? Has the difference been consistent over time?

Tokyo, Japan was on average hotter than the global average, almost by 4 degrees C. This has been consistent over time.

“How do the changes in your city’s temperatures over time compare to the changes in the global average?”

The rate of temperature increase is greater in the global average temperature readings when compared to Tokyo, Japan average temperature over time the same time period.

What does the overall trend look like? Is the world getting hotter or cooler? Has the trend been consistent over the last few hundred years?

World is certainly getting warmer based on the data over 150 years.

## REFERENCES

[1] A. I. Moreno, “Moving averages with Python,” *Medium*, 08-Jul-2020. [Online]. Available: <https://towardsdatascience.com/moving-averages-in-python-16170e20f6c>.

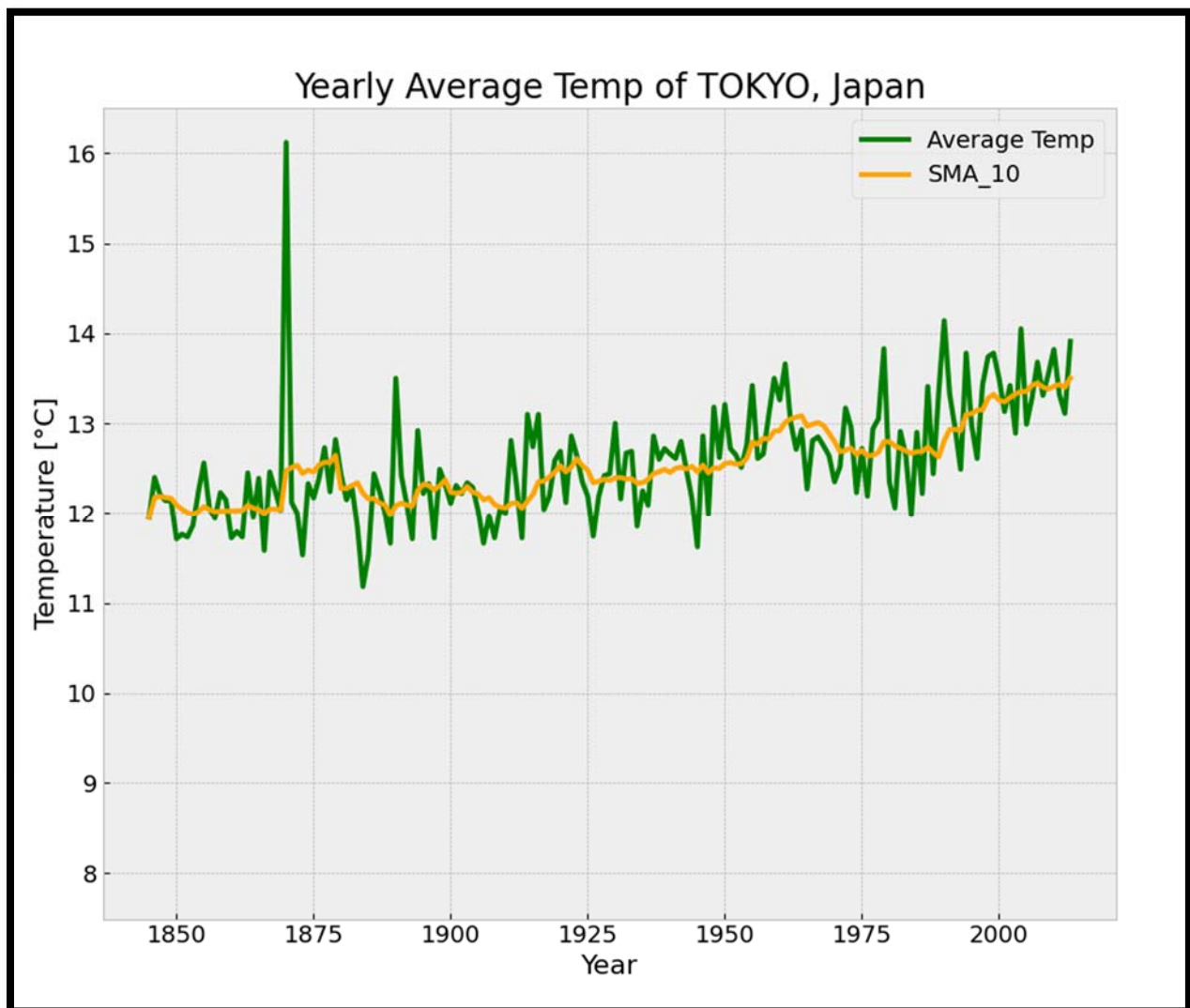


Figure 3: Yearly average temperature of Tokyo, Japan

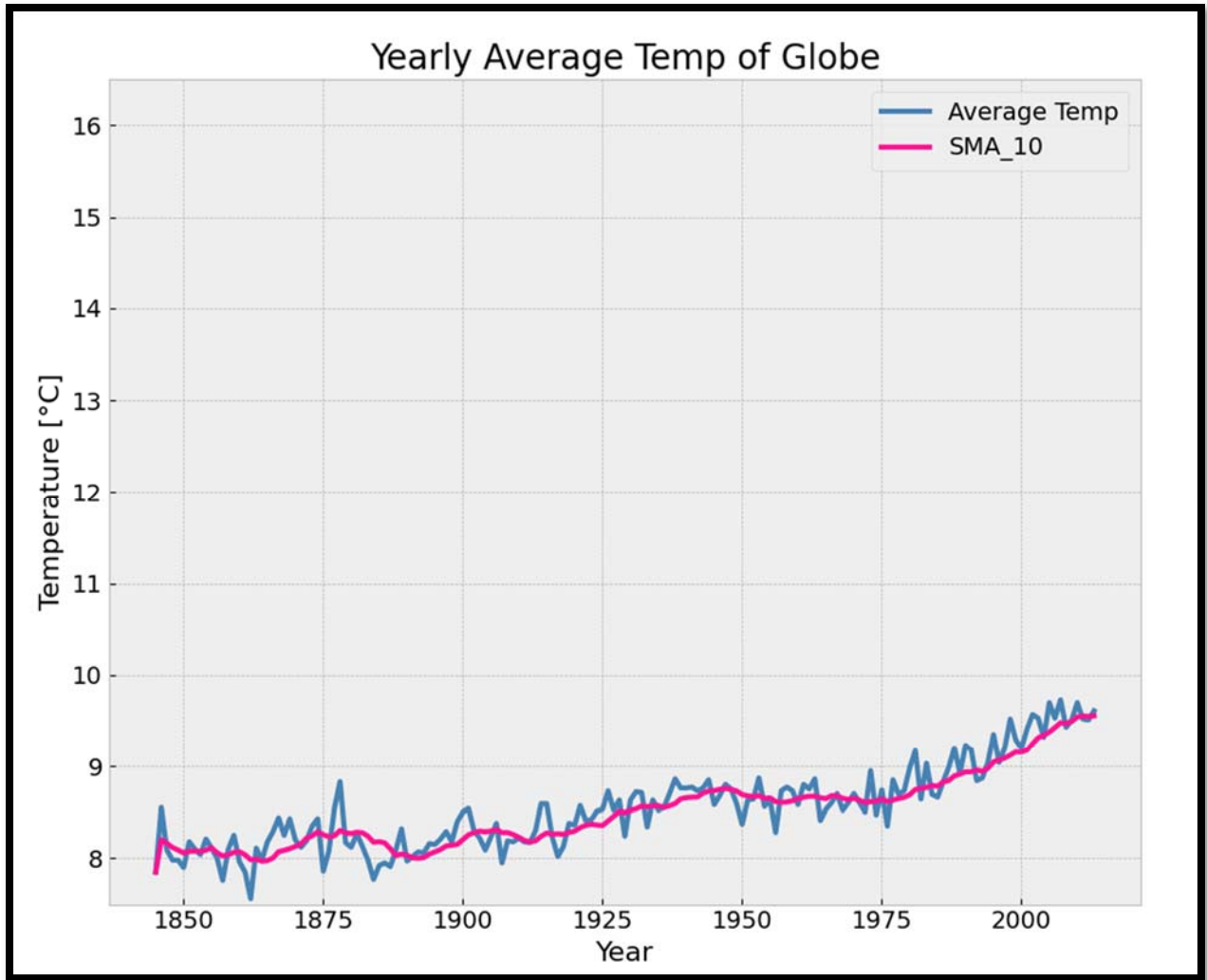


Figure 4: Yearly average temperate of GLOBE

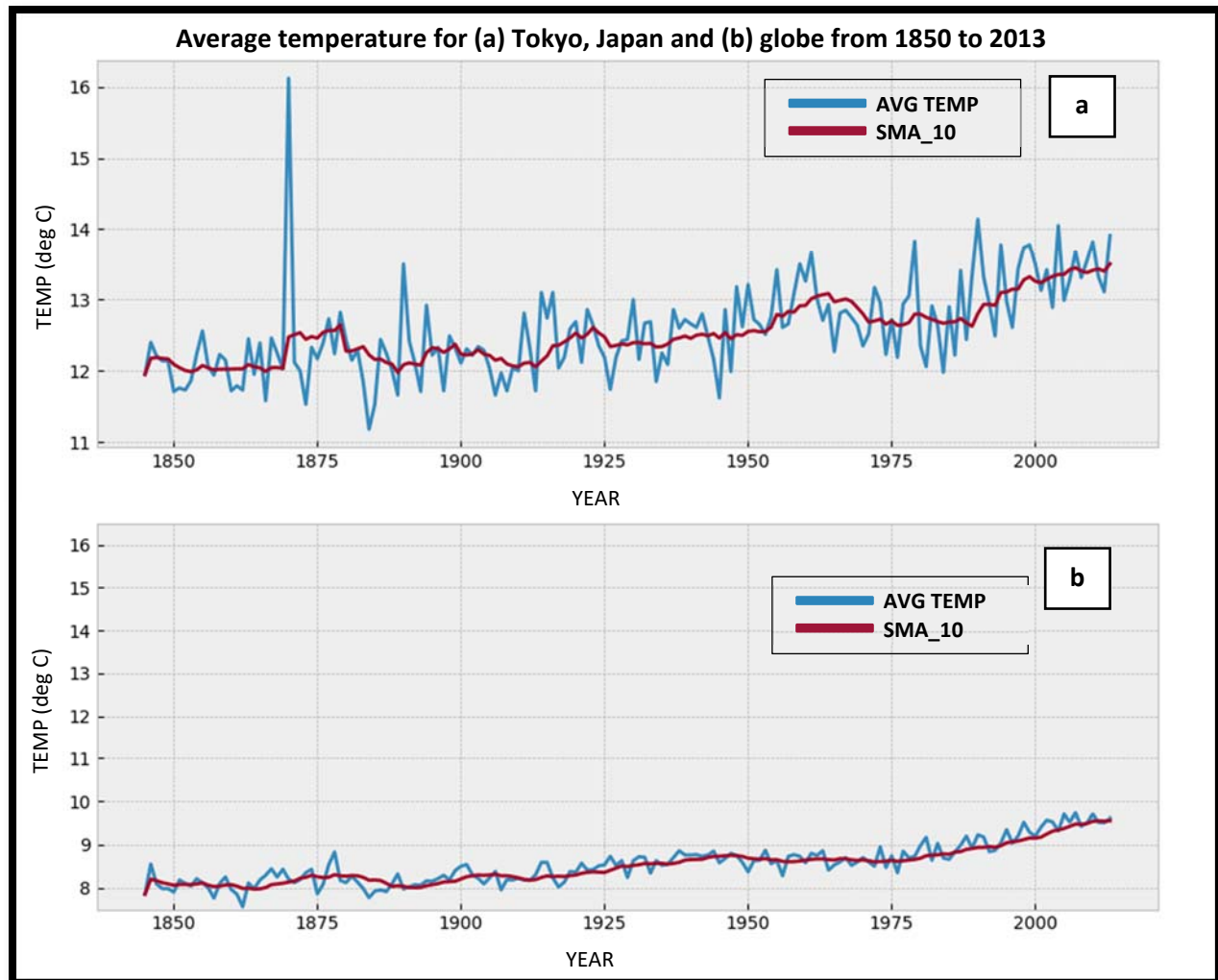


Figure 5: Average temperature for (a) Tokyo, Japan and (b) globe from 1850-2015

## CODE.PY

```
'''
Jhonatan Nagasako
Udacity Data Analysis
Project 1 - Analyzing Global Temps
Purpose:
    Using code and CSV files provide to determine
    moving averages and line graph comparisons

OPPORTUNITY FOR IMPROVEMENT
1. Test scripts
2. Data cleaning (pre-cleaned the docs using excel)

HELP WITH CODING PROJECT IN PYTHON
# project help in PYTHON --> https://towardsdatascience.com/moving-averages-in-python-16170e20f6c
# doing the data analysis in EXCEL would have been EASY -- it was more of a
challenge programming in PYTHON!
```

```

HELP WITH PLOTTING
# plotting help --> https://swcarpentry.github.io/python-novice-gapminder/09-plotting/
# subplot help --> https://bertvandenbroucke.netlify.app/2019/07/10/the-many-ways-to-combine-plots-in-python/
# HELP WITH PLOT AXIS CONTROL -->
https://www.kite.com/python/docs/matplotlib.pyplot.xlim

HELP WITH SQL
# BETWEEN function --> https://www.w3schools.com/sql/sql_between.asp
# WHERE function --> https://www.w3schools.com/sql/sql_where.asp
'''
#####
from IPython.display import display
import pandas as pd

# get csv files from github
df_global_data = pd.read_csv("https://raw.githubusercontent.com/nagaso-
socks/Udacity_Data_Analysis/main/Project%20Explore%20Weather%20Trends/shor
t_global_data.csv", encoding='utf-8')
df_japan = pd.read_csv("https://raw.githubusercontent.com/naga-
socks/Udacity_Data_Analysis/main/Project%20Explore%20Weather%20Trends/toky
o_city_data.csv", encoding='utf-8')

# Verify head and size
print("\nData Import Complete - TOKYO Data")
print(df_japan.head())
print("-----")
print(df_japan.info())

print("\nData Import Complete - GLOBAL Data")
print(df_global_data.head())
print("-----")
print(df_global_data.info())

print("=====")
print("=====minimum data clean=====")
print("=====")
# Remove all NaN from data that will be analyzed also minimum data cleaning
df_japan.dropna()
print(df_japan.head()) #checking that the city column removed from Japan
data

df_global_data.dropna()

print("=====")
print("=====average temps=====")
print("=====")

# set year as index
df_japan.set_index('year', inplace=True)
df_japan.index.name = 'year'
df_japan['japan_average_temperature'] = df_japan.mean(axis=1)
df_japan = df_japan[['japan_average_temperature']]
print(df_japan.head()) # check work

```

```

df_global_data.set_index('year', inplace=True)
df_global_data.index.name = 'year'
df_global_data['global_average_temperature'] = df_global_data.mean(axis=1)
df_global_data = df_global_data[['global_average_temperature']]
print(df_global_data.head()) # check work

print("=====")
print("=====create plots=====")
print("=====")

# plotting help --> https://swcarpentry.github.io/python-novice-
gapminder/09-plotting/

# create plots
import matplotlib.pyplot as plt

# matplotlib inline
plt.style.use('bmh')

print("=====")
print("=====SIMPLE moving average plots=====")
print("=====")

# SIMPLE moving average = SMA of 10 years
df_japan['SMA_10'] = df_japan.japan_average_temperature.rolling(10,
min_periods=1).mean()
df_global_data['SMA_10'] =
df_global_data.global_average_temperature.rolling(10, min_periods=1).mean()

#JAPAN TEMP SMA
# colors for the line plot
colors = ['green', 'orange']

# line plot - the yearly average japan temp
df_japan[['japan_average_temperature', 'SMA_10']].plot(color=colors,
linewidth=3, figsize=(12,6))

# modify ticks size
plt.xticks(fontsize=14)
plt.yticks(fontsize=14)
plt.legend(labels=['Average Temp', 'SMA_10'], fontsize=14)

# range axis control
# HELP WITH PLOT AXIS CONTROL -->
https://www.kite.com/python/docs/matplotlib.pyplot.xlim
plt.ylim(7.5, 16.5)

# title and labels
plt.title('Yearly Average Temp of TOKYO, Japan', fontsize=20)
plt.xlabel('Year', fontsize=16)
plt.ylabel('Temperature [°C]', fontsize=16)

#-----
# GLOBAL TEMP SMA
# colors for the line plot
colors = ['steelblue', 'deeppink']

```



```

# line plot - the yearly accumulated global temp
df_global_data[['global_average_temperature', 'SMA_10']].plot(color=colors,
linewidth=3, figsize=(12,6))

# modify ticks size
plt.xticks(fontsize=14)
plt.yticks(fontsize=14)
plt.legend(labels=['Average Temp', 'SMA_10'], fontsize=14)

# range axis control
plt.ylim(7.5, 16.5)

# title and labels
plt.title('Yearly Average Temp of Globe', fontsize=20)
plt.xlabel('Year', fontsize=16)
plt.ylabel('Temperature [°C]', fontsize=16)

# ===== COMBINE BOTH PLOTS =====
# subplot help: https://bertvandenbroucke.netlify.app/2019/07/10/the-many-ways-to-combine-plots-in-python/

fig, ax = plt.subplots(2, 1)

ax[0].plot(df_japan)
ax[1].plot(df_global_data)
# range axis control
plt.ylim(7.5, 16.5)

plt.show()

```