Submission - Falko Sieverding - Exploring Weather Trends

Goal of the project

I wil analyze local and global temperature data and compare the temperature trends.
For the porject, I choose the city, I am currently living in, Athens, Greece, Europe to visualize and describe the similarities and differences between the local and global temperature trends.

Steps taken

Data extraction from database using SQL

The Dataset

For this project I was provided with a database of 3 tables via a webinterface with the following schema:

1. city_list - Fields: city \ country
2. city_data - Fields: year \ city \ country \ avg_temp (°C)
3. global_data - This contains the average global temperatures by year (°C)

The SQL Queries

1. One Query per table:
   1.1. table city_data

```
SELECT year, avg_temp AS avg_temp_athens FROM city_data WHERE city='Athens'
```

- result count: 261

1.2. table global_data

```
SELECT year, avg_temp AS avg_temp_global FROM global_data
```

- result count: 266

2. Even one step further would be a FULL (OUTER) JOIN (Returns all records when there is a match in either table) to get all results.
   2.1. Full Outer join on table city_data and table global_data

```
SELECT city_data.year, global_data.year, global_data.avg_temp AS avg_temp_global,
city_data.avg_temp AS avg_temp_athens FROM global_data FULL JOIN city_data ON
global_data.year = city_data.year WHERE city_data.city='Athens';
```

- result count: 261

3. As the Full join does not have the expected result count of 266 or more, something is going wrong. I will go on with the 2 SQL queries (one on each table) and join the dataframes later.

Manipulation of data

I choose to manipulate the data in a jupyter python notebook (attached here and also on github), in which I use Pandas Libary for python to manipulate the data.

1. After I have uploaded the CSV files to my GitHub Repo and then read those into dataframes with the `pandas.read_csv()` function, I checked the data for wholes and duplicates, but could not find any, just that the datasets were not fully overlapping and the global dataset had some datapoints more than the Athens one.

2. Then I used the pandas inherent function `pandas.DataFrame.rolling(window=Windowssize)`, which provides a rolling window calculations, with the "mean" aggregation `pandas.DataFrame.rolling(window=Windowssize).mean()`. This I did for the local and global dataframe each with a window size of 5 and 10 years.

   - example code for adding a 5 year Simple Moving Average (SMA) into my dataframe:

```
pandas.DataFrame['avg_temp_Athens_SMA_5']=pandas.DataFrame.iloc[:,1].rolling(window=5).mean()
```

3. After these steps I had 2 dataframes (global and local_Athens) with a 5 and 10 year SMA

Visualization of data

For the visualization of the data I choose to use the python libary matplotlib.
On the x-axis the years in steps of 10 are shown.
The y-axis shows the temperature data.
The used colors are explained in the Legend.

1. The first line plot (fig 1) shows the Simple Moving Average with a windowsize of 5 and 10 years for Athens and the global average temperature.

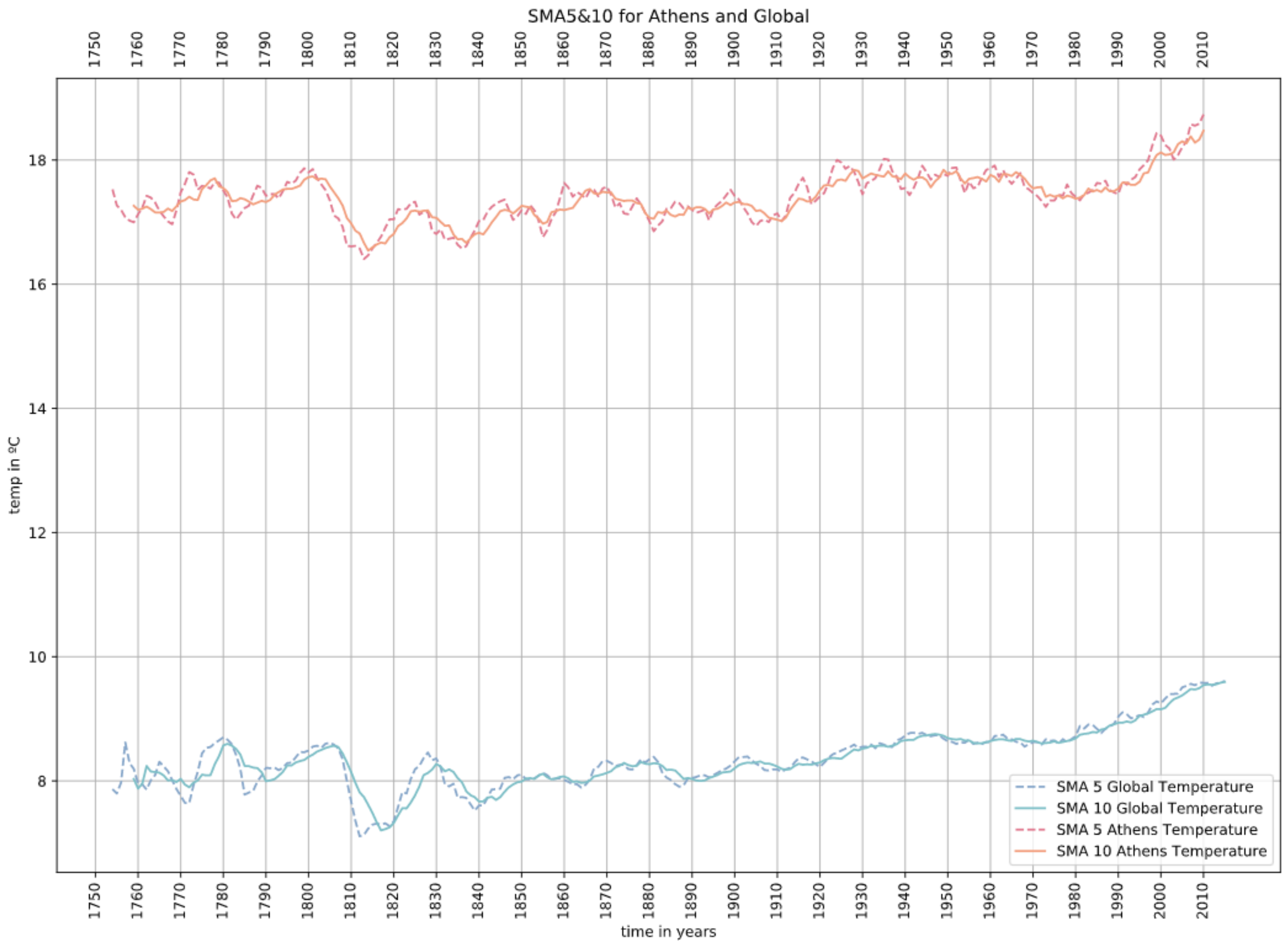
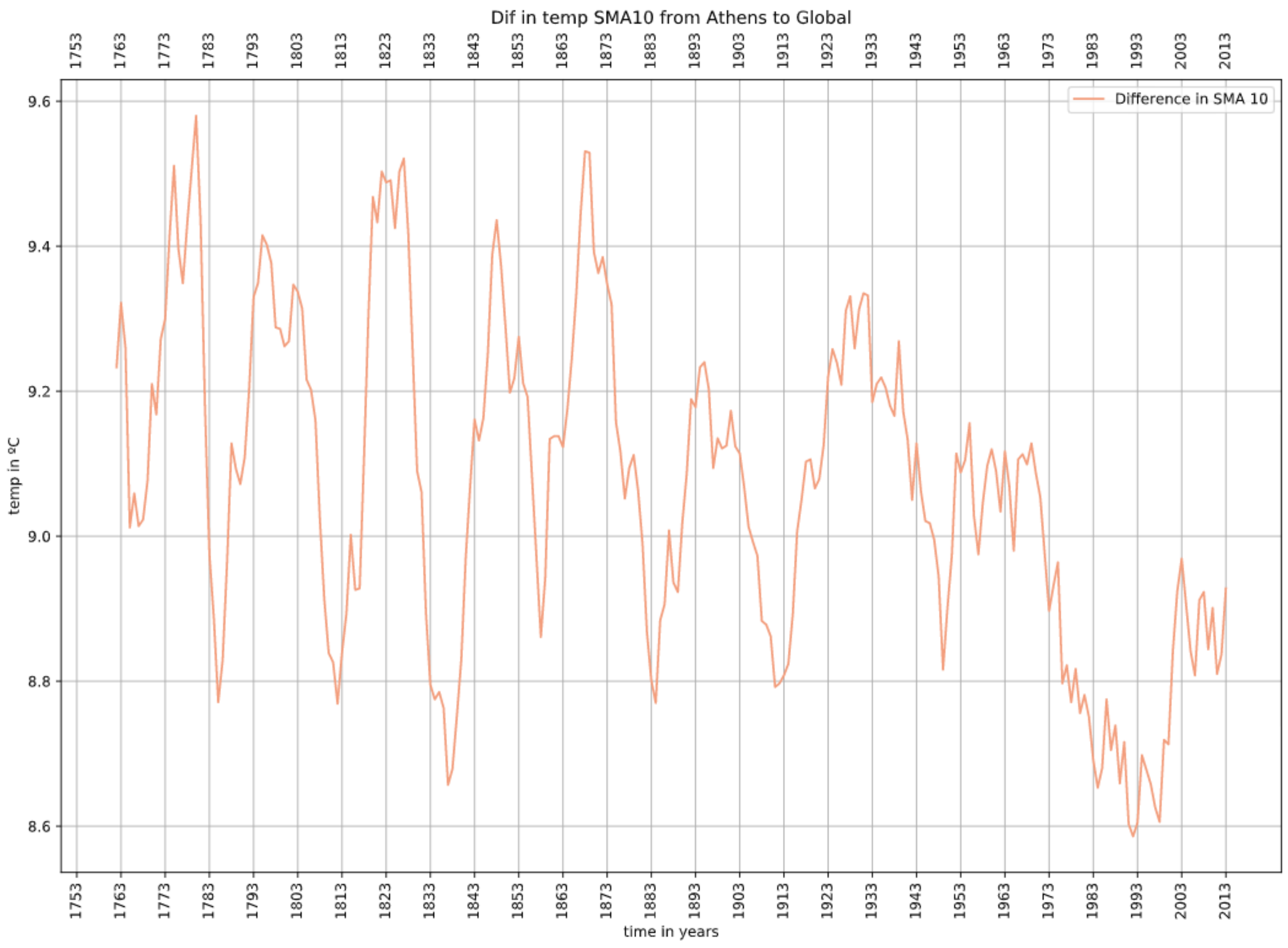


fig 1: SMA 5 and 10 years plotted for average temperatures in Athens and globaly

2. To have a better view and possibility for comparison, I printed and calculated the difference between the avergare temp in Athens against what globaly shows.



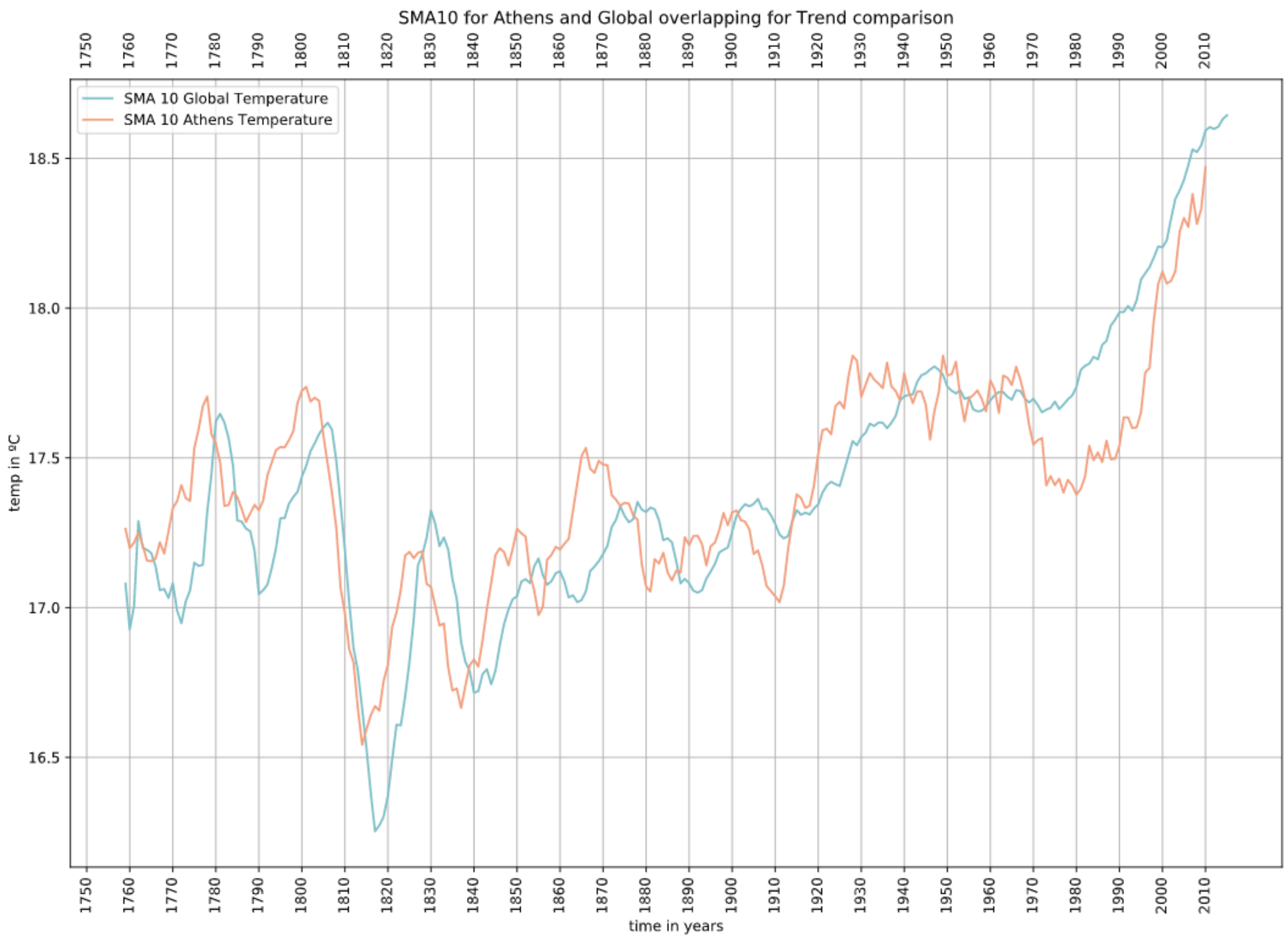
> fig 2: Difference between the SMA 10 of the avergae temperature in Athens compared to the global average

The calculations resultes in:

- Mean of the avergae yearly temp: 9.052490421455937 ºC
- Mean of the average SMA 5 temp: 9.05139299610896 ºC
- Mean of the average SMA 10 temp: 9.052396825396832 ºC

==> I will use as a correction for a plot the average difference of 9.05 ºC between the average temperature globaly to that of Athens.

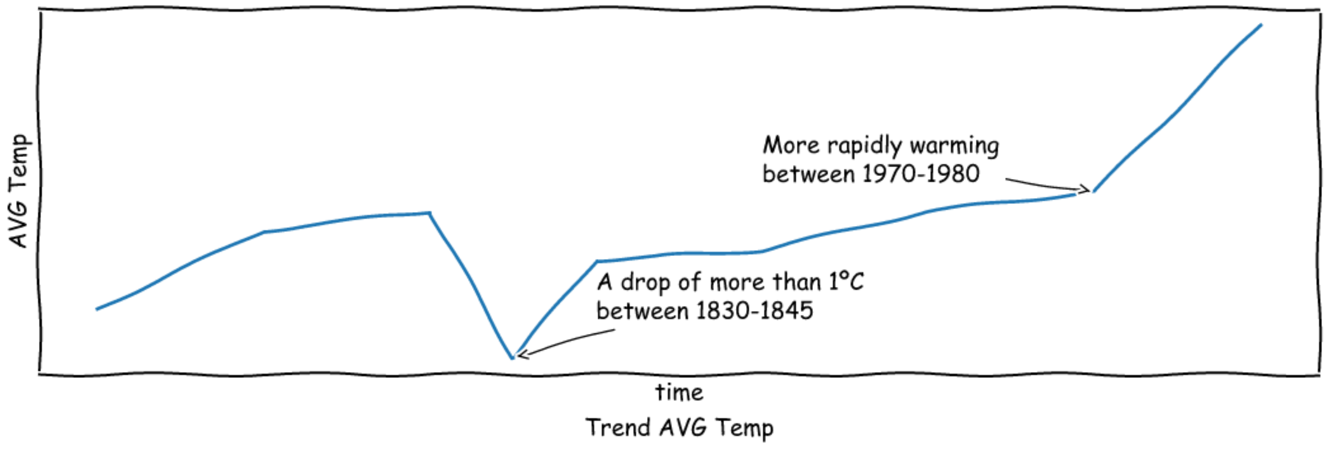
3. To better compare the local and global trend I have arranged the correction z-axis shift for the global temperature in this plot



> fig 3: SMA 10 years plotted for average temperatures in Athens and globaly for trend comparison

Observations

The Trend of the Moving Average I will show in an abstracted form in fig 4 to show also my observations.



> fig 4: Displaying the trend of the local and global temp average over time

1. As we have seen in our calculation and in fig 2, the difference between the global average temperature or its Moving Averge over 5 or 10 years stays within 1ºC and is in Average roughly 9ºC. That means the average temperature in Athen localy is 9ºC higher then the global Average, which is explained through Athens location on 37° Latitude, similar as San Francisco, USA. (A data Project for later days).
2. The Trend for the local and the global temperature are very similar.

3. Between 1830 and 1845 was a short period of very low average temperatures localy in Athens and globaly periods.
4. Between 1970 and 1980 the local and global temperature average has grown more rapidly than before.
5. The Average temperature after 2010 is roughly 1°C higher localy and globaly than the measurements around 1760.

Definitions

Simple Moving Average (SMA) : Simple Moving Average (SMA) uses a sliding window to take the average over a set number of time periods. It is an equally weighted mean of the previous n data.
Full Outer Join : in SQL the FULL OUTER JOIN combines the results of both left and right outer joins and returns all (matched or unmatched) rows from the tables on both sides of the join clause.

Attachments

1. Link to the GitHub Repo
2. The used Jupyter Notebook

Jupyter Notebook as python code

```
# To add a new cell, type '# %%'
# To add a new markdown cell, type '# %% [markdown]'
# %% [markdown]
# # Exploring Weather Trends - Athens, GR compared to the World - Udacity Project
#
# For the porject, I choose the city, I am currently living in, [Athens, Greece,
Europe](https://osm.org/go/xxSiEUQ-) to visualize and describe the similarities
and differences between the local and global temperature trends.
# %% [markdown]
# # The Dataset
#
# Within the project we have been given access to a database via a webinterface
with the following schema:
# - three tables in the database:
# 1. city_list - Fields: city \ country
# 2. city_data - Fields: year \ city \ country \ avg_temp (ºC)
# 3. global_data - This contains the average global temperatures by year (ºC).
# %% [markdown]
# # The Queries
#
# 1. For explaration I entered the SQL query:
# ```
# SELECT * FROM city_list WHERE country='Greece'
# ```
# which showed me that the only dataset for the country Greece came from Athens,
Greece, my chosen target.
#
# 2. As next step I issued the query to the city_data table:
# ```
# SELECT * FROM city_data WHERE city='Athens'
```

```
# ```
# This resulted in 261 results, which I downloaded as CSV file for later use.
#
# 3. As last step I queried the global_data table with the query:
# ```
# SELECT * FROM global_data
# ```
# with 266 results, also downloaded as CSV file.
#
# 4. The different result sets show us that we have to view the datasets before we
work with them.
# They will be uploaded to github next to this notebook to work with them.
# 4.1. [city_data_Athens]
(https://raw.githubusercontent.com/universalamateur/Udacity_Data_Analyst_Nanodegre
e/main/Exploring%20Weather%20Trends%20-%20Project/city_data_WHERE_city_Athens.csv)
# 4.2. [global_data]
(https://raw.githubusercontent.com/universalamateur/Udacity_Data_Analyst_Nanodegre
e/main/Exploring%20Weather%20Trends%20-%20Project/global_data.csv)
# %% [markdown]
# # The Alternativ Queries
# - A different Aproach would be to do some work directly in the query:
# 1. Renaming Columns and using only what we need:
# 1.1. city_data
# ```
# SELECT year, avg_temp AS avg_temp_athens FROM city_data WHERE city='Athens'
# ```
# - result count: 261
# 1.2. global_data
# ```
# SELECT year, avg_temp AS avg_temp_global FROM global_data
# ```
# - result count: 266
# 2. Even one step further would be a FULL (OUTER) JOIN (Returns all records when
there is a match in either table) to get all results.
# 2.1. one direction
# ```
# SELECT city_data.year, global_data.year, global_data.avg_temp AS
avg_temp_global, city_data.avg_temp AS avg_temp_athens FROM global_data FULL JOIN
city_data ON global_data.year = city_data.year WHERE city_data.city='Athens';
# ```
# - result count: 261
# 2.2. other direction
# ```
# SELECT city_data.year, global_data.year, global_data.avg_temp AS
avg_temp_global, city_data.avg_temp AS avg_temp_athens FROM city_data FULL JOIN
global_data ON city_data.year = global_data.year WHERE city_data.city='Athens';
# ```
# - result count: 261
# 3. As the Full join does not have the expected result count of 266, something is
going wrong. I will go on with the 2 SQL queries and join the dataframes later.
#

# %%
#Importing the necessary Python modules for our analysis
```

```
import pandas as pd
import matplotlib.pyplot as plt

# %% [markdown]
# # Preparation of the temp data for Athens
# 1. We load the temperature data of Athens into a Dataframe
# 2. We rename the avg temp data column to make sense with a possible merge
# 3. We drop unnecessary columns
# 4. we set the year as index

# %%
tempDataAtehns=pd.read_csv("https://raw.githubusercontent.com/universalamateur/Uda
city_Data_Analyst_Nanodegree/main/Exploring%20Weather%20Trends%20-
%20Project/city_data_WHERE_city_Athens.csv")
tempDataAtehns.rename(columns={'avg_temp':'avg_temp_athens'}, inplace=True)
tempDataAtehns.drop(columns=['city', 'country'], inplace=True)
tempDataAtehns.set_index('year')
print('The Shape of the Dataframe for the average temperatures in Athens, with
which we will work on is: ',tempDataAtehns.shape)
print(tempDataAtehns.head())
print(tempDataAtehns.tail())

# %% [markdown]
# # Preparation of the global temp data
# 1. We load the temperature data into a Dataframe
# 2. We rename the avg temp data column to make sense with a possible merge
# 3. we set the year as index

# %%
tempDataGlobal=pd.read_csv("https://raw.githubusercontent.com/universalamateur/Uda
city_Data_Analyst_Nanodegree/main/Exploring%20Weather%20Trends%20-
%20Project/global_data.csv")
tempDataGlobal.set_index('year')
tempDataGlobal.rename(columns={'avg_temp':'avg_temp_global'}, inplace=True)
print('The Shape of the Dataframe for the average temperatures globaly, with which
we will work on is: ',tempDataAtehns.shape)
print(tempDataGlobal.head())
print(tempDataGlobal.tail())

# %% [markdown]
# # Checking Data quality
# ## Chacking for duplicates in the year column

# %%
duplicateRowsTempDataAtehns = tempDataAtehns[tempDataAtehns.duplicated('year')]
print("Duplicate Rows Athens DF :")
print(duplicateRowsTempDataAtehns)
duplicateRowsTempDataGlobal = tempDataGlobal[tempDataGlobal.duplicated('year')]
print("Duplicate Rows Global SF :")
print(duplicateRowsTempDataGlobal)

# %% [markdown]
# ## Chacking on missing datapoints
# %%
print("Rows with NaN in Athens DF :")
print(tempDataAtehns[tempDataAtehns.isnull().any(axis=1)])
print("Rows with NaN in Global DF :")
print(tempDataGlobal[tempDataGlobal.isnull().any(axis=1)])

# %% [markdown]
# # Checking on uniterrupted years

# %%
tempYear=tempDataAtehns.at[0,'year']-1
counter=0
for row in tempDataAtehns.iterrows():
    if(row[1]['year']-tempYear!=1):
        print(row)
    tempYear=row[1]['year']
    counter+=1
print(counter,'rows were checked if a whole was present in the row of years in
Athens Data')
tempYear=tempDataGlobal.at[0,'year']-1
counter=0
for row in tempDataGlobal.iterrows():
    if(row[1]['year']-tempYear!=1):
        print(row)
    tempYear=row[1]['year']
    counter+=1
print(counter,'rows were checked if a whole was present in the row of years in
Global Data')

# %% [markdown]
# # Calculating and Adding the Simple Moving Average (SMA) with a 5 and 10 year
window
#

# %%
tempDataAtehns['avg_temp_Athens_SMA_5'] =
tempDataAtehns.iloc[:,1].rolling(window=5).mean()
tempDataAtehns['avg_temp_Athens_SMA_10'] =
tempDataAtehns.iloc[:,1].rolling(window=10).mean()
tempDataAtehns.head(10)


# %%
tempDataGlobal['avg_temp_global_SMA_5'] =
tempDataGlobal.iloc[:,1].rolling(window=5).mean()
tempDataGlobal['avg_temp_global_SMA_10'] =
tempDataGlobal.iloc[:,1].rolling(window=10).mean()
tempDataGlobal.head(10)

# %% [markdown]
# # Visualization of the temperature data

# %%
plt.figure(figsize=[15,10])
plt.grid(True)
plt.xlabel("time in years")
plt.ylabel("temp in ºC")
plt.title("SMA5&10 for Athens and Global")
plt.xticks((tempDataGlobal.index.tolist()[::10]), (tempDataGlobal['year'].tolist()
[::10]), rotation=90, visible=True)
plt.plot(tempDataGlobal['avg_temp_global_SMA_5'],label='SMA 5 Global Temperature',
linestyle='dashed', color='#87a7ca')
plt.plot(tempDataGlobal['avg_temp_global_SMA_10'],label='SMA 10 Global
Temperature', color='#7dbfc7')
plt.plot(tempDataAtehns['avg_temp_Athens_SMA_5'],label='SMA 5 Athens Temperature',
linestyle='dashed', color='#e1798f')
plt.plot(tempDataAtehns['avg_temp_Athens_SMA_10'],label='SMA 10 Athens
Temperature', color='#f39e7d')
plt.legend(loc=0)

# %% [markdown]
# # The difference between the Athens and global Temperature in a Plot

# %%
plt.figure(figsize=[15,10])
plt.grid(True)
plt.xlabel("time in years")
plt.ylabel("temp in ºC")
plt.title("Dif in AVG temp from Athens to Global")
plt.xticks((tempDataAtehns.index.tolist()[::10]), (tempDataAtehns['year'].tolist()
[::10]), rotation=90, visible=True)
plt.plot(tempDataAtehns['avg_temp_athens']-
tempDataGlobal['avg_temp_global'],label='Difference in SMA 10', color='#f39e7d')
plt.legend(loc=0)


# %%
plt.figure(figsize=[15,10])
plt.grid(True)
plt.xlabel("time in years")
plt.ylabel("temp in ºC")
plt.title("Dif in temp SMA10 from Athens to Global")
plt.xticks((tempDataAtehns.index.tolist()[::10]), (tempDataAtehns['year'].tolist()
[::10]), rotation=90, visible=True)
plt.plot(tempDataAtehns['avg_temp_Athens_SMA_10']-
tempDataGlobal['avg_temp_global_SMA_10'],label='Difference in SMA 10',
color='#f39e7d')
plt.legend(loc=0)

# %% [markdown]
# # For a better trend comparison
# We

# %%
tempDataMerged=pd.merge(tempDataAtehns, tempDataGlobal)

tempDataMerged['dif_avg_temp']=(tempDataMerged.iloc[:,1]-tempDataMerged.iloc[:,4])
tempDataMerged['dif_avg_temp_SMA_5']=(tempDataMerged.iloc[:,2]-
tempDataMerged.iloc[:,5])
tempDataMerged['dif_avg_temp_SMA_10']=(tempDataMerged.iloc[:,3]-
tempDataMerged.iloc[:,6])
tempDataMerged.head(10)


# %%
print('Mean of the avergae yearly temp: ',tempDataMerged['dif_avg_temp'].mean())
print('Mean of the average SMA 5 temp:
',tempDataMerged['dif_avg_temp_SMA_5'].mean())
print('Mean of the average SMA 10 temp:
',tempDataMerged['dif_avg_temp_SMA_10'].mean())

# %% [markdown]
# # Dif Result
# With these calculations we will defince the avergae temp dif between global and
Athens as 9.05 degrees and use this to plot again.

# %%
plt.figure(figsize=[15,10])
plt.grid(True)
plt.xlabel("time in years")
plt.ylabel("temp in ºC")
plt.title("SMA10 for Athens and Global overlapping for Trend comparison")
plt.xticks((tempDataGlobal.index.tolist()[::10]), (tempDataGlobal['year'].tolist()
[::10]), rotation=90, visible=True)
plt.plot(tempDataGlobal['avg_temp_global_SMA_10']+9.05,label='SMA 10 Global
Temperature', color='#7dbfc7')
plt.plot(tempDataAtehns['avg_temp_Athens_SMA_10'],label='SMA 10 Athens
Temperature', color='#f39e7d')
plt.legend(loc=0)


# %%
plt.figure(figsize=[15,10])
plt.grid(True)
plt.xlabel("time in years")
plt.ylabel("temp in ºC")
plt.title("SMA5 for Athens and Global overlapping for Trend comparison")
plt.xticks((tempDataGlobal.index.tolist()[::10]), (tempDataGlobal['year'].tolist()
[::10]), rotation=90, visible=True)
plt.plot(tempDataGlobal['avg_temp_global_SMA_5']+9.05,label='SMA 5 Global
Temperature', color='#87a7ca')
plt.plot(tempDataAtehns['avg_temp_Athens_SMA_5'],label='SMA 5 Athens Temperature',
color='#e1798f')
plt.legend(loc=0)

# %% [markdown]
# # Extreme Points
# Here are the Min and Max numbers

# %%
print('Min Avergare Temp Globaly: ')
print(tempDataGlobal.iloc[tempDataGlobal['avg_temp_global'].idxmin()])
print()
print('Max Avergare Temp Globaly: ')
print(tempDataGlobal.iloc[tempDataGlobal['avg_temp_global'].idxmax()])
print()
print('Min Trend Temp Globaly: ')
print(tempDataGlobal.iloc[tempDataGlobal['avg_temp_global_SMA_10'].idxmin()])
print()
print('Max Trend Temp Globaly: ')
print(tempDataGlobal.iloc[tempDataGlobal['avg_temp_global_SMA_10'].idxmax()])
print()
print('Min Avergare Temp Athens: ')
print(tempDataAtehns.iloc[tempDataAtehns['avg_temp_athens'].idxmin()])
print()
print('Max Avergare Temp Globaly: ')
print(tempDataAtehns.iloc[tempDataAtehns['avg_temp_athens'].idxmax()])
print()
print('Min Trend Temp Athens: ')
print(tempDataAtehns.iloc[tempDataAtehns['avg_temp_Athens_SMA_10'].idxmin()])
print()
print('Max Trend Temp Athens: ')
print(tempDataAtehns.iloc[tempDataAtehns['avg_temp_Athens_SMA_10'].idxmax()])


# %%
with plt.xkcd():
    fig = plt.figure(figsize=[15,10])
    ax = fig.add_axes([0.15, 0.1, 0.7, 0.3])

    ax.set_xticks([])
    ax.set_yticks([])
    x = [1,  2,3,3.5,  4,5  ,6,7,8]
    y = [1,1.8,2,0.5,1.5,1.6,2,2.2,3.9]

    ax.annotate(
        'A drop of more than 1ºC\nbetween 1830-1845',
        xy=(3.5, 0.5), arrowprops=dict(arrowstyle='->'), xytext=(4, 0.9))


    ax.annotate(
        'More rapidly warming\nbetween 1970-1980',
        xy=(7, 2.2), arrowprops=dict(arrowstyle='->'), xytext=(5, 2.3))

    ax.plot(x,y)

    ax.set_xlabel('time')
    ax.set_ylabel('AVG Temp')
    fig.text(
        0.5, 0.05,
        'Trend AVG Temp',
        ha='center')


# %%
```