Project 1: Explore Weather Trends

Name: Tuan Bui

Date: April 28, 2020

1 . Outlines:

- What tools did you use for each step? (Python, SQL, Excel, etc) I used SQL to extract data and then used Python for manipulate and plot the data.
- How did you calculate the moving average? Suppose that we have a series x indexed by i. Let w be the window of the moving average, then the value of w moving average series at index n is computed as

$$MA_n = \frac{\sum_{i=n-w+1}^{n} x_i}{w}$$

- What were your key considerations when deciding how to visualize the trends? The key considerations that I took when visualize the trends are:
  - remember to take out the NaN values appear after calculating moving average.
  - use dash lines for trends so that they don't make the plot too dense.
  - first record in HoChiMinh city and global are not the same

2. Manipulating data

Load data

To load data from the database, use the following SQL query and then download the dataset as CSV file

select * from global_data

select * from city_list

select * from city_data

Now use Python to load the dataset form hard disk

In [1]:
```
import pandas as pd
city_list = pd.read_csv('city_list.csv',header = 0)
city_data = pd.read_csv('city_data.csv',header = 0)
global_data = pd.read_csv('global_data.csv',header = 0)
```

In [2]:
```
# Extract temperature data for Ho Chi Minh city, Vietnam
HCM = city_data[city_data['city']=='Ho Chi Minh City']
[['year','avg_temp']]
HCM.head()
```

Out[2]:

	year	avg_temp
25763	1825	27.11
25764	1826	NaN
25765	1827	NaN
25766	1828	NaN
25767	1829	NaN

Check if there are some missing values

In [3]:
```
print(sum(HCM['avg_temp'].isna()))
print(sum(global_data['avg_temp'].isna()))
```

```
15
0
```

There are some missing values in HCM dataset so we impute the NaN using filling forward method

In [4]:
```
HCM.fillna(method = 'ffill',inplace = True)
```

Calculate moving average

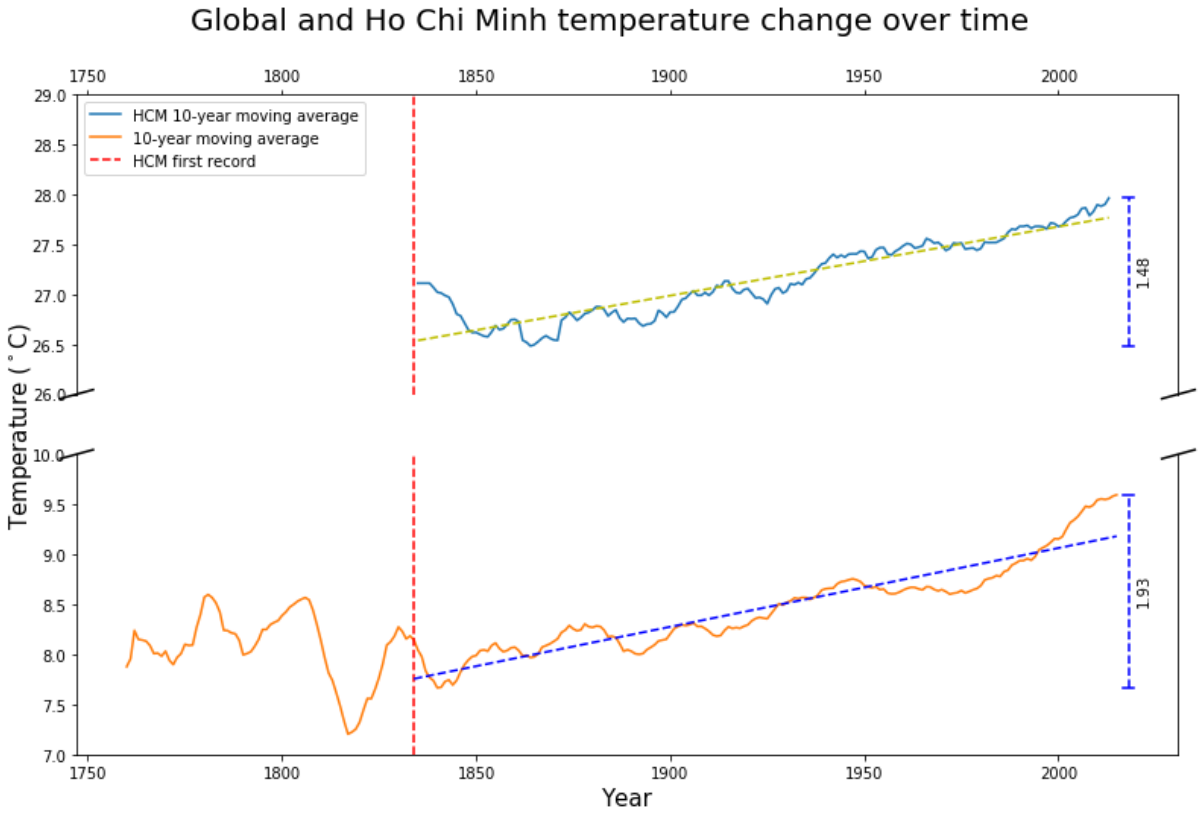
In [5]:
```
# set window for moving average
w = 10
HCM['MA'] = HCM['avg_temp'].rolling(window=w).mean()
global_data['MA'] =
global_data['avg_temp'].rolling(window=w).mean()
HCM = HCM.iloc[w:]
global_data = global_data.iloc[w:]
```

3. Visualization: Line chart

In [7]:

```
import matplotlib.pyplot as plt
import numpy as np
from numpy.polynomial import Polynomial as P

f,(ax,ax2) = plt.subplots(2,1,sharex=True, facecolor='w',figsize=(13,8))
ax.plot(HCM['year'],HCM['MA'],global_data['year'],global_data['MA'])
ax.axvline(1834, 0, 30, label='HCM first record',ls = '--',c='r')

# Calculate and plot the trend lines
Z_g = np.polyfit(global_data['year'],global_data['MA'],1)
p_g = np.poly1d(Z_g)
ax.plot(global_data['year'],p_g(global_data['year']),"b--")

Z_l = np.polyfit(HCM['year'],HCM['MA'],1)
p_l = np.poly1d(Z_l)
ax.plot(HCM['year'],p_l(HCM['year']),"y--")

ax.legend(['HCM 10-year moving average','10-year moving average','HCM first record'],loc='upper left')

ax.set_ylim(26,29)
ax.locator_params(axis='y', nbins=6)
ax2.plot(HCM['year'],HCM['MA'],global_data['year'],global_data['MA'])
ax2.axvline(1834, 0, 30, label='HCM first record',ls = '--',c='r')

# Calculate and plot the trend lines
global_34 = global_data[global_data['year']>=1834]
Z_g = np.polyfit(global_34['year'],global_34['MA'],1)
p_g = np.poly1d(Z_g)
ax2.plot(global_34['year'],p_g(global_34['year']),"b--")

Z_l = np.polyfit(HCM['year'],HCM['MA'],1)
p_l = np.poly1d(Z_l)
ax2.plot(HCM['year'],p_l(HCM['year']),"y--")

# Min and Max
min_HCM = min(HCM['MA'])
max_HCM = max(HCM['MA'])
min_g = min(global_34['MA'])
max_g = max(global_34['MA'])

ax.axhline(min_HCM,xmin = 0.95,xmax=0.96,ls = '-',c='b')
ax.axhline(max_HCM,xmin = 0.95,xmax=0.96,ls = '-',c='b')
ax.plot((2018,2018), (min_HCM,max_HCM),ls = '--',c='b')
ax.text(2020,27.3,round(max_HCM-min_HCM,2),rotation=90)

ax2.axhline(min_g,xmin = 0.95,xmax=0.96,ls = '-',c='b')
ax2.axhline(max_g,xmin = 0.95,xmax=0.96,ls = '-',c='b')
ax2.plot((2018,2018), (min_g,max_g),ls = '--',c='b')
ax2.text(2020,8.7,round(max_g-min_g,2),rotation=90)

ax2.set_ylim(7,10)
ax2.locator_params(axis='y', nbins=6)
```

```
ax.spines['bottom'].set_visible(False)
ax2.spines['top'].set_visible(False)
ax.yaxis.tick_left()
ax.tick_params(labeltop=True)
ax.xaxis.tick_top()

d = .015  # how big to make the diagonal lines in axes coordinates
# arguments to pass to plot, just so we don't keep repeating them
kwargs = dict(transform=ax.transAxes, color='k', clip_on=False)
ax.plot((-d, +d), (-d, +d), **kwargs)        # top-left diagonal
ax.plot((1 - d, 1 + d), (-d, +d), **kwargs)  # top-right diagonal

kwargs.update(transform=ax2.transAxes)  # switch to the bottom axes
ax2.plot((-d, +d), (1 - d, 1 + d), **kwargs)  # bottom-left diagonal
ax2.plot((1 - d, 1 + d), (1 - d, 1 + d), **kwargs)  # bottom-right diagonal

f.suptitle('Global and Ho Chi Minh temperature change over time',fontsize=20)

f.add_subplot(111, frameon=False)
# hide tick and tick label of the big axis
plt.tick_params(labelcolor='none', top=False, bottom=False, left=False, right=False)
plt.xlabel("Year",fontsize=15)
plt.ylabel("Temperature ($^\circ$C)",fontsize=15)

plt.show()
```



4. Observations:

Similarities:

- The temperature in Ho Chi Minh and global are both increasing over time
- The fluctuation in both graphs seems to be very similar. At a pacticular period, temperatures were having the same behavior in both Ho Chi Minh city and global

Differences:

- Observing the slope of two trend lines we see that global's temperature is increasing faster than Ho Chi Minh city's
- Since the first record in Ho Chi Minh city to date, temperature change in Ho Chi Minh (1.48) is less than global (1.93)