Simple model of cause-response relationship between atmosphere CO2 contents an global lower troposphere temperature anomaly

March 7, 2019

1 Introduction

The goal of this analysis is to establish simple and objective relationship between atmosphere CO2 contents and lower troposphere temperature anomaly, based od actual measured data. Only two datasets will be used, CO2 concentration measured each month at Mauna Loa Observatory (1958-2017) and sattelite based UAH lower troposphere temperature anomaly.

Datasets are obtained at following urls.

CO2 dataset: https://www.kaggle.com/ucsandiego/carbon-dioxide

UAH dataset: http://vortex.nsstc.uah.edu/data/msu/v6.0/tlt/uahncdc_lt_6.0.txt

2 Methodology

2.1 Importing data and quick look

After loading of datasets into pandas dataframes we are going to perform basic data cleanup (combine year and month columns into datetime object, assign it to dataframe index, rename columns and keep only those we are interested in - CO2 ppm concentration and global temperature anomaly). Further, in order to simplify dataframes, Mauna CO2 concentration (ppm) column will be renamed to *CO2*, and global lower troposphere temperature anomaly column will be renamed to *Temperature*.

In [1]:
```
import pandas as pd
import matplotlib.pyplot as plt
from pandas.plotting import register_matplotlib_converters
register_matplotlib_converters()
%matplotlib inline
```

In [2]:
```
co2 = pd.read_csv('datasets/co2.csv')
co2 = co2.assign(Date=pd.to_datetime(co2[['Year', 'Month']].assign(day=1)))
co2.set_index('Date', inplace=True)
co2 = co2[['Carbon Dioxide (ppm)']]
co2.rename(columns={"Carbon Dioxide (ppm)": "CO2"}, inplace=True)
```

```
In [3]: temp = pd.read_csv('datasets/uahncdc_lt_6.0.csv', delim_whitespace=True)
        temp.rename(columns={"Mo": "Month"}, inplace=True)
        temp = temp.assign(Date=pd.to_datetime(temp[['Year', 'Month']].assign(day=1)))
        temp.set_index('Date', inplace=True)
        temp = temp[['Globe']]
        temp.rename(columns={"Globe": "Temperature"}, inplace=True)
```

Now we are going to merge both dataframes into single dataframe, per their index (year/month):

```
In [4]: df = pd.concat([co2, temp], axis=1)
        df.dropna(inplace=True)
```

```
In [5]: df.head(3)
```

Out[5]:

Date	CO2	Temperature
1978-12-01	334.95	-0.36
1979-01-01	336.23	-0.33
1979-02-01	336.76	-0.27

Let's take a quick look at correlation and covariance between CO2 concentration and temperature anomaly:

```
In [6]: df.corr()
```

Out[6]:

	CO2	Temperature
CO2	1.00000	0.61243
Temperature	0.61243	1.00000

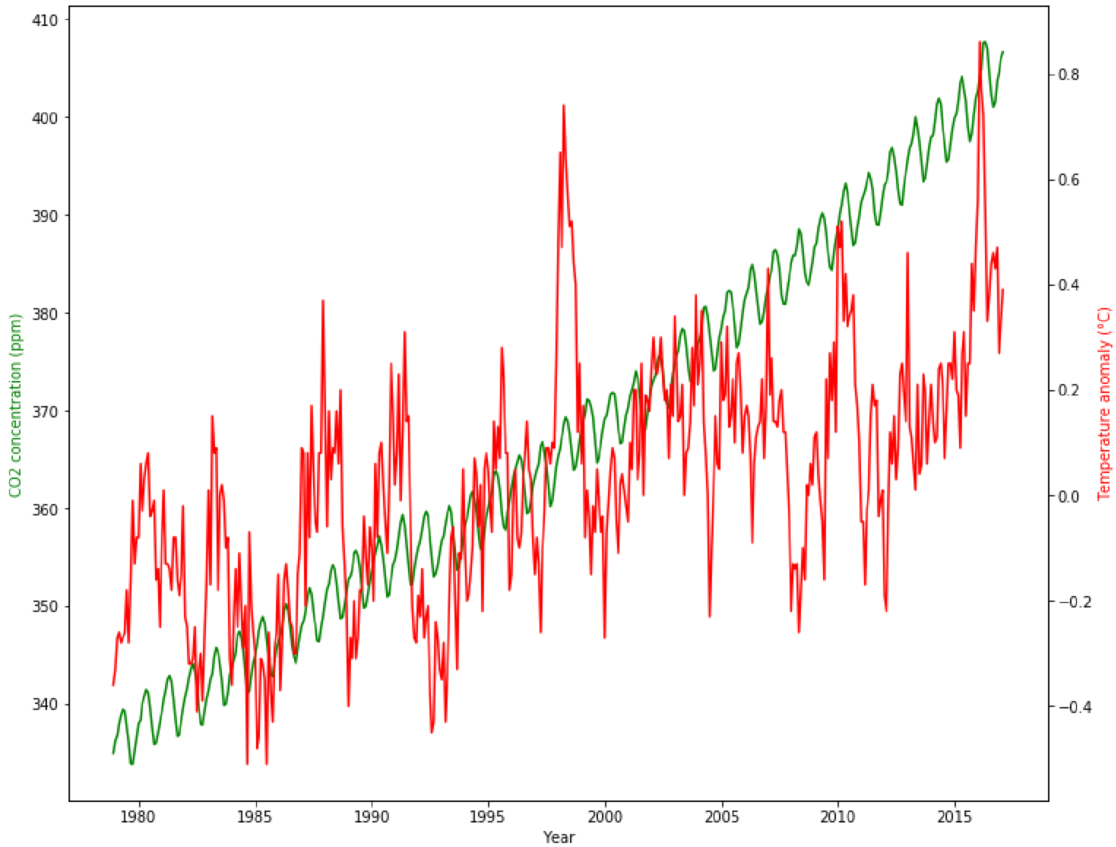
```
In [7]: df.cov()
```

Out[7]:

	CO2	Temperature
CO2	389.047209	2.767783
Temperature	2.767783	0.052499

Now we will plot both datasets at the same graph (green line is CO2 concentration and red line is temperature anomaly).

```
In [8]: fig = plt.figure(figsize=(12,10))

        ax = fig.add_subplot(111)
        ax.plot(df.index, df['CO2'], color='green')
        ax.set_xlabel('Year')
        ax.set_ylabel('CO2 concentration (ppm)', color='green')

        ax2 = ax.twinx()
        ax2.plot(df.index, df['Temperature'], color='red')
        ax2.set_ylabel('Temperature anomaly (řC)', color='red')


        plt.show()
```



2.2 Regression analysis

The objective of regression analysis is to determine the best fit polynomial function to available datasets. Here, 2nd order polynomial function is chosen.

$$y = ax^2 + bx + c \tag{1}$$

In [9]:
```
import numpy as np
import matplotlib.dates as mdates
```

In [10]:
```
x = mdates.date2num(df.index)
```

In [11]:
```
co2_p2 = np.polyfit(x, df['CO2'],2)
```

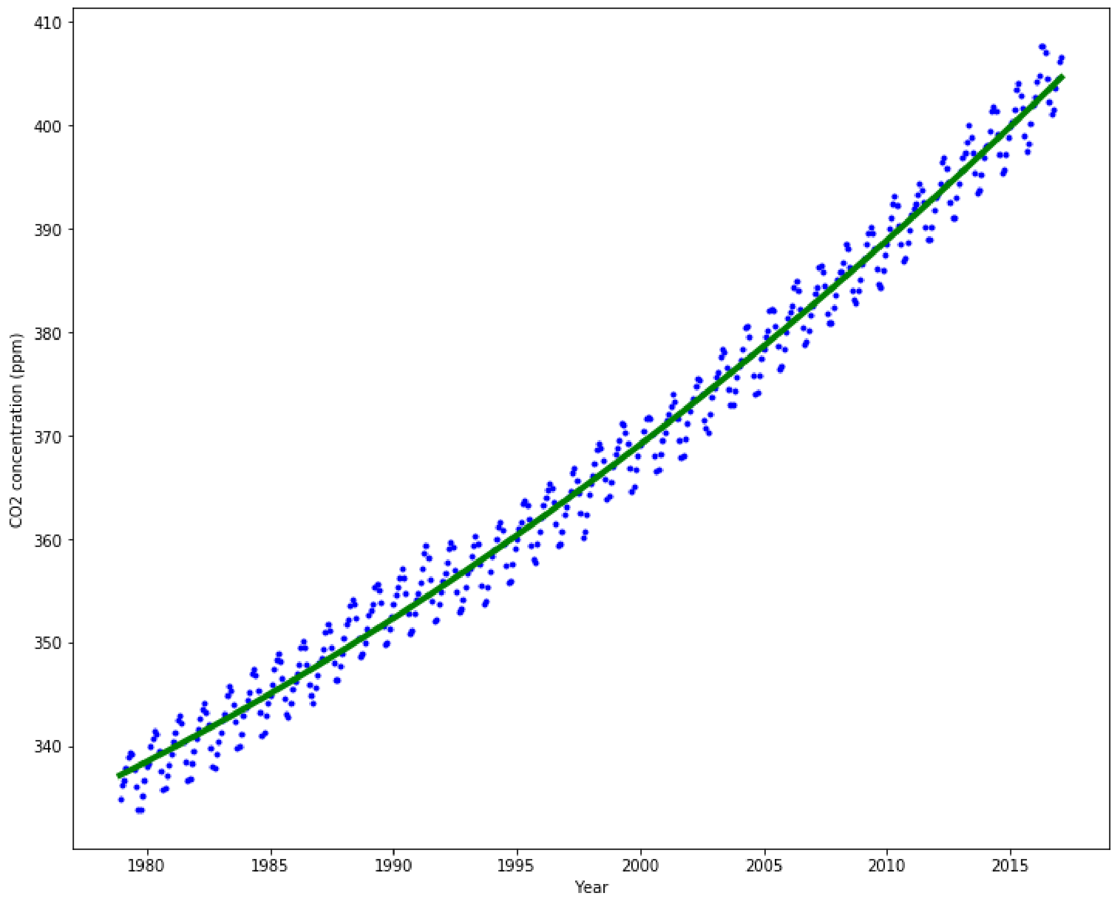
Print polynomial coefficients (a, b, c) for CO2 function:

In [12]:
```
print(co2_p2)
```

```
[ 1.10288786e-07 -1.56050409e-01  5.55124342e+04]
```

Plot CO2 polynomial function against data:

In [13]:
```
fig = plt.figure(figsize=(12,10))

ax = fig.add_subplot(111)
ax.plot(df.index, df['CO2'], 'b.')
ax.plot(df.index, np.polyval(co2_p2,x), color='green', lw=4)
ax.set_xlabel('Year')
ax.set_ylabel('CO2 concentration (ppm)')

plt.show()
```



In [14]:
```
temperature_p2 = np.polyfit(x, df['Temperature'],2)
```

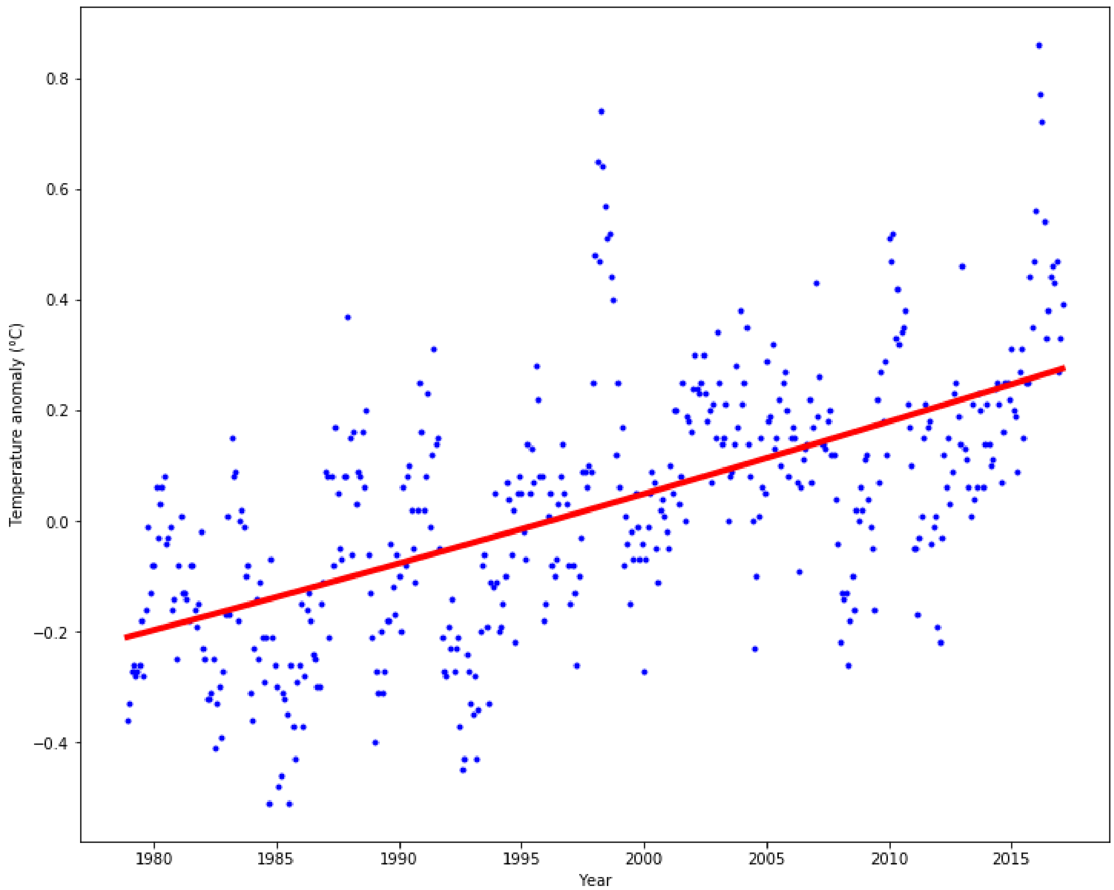
Print polynomial coefficients (a, b, c) for temperature function:

In [15]:
```
print(temperature_p2)
```

```
[ 1.94148069e-10 -2.48439667e-04  7.79439496e+01]
```

Plot temperature polynomial function against data:

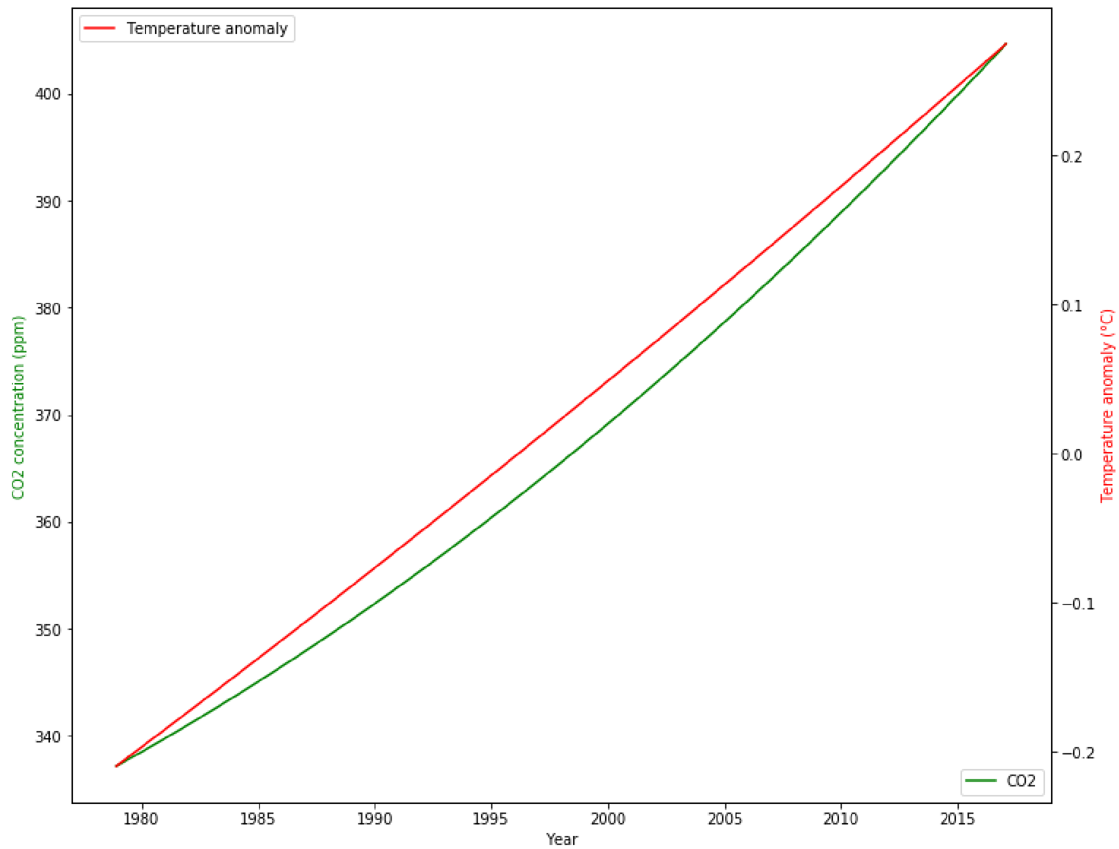
In [16]:
```
fig = plt.figure(figsize=(12,10))

ax = fig.add_subplot(111)
ax.plot(df.index, df['Temperature'], 'b.')
ax.plot(df.index, np.polyval(temperature_p2,x), color='red', lw=4)
ax.set_xlabel('Year')
ax.set_ylabel('Temperature anomaly (řC)')

plt.show()
```



Plot both CO2 and temperature polynomial functions on same graph:

In [17]:
```
fig = plt.figure(figsize=(12,10))

ax = fig.add_subplot(111)
ax.plot(df.index, np.polyval(co2_p2,x), color='green')
ax.legend(['CO2'], loc=4)
ax.set_xlabel('Year')
ax.set_ylabel('CO2 concentration (ppm)', color='green')
```

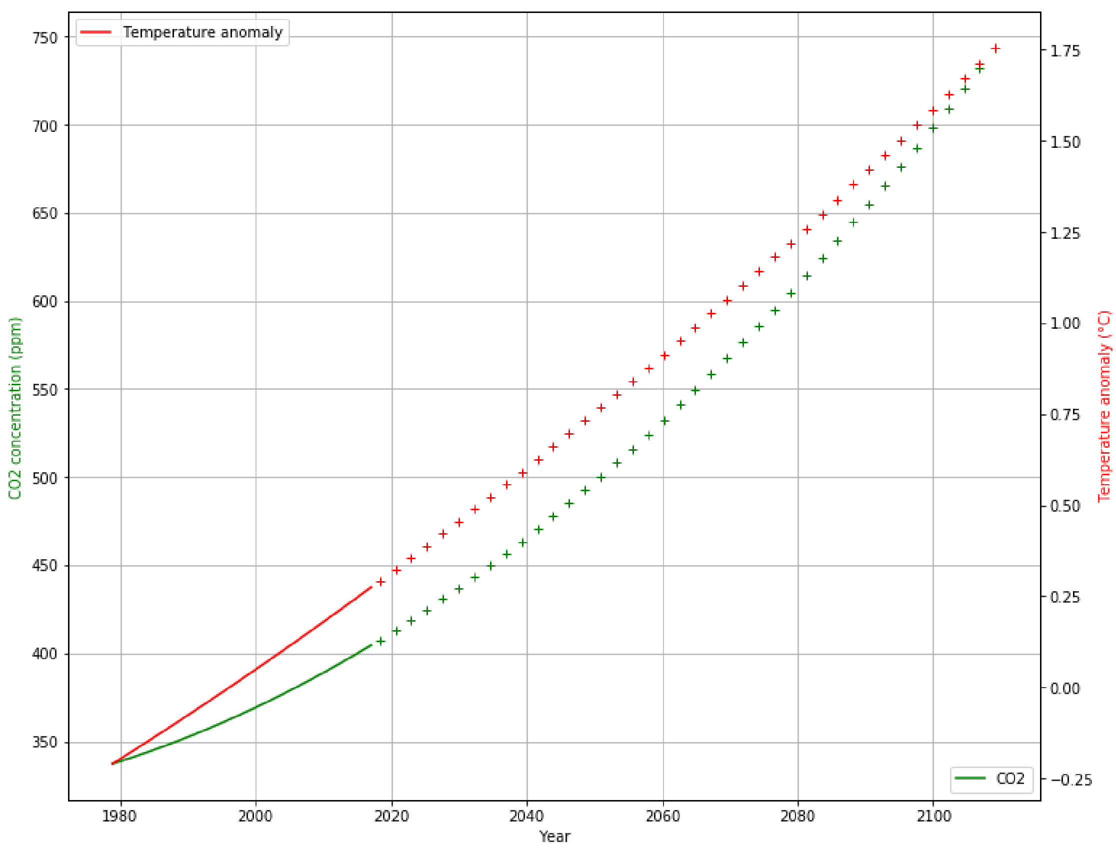
```
ax2 = ax.twinx()
ax2.plot(df.index, np.polyval(temperature_p2,x), color='red')
ax2.legend(['Temperature anomaly'], loc=2)
ax2.set_ylabel('Temperature anomaly (řC)', color='red')

plt.show()
```



We will now extrapolate polynomial functions and plot extrapolate data. Here, we assume that we will see the same exponential rate of CO_2 increase in the future and want to find expected temperature response for such increase in CO_2 concentration.

```
In [18]: fig = plt.figure(figsize=(12,10))

         ax = fig.add_subplot(111)
         ax.plot(df.index, np.polyval(co2_p2,x), color='green')
         ax.legend(['CO2'], loc=4)
         ax.grid(True)
         ax.set_xlabel('Year')
         ax.set_ylabel('CO2 concentration (ppm)', color='green')

         for x1 in np.linspace(736800, 770000, 40):
```

```
    xdate = mdates.num2date(x1)
    xdate = xdate.replace(tzinfo=None)
    plt.plot(xdate, np.polyval(co2_p2,x1), 'g+')

ax2 = ax.twinx()
ax2.plot(df.index, np.polyval(temperature_p2,x), color='red')
ax2.legend(['Temperature anomaly'], loc=2)
ax2.set_ylabel('Temperature anomaly (řC)', color='red')

for x1 in np.linspace(736800, 770000, 40):
    xdate = mdates.num2date(x1)
    xdate = xdate.replace(tzinfo=None)
    plt.plot(xdate, np.polyval(temperature_p2,x1), 'r+')

plt.show()
```




```
In [19]: fig = plt.figure(figsize=(12,10))

         ax = fig.add_subplot(111)
         ax.plot(df.index, np.polyval(co2_p2,x), color='green')
```

```
ax.legend(['CO2'], loc=4)
ax.grid(True)
ax.set_xlabel('Year')
ax.set_ylabel('CO2 concentration (ppm)', color='green')

for x1 in np.linspace(736800, 850000, 40):
    xdate = mdates.num2date(x1)
    xdate = xdate.replace(tzinfo=None)
    plt.plot(xdate, np.polyval(co2_p2,x1), 'g+')

ax2 = ax.twinx()
ax2.plot(df.index, np.polyval(temperature_p2,x), color='red')
ax2.legend(['Temperature anomaly'], loc=2)
ax2.set_ylabel('Temperature anomaly (řC)', color='red')

for x1 in np.linspace(736800, 850000, 40):
    xdate = mdates.num2date(x1)
    xdate = xdate.replace(tzinfo=None)
    plt.plot(xdate, np.polyval(temperature_p2,x1), 'r+')

plt.show()
```