Introduction

Are you a climate change denier or believer? This assignment requires you to do simple (and primitive) analysis on London weather data from 1941 to 2013. Is the seasonal weather in London becoming warmer? Is there more or less seasonal snowfall, snow on the ground, precipitation and rain? You are given the weather data, constructed mostly from a combination of data measurements made at the London International Airport (1941-2005), from an automated weather station (2006-2013) and from a volunteer (2003-2006). This data is contained in a mat file \texttt{london\_weather\_1941\_2013.mat}, available on the course webpage.

This file has the following data: year, month, element and data for up to 31 days (data element 32 specifies the number of days), saved in a 4D array, named \texttt{climate}. The data is for years 1941 to 2013, 12 months a year (29 days in February in leap years), 7 data elements (types of data) followed by the data for the specified element and month for some year. There are 7 data elements stored for most days:

1. max daily temperature degrees (0.1C)
2. min daily temperature degrees (0.1C)
3. ave daily temperature degrees (0.1C)
4. total rainfall (0.1mm)
5. total snowfall (0.1mm)
6. total precipitation (0.1mm)
7. snow on the ground (1.0cm)
Sometimes data elements are missing (not recorded or lost). In this case, nan (not a number) was recorded as the value. Note that you cannot use nan numbers in your calculations (any calculation with nan is nan). Thus, to read and compute the mean and standard deviation of the temperature data for a particular month and year, we would use:

% read the number of days in the month for year
LAST=climate(year,month,3,32)
% Read 1:LAST elements of data.
temperature_data(1:LAST)=squeeze(climate(year,month,3,1:LAST));
% copy only non-nan temperature data to the same variable
temperature_data=temperature_data(~isnan(temperature_data));
% average the non-nan data and save on array average_data.
% The temperature element is 1 in this array
monthly_mean_data(year,month,1)=mean(temperature_data(:));
monthly_std_data(year,month,1)=std(temperature_data(:));

Do the following three tasks for this assignment:

1. Task 1: Compute the average seasonal temperature, rainfall, precipitation, snowfall and snow on the ground values (elements 3, 4, 5, 6 and 7). The 4 seasons each year are fall (Sept. 22 to Dec 21st), winter (Dec 22nd to March 21st), spring (March 22nd to June 21st) and summer (June 22nd to Sept. 21st). You will need to compute the seasonal averages for elements 3, 4, 5, 6 and 7 for each year for your program. Note that the winter season spans two years. So winter 1942 data is from Dec 22nd, 1941 to March 21st, 1942. Thus you have to use:

    season1=squeeze(temp_data(year-1,december,element,...
                        first_day_of_season:days_in_month(december)));

    to get the data from December year-1 for the winder average for year/ January’s data is simple to compute:

    season2=squeeze(temp_data(year,january,element,1:days_in_month(january)));

    while the data for February depends on whether the year is a leap year or not:
season3=squeeze(temp_data(year,february,element,... 1:(days_in_month(february)+leapyear(year+base_year))));

Here **leapyear** is a MatLab function that returns true (1) if the year argument is a leap year and false (0) otherwise. Of course the March data can be obtained as:

season4=squeeze(temp_data(year,march,element,1:last_day_of_season));

The variables **first_day_of_season** and **last_day_of_season** are 22 and 21 respectively. Of course, **season1** to **season4** may have **nan** values and these must be removed from the season vectors. A column vector can be constructed from **season1** to **season4** and used to compute the mean and standard deviation values.

2. Task 2: Perform 1st order (linear) regression on these seasonal average datasets (using **polyfit** and **polyval**). For each month for the years 1942 to 2013 plot these curves as solid coloured lines.

Use colors: red (temperature), green (rainfall) blue (snowfall), cyan (ground snow) and magenta (precipitation) for your plots. Variable **plot_colors** has already been set in **ass4_shell_2016.m**. Use variable **plot_color** with these colour values for element indices of 1 to 5 in your plot command. Use **axis([1942 2013 -20 35])**; to fit the plots to single axes.

Note that the command:

```matlab
set(0,'defaultlinelinewidth',2.0)
```

at the beginning of the shell program sets the default width of all plotted lines to 2.

You should use **polyfit/polyval** and **plot** to generate and display these graphs. Use **hold on** and **hold off** as necessary. Each graph will be for one of the seasons for the 73 years from 1942 to 2013. Yes, winter overlaps a year boundary. For example, to compute the winter average temperature for 1960 we use all non-nan data from Dec 22nd, 1959 to March 21st, 1960.

Each season graph will have 5 straight lines, one for each element. Also, the graphs should have legends, telling the viewer what each line is for. Note that precipitation
is a weighted average of rainfall and snowfall, where 1.0cm of snow is counted as equivalent to 0.1cm of rain. Be aware that sometimes different element values can be the same. For example, for summer, there is no snowfall or snow on the ground (so these are always 0.0) and since there is no snow, the precipitation is completely due to rain (and so the two are always equal). Since identical lines are plotted on top of each other, the colour for such a set of lines will be that of the last line drawn. Figure 1 shows a possible graph for

![Graph of Winter Seasonal Weather Data](image)

Figure 1: Straight line fits of the average temperature, rainfall, precipitation, snowfall and snow on the ground for 1941 to 2013 for the Winter season.

3. Task 3: Plot out the actual data for each year as an error plot. Use the standard deviation values you computed when you were computing the means as the error bars. Use the error bar form `errorbar(X, Y, E)` where X and Y are the x and y axes (the year
versus the polyvals for those year as computed using polyfit) and \( E \) is the corresponding standard deviation data. Figure 2 shows the winter plot. Notice that there is a lot of variable in the adjacent yearly data for all the seasons. In these plots, let MatLab determine the \( y \) axis values (do not use \texttt{axis}).

![Error bars for Winter: 1942 to 2013](image)

**Figure 2**: Error bar plots of the average temperature, rainfall, precipitation, snowfall and snow on the ground for 1941 to 2013 for the Winter season.

You are provided with a shell of the program you must write, \texttt{ass4\_shell\_2016.m}, on the course webpage. This program has many useful predefined variables and its read the initial climate data for you. Fill in your code in the 3 places indicated to complete this assignment.