Temperature analysis

MNXB01 Project

Alexander Ekman, Jessamy Mol, Emelie Olsson, Shi Qiu

1 Introduction

Temperature measurements are made all over the world every day. The data is used for a lot of different purposes, e.g. weather broadcasts, but the one we all think about is probably confirming global warming and the greenhouse effect.
SMHI (Swedish Metrological and Hydrological Institute) measures the temperature on various locations. This data can be found on their website [1]. The temperatures are measured between one and three times per day. Several locations has datasets that started about 60 years ago (e.g. Lund), but the Uppsala dataset contains average daily temperatures since 1722. In this report we analyze this data in ROOT and present the results. These results are: the temperature of a given day, and the beginning of the seasons, the warmest and coldest day of each year, the extrapolation of the mean temperature of each year .
As mentioned above, ROOT is used for the data analysis, which is a scientific software written in C++. Our code is structured in several different parts, with a class that contains all our functions. The common functions are: read data (described in appendix A), print data, detect leap years and one that returns the day number given the date. In addition, each of the four produced results has a separate .cpp-file.

2 The temperature of a given day

Knowing the temperature of a given day does not confirm global warming by itself, but it can be interesting with more detailed data on certain days for other reasons. For example, is the weather the same on midsummer and christmas?
Two functions were created, both giving the temperature of a certain day. The difference is that one takes the date (month, day) as input and the other one takes the day number as input. To make it as simple as possible, the function that takes day number as input first coverts it into month and day, so that the functions thereafter are identical.
The algorithm itself is straightforward. First, the dates in the data that matches the input date are found. Then, the temperatures for all years are stored in a vector, which is plotted in a histogram. This process is shown in a flow chart in Fig. 1.
A histogram for the 19^{th} of July in Uppsala, for the years from 1722 to 2013, is shown in Fig. 2. From the histogram it is possible to get both the mean and the standard deviation. For Fig. 2, the mean is 16.88 and the standard deviation is 2.96. If we want to know the probability for a certain temperature on the given day, we can use the mean and the standard deviation and

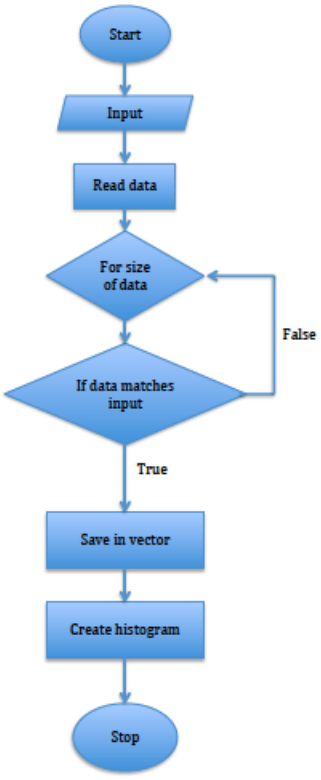


Figure 1: Flow chart for the function that gives the temperature of a given day.

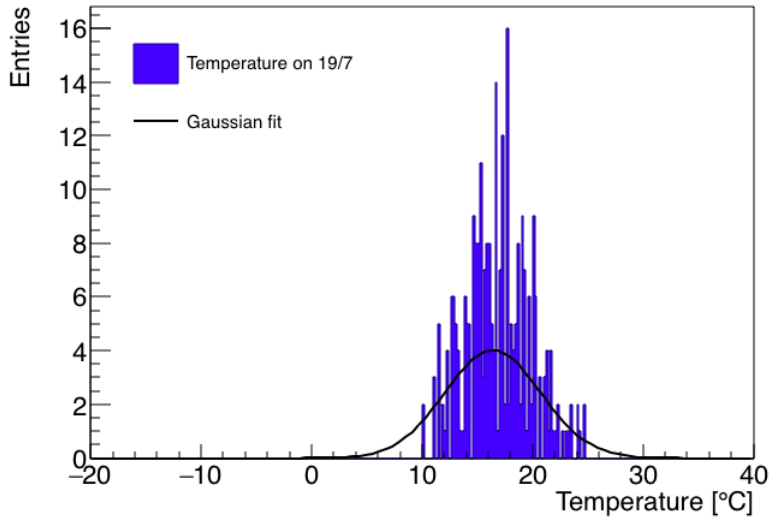


Figure 2: The temperatures on July 19$^{\text{th}}$ in Uppsala for the years between 1722 and 2013. The black line is a Gaussian fitted to the histogram.

assume a Gaussian distribution. The black line
in Fig. 2 is a Gaussian fitted to the histogram, to see if it is a reasonable assumption. For Uppsala, the data contains enough years to give 274 counts, so the Gaussian fit should be sensible. This is not true, given that the fit has a χ^2 of 2, which is not good at all. Some of the other data sets would give a lot less data points, as in Fig. 3. The histogram contains only 35 entries for the 19$^{\text{th}}$ of July in Lund, and the Gaussian fit is definitely not a good approximation. Therefore, the conclusion is that it is not possible to calculate the probability of a certain temperature by approximating with a Gaussian.

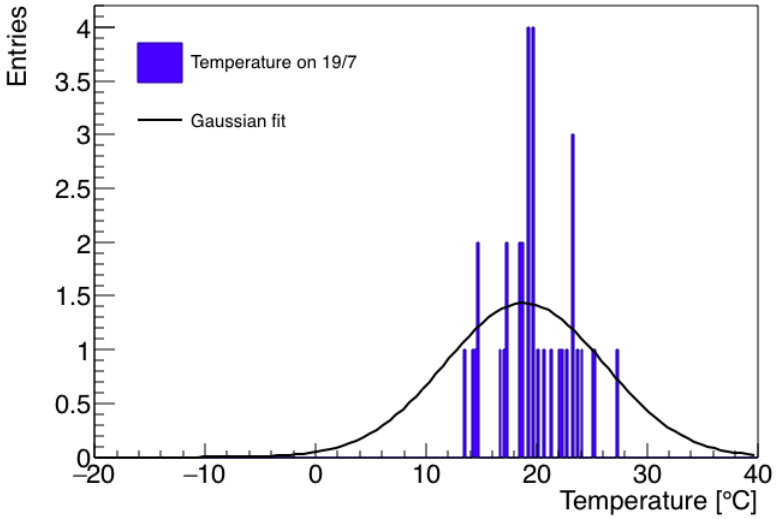
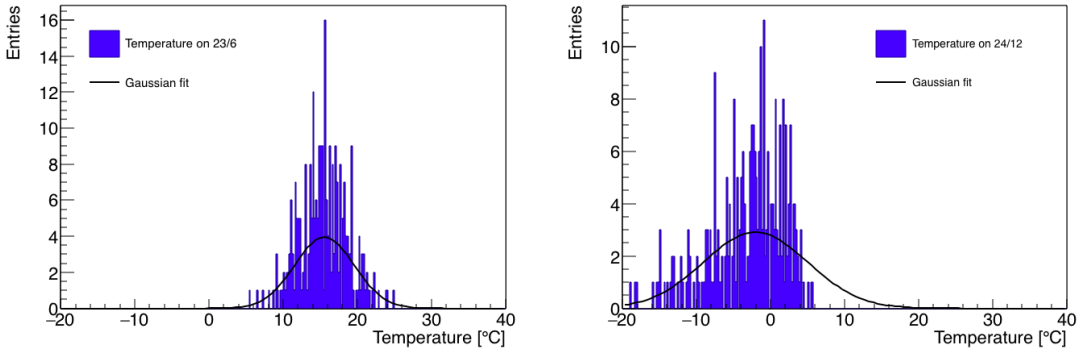


Figure 3: The temperatures on July 19$^{\text{th}}$ in Lund between 1961 and 2014. The black line is a Gaussian fitted to the data.

Midsummer is celebrated in Sweden on a day between June 19$^{\text{th}}$ and June 26$^{\text{th}}$. Historically, beginning in the 4th century, Midsummer was celebrated on the 24$^{\text{th}}$ of June. Therefore, June 23$^{\text{rd}}$ is chosen for the comparison since we celebrate on midsummers eve (we now celebrate on the Friday between June 19$^{\text{th}}$ and June 26$^{\text{th}}$). The weather on midsummers eve is compared to the

weather on christmas eve, both using the dataset from Uppsala, seen in Fig. 4. Clearly there is a large difference in the temperatures on midsummer and christmas eve. The mean for midsummers eve (Fig. 4(a)) is 15.4 °C and the mean for christmas eve (Fig. 4(b)) is -3.1 °C.



(a) Midsummers eve

(b) Christmas eve

Figure 4: The temperatures on midsummers eve (a) and christmas eve (b), in Uppsala, for the years between 1722 and 2013.

3 The beginning of the seasons

A recent article by SVT Nyheter showed which season is was on the 1st of November for different areas in Sweden, as can be seen in Fig. 5. According to this article it was already winter in the north of Sweden while in the most south part, including Lund, it was still summer. This did not seem to correspond to what one might observe when looking out the window in Lund, the ground was already covered with red leaves even though it was still summer according to SVT Nyheter. So which definitions of the season are there? Using one of these definitions when do the seasons start in Lund and how does the start of each season vary over a number of years?

Figure 5: The season in different areas in Sweden on the 1st of November. Image from [2].

3.1 Definition of the Seasons

Sveriges meteorologiska och hydrologiska institut (SMHI) uses two different definitions of the seasons, a definition based on the calendar and a meteorological definition [3]. According to the definition based on the calendar spring is always from March to May, summer from June to August, fall from September to November and winter from December to February. This is clearly the same for each year and is not dependent on temperature of any other parameter. The

meteorological definition of the seasons is based on the average temperature of each day and the seasons can therefore be start on different dates each year. The meteorological winter starts when the average temperature of a day is equal to or below zero for 5 days in a row. The first day of winter is then the first of these 5 days. The definition for the meteorological summer is similar, but the average temperature needs to be 10 degrees or higher. The meteorological spring starts when the average temperature of a day is between zero and 10 degrees for 7 days in a row. The definition for the meteorological fall is similar, but the temperature only need to be between zero and 10 degrees for 5 days in a row. The seasons do of course need to occur in the right order, which means that spring has to start after winter and fall has to start after summer. SMHI's meteorological definition of the seasons will be used for the data analyses.

3.2 Method

The temperature data of Lund from Sveriges meteorologiska och hydrologiska institut (SMHI) will be used to determined to beginning of the meteorological season for the years in the dataset. The code used to produce the results on the beginning of the seasons can be found in `tempSeasons.cpp`, a number of member functions defined in `tempTrender.cpp` are also used. The main function, which calls all other functions, of the section is `tempTrender::startDaySeasons`. The first thing this function does is reading the data on Lund into a 2D vector with the use of `tempTrender::readData`. Since the meteorological definition of the seasons depends on the average temperature of a day, this is then calculated for every day in the dataset by `calcAverageTemp`. Using the resulting average temperatures each first day of a season is determined for all seasons and years in the dataset. This is done using the function `tempTrender::beginningWinterSummer` for both winter and summer and `tempTrender::beginningSpringFall` for both spring and fall. These functions are member variables since they use a member funtion `tempTrender::getDayOfYear`. The general flowchart of how each first day of all seasons is determined can be seen in 6. With the right temperature the temperature condition for a certain season is meant, which is smaller or equal to 0 degrees for winter, greater or equal to 10 degrees for summer and between 0 and 10 degrees for spring and fall. With the right day the condition on the number of subsequent is meant, which is 7 days for spring and 5 days for all other seasons. When a new season starts differs a lot between the seasons, for summer the only condition is that it occurs in a different year than the previous summer, because there is one summer per year. For spring and fall there is another condition that also needs to be satisfied, namely spring needs to occur after the first day of winter and fall needs to occur after the first day of summer. Because winter can cover part of the end of one year as well as the beginning of the next, winter does not always start once per year. There are four ways the first day of two consecutive winters can succeed each other.

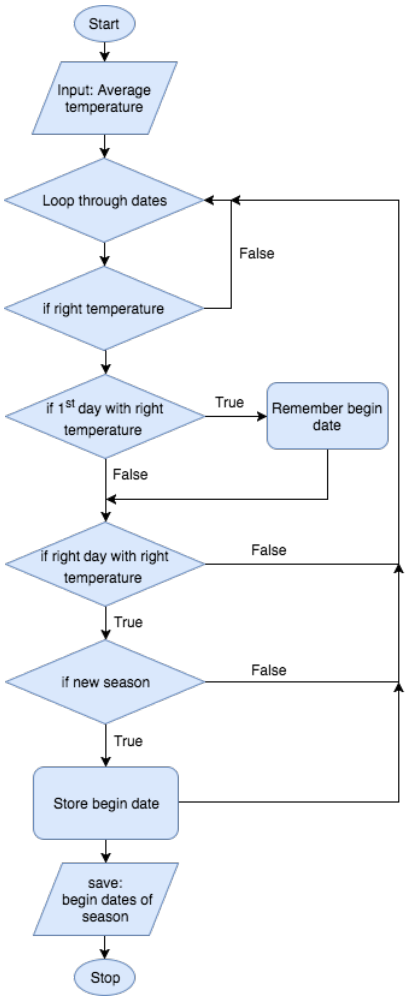


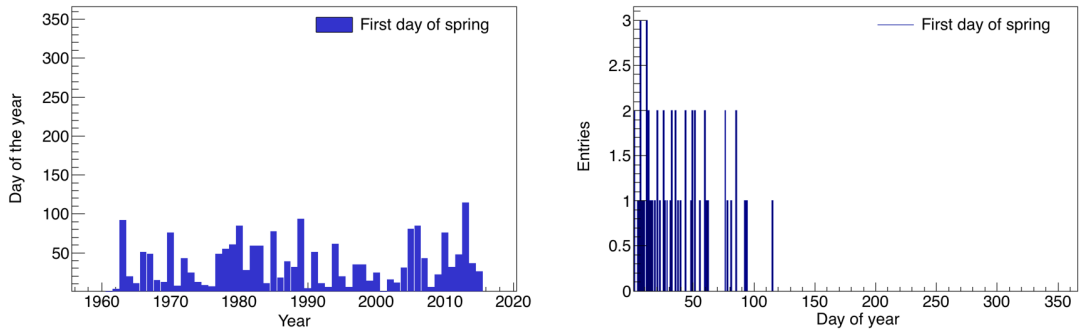
Figure 6: The code used for the calculation of the start of a season represented in a flowchart.

1. The successive winters are both early in the year (before June)
2. The successive winters are both late in the year (after June)
3. The successive winters are in the same year (one early in the year, the other late)
4. The successive winters skip a year (one winter late in the year, the next early)

If one of these cases is satisfied it is a new winter. Using the result for each season a bar graph showing the first day of the season as a function of the year is created by `plotSeason`. Histograms of the results are also created for each season using `histogram`.

3.3 Results and Conclusion

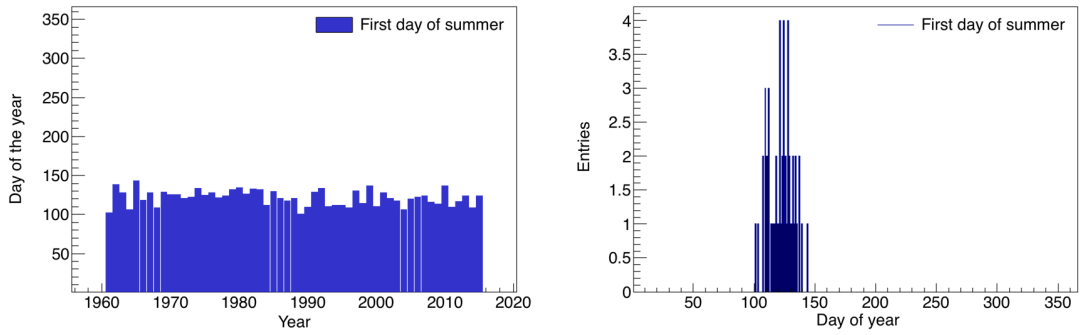
The first day of a season is plotted in a bar graph as a function the year. The date of the beginning of the season is represented by the day of the year (i.e. from 1 to 365 or 366 for a leap year). A histogram with the first days of the season is also plotted. The result for spring can be seen in Fig. 7.



(a) Bar graph **(b)** Histogram

Figure 7: The first day on which spring starts for each year in Lund is shown in (a). While (b) shows the number of times spring starts on a certain day in the year.

There is no clear trend visible in the bar graph, clear increase or decrease as a function of year. There is however quite a large difference between the start of spring for different years. In the histogram this spread can also be seen. Spring is most frequently at the beginning of the year.

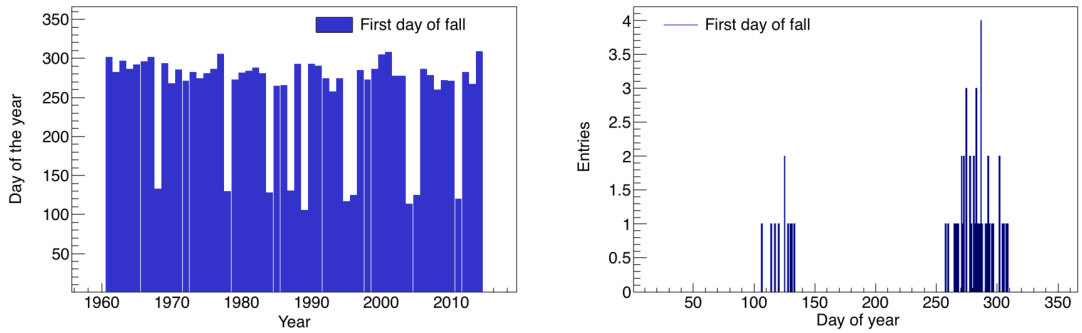


(a) Bar graph **(b)** Histogram

Figure 8: The first day on which summer starts for each year in Lund is shown in (a). While (b) shows the number of times summer starts on a certain day in the year.

The results for summer are shown in Fig. 8. Again there is no trend in the first day of summer as a function of year. It is however interesting that the first day of summer does not vary much per year, it is very consistent. This can also be seen in the histogram, which shown a fairly narrow peak.

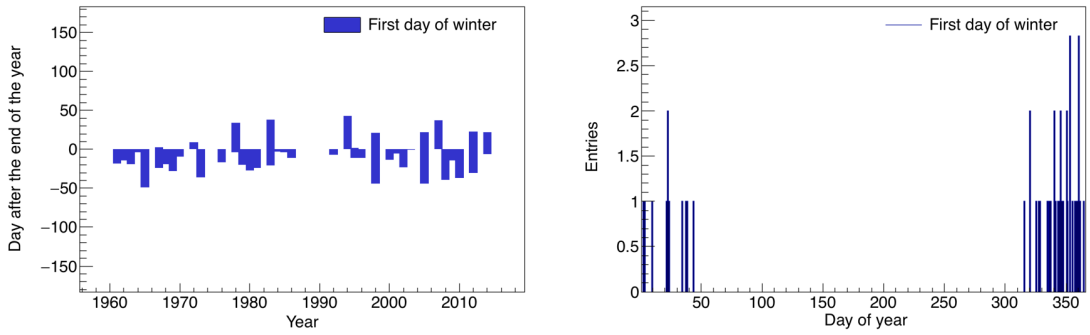
Fig. 9 shows the result for fall. The bar graph shows a strange spread in the first day of fall which is not clearly correlated to the year. The histogram shows the peculiar divide of the first day of fall in two peaks, one between the 100^{th} and 150^{th} day and the other at the 250^{th} to the



(a) Bar graph **(b)** Histogram

Figure 9: The first day on which fall starts for each year in Lund is shown in (a). While (b) shows the number of times summer starts on a certain day in the year.

300^{th} day of the year.



(a) Bar graph **(b)** Histogram

Figure 10: The first day on which winter starts for each year in Lund is shown in (a). The day is given relative to the start of a new year, all negative numbers are before the 1^{st} of January and all the positive numbers are on or after the 1^{st} of January. The number of times spring starts on a certain day in the year in given in (b).

The bar graph and histogram for winter are shown in Fig. 10. The bar graph shows the first day of winter relative to the start of the year, negative numbers are at the end of the year while positive numbers correspond to a day at the beginning of the year. There is no clear trend visible. However the graph shows that not every year has a winter according to the used definition, only 44 winters where found in 54 full years. The histogram shows that winter occurs at most frequently at the end of the year.

4 The warmest and coldest day of a year

A useful information of the study of the temperature in Sweden is to look for the warmest and the coldest day at different locations in Sweden. The warmest and coldest date fluctuates every year, while it is assumed that its overall distribution is Gaussian-like. In this section, we use provided data from all locations to explore the distribution of the warmest and the coldest day for each year.

4.1 Method

In order to get the highest and the lowest temperature, the temperature data from the same year is scanned through. At the first day of a year, the minimum and maximum temperature is assigned to the temperature of the first day. If the temperature of the second day is either larger or smaller than the maximum or the minimum temperature, the maximum or the minimum temperature is updated, respectively. By comparing the temperature of consecutive days with the constantly updated minimum and maximum temperature, the warmest and coldest day of a year can be determined. The complexity of such algorithm is linear with respect to the data size. As the date is stored in the normal calendar format, year-month-day, it is necessary to convert it into the day of the year so as to quantify the date. This is done by creating the `getDayOfYear` function in `tempTrender.cpp`, which takes into account of leap years.
The distribution of the warmest and coldest day of a year are plotted in the same histogram and are fitted to a Gaussian. It can be seen in the result section below that the distribution of the coldest day of a year ranges from the end of November to the start of March, so that the histogram is not continuous, but scattered at the two ends of the histogram. In order to fit a Gaussian to it, which joins the two ends, two histograms for the coldest day are created and each one is filled with the complete data of the coldest day, as shown in Fig. 11.

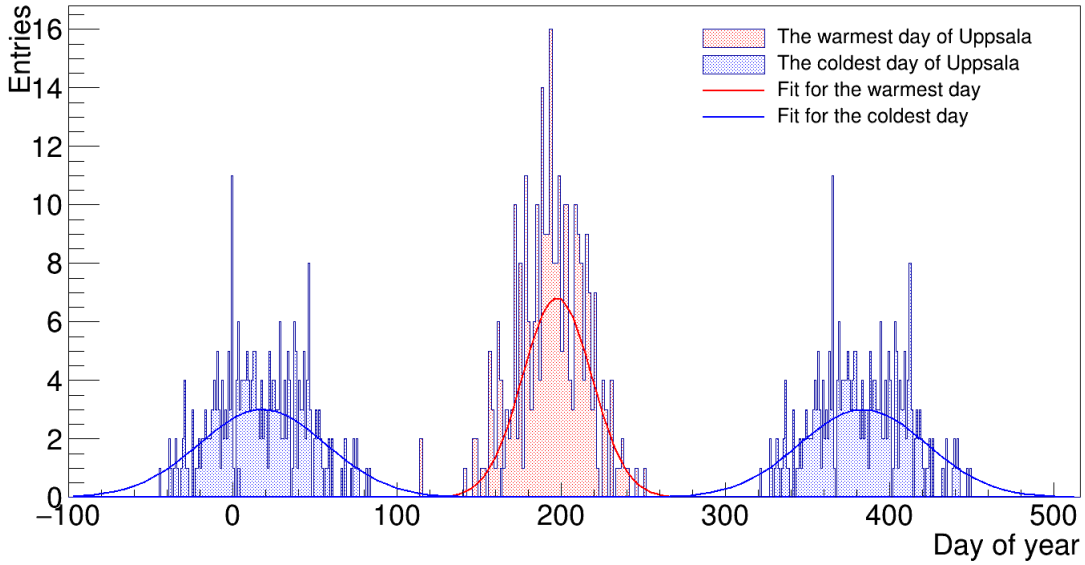


Figure 11: A demonstration of the method we used to fit the distribution of the coldest day of a year.

4.2 Results

The distribution of the warmest and the coldest day for Uppsala, Lund and Lulea are shown in Fig. 12, 13 and 14, respectively. The number of years in Uppsala data is around six times more than that in other data files. This gives a lot more entries to the plot of Uppsala, so that its Gaussian fit has the best χ^2 value. Since other data files generally only have 50 to 60 entries in total, the bin number is reduced from 366 for Uppsala to 50. Due to insufficient number of data, the χ^2 for other locations other than Uppsala generally varies significantly with the choice of the number of bins. We pick Lund and Lulea as two sample places to show the plot of the warmest- and coldest-day distribution, with Lund corresponding to one of the places at the lowest latitude and Lulea corresponding to the place at the highest latitude from the data provided. The results of the mean of the warmest and the coldest with corresponding uncertainty is summarized in Table 1. The order of the place in the table is chose to start from low latitude to high latitude. However, there is no clear relation between the mean value of the warmest and coldest day to the latitude.

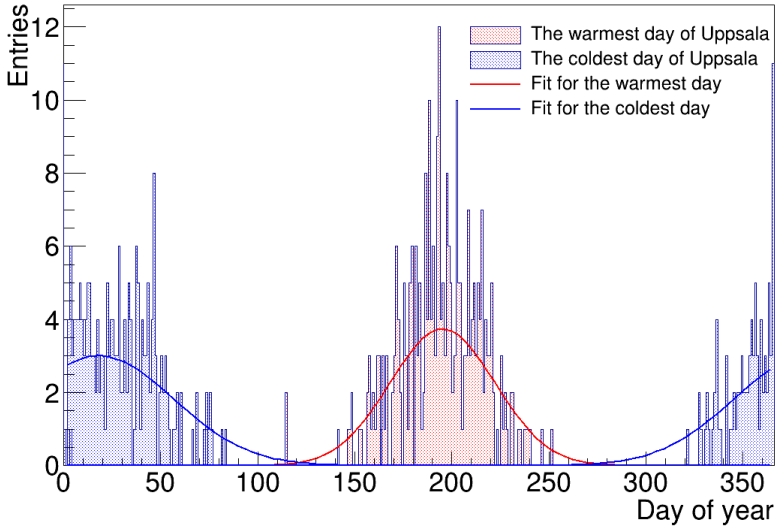


Figure 12: The distribution of the warmest and the coldest day for each year in Uppsala from 1722 to 2013. A Gaussian function is fit to each of the distribution with χ^2/dof determined to be 0.90. Note that the temperature for some years was not recorded or partially recorded for Uppsala. Only the data that are strictly from Uppsala are read by the `readData` function.

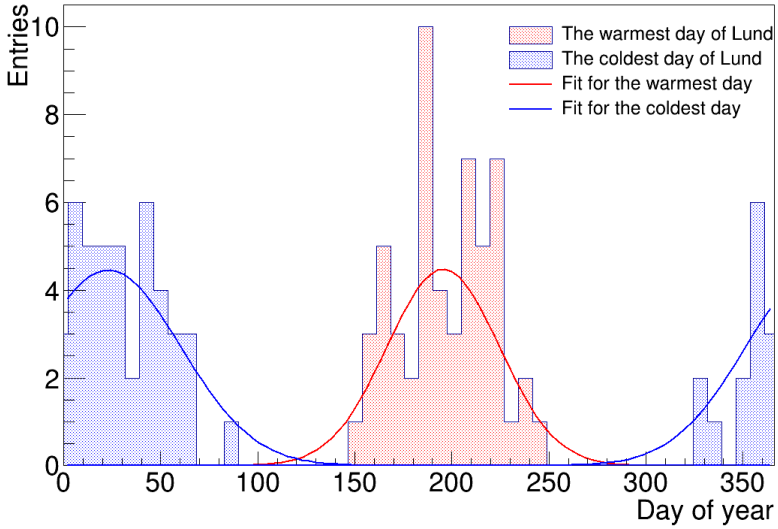


Figure 13: The distribution of the warmest and the coldest day for each year in Lund from 1961 to 2015. A Gaussian function is fit to each of the distribution with χ^2/dof determined to be 1.18.

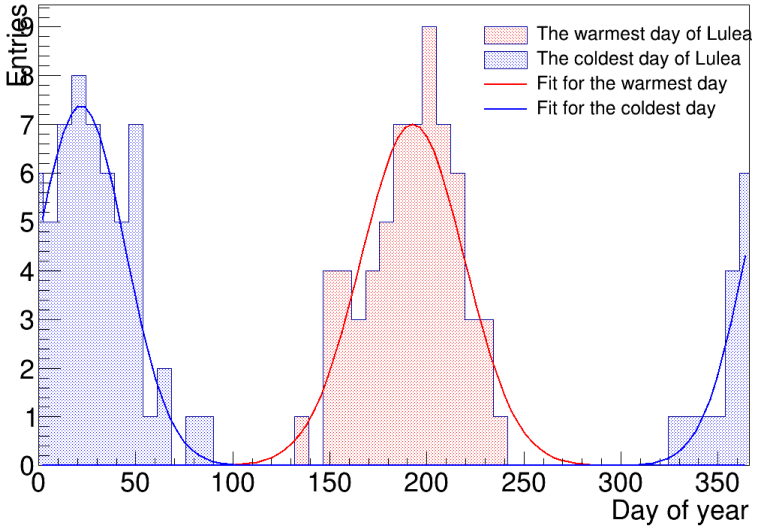


Figure 14: The distribution of the warmest and the coldest day for each year in Lulea from 1951 to 2015. A Gaussian function is fit to each of the distribution with χ^2/dof determined to be 0.37.

	Mean of the warmest day	Uncertainty of mean of the warmest day	Mean of the coldest day	Uncertainty of the mean of the coldest day
Falsterbo	198.9	3.8	28.6	5.4
Lund	195.5	5.7	22.9	4.9
Visby	197.8	4.0	44.3	9.0
Boras	188.6	4.9	29.0	13.4
Soderarm	209.1	2.9	34.6	3.7
Karlstad	199.9	4.8	25.5	11.4
Uppsala	195.1	2.4	17.9	3.7
Falun	196.8	4.8	19.1	7.0
Umea	200.1	6.0	29.6	4.7
Lulea	192.7	4.4	21.8	3.2

Table 1: Calculation of the mean of the warmest day and the coldest day from the Gaussian fit. The uncertainty for each mean value is evaluated. The order of the place is chosen to start from low latitude to high latitude.

5 Extrapolation of the mean temperature

Global climate change and the greenhouse effect is often a reocurring political subject. From 1722-2013, the Swedish Meteorological and Hydrological Institute (SMHI) has collected daily temperature measurements from uppsala, Sweden. This data will be used to show the distribution of the yearly average temperature in uppsala from 1723-2013. Further more, analysis of this data will provide a moving average of the yearly average temperature. The moving average will be fitted with a $a(x - 1840) \cdot cos(bx)$ function, trying to se if the future temperature is not only increasing, but actually oscillating more violently correlated to the human development since the industrial revolution in 1840.

5.1 Method

Dailytemperatures were read from raw data and put into a multi dimensional vector according to the readData function. After this was called from `tempExtrap` a simple for loop converted this

into a 2-D data vector containing year and temperature, with one entry for each day. This data vector was made in two different variants. One containing all values from the uppsala data set, assuming a measurement was taken for every day. Another containing only the measurements from the uppsala location.
The next step was to calculate the yearly averages from the data. For the uppsala only data, this was done by `averages1`. This iterated through all the elements of the data vector. If the year of the current element was not larger than the previous, then the temperature of the current element was added to a temperature sum and the number of days counted was incremented. When the current year became larger than the previous, the previous year was pushed into a dummy vector along with the average temperature. This dummy vector was then pushed into another vector, creating a 2D averages vector containing one element per year with that year's average temperature.
For the complete set of data, the first year, 1722, was skipped because its measurement started after the 1st of January. After 1722 the data vector was iterated in steps of the years with two nested for loops. In the outer loop, if the year was a leap year, an inner for loop iterated from day 1 to 366 and pushed the average temperature and the year into a dummy vector, into an averages vector. And similarly if the year was not a leap year, it was then iterated 365 times and the average was calculated appropriately.
With the data of yearly averages at hand, it was processed for plotting. To do this, the total average temeperature from all the yearly average temperatures was calculated. With a for loop iterating through the data elements and simple if statements, the average temperatures and corresponding year were pushed into one vector containing all temperatures above the total average, and one vector containing those below. These two new vectors are not used for further analysis, but purely used for plotting.
The average temperature vector was then used to calculate a moving average. The goal of this moving average is to take the yearly averages in groups and take the average yearly temperature for that group. This will make it easier to fit. Similarly ot the other average functions, this iterated through the average vector and for each element added the temperature. If the iteration counter was the size of the desired group size, then the average of the average temperatures was pushed into a vector for the y-values. For the x-values, the current year, the middle of the group, corresponded to the initial year, plus the number of groups so far times the group size. Plus half a group size. If the number of years is not divisible by the group size then the last years will not be represented. This is accomodated by outside the for loop push the current value of the sum.
These moving average y- and x-values were then put into a `TGraph` object, and a fit function was added to this graf. The function was of the form $A * (x - 1840) * cos(B * x))$, where A and B are parameters. The graph, the fit were then plotted together with the above and belov vectors, as bar graphs, mentioned earlier.
The main function `tempEx` then returned the value of the fit function evaluated at the desired year.

5.2 Results and Conclusion

The results show a somewhat periodically oscillating average yearly temperature around 5.9 Celcius. The fit achieves a $\chi^2 = 1.2$, which is surprisingly good for such a simplistic model. Extrapolating with this function, $0.005(x - 1840)cos(0.125x)$ gives an average temperature of 5.88 Celcius year 2050. In conclusion, the yearly temperatures do in fact oscillate over time. However, it is clearly seen in the data that at in the more recent years have been concistently warmer and a more proper model, using global data could most likely show proof of global warming. It is interesting to see that such a simple model as ours does make a good fit with the data. It is also interesting that such a temperature increase can be seen in data just obtained in the Stockholm region.

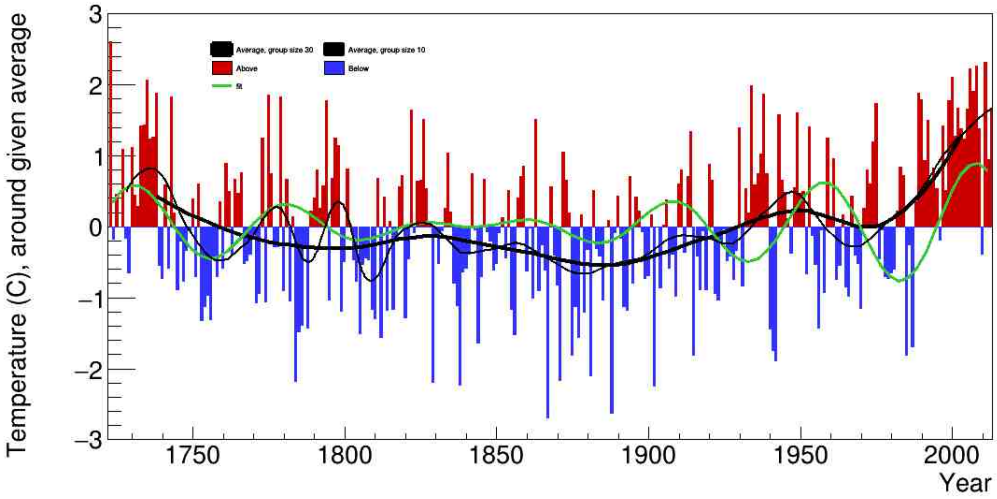


Figure 15

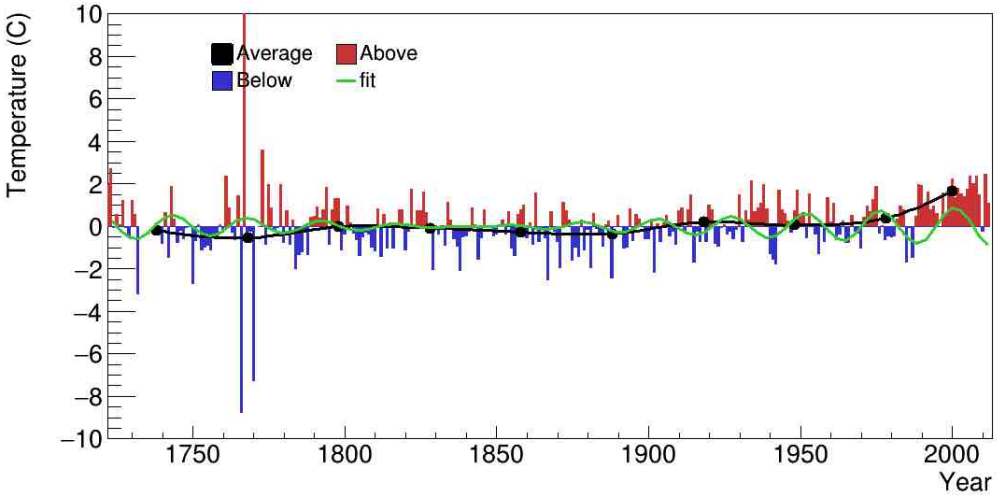


Figure 16

A Read data function

The temperature data are provided for various locations, including Falsterbo, Lund, Visby, Boras, Soderarm, Karlstad, Uppsala, Falun, Umea and Lulea, in Sweden. The period of the record ranges from 1951 or 1961 to 2015 for all locations except for Uppsala. Uppsala data has a much longer record of the temperature, which starts from 1722 and ends at 2013 for the data provided. It should be noted that the Uppsala data are not strictly from Uppsala and a small portion of the data is from nearby regions such as Stockholm, Betna and Risinge. The location information is recorded in the so called dataID column in the Uppsala, with 1 and 2 to 6 representing data recorded in Uppsala and other five locations, respectively.
To obtain the data from all locations except for Uppsala, we used `getline` function to read the data file line by line and chose delimiters in the correct order that corresponds to the data file, since the data are separated by different delimiters, such as "-" used to separate date, and ":" used to separate time. Each data file except for Uppsala has some text to describe the data file at the beginning. It can be skipped by using "Datum" as an indication of the start of the data since "Datum" always occurs above the row of actual data. The data are stored in a two dimensional vector.
Each column of the data from Uppsala is separated by a space. We can directly use `>>` to output the data for each row and pass it to a two dimensional vector. It should be noted that our general data output function, the one that can be used to read all data file, only includes temperature data strictly from Uppsala, while we have another separate read data function specifically for Uppsala data that read all data from it. The reason for writing another dedicated read data function for the entire Uppsala data is due to the need for the analysis of the mean temperature of each year.
A summary of the output format of the data read function is shown in Table 2. Since we want the output format from the general data read function being compatible for the same code, it is important to match the format of them. The Uppsala data do not have the time and air quality data as other data files. Therefore, the time columns are filled with fictitious data, "-1", and the air quality is filled with "-N". Note that some text occasionally occurs at end of some rows for data from all location except for Uppsala, which is inevitably recorded at the last column of the vector, but it has no effect on the data analysis.

	Format of the output data								
Data except for Uppsala	Year	Month	Day	Hour	Min	Sec	Temp	Air	Some text
	1961	01	01	12	00	00	0.4	G	Kvaliteskontroll...
Uppsala Data	Year	Month	Day	Hour	Min	Sec	Temp	Air	Temp (urban corr.)
	1722	1	12	-1	-1	-1	1.9	-N	1.8

Table 2: A summary of the output format of the data read function. The hour, min, sec and air quality data are not recorded for Uppsala. In order to match the same output format as other data, all three time data are filled with "-1" and the air quality is filled with "-N" for Uppsala. Note that Upssala has one more column of temperature data, the temperature with urban correction, which is put to the last column of the output data vector.

References

[1] Sveriges meteorologiska och hydrologiska institut (SMHI). Dataset. https://www.smhi.se/klimatdata/meteorologi/temperatur.

[2] SVT Nyheter. Vintern 2017-2018. https://www.svt.se/vader/vintern20172018. Accessed: 2017-11-09.

[3] Sveriges meteorologiska och hydrologiska institut (SMHI). Årstider. https://www.smhi.se/kunskapsbanken/meteorologi/arstider-1.1082. Accessed: 2017-11-09.