**Phys 5053 – Exercise 5**

**Discussion question: Lake Temperatures**

The course web page contains data on the surface temperature records of seven lakes (courtesy of Dr. Josef Werne, LLO). The temperature data were derived from the NASA satellite data collected for a 1 $km^2$ pixel in each lake. This being a real-word dataset, it is not perfect. The values listed as NaN represent missing values (due to cloud cover). A number of values are obviously outside of the reasonable range, most probably due to partial cloud cover at the time of the measurement. The task is to determine the average annual, December – March, and June – September temperatures for the lake, as well as their uncertainties, while accounting for the fact that many months may have very few data points. The data needs to be screened for outliers and a reasonable method needs to be applied for exclusion of outliers from the dataset. For example, in Lake Ginghai on the Tibetan Plateau, it is impossible to have temperatures of 8.4 $^oC$ in July (too cold) or 22.3 $^oC$ in March (too warm).

The 'Course' web page of the course web site contains several routines that will save you time.

The script '*LakeTemperature.m*' is essentially a solution to the problem (my way, you are certainly welcome to suggest yours). If you prefer to use it, make sure you understand what it does. For convenience, the file is split into Matlab cells. You can execute each cell separately to evaluate its results. The script uses several external functions that I wrote to make things easier and that can be used in other exercises as well.

*sergeiRemoveNaNs.m* removes the rows containing invalid data points (NaN)

*sergeiRemovOutliers.m* applies the Chauvenet's criterion (modify the criterion and see how this affects the results!).

*sergeiRunning.m* computes the running mean and variance.

These programming routines are written as Matlab '<u>functions</u>', i.e. they can be executed just like Matlab commands. For this to work, you need to copy the corresponding .m files into the Matlab program directory on your computer. Be sure to also open the files in Matlab's Editor and study them to understand what these scripts do.

This is a lot of programming, so let's proceed in steps:

1. Choose a lake for which you will perform the analysis and import the data into Matlab. My script assumes that the data will be stored in a 'data' variable. Look at the dataset and see why it is impossibly to do the analyses without a series of preparations.
2. The time vector is presented in an unwieldy year-month-day format. Convert it to something more mathematically tractable, e.g. by running the first cell of my script (which uses the Matlab '*juliandate'* command).

3. Plot the data as a time series. Which type of plot is best to plot this kind of data? This plot is an important starting point. You will be comparing your results against this plot.
4. Calculate the average temperature for your lake. Can you do this? What is the value? Does it make sense?
5. Eliminate the invalid data points, those pesky NaNs. If you are using my script and the function sergeiRemoveNaNs.m, try experimenting with criteria.
6. Plot the data again and compare against your first plot.
7. Re-calculate the average temperature.
8. For calculating the annual average, my script merges the data for multiple years into one, to have better statistics. Without looking into the script, think how you would do this? You can run my script if you prefer doing that as well.
9. Now, let's calculate the running mean and variance. (Matlab has a 'smooth' command that may be used for doing this but my script does not use it.). How running mean is different from the "ordinary" mean? Can you write a mathematical expression for the running mean? For the running variance?
10. Plot the running mean and compare it against the original data.
11. Now perform data rejection, e.g. the Chauvenet's criterion.
12. Plot the rejected data points and see if their rejection makes sense.
13. Re-calculate the running mean and compare with your previous result.
14. Re-calculate the "global" mean and compare with your previous results.
15. Plot your running mean plus/minus two sigma (or use my script to do that) and compare your result against the original data points. Do the results make sense? How can you improve the procedure?