Pre-Lab 2: Local Climate - UBC Weather Station Data

Lab Overview

This week, you will investigate some basic climate related variables as they were recorded right here at the UBC weather station. In this pre-lab, you will download the data, read a short introduction on specific matlab skills we will need, and complete the online quiz.

For submission

1. Online worksheet to be completed on Connect

1. Download UBC Weather Station Data

You can download any of the data recorded by the UBC Weather Station from the website located here: http://137.82.254.122/PlotTraces.aspx

Spend a little time playing around with the interface and looking at the options available to you. Notice which variables are available for you to look at and the date ranges you can select.

Download the maximum available **daily** time series for the variable *Air Temperature*. To do this, you'll have to change the resolution to "Day" and set the date range to be as large as possible.

This should give you one plain text file. Please bring it to the lab this week.

Make sure to download the data **before you come to the lab**; the website is not very powerful, and may go down if everyone attempts to download the data at the same time.

2. MATLAB Skills part 1: Mask, the *find()* function and NaNs.

There is nothing to submit for this section of the lab, but it will be helpful for you to read through.

In lab 1, we saw how to select or get rid of specific values, or ranges of values, using masks. An equivalent way to do this is to use the function `find()`. It is a very useful one to know how to use, and it will likely help you in the upcoming labs. Say you are given a vector `A` and you are interested in finding a specific value, for example `10`. You want to know where all the 'tens' in your vector are, and how many of them there are. To use the `find()` function you can enter

```
find(A==10)
```

which will return the index locations of all occurrences of `10` in the vector `A`. If you want to know how many 'tens' there are, you could enter

```
size(find(A==10))
```

which just gives you the size of the vector returned by first call, ie. the number of 'tens' in your vector. Remember that if you want to find all the values not equal to 10 in A, you can use the "~" symbol to negate a statement:

```
find(A~=10)
```

which will return the index locations of all numbers unequal to `10` in the vector `A`.

Note that `A==10` is a mask, i.e., it is the same size as A and contains the logical value 1 where `A==10` is true and 0 otherwise. The function `find()` transforms this vector of logical values into a list of indices, which are the indices for which the condition is true, i.e. where the mask is equal to 1. If you want to get these values, and not just their index, you can just pass the index list to the vector A:

```
A(find(A~=10))
```

Note that this is equivalent to

```
A(A~=10)
```

Searching for occurrences of `NaN` is slightly different. Because a `NaN` is not actually a number, you cannot use it in logical comparisons. So the expression `A==NaN` would return an error. In place of a logical equivalency, you can use the function `isnan()`. For example, to find the indices associated with each `NaN` in your array, you can use

```
find(isnan(A))
```

Similarly, to find the indices associated with each value that is not a NaN in your array, you can use:

```
find(~isnan(A))
```

Again, to get the values that are not NaNs, both `A(find(~isnan(A)))` and `A(~isnan(A))` will do the job.

Finally, remember that to combine two conditions, you can use the symbol “&” for AND and the symbol “|” for OR. For example, `A(~isnan(A) & A<0)` will return all values of A which are not NaN values and which are negative. `A(A>=0 | A==-1)` will return all positive values (not strictly, i.e. including 0) or values equal to -1 in A.

3. MATLAB Skills part 2: for loop.

This Section gives a brief introduction to the concept of “for loop” in matlab. Don’t forget to answer the online quiz to become a loop master!

In lab 1 part 2, we used a for loop to create a date vector associated the the MEI index, but did not really go over what loops are. The structure of a loop in matlab is always as follow:

```
for i = min:incr:max

    execute some command

end
```

A for loop always starts with “for” followed by the name of the loop index, which I called “`i`” in this case (but you can give it any name). The loop index will be varied from the value `min` to the value `max` with the increment `incr`. We will often use increment of 1 in which case you can simply write.
Below this line starting the loop, you can write any command needed. These commands will be executed for each value of the loop index.
`end` simply indicates where the loop ends.

For example, preallocate a 10-element vector x, and calculate five values:

```
x = ones(1,10);

for n = 2:6

    x(n) = 2 * x(n - 1);

end
```

Write down (on some paper) what you think the vector x should be after this loop has been executed. Then, execute the lines above on matlab and verify your answer. Do the same with the next example:

```
x = ones(1,6);

for n = 1:size(x,2)/2
```

```
    x(n) = n/3;
end
```

Last, we may have to use nested loop in matlab, i.e., a for loop inside a for loop. The structure is the same, **but it is very important to make sure that you have different names for the indices of your two loops:**

```
for i = min1:incr1:max1
    execute some command
     for j = min2:incr2:max2
         execute some command
     end
     execute some command
end
```

Nested loop will often be used, for example, to create or modify matrices. For example, try to guess what the matrix A would be if one execute the following command lines:

```
A=zeros(4,2);
for i = 1:2:size(A,1)
     for j = 1:2
         A(i,j)=2*i+0.5*j;
     end
end
```

Again, verify your answer in matlab to make sure you understood. Don't forget to answer the online quiz on connect!