

Problem: You were given 12,179 daily average wind speed measurements for a site from 1942 through 2013 (with some significant and annoying gaps in time!). Your PI wants monthly (*within years*) and yearly averages for these wind speed data.

Goal: Write an R function that will calculate all of these averages for you in seconds, instead of spending your whole weekend calculating them by hand.

New Functions to know:

1. unique(). Returns an object that is the same as the object given as input EXCEPT with all duplicate entries removed.

2. cbind(). Binds two or more vectors, separated by commas, into a single matrix by making the vectors into columns in that matrix.

Steps to take:

1. Start with the function skeleton we’ve already shown you.

2. START BY TRYING TO CALCULATE YEARLY AVERAGES FIRST. Make two empty vectors, one to store the years you’re calculating averages for, and another one for the averages themselves. Having both with make the results clearer because you’ll know which average goes with which year.

3. Use unique() to create a pool of years you have data from.

4. Use a for() loop to go year by year using this pool. Find (using a which call) the rows of data from the current year, use mean() to average those data, and then store the result in your storage vectors for year and averages.

5. One challenge of using a pool that isn’t a sequence starting at one in a for() loop is that it makes storing data using indexing harder than usual. One way around this problem is to use a “counter.” A counter works like this:

year.counter = 0

for (i in years) {

year.counter = year.counter + 1

…

}

If you set up a counter like this, you can use it for indexing the same way you’d have normally used the variable object!

6. Once you have a function that works for yearly averages, try creating a similar one for calculating monthly averages. Remember that we want monthly averages WITHIN years! Try using two expressions inside of your which() call to do this, one for finding data from the current year and a second one for finding data from the current month.

7. Once you have two functions, one for each type of average, see if you can combine the two functions into a single function. This is the hardest exercise for a reason!

Function testing:

Use the data sheet “averaging.data.csv” to test your function. Save the data sheet to your working directory (ask for help if you don’t know how to do this). Then, load the data sheet by running this code:

data1 = as.matrix(read.csv(“averaging.data.csv”))

You can view your data sheet before you run your function on it by using View()

View(data1)