MEMORANDUM - R coding tips
To: Students in Big Data Analytics

Subject: Don't use Loops in R, and how to avoid them

From: Prof. Roger Bohn. Rbohn@ucsd.edu
Date: June 7, 2018

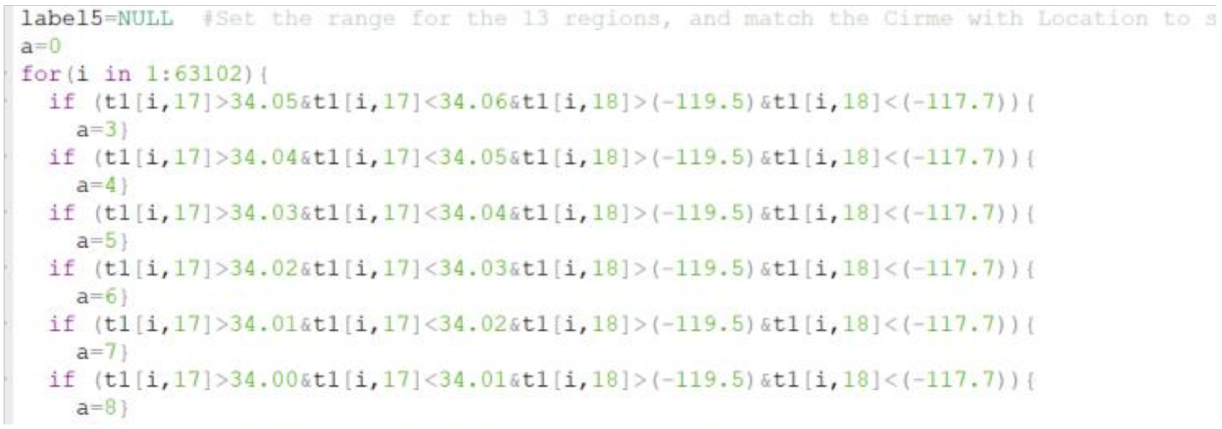
Loops in R are slow. The language is designed to not need loops. Instead, R operates on data *en masse*, meaning that all observations are processed in a single command. Almost the only situation where loops are needed is when you have more than 1,000,000 observations, and not all calculations will fit into memory at the same time. In that situation you may decide to process your data in chunks, and you will need a loop to go through all the needed chunks. For example, you might process 200,000 observations at a time, and run a loop that goes through 5 iterations to cover the entire 1,000,000 in the original data set.

Here is some code that was written using a giant loop. It marches through data in the variable t1, one row at a time. Each row is tested six times, to assign it to group 3 through 8.

This loop is completely *unnecessary*. The entire loop can be replaced by almost identical code, but working on ALL rows at the same time! This will run MUCH faster. It is also easier to understand, and less prone to bugs. The R language was designed to use vectors and arrays as much as possible, and you should take advantage of this.

More detailed explanation.

Here is code to assign events to different pollution zones, using the latitude and longitude of each event to decide what zone it belongs to. t1[,17] is the latitude of the event, while t1[,18] is the longitude.



```
label5=NULL  #Set the range for the 13 regions, and match the Cirme with Location to s
a=0
for(i in 1:63102){
  if (t1[i,17]>34.05&t1[i,17]<34.06&t1[i,18]>(-119.5)&t1[i,18]<(-117.7)){
    a=3}
  if (t1[i,17]>34.04&t1[i,17]<34.05&t1[i,18]>(-119.5)&t1[i,18]<(-117.7)){
    a=4}
  if (t1[i,17]>34.03&t1[i,17]<34.04&t1[i,18]>(-119.5)&t1[i,18]<(-117.7)){
    a=5}
  if (t1[i,17]>34.02&t1[i,17]<34.03&t1[i,18]>(-119.5)&t1[i,18]<(-117.7)){
    a=6}
  if (t1[i,17]>34.01&t1[i,17]<34.02&t1[i,18]>(-119.5)&t1[i,18]<(-117.7)){
    a=7}
  if (t1[i,17]>34.00&t1[i,17]<34.01&t1[i,18]>(-119.5)&t1[i,18]<(-117.7)){
    a=8}
```

Rewrite this code to use arrays instead of a loop. Basically keep the same code, but *REMOVE the i index* and remove the loop.

How to rewrite without a loop:

Create a data frame with 63102 rows and 13 columns. Let's name it *Region*. All values are 0 (or FALSE) at the start.

```
Region <- matrix(0, nrow = 63102, ncol = 13) #Initialize the
matrix.

Region[,4 ] <- (t1[,17] > 34.05) & (t1[,17] < 34.06) & ( t1[,18] >
-119.5 ) & (t1[,18] < -117.7)

sum(Region[,4]) #Check how many 1s in column 4

#Similar for Region[,5] and the remaining columns.
Region[,5 ] <- same for the next set of coordinates
Region[,6 ] <- same for next set of coordinates
#Continue through all regions/columns
```

Now you have 13 columns of dummy variables. The value is 1 if that row is geographically in that region. So each row should have exactly one 1, the rest 0.

Optional, but to be safe you can test for problems by checking that each row has the correct number of 0s and 1s:

```
1 == rowSums(Region) # Should return a vector of all 1s. Note
spelling of rowSums

0   %in% (1 == rowSums(Region)) # This should be FALSE, meaning
that every row has exactly one  value of 1.
# Explanation: A %in%  B returns TRUE if every element of A is
somewhere in B.
```

```
# We could also check that every row has exactly 12 values of 0.
```

Of course when writing code, test it on small problems (e.g. 10 rows), and check that the code is doing what you want it to. Before you unleash it on the entire data set at once.

Other examples

This example showed how to avoid loops for testing logical expressions. Of course, other languages use loops for many different purposes. Almost all such loops can, and should, be replaced in R by solutions that don't use loops. Many functions and methods exist in R to do operations easily and efficiently without them. See all the references on how to write R. My starting list of resources is at https://bda2020.wordpress.com/2018/04/10/resources-for-r-language/ A good recent textbook, with some newer techniques, is *R for Data Science,* which is available as a web site at http://r4ds.had.co.nz/index.html.