Stat243: Problem Set 5, Due Monday Oct. 19

October 12, 2015

Comments:

- This covers Unit 6.
- It's due at the start of class on 10/19.
- As usual, your solution should mix textual description of your solution, code, and example output.
- Please note my comments in the syllabus about when to ask for help and about working together.
- Please give the names of any other students that you worked with on the problem set.

Questions

1. Here we'll consider the effects of adding together numbers of very different sizes. Let's consider adding the number 1 to 10000 copies of the number 1×10^{-16}. Mathematically the answer is obviously $1 + 1 \times 10^{-12} = 1.000000000001$ by multiplication, but we want to use this as an example of the accuracy of summation with numbers of very different magnitudes, so consider the sum $1 + 1 \times 10^{-16} + \cdots + 1 \times 10^{-16}$.

   (a) How many digits of accuracy are the most we can expect of our result (i.e., assuming we don't carry out the calculations in a dumb way). In other words, if we store 1.000000000001 on a computer, how many digits of accuracy do we have? This is not a trick question.

   (b) In R, create the vector $x = c(1, 1 \times 10^{-16}, \ldots, 1 \times 10^{-16})$. Does the use of *sum()* give the right answer up to the accuracy expected from part (a)?

   (c) Do the same as in (b) for Python. For Python the *Decimal()* function from the decimal package is useful for printing additional digits. Also, this code will help you get started in creating the needed vector:

   ```
   import numpy as np

   vec = np.array([1e-16]*(10001))
   ```

   (d) Use a *for* loop to do the summation, $((x_1 + x_2) + x_3) + \ldots$. Does this give the right answer? Now use a *for* loop to do the summation with the 1 as the last value in the vector instead of the first value. Right answer? If either of these don't give the right answer, how many decimal places of accuracy does the answer have? Do the same in Python.

    (e) What do the results suggest about how R's *sum()* function works? Is it simply summing the numbers from left to right?

    (f) Extra credit: Figure out what R's *sum()* function is doing, either by finding documentation or by looking at the actual code used.

   This should suggest that the *sum()* function works in a smart way, but that if you calculate the sum manually, you need to be careful in some situations.

2. Explore whether integer calculations in R are faster than floating point calculations (assessing my claim in the notes that in general integer operations are faster). Try out some arithmetic, subsetting vectors and other operations that you think might be interesting.