IS206 Assignment 7

Sharon Morris

3/14/2017

Problem Set 1 Please write a function to compute the expected value and standard deviation of an array of values. Compare your results with that of R's mean and std functions.

```
#Define the function
expValue <- function(E) {
    if (length(E) != 0)    # lenght of E cannot equal zero - cannot divide by zero
        return(sum(E)/length(E))  #lenght is used to determine the length of a list
}

#Validate the function using built in mean
validate <- c(1, 3, 10, 15, 30, 40)
expValue(validate) == mean(validate)
```

```
## [1] TRUE
```

Standard deviation

$$\sigma = \sqrt{\frac{\sum_{i=1}^{n} (x_i - \bar{x})^2}{n-1}}$$

```
SD <- function(E) {
    if ((length(E)-1) !=0)
        return(sqrt (sum( (E-expValue(E) ) ^2 ) / (length(E)-1)))
}

#Validate using built in sd
(SD(validate) == sd(validate))
```

```
## [1] TRUE
```

Now, consider that instead of being able to neatly fit the values in memory in an array, you have an infinite stream of numbers coming by. How would you estimate the mean and standard deviation of such a stream? Your function should be able to return the current estimate of the mean and standard deviation at any time it is asked. Your program should maintain these current estimates and return them back at any invocation of these functions. (Hint: You can maintain a rolling estimate of the mean and standard deviation and allow these to slowly change over time as you see more and more new values).

Estimate the mean and standard deviation of an infinite stream of numbers coming by.

```
#define variables
glength <- 0
sum  <- 0
sumSq <- 0
xBar <- 0
SD <- 0

#function with variables
gF <- function() {
    glength <- 0
    sum <- 0
```

```
    xBar <- 0
    sumSq <- 0
    SD <- 0

}

#function prints the sample size, mean and standard deviation
pRolngStats <- function() {
  print(sprintf("n = %s; mean = %s;  sd = %s", glength, xBar, SD))

}


rolngStats <- function(vector) {
    for (element in vector) {        #numbers in the vector
        glength <<- glength + 1        #increment value in the vector by 1
        sum <<- sum + element          #old value plus new value
        xBar <<- sum/glength           #calculate the mean
        sumSq <<- sumSq + element^2

        SD <<- sqrt((glength)* sumSq - sum^2)/(glength) #calculate standard deviation

        if (glength %% 10000 == 0) pRolngStats()
    }

    return (c("n" = glength, "mean" = xBar, "sd" = SD))

}
```

```
#Test
gF()
    vector <- rnorm(n = 100000, mean = 500, sd = 2.5) #generates n with mean and SD
    rolngStats(vector)
```

```
## [1] "n = 10000; mean = 499.987247563538;  sd = 2.51368420796703"
## [1] "n = 20000; mean = 499.985370746129;  sd = 2.49678787631166"
## [1] "n = 30000; mean = 499.985205127313;  sd = 2.50763340960046"
## [1] "n = 40000; mean = 499.991573024297;  sd = 2.51287467078139"
## [1] "n = 50000; mean = 499.994741678673;  sd = 2.50998029410591"
## [1] "n = 60000; mean = 499.993243593469;  sd = 2.51082235154972"
## [1] "n = 70000; mean = 499.995467860499;  sd = 2.50891713535777"
## [1] "n = 80000; mean = 499.994633422286;  sd = 2.50908075044739"
## [1] "n = 90000; mean = 499.994281716431;  sd = 2.50893233058911"
## [1] "n = 1e+05; mean = 499.997059308865;  sd = 2.50859492383685"
```

```
##            n         mean           sd
## 1.000000e+05 4.999971e+02 2.508595e+00
```

Test

```
mean(vector) #mean
```

```
## [1] 499.9971
```

```
sd(vector)   #standard deviation
```

```
## [1] 2.508607
```

References

http://stackoverflow.com/questions/3655717/given-a-random-variable-with-p

http://stackoverflow.com/questions/10657503/find-running-median-from-a-s

http://jonisalonen.com/2013/deriving-welfords-method-for-computing-varian