STA 250: Homework # 2

Valerie Regalia
SID# 997214200

February 11, 2014

Technique 1

What I did

In this technique, I first used the shell to extract the Arrival Delays from each file given, and I then combined them into 11 different files that contained just the delays. In R, I created a function f that read in a given file, and calculated its mean, standard deviation, and median. I used parallel package, and, with 4 nodes, used clusterApply() to call my function on each file. Once that had been done, I took the weighted means of all 10 estimates for the mean, standard deviation, and median.

Estimates

Mean: 6.200016,
Standard Deviation: 30.93905,
Median: -1.168566.

Files

For this technique, I used all files except for "2002.csv", since I was unable to get the 2002 file to work with my code.

Time

user	0.039
system	0.035
elapsed	33.015

Code

```
library(parallel)

filenames = c("arr1", "arr2", "arr3", "arr2003", "arr2004", "
    arr2005", "arr2006", "arr2007", "arr5", "arr6")

cl = makeCluster(4, "FORK")
```

```
avgs = numeric(10)
stdevs = numeric(10)
medians = numeric(10)
lengths = numeric(10)


x = clusterApply(cl, filenames, f)
time = system.time(clusterApply(cl, filenames, f))

  for(k in 1:10){
    t = x[[k]]
    avgs[k] = t[[1]]
    stdevs[k] = t[[2]]
    medians[k] = t[[3]]
    lengths[k] = t[[4]]
  }

avg = weighted.mean(avgs, lengths) # 6.200016
stdev = weighted.mean(stdevs, lengths) # 30.93905
med = weighted.mean(medians, lengths) # -1.168566




f = function(fn){
  delays = readLines(fn)
  delays = as.numeric(delays)
  avg = mean(delays, na.rm = TRUE)
  stdev = sd(delays, na.rm = TRUE)
  med = median(delays, na.rm = TRUE)
  l = length(delays)

  rm(delays)

  return(list(avg, stdev, med, l))

}
```

Technique 2

What I did

For this technique, I used the same 11 parsed files from the shell that contain only the Airline Delays. I installed the package FastCSVSample that was given to us (*Thanks, Duncan!!*), and loaded it into R. Next, I wrote a function that takes in a file name, and uses csvSample() to sample 3000 observations from that file and calculate the mean, median, and standard deviation of those observations. This is looped over 50 times, and the averages of these 50 estimates is taken for each the mean, median, and standard deviation. These averages are returned. Using the parallel package, I opened 10 nodes and used clusterApply() on a vector of file names to call the function on each file. I took the means over all 10 estimates for each of the mean, median, and standard deviation *again* to attain a final estimate for each.

Estimates

Mean: 6.2964,
Standard Deviation: 32.2935,
Median : -1.445.

Files

For this technique, I used all files except for "2002.csv", since I was unable to get the 2002 file to work with my code.

Time

user	0.258
system	0.242
elapsed	245.693

$\longrightarrow$ Please note: The time for this technique took much longer than for technique 1. This would be because sampling takes much more time than just using the actual raw data.

Code

```
library(parallel)


filenames = c("arr1", "arr2", "arr3", "arr2003", "arr2004", "
    arr2005", "arr2006", "arr2007", "arr5", "arr6")


fns = rep(filenames, times = 100)

cl = makeCluster(10, "FORK")
x = clusterApply(cl, filenames, samplecalc)
system.time(clusterApply(cl, filenames, samplecalc))


means = numeric(10)
sds = numeric(10)
medians = numeric(10)

for(k in 1:10){
  t = x[[k]]
  means[k] = t[[1]]
  sds[k] = t[[2]]
  medians[k] = t[[3]]
}

mean.est = mean(means)
sd.est = mean(sds)
median.est = mean(medians)


#1000 sample size:{ mean = 6.2992, median = -1.393, sd = 31.6652,
     system.time =   #user  system elapsed
#0.218    0.222 238.608    }


#Sample size: 3000 : {mean = 6.2964, median = -1.445, sd =
    32.2935, time:     user  #system elapsed
```

```
#0.258    0.242  245.693   }



samplecalc = function(fn, times = 50){

  setwd("~/Documents/UCD/2013-2014/WINTER-2014/STA-250/Homework2"
      )

  means = numeric(times)
  sds = numeric(times)
  medians = numeric(times)


  for(i in 1:times){
    x = csvSample(fn, 3000)
    x = as.numeric(x)
    means[i] = mean(x, na.rm = TRUE)
    sds[i] = sd(x, na.rm = TRUE)
    medians[i] = median(x, na.rm = TRUE)
  }

  mean.est = mean(means)
  sd.est = mean(sds)
  median.est = mean(medians)

  return( list( mean.est, sd.est, median.est ) )


}
```