HW2 Report

Alice Mee Seon Chung

10/9/2017

Part 1.6 Compare_outcome

We make a new function called `compare_outcomes`, that takes as input an iteration number and compute the mean and std of the values in the chain for a and return those output in matrix. We can test this function for 1,000, 10,000, and 100,000 iterations as different input values. Here, we set our true value A as 5.

```
trueA <- 5
trueB <- 0
trueSd <- 10
sampleSize <- 31

# create independent x-values
x <- (-(sampleSize-1)/2):((sampleSize-1)/2)
# create dependent values according to ax + b + N(0,sd)
y <-  trueA * x + trueB + rnorm(n=sampleSize,mean=0,sd=trueSd)
```

```
# compare outcomes function
# define function for comparing 10 iterations of run_metropolis_MCMC function
compare_outcomes <- function(iterations) {
  # set a initial matrix
  initialmatrix <- array(dim = c(2,10))
  burnIn = iterations * 0.5
  for (i in 1:10) {
    # set a andomly selected start values for a, b and sd
    startvalues = c(runif(1, 0, 10), rnorm(1, 0, 5), runif(1, 0, 30))
    # run run_metropolis_MCMC with startvalues and iterations
    chain = run_metropolis_MCMC(startvalues, iterations)
    # compute the mean of the values into the initialmatrix[1,i]
    initialmatrix[1,i] <- mean(chain[-(1:burnIn),][,1])
    # compute the standard deviation of the values into the initialmatrix[2,i]
    initialmatrix[2,i] <- sd(chain[-(1:burnIn),][,1])
  }
  # return the intialmatrix
  return(initialmatrix)
}
```

Here are the results of different numbers of iterations. First iteration number is 1,000 and the matrix with mean and sd are below,

```
results1 <-compare_outcomes(1000)
print(results1)
```

```
##           [,1]     [,2]      [,3]      [,4]      [,5]      [,6]      [,7]
## [1,] 5.2660049 5.358748 5.2158977 5.2603134 5.1654514 5.2662511 5.3463850
## [2,] 0.2315181 0.216174 0.2228972 0.2159574 0.2102375 0.3125827 0.2616422
##          [,8]      [,9]     [,10]
## [1,] 5.282987 5.3669830 5.1835718
## [2,] 0.186703 0.1842234 0.2472624
```

Second iteration number is 10,000 and the matrix with mean and sd are below,

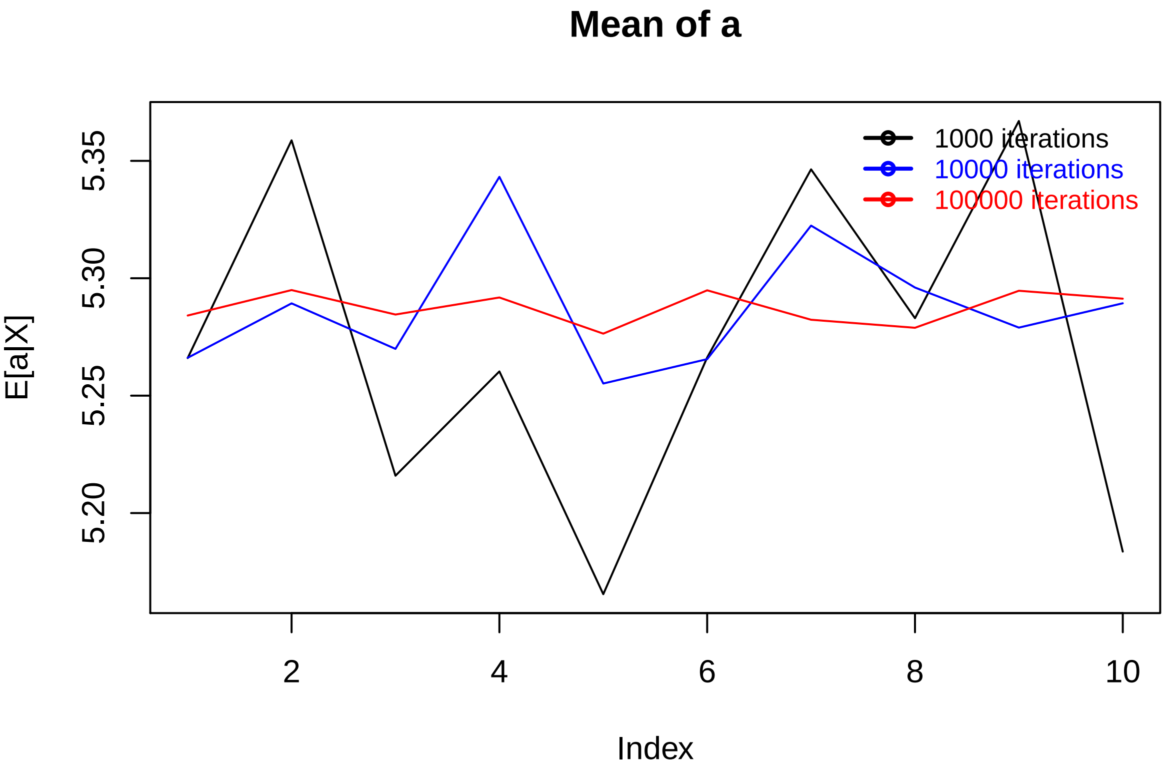
```
results2 <-compare_outcomes(10000)
print(results2)
```

```
##           [,1]      [,2]      [,3]      [,4]      [,5]     [,6]      [,7]
## [1,] 5.2661311 5.2893107 5.2699271 5.3431861 5.2551765 5.265549 5.3224214
## [2,] 0.2435008 0.2485823 0.2361065 0.2439546 0.2370672 0.250416 0.2809372
##           [,8]      [,9]     [,10]
## [1,] 5.2960444 5.2789774 5.2893424
## [2,] 0.2141447 0.2260919 0.2772752
```

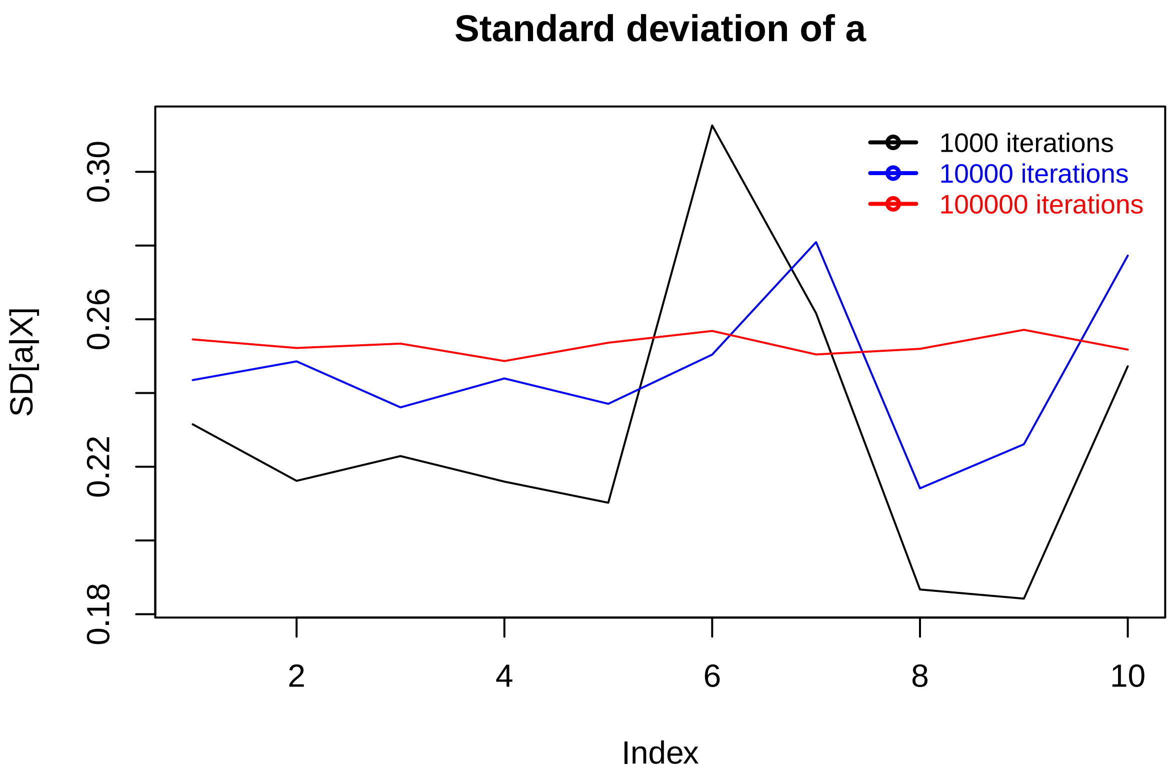
Third iteration number is 100,000 and the matrix with mean and sd are below,

```
results3 <-compare_outcomes(100000)
print(results3)
```

```
##           [,1]      [,2]     [,3]      [,4]      [,5]      [,6]      [,7]
## [1,] 5.2841271 5.2949690 5.284543 5.2918088 5.2764074 5.2948665 5.2823313
## [2,] 0.2545447 0.2522071 0.253397 0.2486687 0.2536445 0.2568344 0.2504577
##           [,8]      [,9]     [,10]
## [1,] 5.2789156 5.2946720 5.2912971
## [2,] 0.2519877 0.2571392 0.2517708
```



For easier comparison of above results, let's plot with three lines of means of "a". As you saw from above graph, we can observe that as iteration number increases the trend of the mean of a is less fluctuates and becomes stable. We also can see when the iteration number is 1,000, the mean of mean of "a" is 5.2712594 and when the iteration number is 10,000, the mean of mean of "a" is 5.2876066. Lastly, when the iteration number is 100,000, the mean of mean of "a" is 5.2876066. Thus as iteration number increases, the mean of mea of "a"" also becomes small and stable.



Again let's plot with three lines of standard deviation of "a". As you saw from above graph, we can observe that as iteration number increases the trend of the standard deviation of "a"" is less fluctuates and becomes stable like previous observation. When the iteration number is 1,000, the mean of standard deviation of "a" is 0.2289198 and when the iteration number is 10,000, the mean of standard deviation of "a" is 0.2458076. Lastly, when the iteration number is 100,000, the mean of standard deviation of "a" is 0.2458076. Thus as iteration number increases, the mean of mea of "a"" also becomes small and stable.

To sum up, the deviation of the results decreases and becomes stable, we can conclude that the accuracy of algorithm in finding "a" improves as the number of iterations increases.