Report for question 5,6

Jinjie Zhang

October 12, 2017

The compare_outcomes function

In this question, we make a new function, compare_outcomes, that takes as input an iteration number. It gives the result of mean and sd of the data a. By running this function for three times with different input of iterations as 1000, 10000 and 100000 we can get following results:

```r
source("Source.R")
#load the Source.R file which contains all the functions we've defined
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(trueA,trueB,trueSd,iterations)
{
  adata=matrix(0,10,2)
  for (i in 1:10)
  {
    #construct the random variables for the startvalue
    ParA<-runif(1,trueA-3,trueA+3)
    ParB<-runif(1,trueB-3,trueB+3)
    ParSd<-runif(1,trueSd-3,trueSd+3)
    startvalue<-c(ParA,ParB,ParSd)
    newchain<-run_metropolis_MCMC(startvalue,iterations)
    refinechain<-newchain[-(1:0.5*length(newchain)),]
    adata[i,1]=mean(refinechain[,1])
    adata[i,2]=sd(refinechain[,1])
  }
  return(adata)
}
Final1<-compare_outcomes(trueA,trueB,trueSd,1000)
Final2<-compare_outcomes(trueA,trueB,trueSd,10000)
Final3<-compare_outcomes(trueA,trueB,trueSd,100000)
print(Final1)
```

```
##           [,1]      [,2]
##  [1,] 5.038810 0.3148011
##  [2,] 5.081873 0.2714768
##  [3,] 5.087064 0.2065851
##  [4,] 5.002150 0.3298254
##  [5,] 5.008709 0.4073824
##  [6,] 5.083670 0.1833404
##  [7,] 5.061289 0.1877996
##  [8,] 5.108437 0.2265524
##  [9,] 5.099648 0.1899201
## [10,] 5.163927 0.4702747
```

```
print(Final2)
```

```
##           [,1]      [,2]
##  [1,] 5.078166 0.1945835
##  [2,] 5.062228 0.1756434
##  [3,] 5.060906 0.2327271
##  [4,] 5.087628 0.1888523
##  [5,] 5.087171 0.1886781
##  [6,] 5.071736 0.1756795
##  [7,] 5.063907 0.2102735
##  [8,] 5.058300 0.2058900
##  [9,] 5.066728 0.2044800
## [10,] 5.069867 0.2050635
```

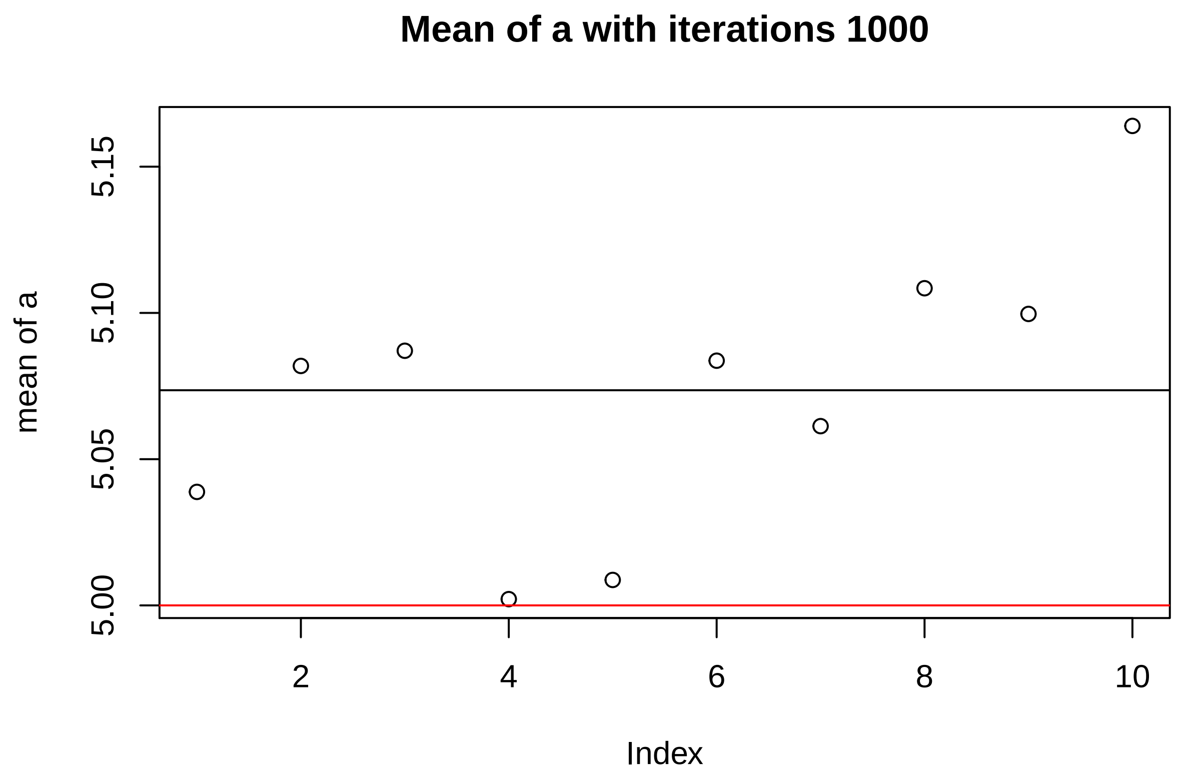
```
print(Final3)
```

```
##           [,1]      [,2]
##  [1,] 5.078664 0.1833910
##  [2,] 5.080387 0.1814040
##  [3,] 5.080003 0.1822779
##  [4,] 5.078508 0.1867724
##  [5,] 5.073646 0.1813393
##  [6,] 5.073090 0.1841466
##  [7,] 5.075102 0.1836620
##  [8,] 5.076541 0.1837015
##  [9,] 5.076374 0.1827932
## [10,] 5.077631 0.1828145
```

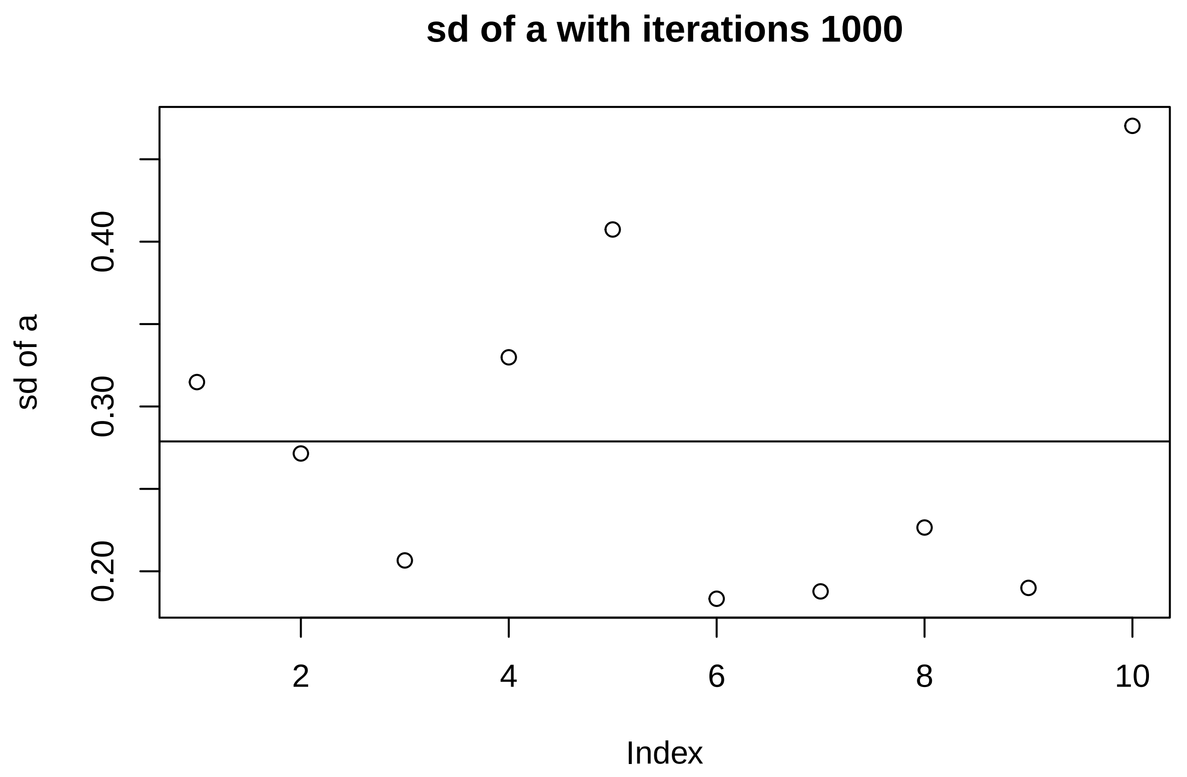
```
axis=c(1:10)
```

The number of iterations is 1000.

```
plot(axis,Final1[,1],main="Mean of a with iterations 1000", xlab="Index",ylab="mean of a")
abline(h = mean(Final1[,1]))
abline(h = trueA, col="red" )
```

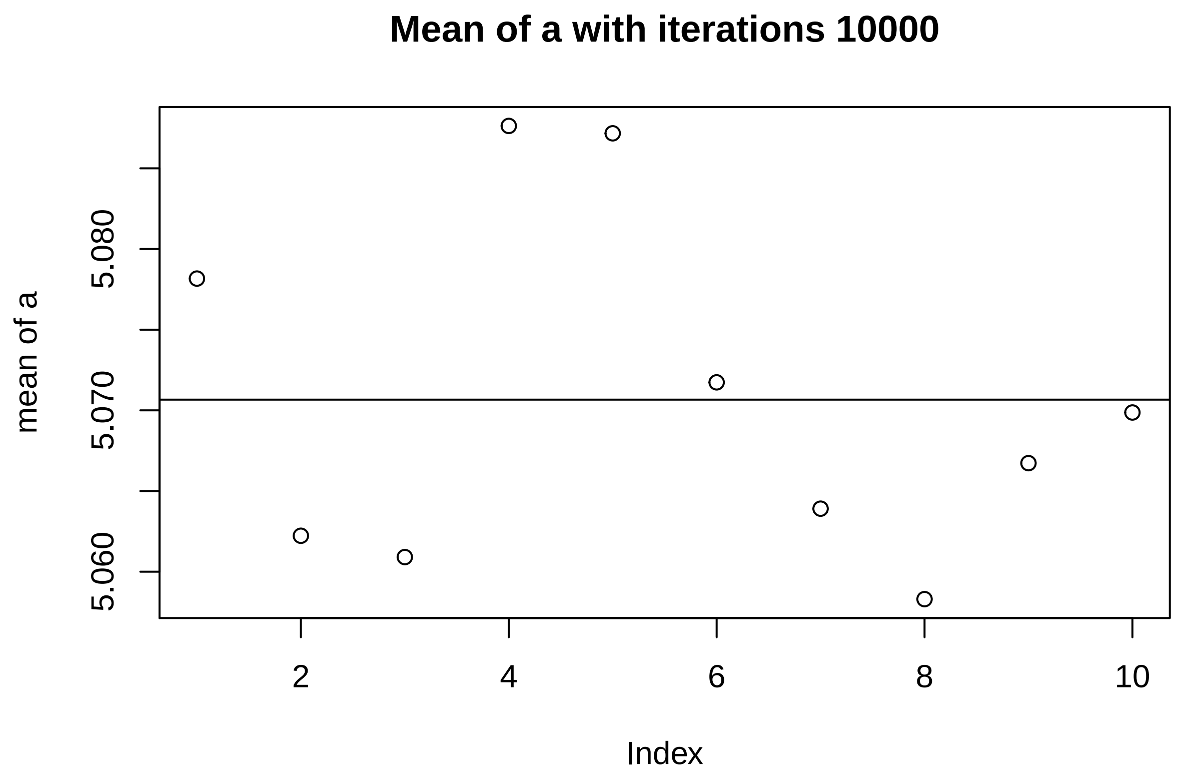


```
plot(axis,Final1[,2],main="sd of a with iterations 1000", xlab="Index",ylab="sd of a")
abline(h = mean(Final1[,2]))
abline(h = 0, col="red" )
```

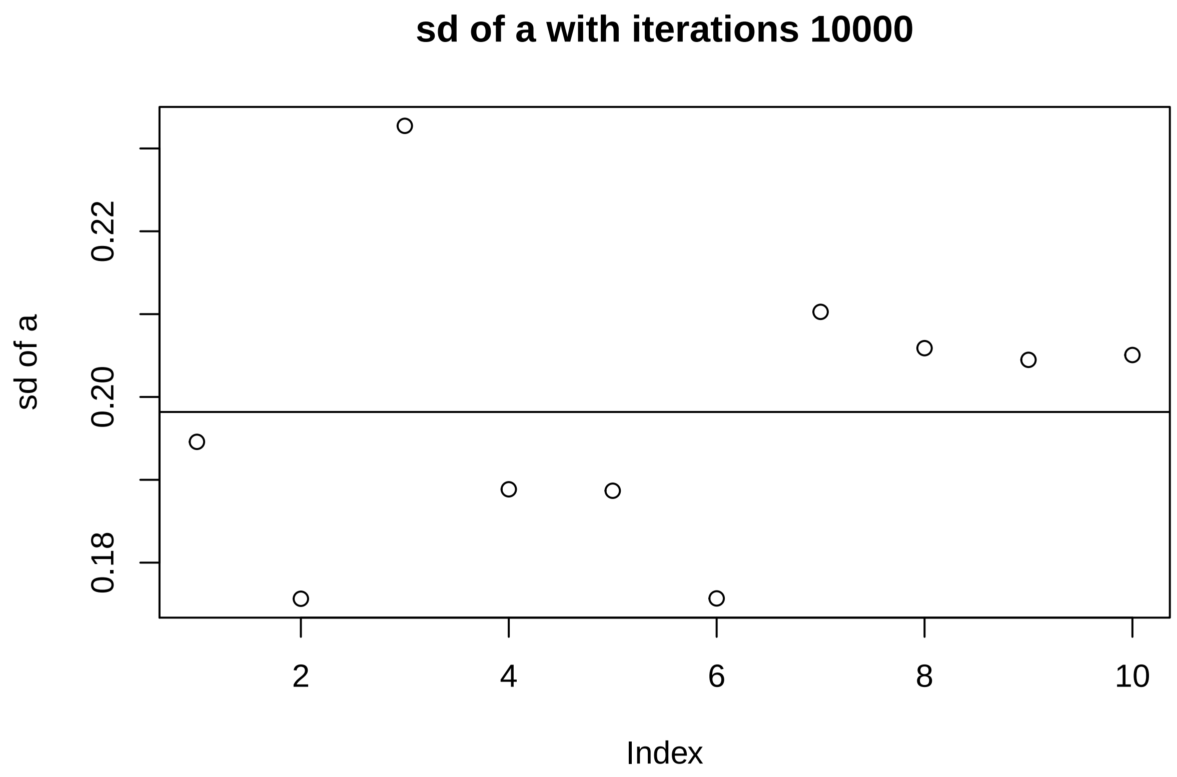


The number of iterations is 10000.

```
plot(axis,Final2[,1],main="Mean of a with iterations 10000", xlab="Index",ylab="mean of a")
abline(h = mean(Final2[,1]))
abline(h = trueA, col="red" )
```

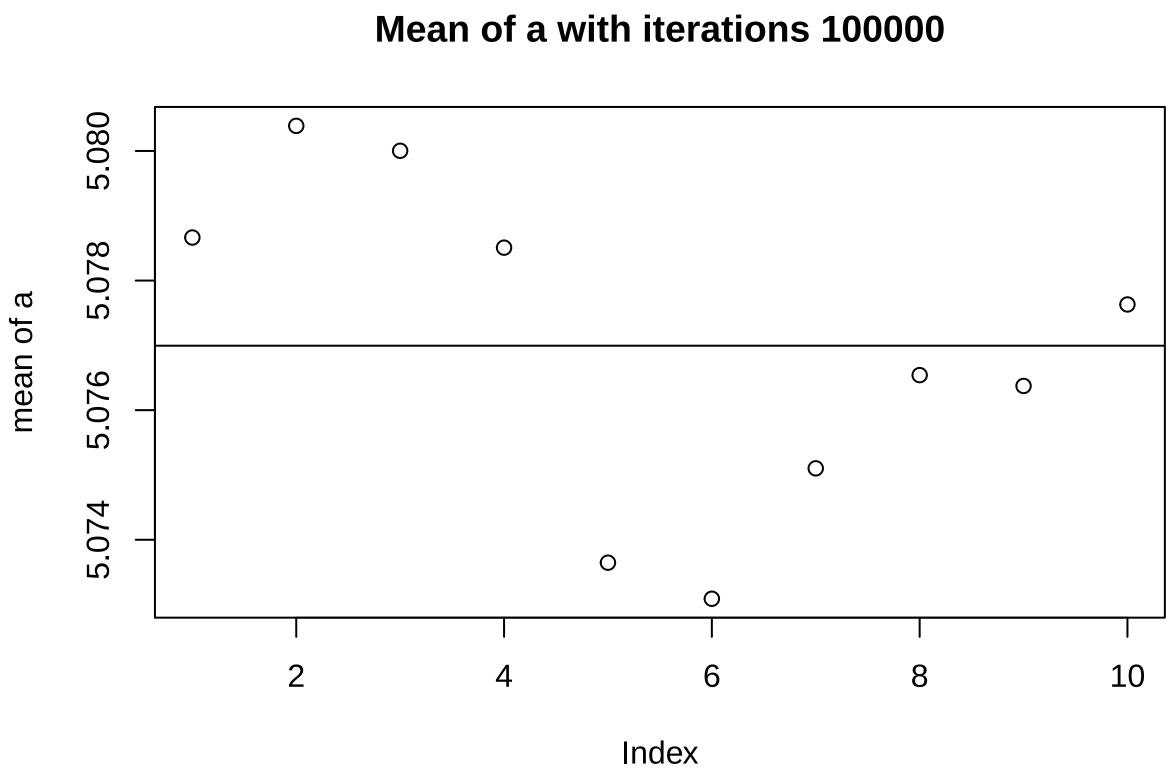


```
plot(axis,Final2[,2],main="sd of a with iterations 10000", xlab="Index",ylab="sd of a")
abline(h = mean(Final2[,2]))
abline(h = 0, col="red" )
```

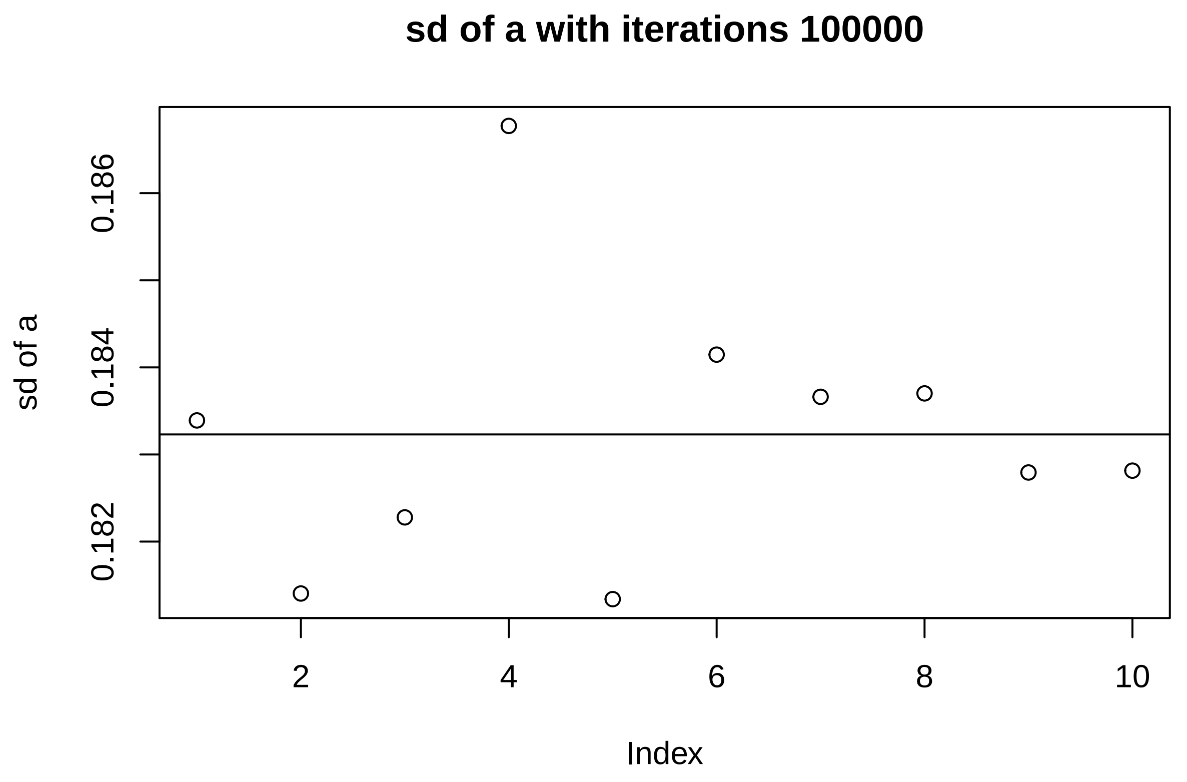


The number iterations is 100000.

```r
plot(axis,Final3[,1],main="Mean of a with iterations 100000", xlab="Index",ylab="mean of a")
abline(h = mean(Final3[,1]))
abline(h = trueA, col="red" )
```



```
plot(axis,Final3[,2],main="sd of a with iterations 100000", xlab="Index",ylab="sd of a")
abline(h = mean(Final3[,2]))
abline(h = 0, col="red" )
```



Conclusion:

The true value of a is 5. For above three different numbers of iterations, we have following results:

1. When the number of iterations is 1000, the mean of approximations is 4.78 and the standard deviation is 0.37;
2. When the number of iterations is $10,000$, the mean of approximations is 4.74 and the standard deviation is 0.21;
3. When the number of iterations is $100,000$, the mean of approximations is 4.74 and the standard deviation is 0.20.

Since the deviation of the predictions decreases, the accuracy of finding a improves as the number of iterations increases.