Assignment 2 part 1.5&1.6

Fangzhou Yang

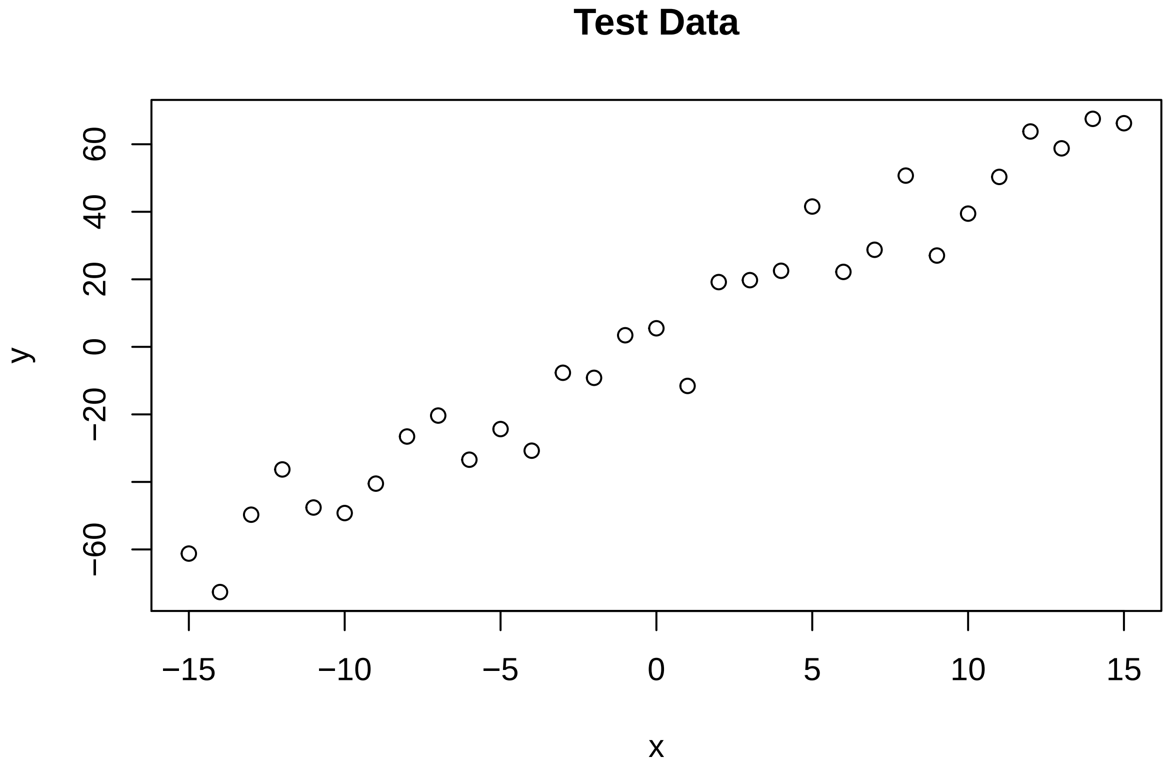
10/12/2017

Main

Firstly, we source all of the functions what we have saved in 1.4, and define variables:

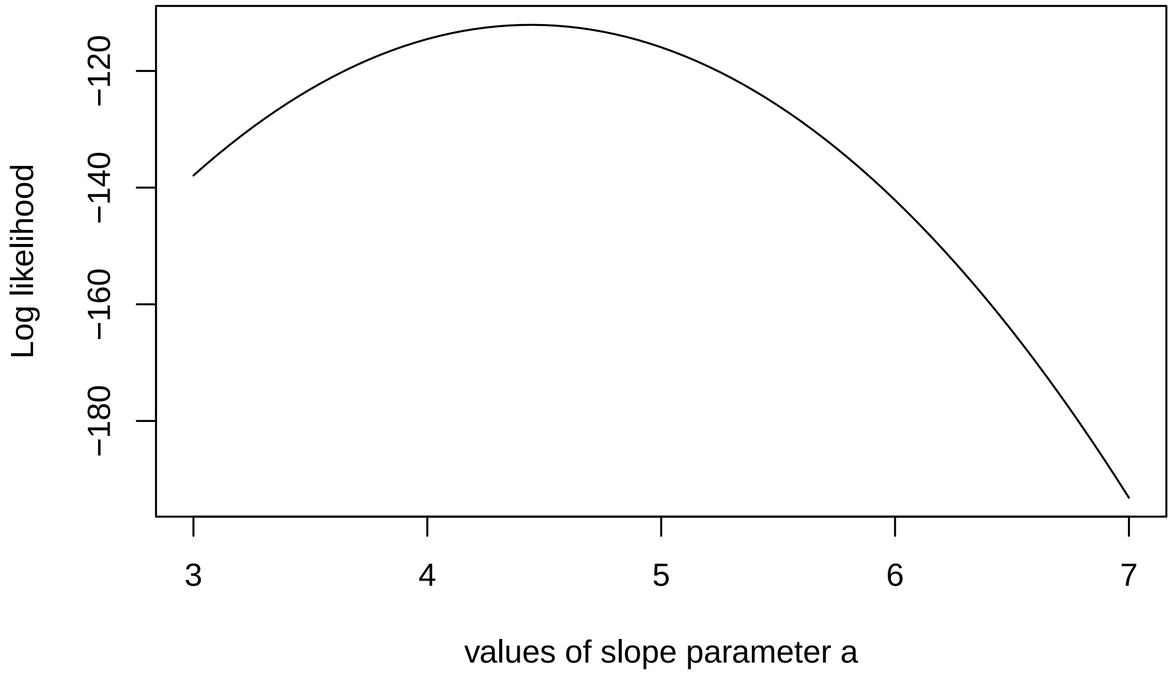
```
source("Source.R")
trueA <- 5
trueB <- 0
trueSd <- 10
sampleSize <- 31

x <- (-(sampleSize-1)/2):((sampleSize-1)/2)
y <-  trueA * x + trueB + rnorm(n=sampleSize,mean=0,sd=trueSd)
plot(x,y,main="Test Data")
```



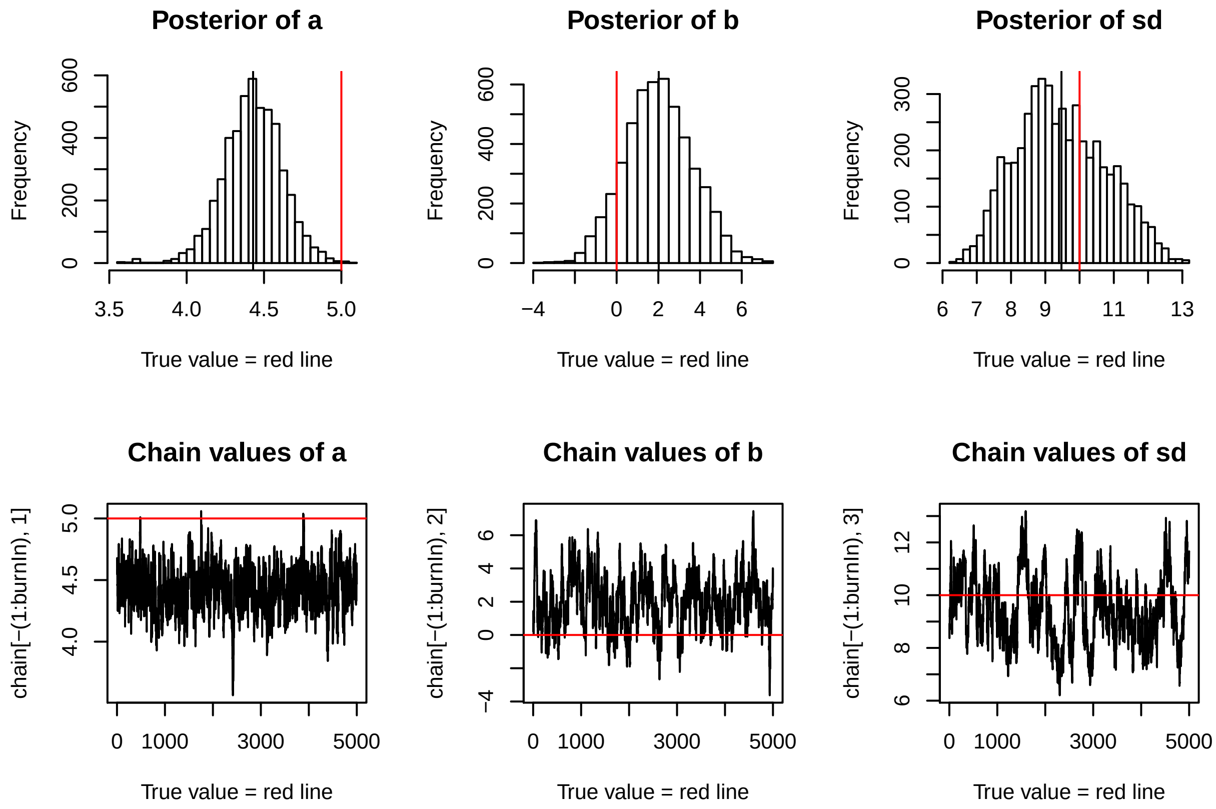
Plot the graph of slope lokelihood:

```
slopelikelihoods=lapply(seq(3, 7, by=.05), slopevalues )
plot (seq(3, 7, by=.05), slopelikelihoods , type="l",
      xlab = "values of slope parameter a", ylab = "Log likelihood")
```



Set value to the variables as below and plot graphs of a,b and sd:

```
startvalue = c(4,0,10)
chain = run_metropolis_MCMC(startvalue, 10000)
burnIn = 5000
acceptance = 1-mean(duplicated(chain[-(1:burnIn),]))
fit_13(chain,burnIn,trueA,trueB,trueSd)
```



```
summary(lm(y~x))
```

```
##
## Call:
## lm(formula = y ~ x)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -18.135  -5.528  -0.666   5.846  17.220
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)   2.1208     1.6017   1.324    0.196
## x             4.4431     0.1791  24.811   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 8.918 on 29 degrees of freedom
## Multiple R-squared:  0.955,  Adjusted R-squared:  0.9535
## F-statistic: 615.6 on 1 and 29 DF,  p-value: < 2.2e-16
```

Write the compare_outcomes function:

```
compare_outcomes<-function(iteration){
  for(i in 1:10){
    a<-runif(1,min=0,max=10)
    b<-rnorm(1)
    sd<-runif(1,min=0,max=20)
    startvalue <- c(a,b,sd)
    chain<- run_metropolis_MCMC(startvalue, iteration)
    meana<-mean(chain[,1])
    sda<-sd(chain[,1])
    print(paste("mean is",meana, "sd is",sda))
  }
}
```

Test the compare_outcomes function use iteration = 1000,10000 and 100000.

```
fit1 <- compare_outcomes(1000)
```

```
## [1] "mean is 4.3921104944019 sd is 0.268489541145208"
## [1] "mean is 4.11389768010548 sd is 0.804153906494014"
## [1] "mean is 4.36818112048496 sd is 0.419694816430997"
## [1] "mean is 4.3538327246555 sd is 0.298325583835206"
## [1] "mean is 4.42223074858848 sd is 0.230463829767608"
## [1] "mean is 4.74736630443803 sd is 0.753217269875604"
## [1] "mean is 4.53881096418305 sd is 0.389982179081424"
## [1] "mean is 4.7817289569434 sd is 0.909922629000403"
## [1] "mean is 4.62331519960324 sd is 0.684158637679515"
## [1] "mean is 4.09223166041191 sd is 0.923121310750425"
```

```
fit2 <- compare_outcomes(10000)
```

```
## [1] "mean is 4.4400006666097 sd is 0.206997633116777"
## [1] "mean is 4.41985574042383 sd is 0.257522154070453"
## [1] "mean is 4.45504813168224 sd is 0.291096046686261"
## [1] "mean is 4.47034284505817 sd is 0.341915401964988"
## [1] "mean is 4.45665614832356 sd is 0.202837827968939"
## [1] "mean is 4.44271932066543 sd is 0.25330133553684"
## [1] "mean is 4.48806068713633 sd is 0.407626035045641"
## [1] "mean is 4.41081320688795 sd is 0.372330431888999"
## [1] "mean is 4.45574334912082 sd is 0.304685843245363"
## [1] "mean is 4.45394739063758 sd is 0.199079004422847"
```

```
fit3 <- compare_outcomes(100000)
```

```
## [1] "mean is 4.44195803810228 sd is 0.188556536127063"
## [1] "mean is 4.44492957693751 sd is 0.190056838373457"
## [1] "mean is 4.44198005652401 sd is 0.190454856383807"
## [1] "mean is 4.44273475691186 sd is 0.215889898193233"
## [1] "mean is 4.44508055574119 sd is 0.20113509191163"
## [1] "mean is 4.43959497623343 sd is 0.195708917674968"
## [1] "mean is 4.44382736702155 sd is 0.188538412404463"
## [1] "mean is 4.45024784573902 sd is 0.193069820826553"
## [1] "mean is 4.44736922907419 sd is 0.220811950174503"
## [1] "mean is 4.44897694077737 sd is 0.203183003337266"
```

From the test, we can see that the mean is around 4.97, and as iteration goes bigger, the mean is getting stable. Therefore we'd like to use a larger iteration to get an accurate outcome, but I think the iteration is no need to be pretty big, since it is unnecesary and run slow.