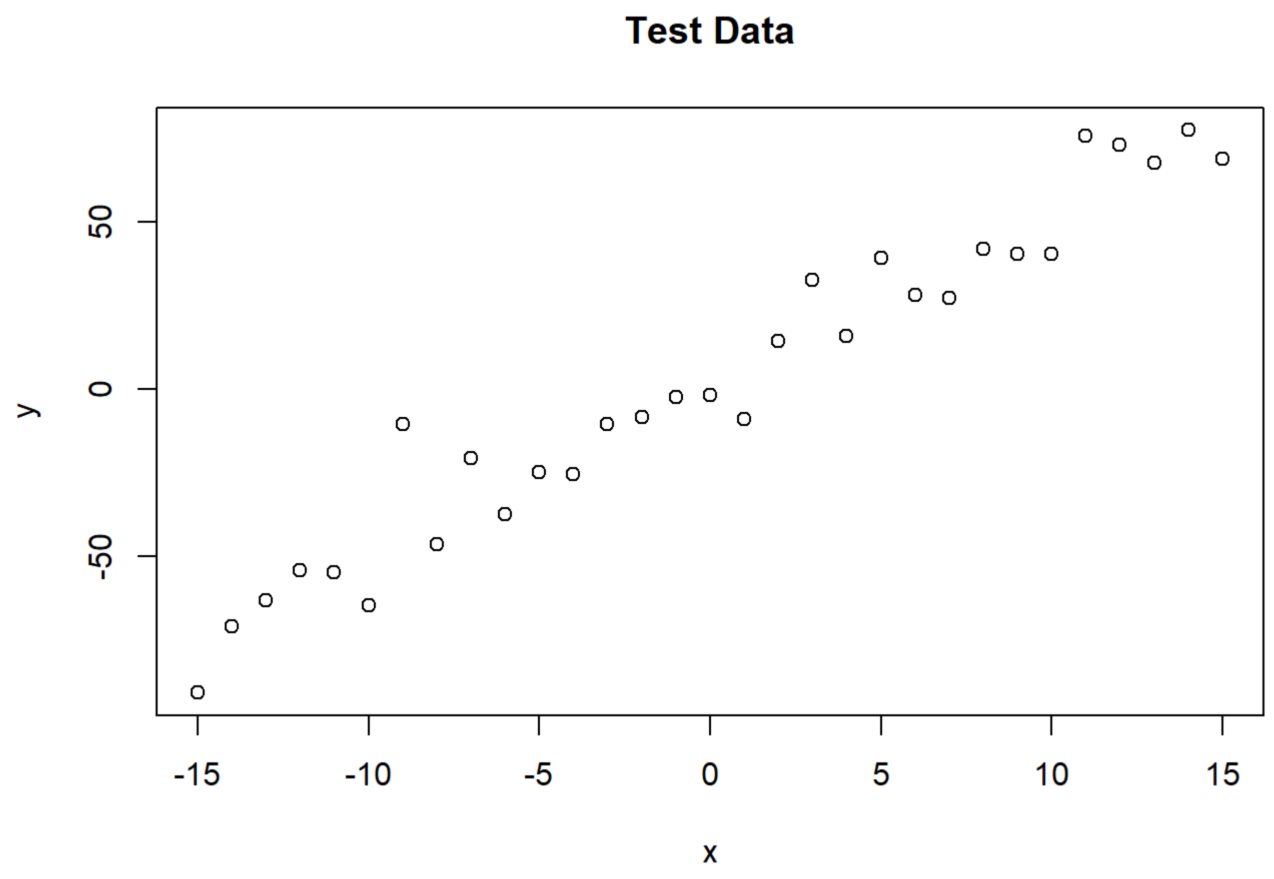
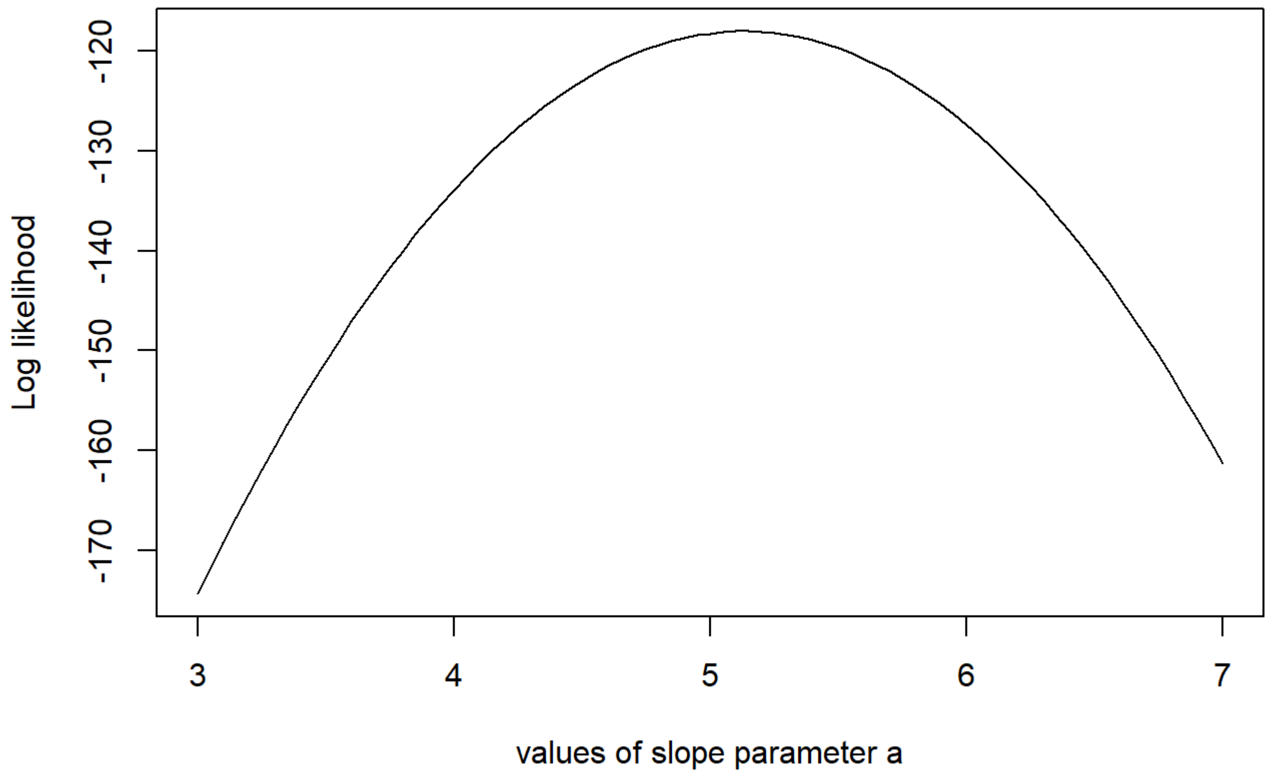
HW2 Zhen Dai

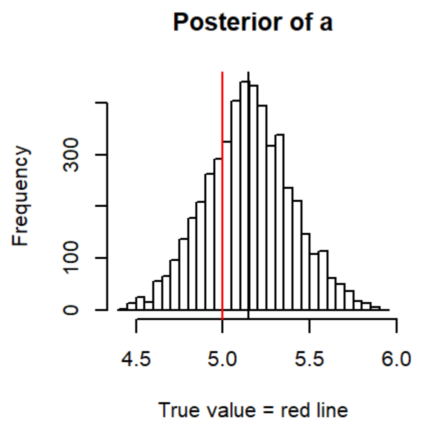
Zhen Dai

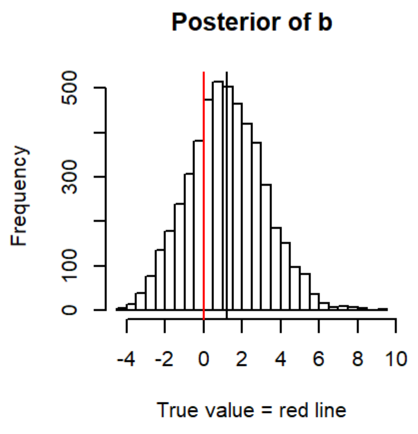
October 22, 2018

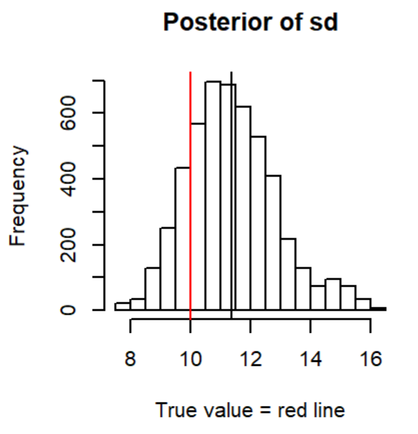
```
source("C:/Users/meteorite619/Documents/GitHub/assignment-2-Eric-Zhen/HW2 Zhen Dai code.R")
```

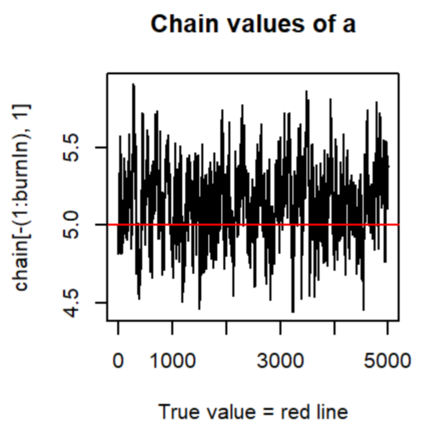

Test Data
y
x
50
0
-50
-15
-10
-5
0
5
10
15

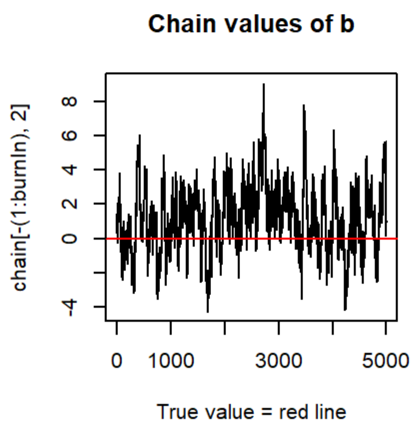

Log likelihood
values of slope parameter a
-120
-130
-140
-150
-160
-170
3
4
5
6
7

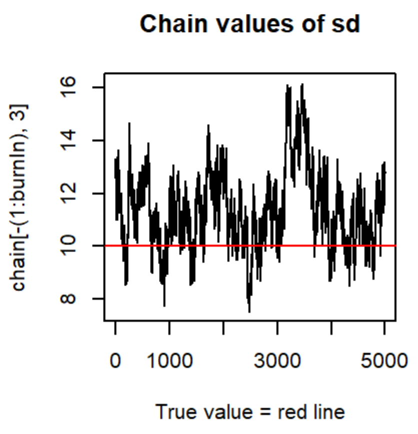












```r
compare_outcomes = function(iterations){
  i = 1
  burnIn = iterations/2
  c = 0
  while (i < 11){
    a = runif(n = 1, min=0, max=10)
    b = rnorm(n = 1, sd = 5)
    sd = runif(n = 1, min=0, max=30)
    startvalue = c(a,b,sd)
    result = run_metropolis_MCMC(startvalue, iterations)
    pred_a = result[-(1:burnIn),1]
    mean_a = mean(pred_a)
    c = c+ mean_a
    sd_a = sd(pred_a)
    t = c(mean_a,sd_a)
    print(t)
    i = i+1
  }
  mean_final = c/10
  return(mean_final)
}


co1 = compare_outcomes(1000)
```

```
## [1] 5.1731604 0.1610864
## [1] 5.1486177 0.2043086
## [1] 5.2208144 0.1972708
## [1] 5.0865174 0.2071053
## [1] 5.1382855 0.3103057
## [1] 5.2787999 0.2324403
## [1] 5.1435792 0.2262881
## [1] 5.0886823 0.2391387
## [1] 5.1375301 0.2527857
## [1] 5.1343319 0.1624121
```

```
co2 = compare_outcomes(10000)
```

```
## [1] 5.1346659 0.2438764
## [1] 5.1303559 0.1987261
## [1] 5.1429373 0.2398011
## [1] 5.1157624 0.2194893
## [1] 5.1190078 0.2351077
## [1] 5.1200546 0.2159517
## [1] 5.1025851 0.2193431
## [1] 5.1470687 0.2352004
## [1] 5.1482588 0.2469505
## [1] 5.1048007 0.2239667
```

```
co3 = compare_outcomes(100000)
```

```
## [1] 5.1356551 0.2328871
## [1] 5.1218393 0.2333857
## [1] 5.1411591 0.2332127
## [1] 5.1269544 0.2309129
## [1] 5.130790 0.235949
## [1] 5.1299557 0.2374672
## [1] 5.1325948 0.2381988
## [1] 5.1303253 0.2280371
## [1] 5.1373795 0.2317666
## [1] 5.1502016 0.2326985
```

```
co4 = compare_outcomes(2000)
```

```
## [1] 5.1428480 0.2499208
## [1] 5.140128 0.209959
## [1] 5.1642535 0.2753717
## [1] 5.1537801 0.1897776
## [1] 5.1711776 0.2249887
## [1] 5.1617615 0.1854843
## [1] 5.1345660 0.2480532
## [1] 5.1519028 0.2081569
## [1] 5.1271278 0.1944868
## [1] 5.1134894 0.2378609
```

```
co5 = compare_outcomes(3000)
```

```
## [1] 5.1208540 0.2551937
## [1] 5.0997004 0.2214241
## [1] 5.1640043 0.2210088
## [1] 5.1412013 0.1895497
## [1] 5.1461905 0.1950478
## [1] 5.0828853 0.2331389
## [1] 5.1269456 0.1980545
## [1] 5.0509876 0.2159671
## [1] 5.1686883 0.2059792
## [1] 5.1277889 0.2418061
```

```
co6 = compare_outcomes(6000)
```

```
## [1] 5.1174728 0.2313784
## [1] 5.1366572 0.2239922
## [1] 5.137412 0.227682
## [1] 5.1418120 0.2114482
## [1] 5.1367043 0.2330168
## [1] 5.0871893 0.2471702
## [1] 5.1350230 0.2078851
## [1] 5.0845125 0.2078622
## [1] 5.0748520 0.2213496
## [1] 5.1195163 0.2501353
```

```
co7 = compare_outcomes(8000)
```

```
## [1] 5.1331399 0.2231901
## [1] 5.0997554 0.2267811
## [1] 5.1335083 0.2303717
## [1] 5.1281220 0.2264248
## [1] 5.1508410 0.2235407
## [1] 5.0928086 0.2741698
## [1] 5.1427067 0.2338609
## [1] 5.1514919 0.2311643
## [1] 5.0946673 0.2276387
## [1] 5.1446557 0.2226232
```

```
co1
```

```
## [1] 5.155032
```

```
co2
```

```
## [1] 5.12655
```

```
co3
```

```
## [1] 5.133685
```

```
co4
```

```
## [1] 5.146103
```

```
co5
```

```
## [1] 5.122925
```

```
co6
```

```
## [1] 5.117115
```

```
co7
```

```
## [1] 5.12717
```

```
co = c(co1, co2, co3, co4, co5, co6, co7) - 5

me = mean(co)
```

The mean of a is 5.1550319 for the iteration of 1000 times. Thus, the estimation error is 0.1550319.

The mean of a is 5.1265497 for the iteration of 10000 times. Thus, the estimation error is 0.1265497.

The mean of a is 5.1336855 for the iteration of 100000 times.Thus, the estimation error is 0.1336855.

The mean of a is 5.1461035 for the iteration of 2000 times.Thus, the estimation error is 0.1461035.

The mean of a is 5.1229246 for the iteration of 3000 times.Thus, the estimation error is 0.1229246.

The mean of a is 5.1171152 for the iteration of 6000 times.Thus, the estimation error is 0.1171152.

The mean of a is 5.1271697 for the iteration of 8000 times.Thus, the estimation error is 0.1271697.

From the results, we see that the predcted value of the slope tends to have a mean around 5.1550319 with standard deviation around 0.22. Also, the outcomes seem to be relatively stable since the means and standard deviations in each of the 10 rounds are close to each other. Also, the values we get for different iterations are also similar which means that the markov chain is close to its stationary distribution when we iterate is over 1000 times. Also, the accurace of this algorithm is pretty good since the average error of the slope is about 0.1326543. Also, this value doesn't change much when we increase the number of iterations. Thus, this algorithm is both accurate and stable.