AdvStDaAn, Worksheet, Week 12

Michael Lappert

20 Juni, 2022

Contents

Task 1

Approximate the integral

$\int_{-1}^{2}\left(x^2 * cos(0.6x) + 0.2\right)dx$

via Monte Carlo simulation.

```
# from solutionsheet:
f = function(x){
  return(x^2*cos(.6*x)+.2)
  }
x = seq(-1,2,.01)
x
```

```
##   [1] -1.00 -0.99 -0.98 -0.97 -0.96 -0.95 -0.94 -0.93 -0.92 -0.91 -0.90 -0.89
##  [13] -0.88 -0.87 -0.86 -0.85 -0.84 -0.83 -0.82 -0.81 -0.80 -0.79 -0.78 -0.77
##  [25] -0.76 -0.75 -0.74 -0.73 -0.72 -0.71 -0.70 -0.69 -0.68 -0.67 -0.66 -0.65
##  [37] -0.64 -0.63 -0.62 -0.61 -0.60 -0.59 -0.58 -0.57 -0.56 -0.55 -0.54 -0.53
##  [49] -0.52 -0.51 -0.50 -0.49 -0.48 -0.47 -0.46 -0.45 -0.44 -0.43 -0.42 -0.41
##  [61] -0.40 -0.39 -0.38 -0.37 -0.36 -0.35 -0.34 -0.33 -0.32 -0.31 -0.30 -0.29
##  [73] -0.28 -0.27 -0.26 -0.25 -0.24 -0.23 -0.22 -0.21 -0.20 -0.19 -0.18 -0.17
##  [85] -0.16 -0.15 -0.14 -0.13 -0.12 -0.11 -0.10 -0.09 -0.08 -0.07 -0.06 -0.05
##  [97] -0.04 -0.03 -0.02 -0.01  0.00  0.01  0.02  0.03  0.04  0.05  0.06  0.07
```

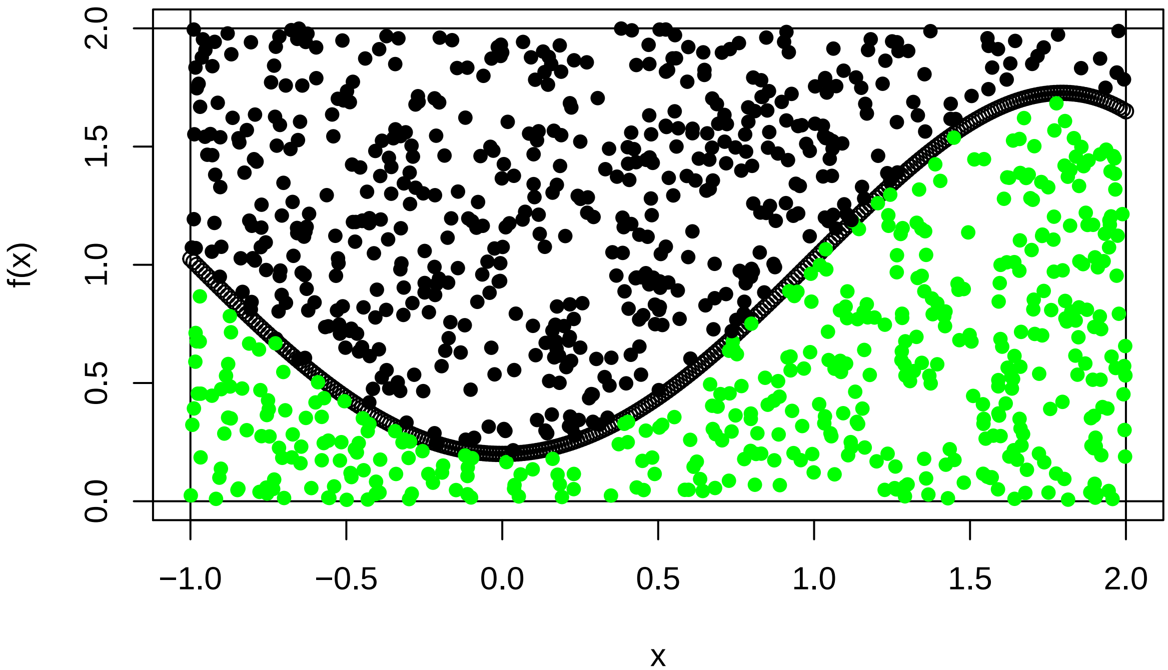
```
## [109]  0.08  0.09  0.10  0.11  0.12  0.13  0.14  0.15  0.16  0.17  0.18  0.19
## [121]  0.20  0.21  0.22  0.23  0.24  0.25  0.26  0.27  0.28  0.29  0.30  0.31
## [133]  0.32  0.33  0.34  0.35  0.36  0.37  0.38  0.39  0.40  0.41  0.42  0.43
## [145]  0.44  0.45  0.46  0.47  0.48  0.49  0.50  0.51  0.52  0.53  0.54  0.55
## [157]  0.56  0.57  0.58  0.59  0.60  0.61  0.62  0.63  0.64  0.65  0.66  0.67
## [169]  0.68  0.69  0.70  0.71  0.72  0.73  0.74  0.75  0.76  0.77  0.78  0.79
## [181]  0.80  0.81  0.82  0.83  0.84  0.85  0.86  0.87  0.88  0.89  0.90  0.91
## [193]  0.92  0.93  0.94  0.95  0.96  0.97  0.98  0.99  1.00  1.01  1.02  1.03
## [205]  1.04  1.05  1.06  1.07  1.08  1.09  1.10  1.11  1.12  1.13  1.14  1.15
## [217]  1.16  1.17  1.18  1.19  1.20  1.21  1.22  1.23  1.24  1.25  1.26  1.27
## [229]  1.28  1.29  1.30  1.31  1.32  1.33  1.34  1.35  1.36  1.37  1.38  1.39
## [241]  1.40  1.41  1.42  1.43  1.44  1.45  1.46  1.47  1.48  1.49  1.50  1.51
## [253]  1.52  1.53  1.54  1.55  1.56  1.57  1.58  1.59  1.60  1.61  1.62  1.63
## [265]  1.64  1.65  1.66  1.67  1.68  1.69  1.70  1.71  1.72  1.73  1.74  1.75
## [277]  1.76  1.77  1.78  1.79  1.80  1.81  1.82  1.83  1.84  1.85  1.86  1.87
## [289]  1.88  1.89  1.90  1.91  1.92  1.93  1.94  1.95  1.96  1.97  1.98  1.99
## [301]  2.00
```

```r
y = f(seq(-1,2,.01))
y
```

```
##   [1] 1.0253356 1.0122173 0.9991025 0.9859949 0.9728980 0.9598156 0.9467512
##   [8] 0.9337083 0.9206905 0.9077013 0.8947440 0.8818223 0.8689394 0.8560988
##  [15] 0.8433039 0.8305579 0.8178642 0.8052260 0.7926465 0.7801291 0.7676768
##  [22] 0.7552927 0.7429801 0.7307420 0.7185815 0.7065015 0.6945051 0.6825952
##  [29] 0.6707747 0.6590465 0.6474136 0.6358786 0.6244445 0.6131138 0.6018895
##  [36] 0.5907741 0.5797703 0.5688807 0.5581079 0.5474545 0.5369229 0.5265156
##  [43] 0.5162351 0.5060837 0.4960638 0.4861778 0.4764279 0.4668164 0.4573455
##  [50] 0.4480174 0.4388341 0.4297979 0.4209107 0.4121746 0.4035916 0.3951636
##  [57] 0.3868925 0.3787802 0.3708285 0.3630392 0.3554141 0.3479548 0.3406630
##  [64] 0.3335403 0.3265884 0.3198088 0.3132029 0.3067723 0.3005184 0.2944424
##  [71] 0.2885459 0.2828301 0.2772962 0.2719455 0.2667791 0.2617982 0.2570038
##  [78] 0.2523971 0.2479790 0.2437504 0.2397123 0.2358657 0.2322112 0.2287498
##  [85] 0.2254821 0.2224089 0.2195309 0.2168486 0.2143627 0.2120737 0.2099820
##  [92] 0.2080882 0.2063926 0.2048957 0.2035977 0.2024989 0.2015995 0.2008999
##  [99] 0.2004000 0.2001000 0.2000000 0.2001000 0.2004000 0.2008999 0.2015995
## [106] 0.2024989 0.2035977 0.2048957 0.2063926 0.2080882 0.2099820 0.2120737
## [113] 0.2143627 0.2168486 0.2195309 0.2224089 0.2254821 0.2287498 0.2322112
## [120] 0.2358657 0.2397123 0.2437504 0.2479790 0.2523971 0.2570038 0.2617982
## [127] 0.2667791 0.2719455 0.2772962 0.2828301 0.2885459 0.2944424 0.3005184
## [134] 0.3067723 0.3132029 0.3198088 0.3265884 0.3335403 0.3406630 0.3479548
## [141] 0.3554141 0.3630392 0.3708285 0.3787802 0.3868925 0.3951636 0.4035916
## [148] 0.4121746 0.4209107 0.4297979 0.4388341 0.4480174 0.4573455 0.4668164
## [155] 0.4764279 0.4861778 0.4960638 0.5060837 0.5162351 0.5265156 0.5369229
## [162] 0.5474545 0.5581079 0.5688807 0.5797703 0.5907741 0.6018895 0.6131138
## [169] 0.6244445 0.6358786 0.6474136 0.6590465 0.6707747 0.6825952 0.6945051
## [176] 0.7065015 0.7185815 0.7307420 0.7429801 0.7552927 0.7676768 0.7801291
## [183] 0.7926465 0.8052260 0.8178642 0.8305579 0.8433039 0.8560988 0.8689394
## [190] 0.8818223 0.8947440 0.9077013 0.9206905 0.9337083 0.9467512 0.9598156
## [197] 0.9728980 0.9859949 0.9991025 1.0122173 1.0253356 1.0384538 1.0515681
## [204] 1.0646748 1.0777701 1.0908503 1.1039116 1.1169501 1.1299621 1.1429435
## [211] 1.1558906 1.1687994 1.1816660 1.1944864 1.2072567 1.2199729 1.2326308
## [218] 1.2452266 1.2577562 1.2702154 1.2826002 1.2949066 1.3071302 1.3192671
## [225] 1.3313130 1.3432639 1.3551154 1.3668633 1.3785036 1.3900318 1.4014439
```

```
## [232] 1.4127354 1.4239022 1.4349399 1.4458442 1.4566109 1.4672355 1.4777138
## [239] 1.4880415 1.4982140 1.5082271 1.5180765 1.5277576 1.5372661 1.5465977
## [246] 1.5557478 1.5647121 1.5734862 1.5820655 1.5904458 1.5986224 1.6065911
## [253] 1.6143472 1.6218864 1.6292042 1.6362961 1.6431577 1.6497845 1.6561720
## [260] 1.6623157 1.6682112 1.6738539 1.6792395 1.6843633 1.6892211 1.6938081
## [267] 1.6981201 1.7021525 1.7059009 1.7093607 1.7125276 1.7153970 1.7179646
## [274] 1.7202258 1.7221762 1.7238113 1.7251269 1.7261183 1.7267812 1.7271112
## [281] 1.7271039 1.7267549 1.7260597 1.7250141 1.7236137 1.7218540 1.7197308
## [288] 1.7172398 1.7143765 1.7111367 1.7075162 1.7035105 1.6991155 1.6943268
## [295] 1.6891403 1.6835518 1.6775569 1.6711516 1.6643316 1.6570928 1.6494310
```

```
plot(x, y, ylim=c(0,2), xlab="x", ylab="f(x)")
abline(h=c(0,2), v=c(-1,2))
n = 1000
hits = 0
for(i in 1:n)
{
    x_ = runif(1,-1,2)
    y_ = runif(1,0,2)
    col="black"
    if(y_<f(x_))
    {
        hits = hits+1
        col="green"
    }
    points(x_, y_, pch=16, col=col)
}
```




```r
# hits/n = I/6  =>  I = hits/n*6
hits/n*6 # Exact Integral: 2.50
```

```
## [1] 2.652
```

Task 2

Proof: The stationary distribution is $(\frac{9}{27}, \frac{8}{27}, \frac{10}{27})$.
Does this Markov model satisfy 'detailed balance'?

---

stationary distribution:

9/27 = 9/27 * 0.0 + 8/27 * 0.5 + 10/27 * 0.5 ok
8/27 = 9/27 * 0.8 + 8/27 * 0.1 + 10/27 * 0.0 ok
10/27 = 9/27 * 0.2 + 8/27 * 0.4 + 10/27 * 0.5 ok

=> $(9/27, 8/27, 10/27)$ is stationary distribution.

detailed balance:

consider state 2 and 3: 8/27 * 0.4 is not 10/27 * 0
=> detailed balance does not hold.

Task 3

Find the equilibrium distribution.

Does this Markov model satisfy 'detailed balance'?

From solution:

Stationary distribution:

```
m = matrix(c(.5, 0, 1, .25, .5, .0, .25, .5, 0), 3)
m
```

```
##      [,1] [,2] [,3]
## [1,]  0.5 0.25 0.25
## [2,]  0.0 0.50 0.50
## [3,]  1.0 0.00 0.00
```

```
m = m%*%m
m
```

```
##      [,1] [,2] [,3]
## [1,]  0.5 0.25 0.25
## [2,]  0.5 0.25 0.25
## [3,]  0.5 0.25 0.25
```

```
m = m%*%m
m
```

```
##      [,1] [,2] [,3]
## [1,]  0.5 0.25 0.25
## [2,]  0.5 0.25 0.25
## [3,]  0.5 0.25 0.25
```

```
# m doesn't change any more
# => (0.5,0.25,0.25) is stationary distribution
# proof this!
```

proof:

0.50 = 0.50 * 0.5 + 0.25 * 0.0 + 0.25 * 1.0 ok
0.25 = 0.50 * 0.25 + 0.25 * 0.5 + 0.25 * 0.0 ok
0.25 = 0.50 * 0.25 + 0.25 * 0.5 + 0.25 * 0.0 ok

detailed balance:

consider state 2 and 3: 0.25 * 0.5 is not 0.25 * 0
=> detailed balance does not hold.

Task 4

Find the equilibrium distribution.

Does this Markov model satisfy 'detailed balance'?

What's the expected number of sunny days per year?

Stationary distribution:

```r
m = matrix(c(0.8, 0.4, 0.2, 0.6), 2)
m
```

```
##      [,1] [,2]
## [1,]  0.8  0.2
## [2,]  0.4  0.6
```

```r
m = m %*% m
m
```

```
##      [,1] [,2]
## [1,] 0.72 0.28
## [2,] 0.56 0.44
```

```r
(m = m %*% m)
```

```
##        [,1]   [,2]
## [1,] 0.6752 0.3248
## [2,] 0.6496 0.3504
```

```r
#           [,1]      [,2]
# [1,] 0.6666667 0.3333333
# [2,] 0.6666667 0.3333333
# -> does not change anymore
# (0.67, 0.33) is the stationary distribution
```

Proof: 0.67 = 0.67*0.8 + 0.33*0.4 = 0.6666667 -> ok 0.33 = 0.67*0.2 + 0.33*0.6 = 0.3333333 -> ok

Detailed balance:
2/3 * 0.2 = 0.1333333 = 1/3 * 0.4 = 0.1333333 -> satisfies detailed balance

The expected number of sunny days per year is 365 * 2/3 = 243.3333333.

Task 5

Recall our salmon selling company example: 6 out of 16 persons signed up.

Infer the sign-up rate with a Markov chain Monte Carlo method. Consider a uniform and a Beta(2, 20) prior on the sign-up-rate and compare your results with the exact results from the theoretical approach using a conjugate prior.

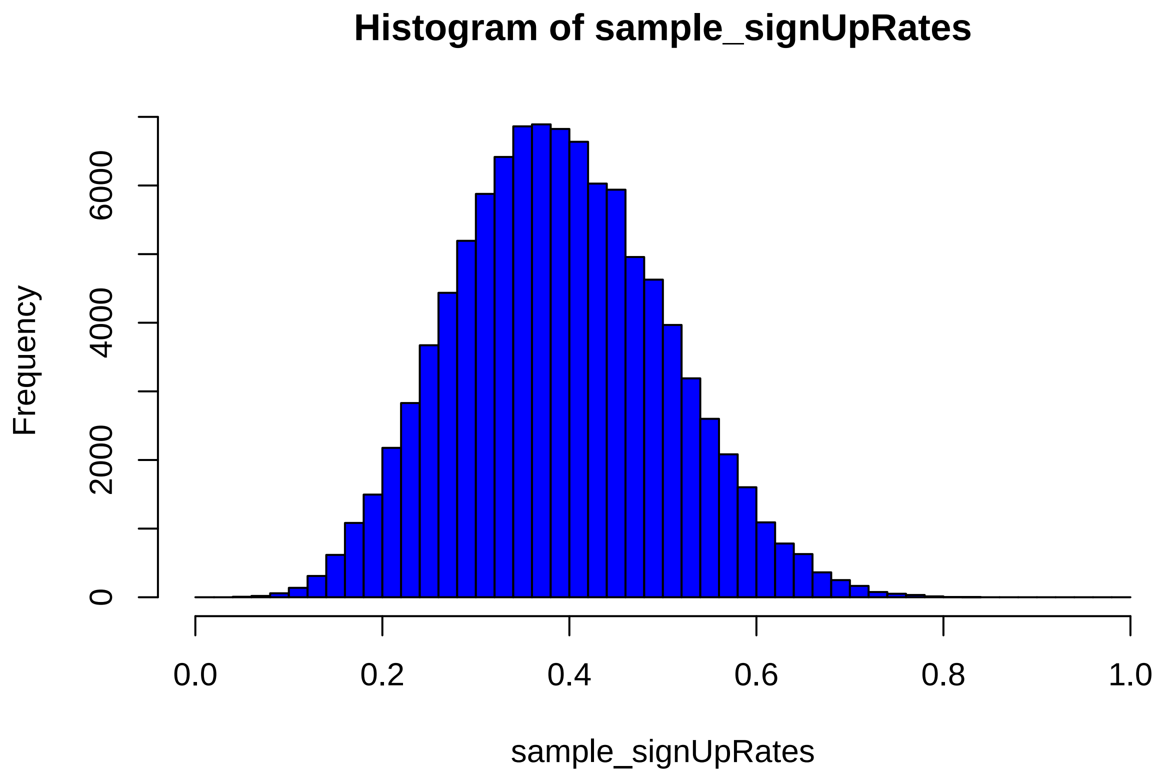
From solution:

uniform prior:

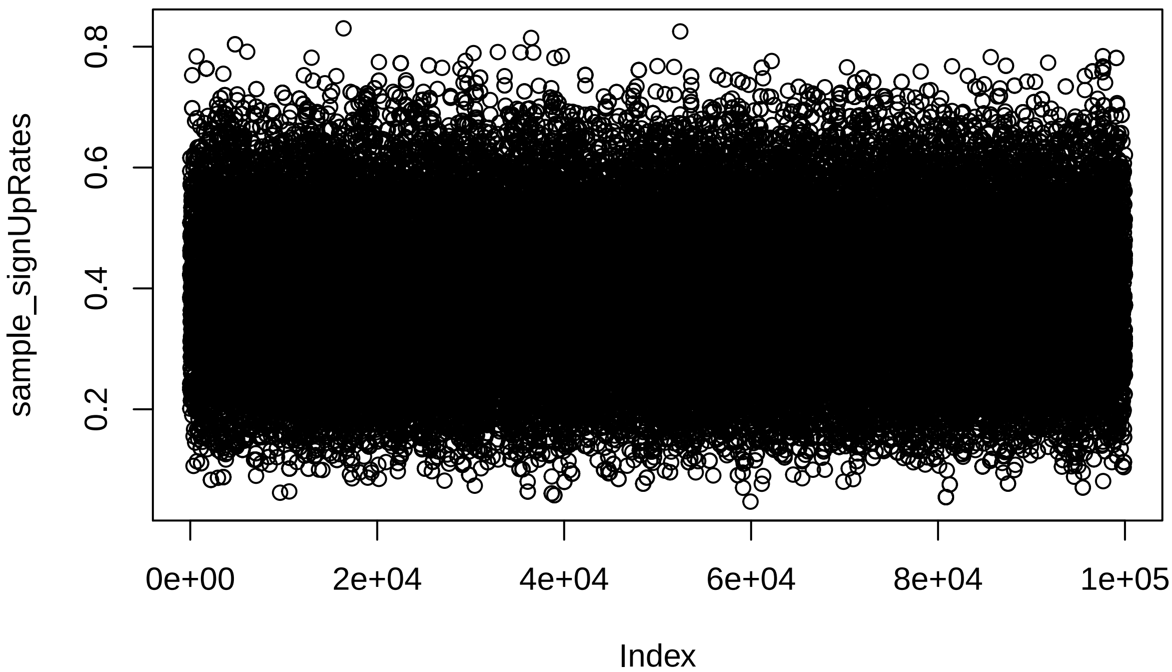
```
# 6 from 16 sign up for salomon
# Baysian Data Analysis via MCMC
nrSignups =  6
nrTotal   = 16

nSimulations = 100000

signUpRate_i = runif(1)
signUpRate_i
```

```
## [1] 0.6866836
```

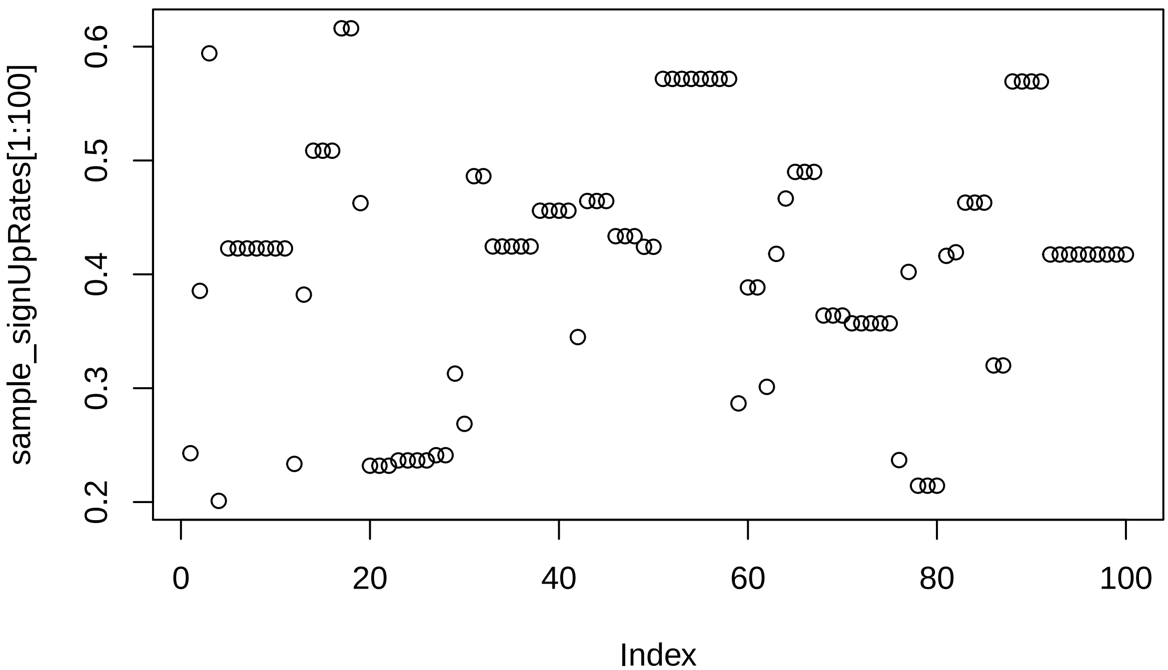
```
sample_signUpRates = rep(NA,nSimulations)

for(i in 1:nSimulations)
{
    signUpRate_prop = runif(1)
    u = runif(1)
    if( u < (dbinom(nrSignups,nrTotal,signUpRate_prop)/dbinom(nrSignups,nrTotal,signUpRate_i)))
    {
        signUpRate_i = signUpRate_prop
    }
    sample_signUpRates[i] = signUpRate_i
}


# chain with high correlation:
hist(sample_signUpRates,breaks=seq(0,1,.02),col="blue")
```



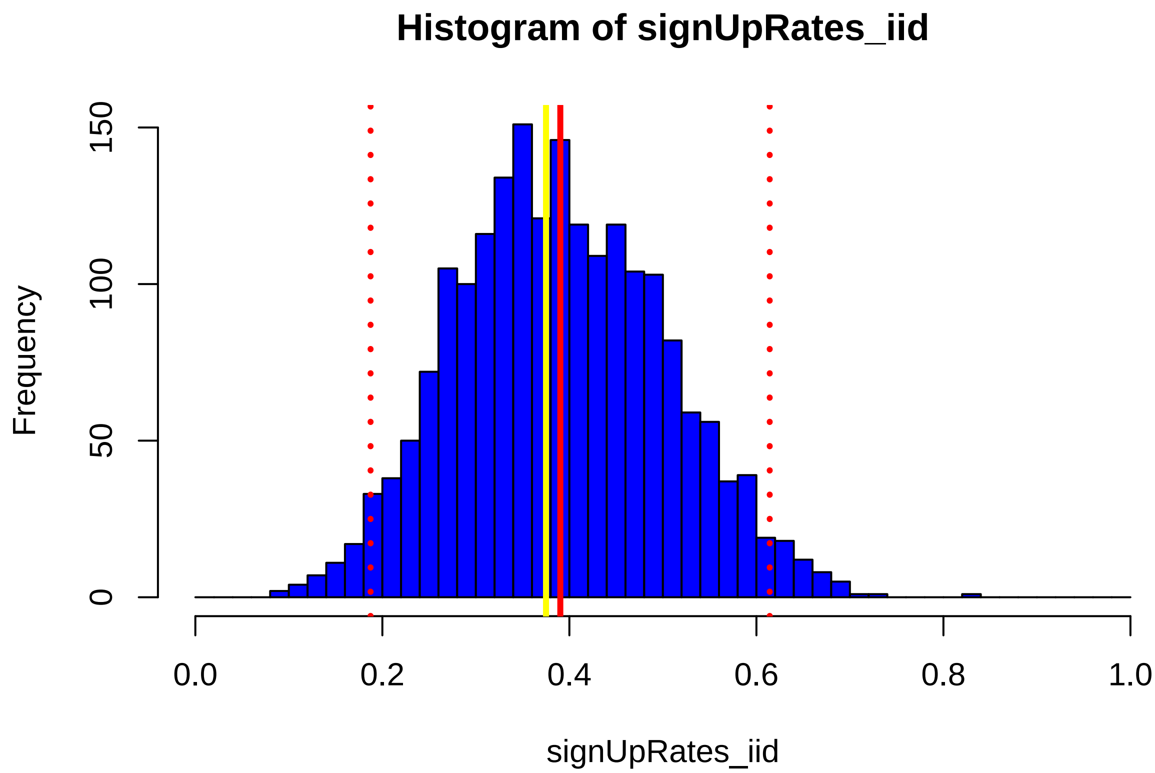

```
plot(sample_signUpRates)
```



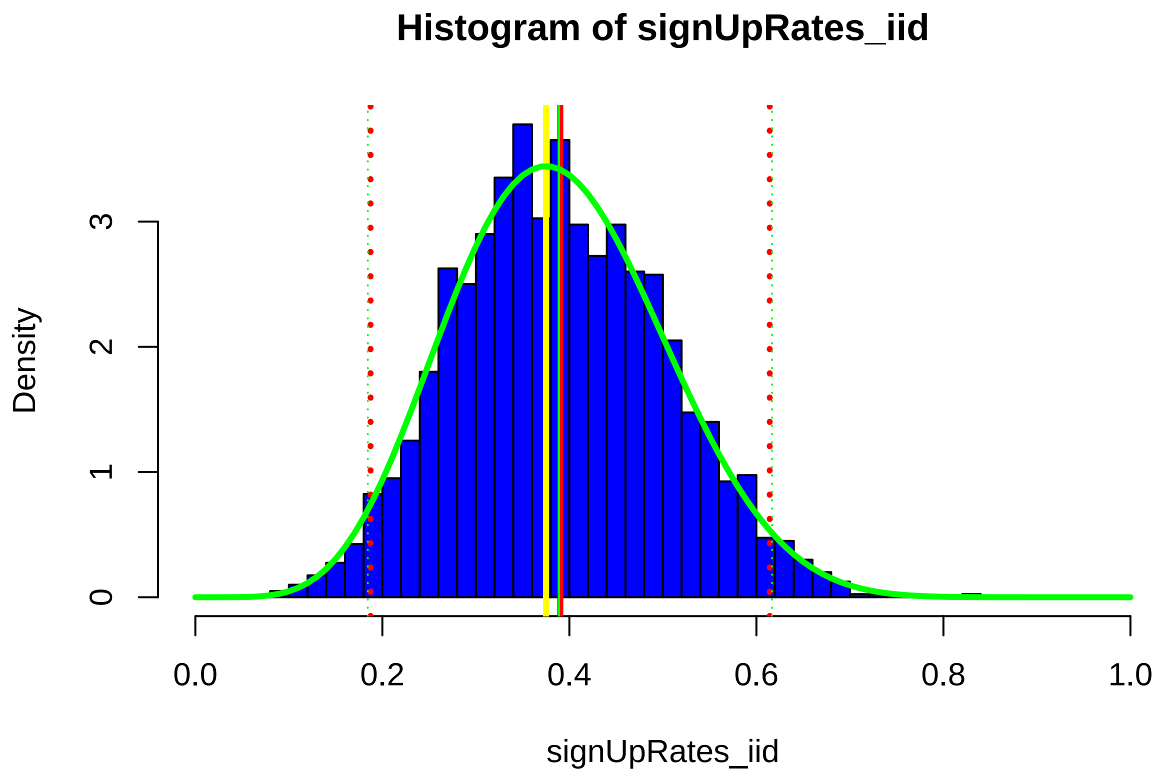

```
plot(sample_signUpRates[1:100])
```




```
# Result:
signUpRates_iid = sample_signUpRates[seq(100,nSimulations,50)]
hist(signUpRates_iid,breaks=seq(0,1,.02),col="blue")
abline(v=nrSignups/nrTotal,col="yellow",lwd=3)
abline(v=mean(signUpRates_iid),col="red",lwd=3)
abline(v=quantile(signUpRates_iid,c(0.025,.975)),col="red",lty=3,lwd=3)
```



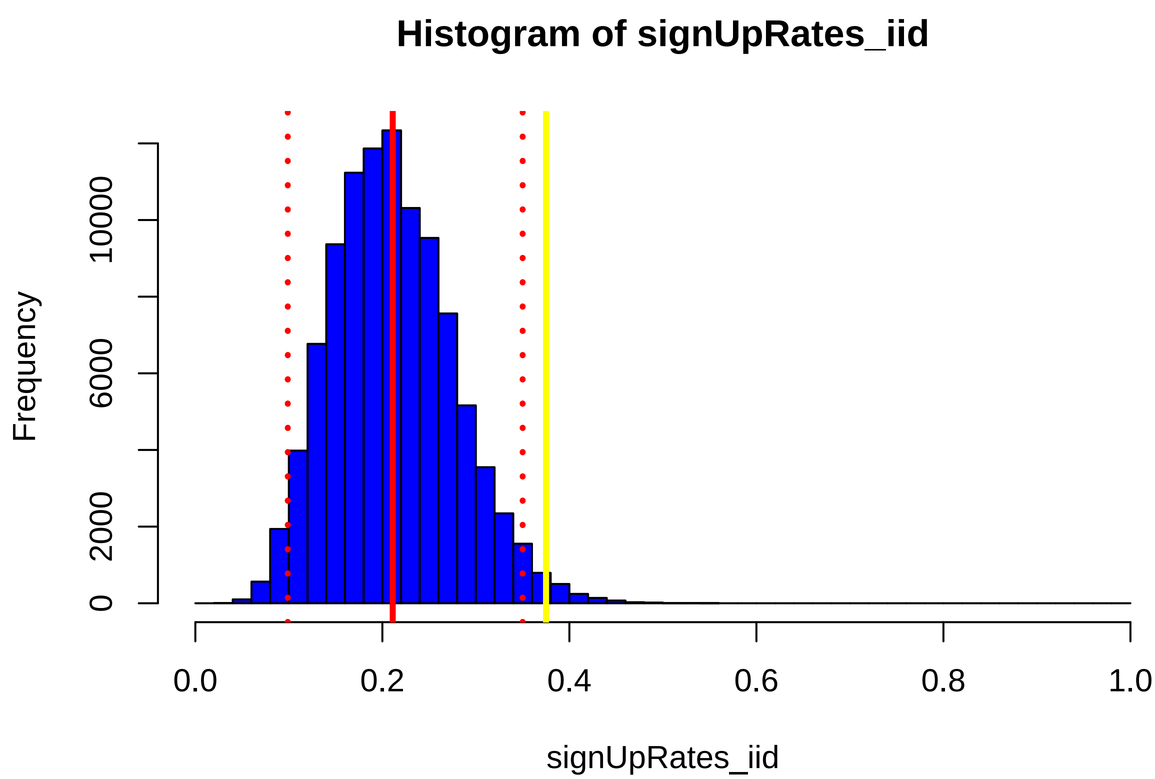

```r
# Compare with theoretical result:
hist(signUpRates_iid,breaks=seq(0,1,.02),col="blue",freq=FALSE)
abline(v=nrSignups/nrTotal,col="yellow",lwd=3)
abline(v=mean(signUpRates_iid),col="red",lwd=3)
abline(v=quantile(signUpRates_iid,c(0.025,.975)),col="red",lty=3,lwd=3)
# theoretical results in green
curve(dbeta(x,1+6,1+10),add=T,lwd=3,col="green")
abline(v=7/18,col="green",lwd=1)
abline(v=qbeta(c(0.025,.975),7,11),col="green",lty=3,lwd=1)
```



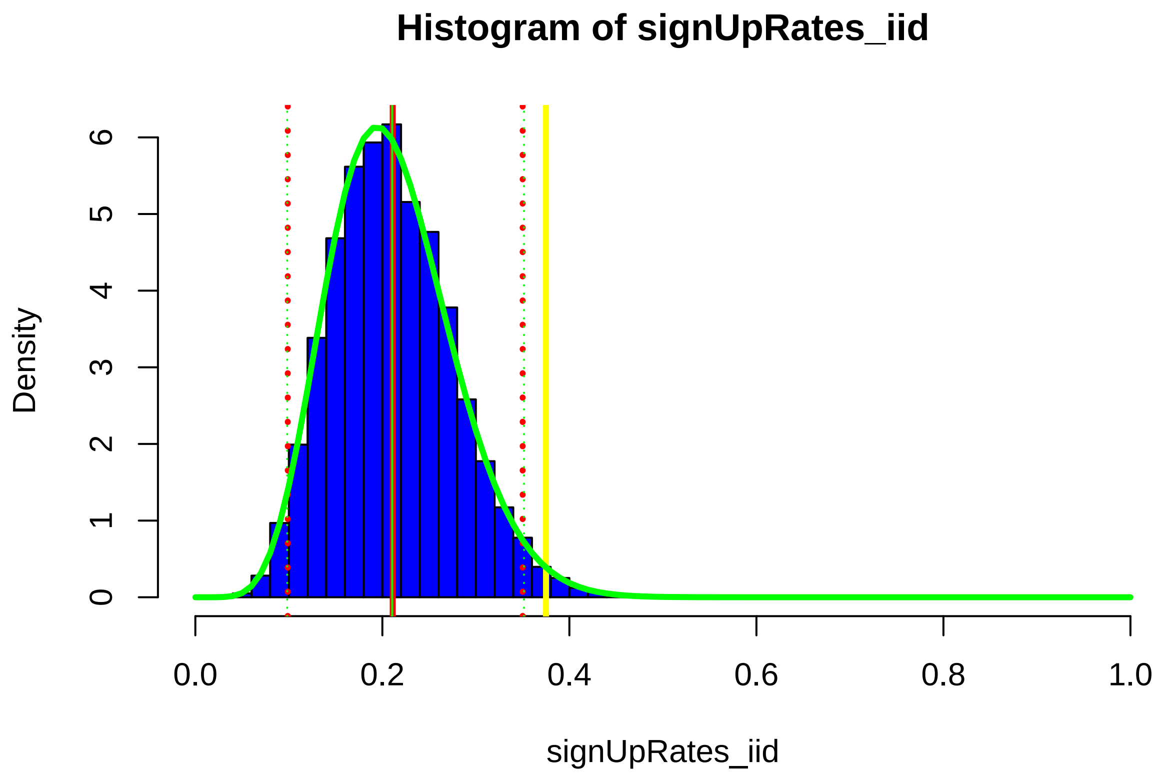
Beta(2, 20)-prior:

```
# 6 from 16 sign up for salomon
# Baysian Data Analysis via MCMC
nrSignups =  6
nrTotal   = 16

nSimulations = 100000

signUpRate_i = rbeta(1, 2, 20)

sample_signUpRates = rep(NA,nSimulations)

for(i in 1:nSimulations)
{
    signUpRate_prop = runif(1)
    u = runif(1)
    if( u < (dbinom(nrSignups,nrTotal,signUpRate_prop)*dbeta(signUpRate_prop,2,20))
             /(dbinom(nrSignups,nrTotal,signUpRate_i)*dbeta(signUpRate_i,2,20)) )
    {
        signUpRate_i = signUpRate_prop
    }
    sample_signUpRates[i] = signUpRate_i
}
```

```
# chain with high correlation:
#hist(sample_signUpRates,breaks=seq(0,1,.02),col="blue")
#plot(sample_signUpRates)
#plot(sample_signUpRates[1:100])

# Result:
signUpRates_iid = sample_signUpRates#= sample_signUpRates[seq(100,nSimulations,50)]
hist(signUpRates_iid,breaks=seq(0,1,.02),col="blue")
abline(v=nrSignups/nrTotal,col="yellow",lwd=3)
abline(v=mean(signUpRates_iid),col="red",lwd=3)
abline(v=quantile(signUpRates_iid,c(0.025,.975)),col="red",lty=3,lwd=3)
```




```
# Compare with theoretical result:
hist(signUpRates_iid,breaks=seq(0,1,.02),col="blue",freq=FALSE)
abline(v=nrSignups/nrTotal,col="yellow",lwd=3)
abline(v=mean(signUpRates_iid),col="red",lwd=3)
abline(v=quantile(signUpRates_iid,c(0.025,.975)),col="red",lty=3,lwd=3)
# theoretical results in green
curve(dbeta(x,2+6,20+10),add=T,lwd=3,col="green")
abline(v=8/38,col="green",lwd=1)
abline(v=qbeta(c(0.025,.975),8,30),col="green",lty=3,lwd=1)
```



Task 6

Let

x = c(3.4, -4.2, -0.7, -2.6, -1.6, -1.2, -2.2, -3.7, -0.9, -3.1)

be some observations with corresponding responses

y = c(5.7, -0.1, 4.7, 2.7, 3.1, 2.3, 3.4, 1.0, 2.2, 1.5).

The values were simulated by the formular: y = 3 + 0.3*x + rnorm(10,0,1).

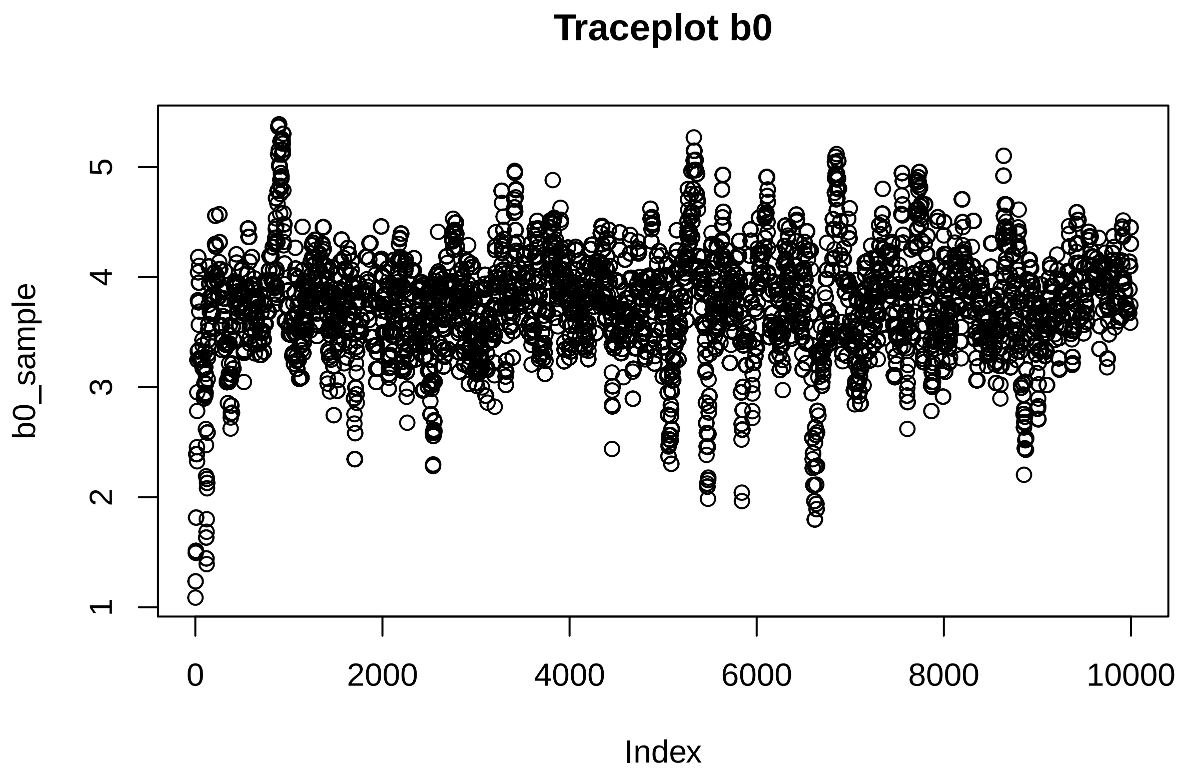
Consider a linear model, i.e. $y_i = \beta_0 + \beta_1 x_i + \epsilon_i$, with Gaussian noise $\epsilon_i \sim Norm(o, \sigma^2)$. Assume a Gamma(1,1)- prior on sigma and normal distributed priors with mean 0 and standard deviation 3 on β_0 and β_1.

Perform a Bayesian analysis: Estimate the posterior mean and credible intervals for β_0, β_1 and sigma.

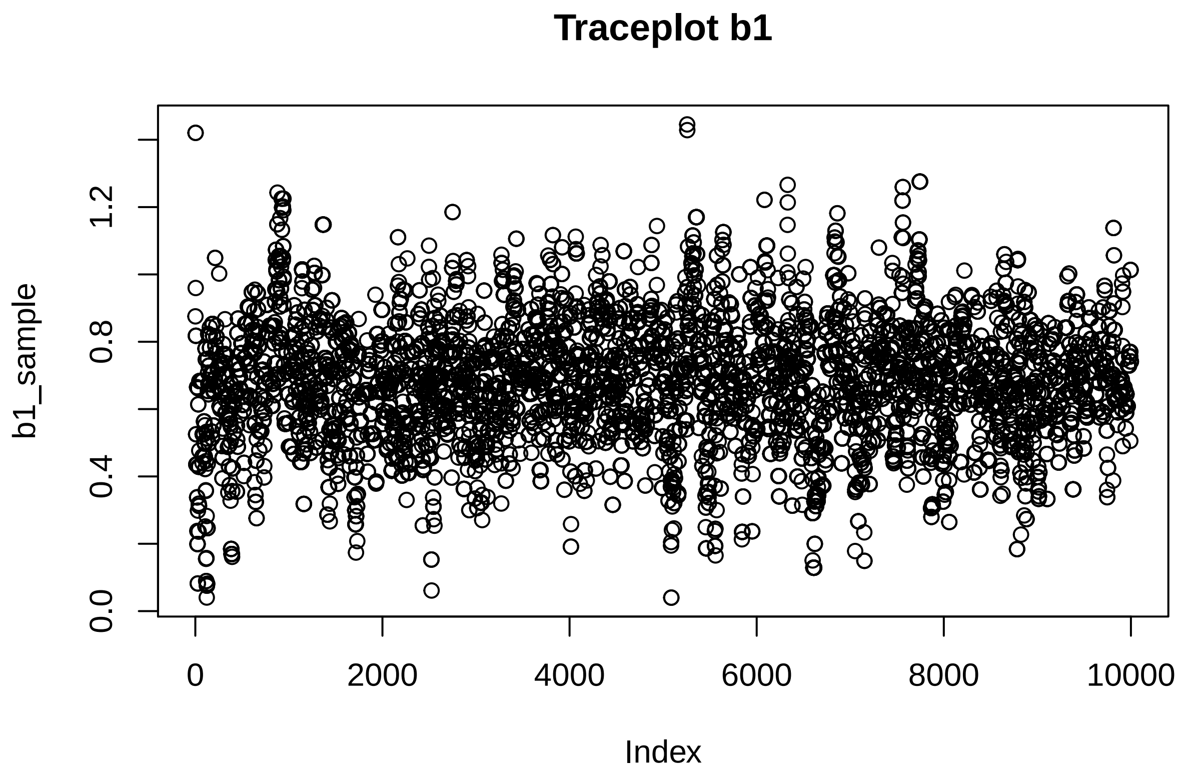
From solution:

```
# Observed data:
obs_x = c(3.4, -4.2, -0.7, -2.6, -1.6, -1.2, -2.2, -3.7, -0.9, -3.1)
obs_y = c(5.7,-0.1, 4.7, 2.7, 3.1, 2.3, 3.4, 1.0, 2.2, 1.5)
```

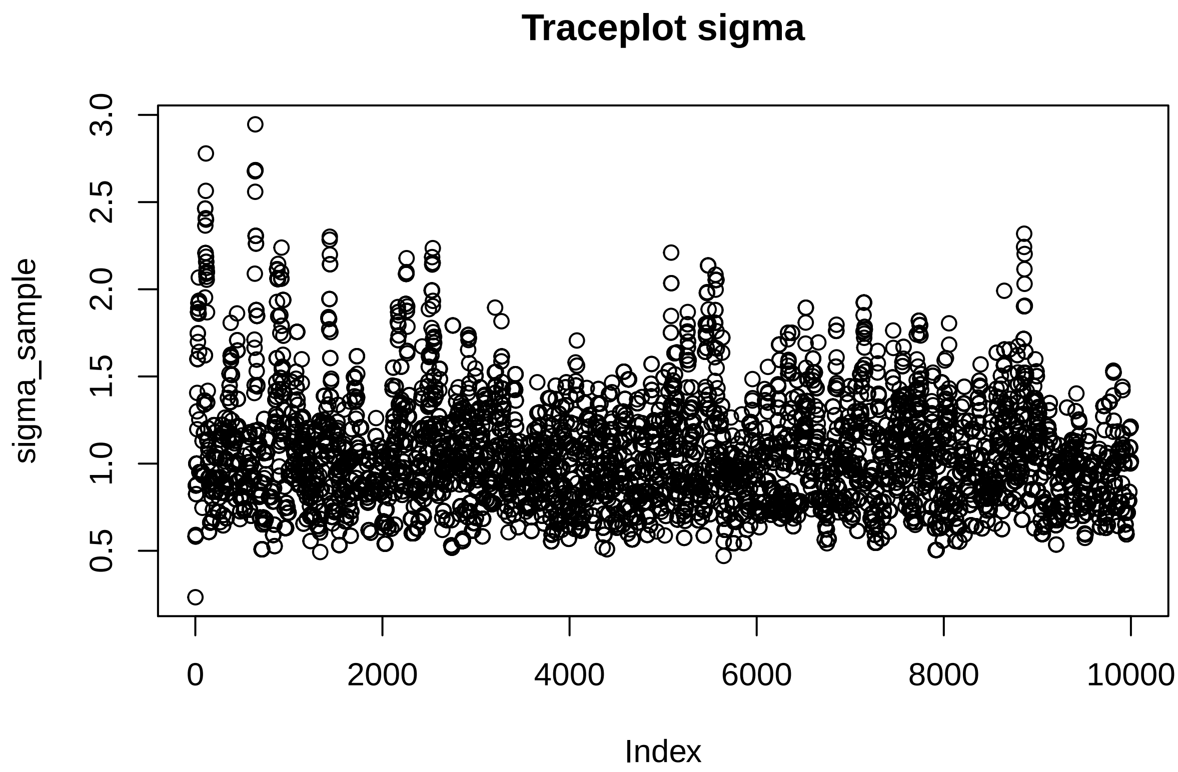
```r
# Log of (unnormalized) posterior density
logPosterior = function(b0_, b1_, sigma_){
   sum(dnorm(b0_+b1_*obs_x - obs_y, 0, sigma_, log=TRUE)) +
    dnorm(b0_, 0, 3, log=TRUE) +
    dnorm(b1_, 0, 3, log=TRUE) +
    dgamma(sigma_, 1, 1, log=TRUE)
}
# Starting value
b0 = runif(1,0,1)
b1 = runif(1,0,1)
sigma = runif(1,0,1)
# Collect sampled values
b0_sample = c()
b1_sample = c()
sigma_sample = c()
# Bayesian Data Analysis via MCMC
for(i in 1:10000) {
    b0_prop = rnorm(1, b0, 0.3)
    b1_prop = rnorm(1, b1, 0.3)
    s_prop = abs(rnorm(1, sigma, 0.3))
    R = exp(logPosterior(b0_prop, b1_prop, s_prop)-logPosterior(b0, b1, sigma))
    u = runif(1)
    if( u <  R)
    {
        b0 = b0_prop
        b1 = b1_prop
        sigma = s_prop
    }
    b0_sample = c(b0_sample,b0)
    b1_sample = c(b1_sample,b1)
    sigma_sample = c(sigma_sample,sigma)
}
# Evaluation
plot(b0_sample, main='Traceplot b0')
```



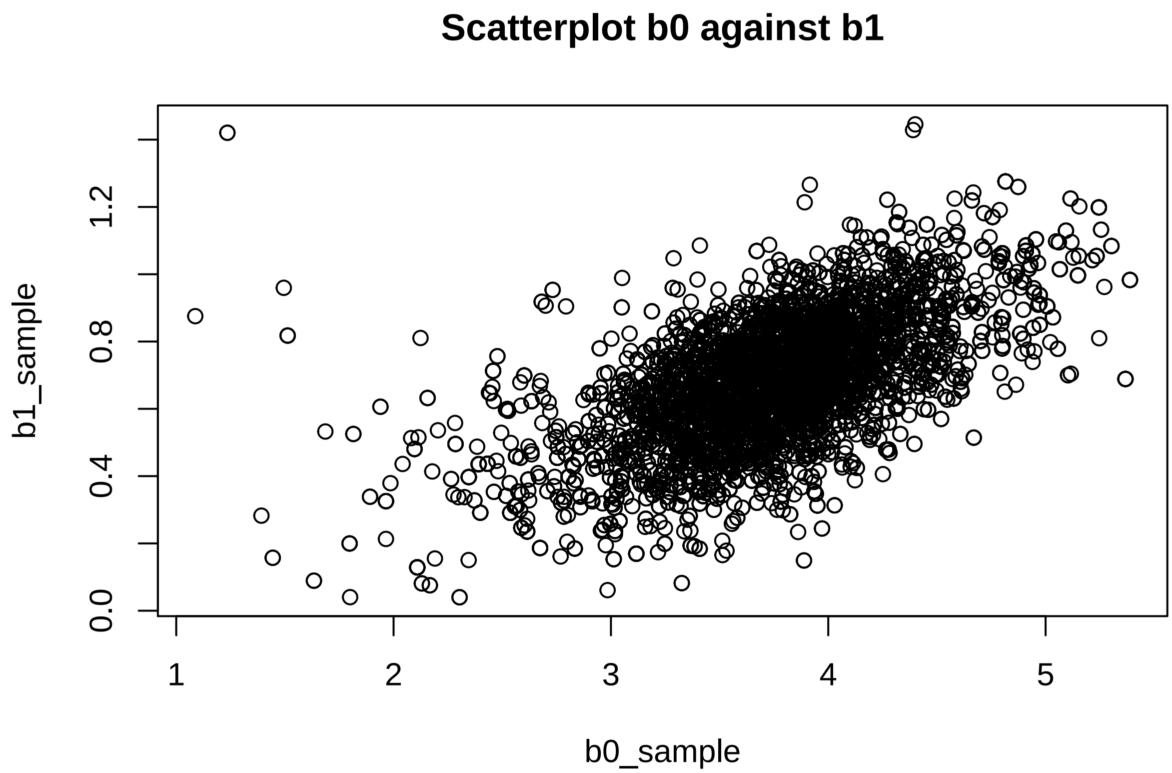

```
plot(b1_sample, main='Traceplot b1')
```



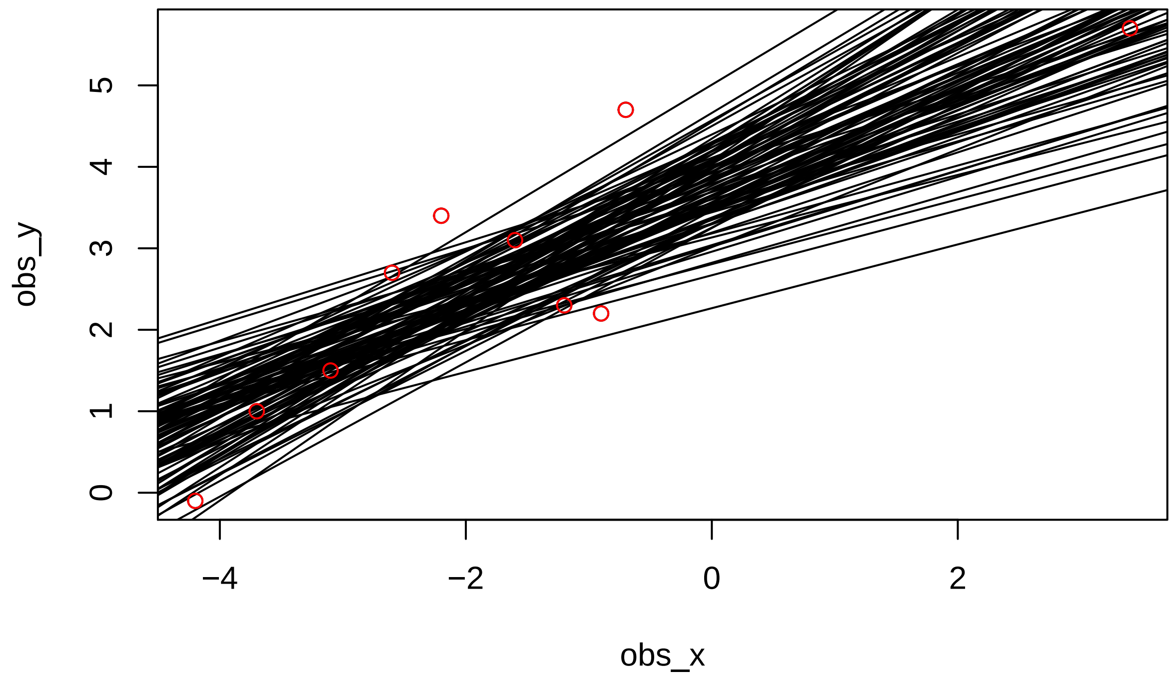

```
plot(sigma_sample, main='Traceplot sigma')
```




```
plot(b0_sample,b1_sample, main='Scatterplot b0 against b1')
```




```r
# Thinning the chains to get iid samples
b0_iid = b0_sample[1:100*100]
b1_iid = b1_sample[1:100*100]
sigma_iid = sigma_sample[1:100*100]
plot(obs_x,obs_y)
for(i in 1:100)
{
  abline(b0_iid[i],b1_iid[i])
}
points(obs_x,obs_y,col='red')
```




```
mean(b0_iid)
```

```
## [1] 3.736836
```

```
quantile(b0_iid,c(0.025,.975))
```

```
##     2.5%    97.5%
## 2.813580 4.561177
```

```
mean(b1_iid)
```

```
## [1] 0.6751743
```

```
quantile(b1_iid,c(0.025,.975))
```

```
##      2.5%     97.5%
## 0.3979859 0.9608822
```

```
mean(sigma_iid)
```

```
## [1] 0.9872558
```

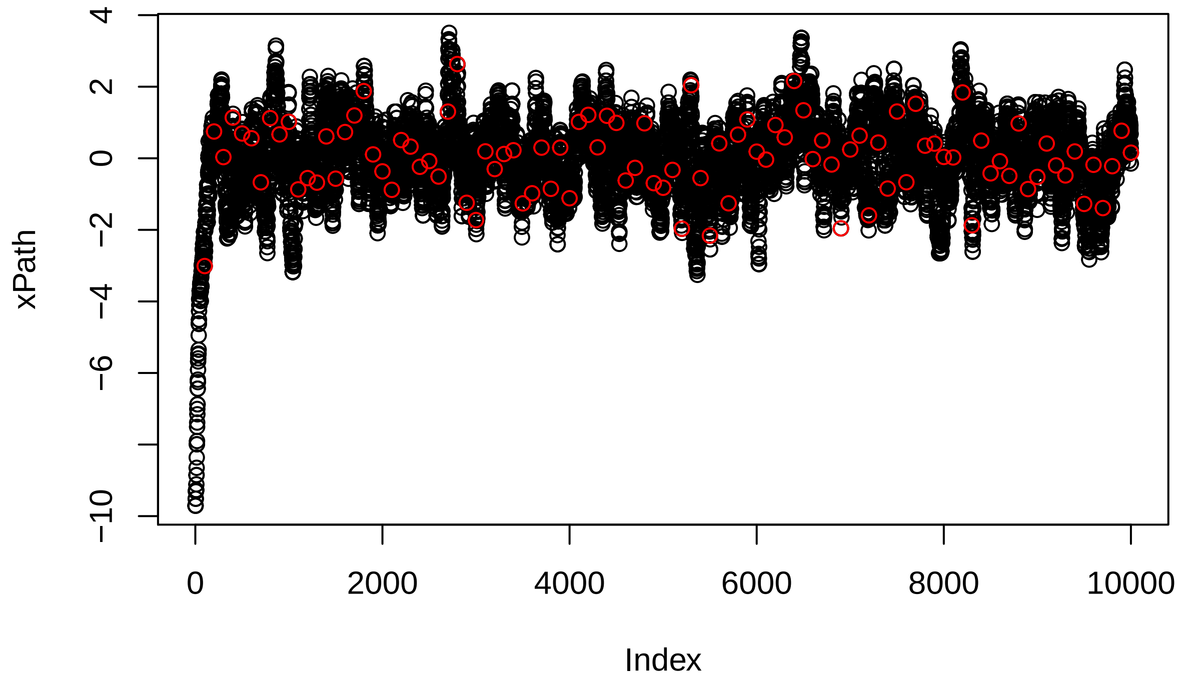
```r
quantile(sigma_iid,c(0.025,.975))
```

```
##      2.5%     97.5% 
## 0.6613533 1.4055060
```

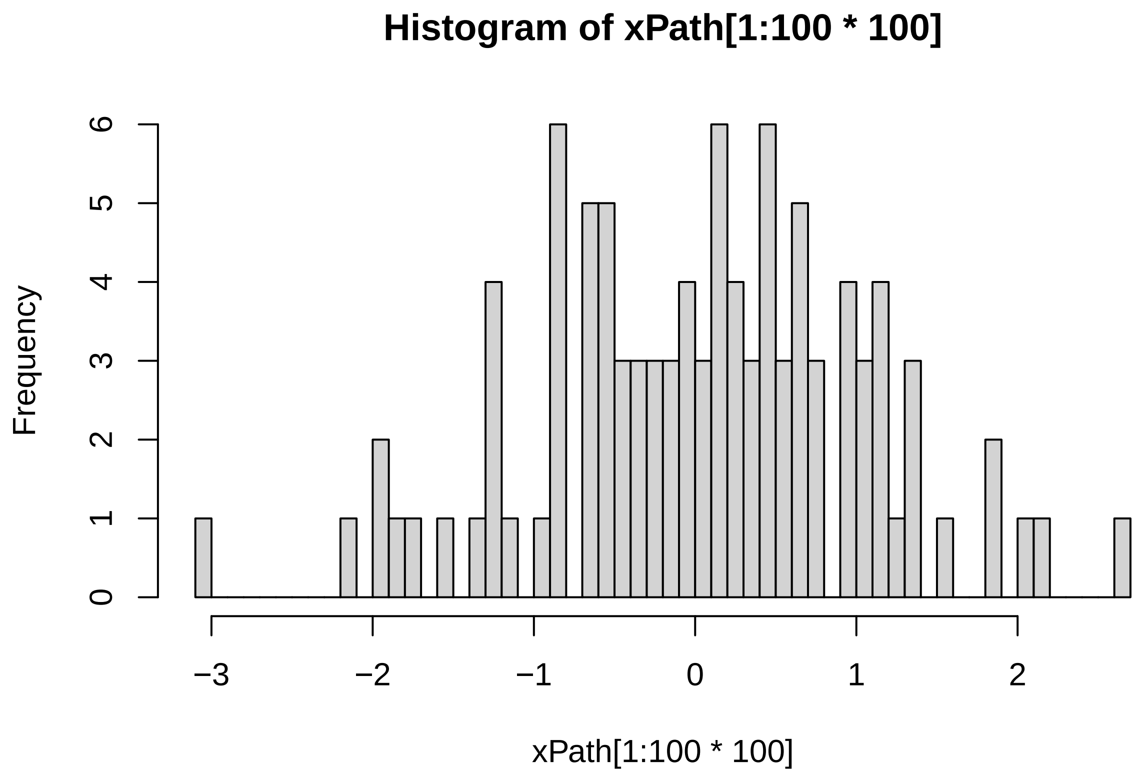
```r
# Compare
lm(obs_y~obs_x)
```

```
## 
## Call:
## lm(formula = obs_y ~ obs_x)
## 
## Coefficients:
## (Intercept)        obs_x  
##      3.8120       0.6917
```

```r
# Code from slide 60 of week 13
x = -10
s = 0.5
xPath = c()

for(i in 1:10000) {
  xProp = x + runif(1,-s,s)
  if(exp(-xProp^2/2)/exp(-x^2/2) > runif(1)) {
    x = xProp
    }
  xPath = c(xPath,x)
}

plot(xPath)
points(1:100*100, xPath[1:100*100], col="red")
```




```
hist(xPath[1:100*100], breaks = 50)
```




```r
runif(1)
```

```
## [1] 0.377749
```

Task 2

Question Task

|>Q1: |»A1:
