STATS 551 - HW4

Zhen Qin

1.

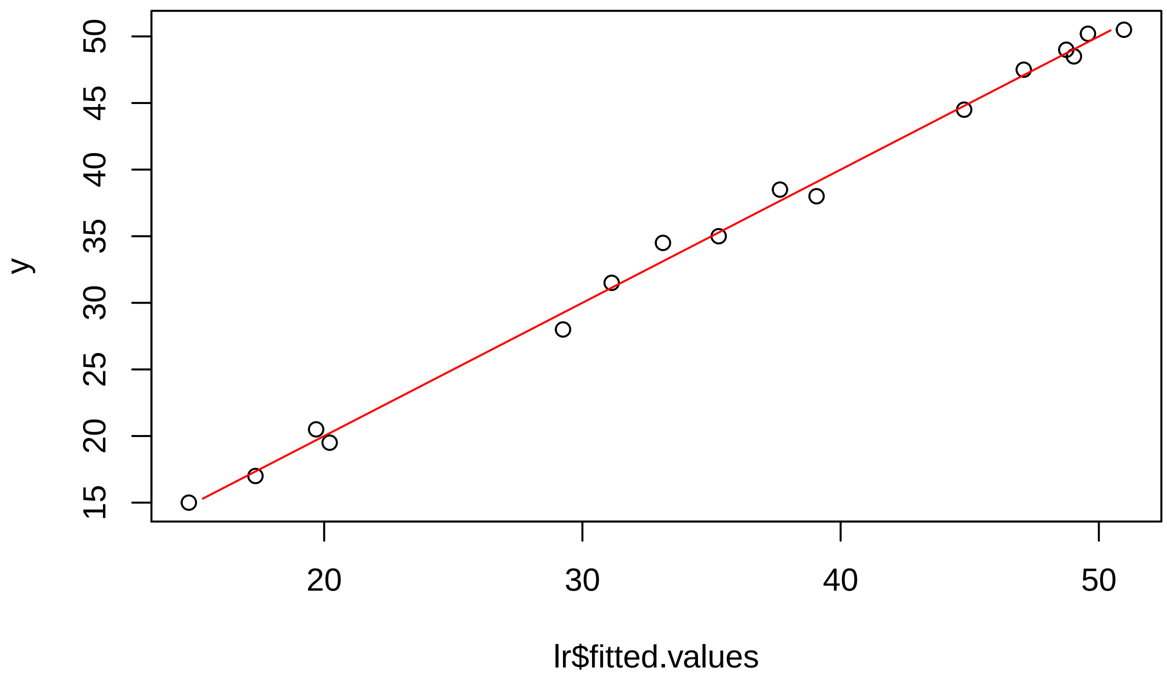
1.

For ordinary linear model, $(y_i|\beta_i, X) \sim N(\beta_1 x_{i1} + ... + \beta_k x_{ik}, \sigma^2)$, $p(\beta, \sigma^2|X) \propto \sigma^{-2}$. According to the theory, the rank of predictors is 9, $n = 16 > 9$. The posterior is $(\beta|\sigma, y) \sim N(\hat{\beta}, V_\beta \sigma^2)$, $(\sigma^2|y) \sim inv - \chi^2(5, s^2)$

```r
y=c(49,50.2,50.5,48.5,47.5,44.5,28,31.5,34.5,35,38,38.5,15,17,20.5,19.5)
x1=scale(c(rep(1300,6),rep(1200,6),rep(1100,4)))
x2=scale(c(7.5,9,11,13.5,17,23,5.3,7.5,11,13.5,17,23,5.3,7.5,11,17))
x3=scale(c(0.012,0.012,0.0115,0.013,0.0135,0.012,0.04,0.038,0.032,0.026,0.034,0.041,0.084,0.098,0.092,0
dat =data.frame(y=y,x1=x1,x2=x2,x3=x3)
dat$x12 = dat$x1*dat$x2
dat$x13 = dat$x1*dat$x3
dat$x23 = dat$x2*dat$x3
dat$x11 = dat$x1^2
dat$x22 = dat$x2^2
dat$x33 = dat$x3^3

lr = lm(y~.,dat)
summary(lr)
```

```
##
## Call:
## lm(formula = y ~ ., data = dat)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -1.24936 -0.48773 -0.01876  0.46139  1.38059
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)  39.4882     1.6895  23.372 4.02e-07 ***
## x1           10.3311     3.3294   3.103  0.02104 *
## x2            2.1713     0.4168   5.209  0.00200 **
## x3            4.6043     4.1918   1.098  0.31415
## x12         -10.6027     1.8911  -5.607  0.00137 **
## x13         -30.3799    15.7205  -1.933  0.10149
## x23         -10.8840     2.1820  -4.988  0.00248 **
## x11         -26.0965    13.2783  -1.965  0.09697 .
## x22          -1.8648     0.5760  -3.238  0.01774 *
## x33          -7.2094     3.0578  -2.358  0.05646 .
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 1.161 on 6 degrees of freedom
## Multiple R-squared:  0.9966, Adjusted R-squared:  0.9914
## F-statistic:   193 on 9 and 6 DF,  p-value: 1.085e-06
```

```r
plot(lr$fitted.values,y)
lines(quantile(y, c(0.01, 0.99)), quantile(y, c(0.01, 0.99)), col = "red")
```



```r
X=as.matrix(dat[,-1])
solve(t(X)%*%X)
```

```
##              x1          x2         x3          x12         x13         x23
## x1   6.5329642 -0.18961853  4.2824380 -0.245212388  28.9550888  0.23086909
## x2  -0.1896185  0.09484032 -0.1274592  0.035267247  -0.8010558  0.01641373
## x3   4.2824380 -0.12745917  3.7771549 -0.729532569  16.8612531 -0.53418744
## x12 -0.2452124  0.03526725 -0.7295326  2.585612061  -1.3704955  2.78430401
## x13 28.9550888 -0.80105581 16.8612531 -1.370495470 157.3171829  0.79372682
## x23  0.2308691  0.01641373 -0.5341874  2.784304005   0.7937268  3.17229372
## x11 23.6132555 -0.63838485 13.9527783 -1.383593552 129.3943558  0.33562788
## x22  0.3131513 -0.04076894  0.1991385  0.001249994   1.1947524  0.06052415
## x33  4.9227365 -0.12826365  2.4173648  0.218356300  26.5696131  0.67593378
##              x11          x22        x33
## x1   23.6132555  0.313151259  4.9227365
## x2   -0.6383848 -0.040768936 -0.1282636
## x3   13.9527783  0.199138482  2.4173648
## x12  -1.3835936  0.001249994  0.2183563
## x13 129.3943558  1.194752374 26.5696131
## x23   0.3356279  0.060524147  0.6759338
## x11 106.6534698  0.925913185 21.6564838
## x22   0.9259132  0.075980889  0.2240188
## x33  21.6564838  0.224018804  4.7829546
```

2.

For a mixed-effects model, $p(\beta_0) \propto 1$, $\beta_i \sim N(\mu, \sigma^2), i = 1, ..., 9$.

```
library(rstan)
```

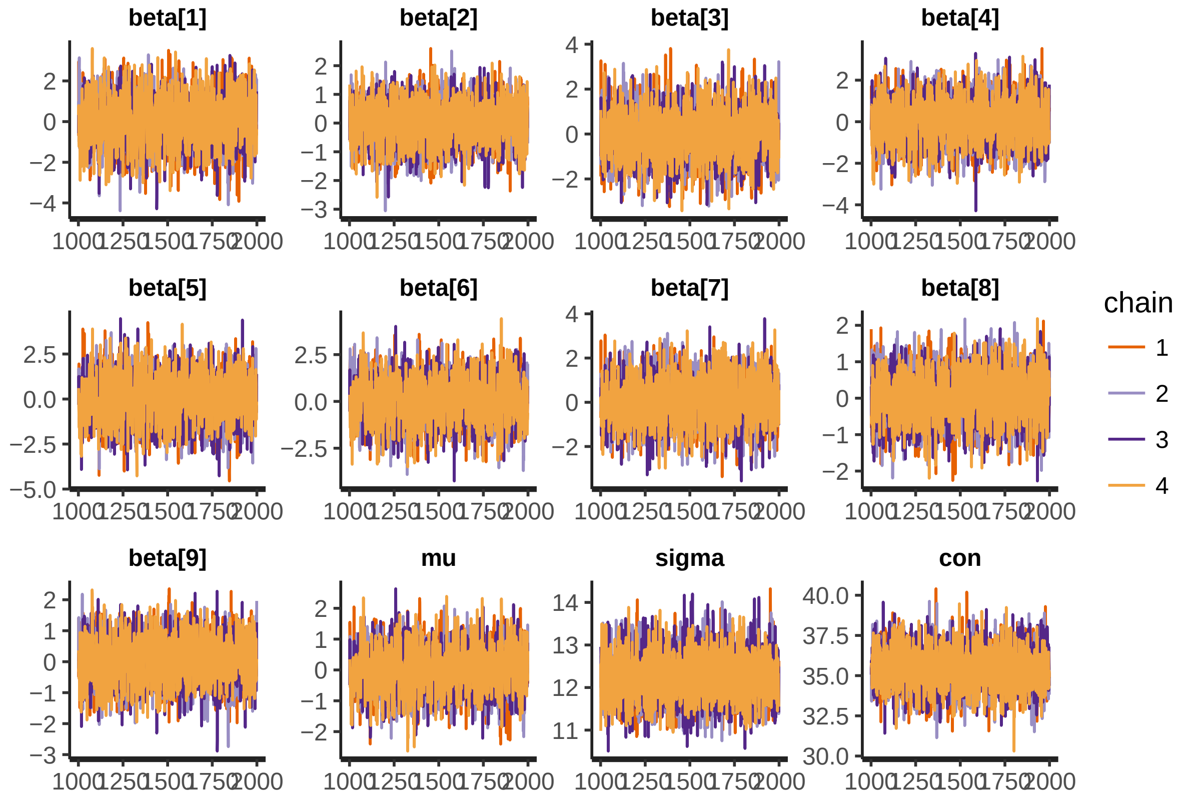
```
## Loading required package: ggplot2

## Loading required package: StanHeaders

## rstan (Version 2.17.3, GitRev: 2e1f913d3ca3)

## For execution on a local, multicore CPU with excess RAM we recommend calling
## options(mc.cores = parallel::detectCores()).
## To avoid recompilation of unchanged Stan programs, we recommend calling
## rstan_options(auto_write = TRUE)
```

```
library(ggplot2)
library(StanHeaders)
hlr =stan(file = "hlr.stan", data =list(N =length(y), y = dat$y, x = dat[,-1],
                                        tau =1,iter = 1000, chains = 4, refresh = 0))
```

```
## In file included from D:/R/R-3.4.4/library/BH/include/boost/config.hpp:39:0,
##                  from D:/R/R-3.4.4/library/BH/include/boost/math/tools/config.hpp:13,
##                  from D:/R/R-3.4.4/library/StanHeaders/include/stan/math/rev/core/var.hpp:7,
##                  from D:/R/R-3.4.4/library/StanHeaders/include/stan/math/rev/core/gevv_vvv_vari.hpp:
##                  from D:/R/R-3.4.4/library/StanHeaders/include/stan/math/rev/core.hpp:12,
##                  from D:/R/R-3.4.4/library/StanHeaders/include/stan/math/rev/mat.hpp:4,
##                  from D:/R/R-3.4.4/library/StanHeaders/include/stan/math.hpp:4,
##                  from D:/R/R-3.4.4/library/StanHeaders/include/src/stan/model/model_header.hpp:4,
##                  from file23507be3410b.cpp:8:
## D:/R/R-3.4.4/library/BH/include/boost/config/compiler/gcc.hpp:186:0: warning: "BOOST_NO_CXX11_RVALUE
##  #  define BOOST_NO_CXX11_RVALUE_REFERENCES
##  ^
## <command-line>:0:0: note: this is the location of the previous definition
##
## SAMPLING FOR MODEL 'hlr' NOW (CHAIN 1).
##
## Gradient evaluation took 0 seconds
## 1000 transitions using 10 leapfrog steps per transition would take 0 seconds.
## Adjust your expectations accordingly!
##
##
## Iteration:    1 / 2000 [  0%]  (Warmup)
## Iteration:  200 / 2000 [ 10%]  (Warmup)
## Iteration:  400 / 2000 [ 20%]  (Warmup)
## Iteration:  600 / 2000 [ 30%]  (Warmup)
## Iteration:  800 / 2000 [ 40%]  (Warmup)
## Iteration: 1000 / 2000 [ 50%]  (Warmup)
## Iteration: 1001 / 2000 [ 50%]  (Sampling)
## Iteration: 1200 / 2000 [ 60%]  (Sampling)
## Iteration: 1400 / 2000 [ 70%]  (Sampling)
## Iteration: 1600 / 2000 [ 80%]  (Sampling)
## Iteration: 1800 / 2000 [ 90%]  (Sampling)
## Iteration: 2000 / 2000 [100%]  (Sampling)
##
##  Elapsed Time: 0.148 seconds (Warm-up)
##                 0.09 seconds (Sampling)
##                 0.238 seconds (Total)
##
##
## SAMPLING FOR MODEL 'hlr' NOW (CHAIN 2).
##
## Gradient evaluation took 0 seconds
## 1000 transitions using 10 leapfrog steps per transition would take 0 seconds.
## Adjust your expectations accordingly!
##
##
## Iteration:    1 / 2000 [  0%]  (Warmup)
## Iteration:  200 / 2000 [ 10%]  (Warmup)
## Iteration:  400 / 2000 [ 20%]  (Warmup)
## Iteration:  600 / 2000 [ 30%]  (Warmup)
## Iteration:  800 / 2000 [ 40%]  (Warmup)
## Iteration: 1000 / 2000 [ 50%]  (Warmup)
## Iteration: 1001 / 2000 [ 50%]  (Sampling)
## Iteration: 1200 / 2000 [ 60%]  (Sampling)
## Iteration: 1400 / 2000 [ 70%]  (Sampling)
## Iteration: 1600 / 2000 [ 80%]  (Sampling)
## Iteration: 1800 / 2000 [ 90%]  (Sampling)
## Iteration: 2000 / 2000 [100%]  (Sampling)
##
##  Elapsed Time: 0.168 seconds (Warm-up)
##                 0.104 seconds (Sampling)
##                 0.272 seconds (Total)
##
##
## SAMPLING FOR MODEL 'hlr' NOW (CHAIN 3).
##
## Gradient evaluation took 0 seconds
## 1000 transitions using 10 leapfrog steps per transition would take 0 seconds.
## Adjust your expectations accordingly!
##
##
## Iteration:    1 / 2000 [  0%]  (Warmup)
## Iteration:  200 / 2000 [ 10%]  (Warmup)
## Iteration:  400 / 2000 [ 20%]  (Warmup)
## Iteration:  600 / 2000 [ 30%]  (Warmup)
## Iteration:  800 / 2000 [ 40%]  (Warmup)
## Iteration: 1000 / 2000 [ 50%]  (Warmup)
## Iteration: 1001 / 2000 [ 50%]  (Sampling)
## Iteration: 1200 / 2000 [ 60%]  (Sampling)
## Iteration: 1400 / 2000 [ 70%]  (Sampling)
## Iteration: 1600 / 2000 [ 80%]  (Sampling)
## Iteration: 1800 / 2000 [ 90%]  (Sampling)
## Iteration: 2000 / 2000 [100%]  (Sampling)
##
##  Elapsed Time: 0.135 seconds (Warm-up)
##                 0.125 seconds (Sampling)
##                 0.26 seconds (Total)
##
##
```

```
## SAMPLING FOR MODEL 'hlr' NOW (CHAIN 4).
##
## Gradient evaluation took 0 seconds
## 1000 transitions using 10 leapfrog steps per transition would take 0 seconds.
## Adjust your expectations accordingly!
##
##
## Iteration:    1 / 2000 [  0%]  (Warmup)
## Iteration:  200 / 2000 [ 10%]  (Warmup)
## Iteration:  400 / 2000 [ 20%]  (Warmup)
## Iteration:  600 / 2000 [ 30%]  (Warmup)
## Iteration:  800 / 2000 [ 40%]  (Warmup)
## Iteration: 1000 / 2000 [ 50%]  (Warmup)
## Iteration: 1001 / 2000 [ 50%]  (Sampling)
## Iteration: 1200 / 2000 [ 60%]  (Sampling)
## Iteration: 1400 / 2000 [ 70%]  (Sampling)
## Iteration: 1600 / 2000 [ 80%]  (Sampling)
## Iteration: 1800 / 2000 [ 90%]  (Sampling)
## Iteration: 2000 / 2000 [100%]  (Sampling)
##
##  Elapsed Time: 0.159 seconds (Warm-up)
##                0.103 seconds (Sampling)
##                0.262 seconds (Total)
```

```
traceplot(hlr, pars =names(hlr)[1:12])
```



```
hlrfit = extract(hlr, permuted = TRUE)
```

3.

In question 2, all coefficients are around 0. In question 1, the variance matrix of beta is big, so the results will be less stable. Their conclusion is not acceptable.

4.

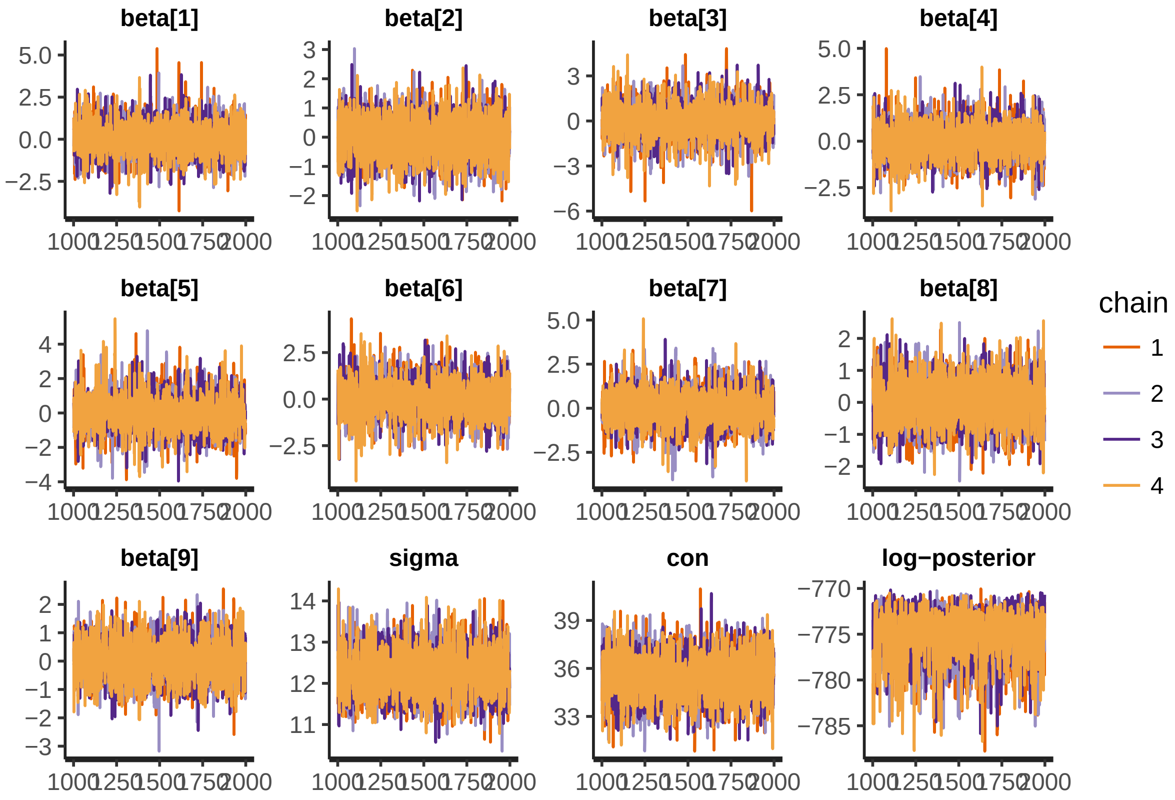
```
t4lr =stan(file = "t4lr.stan", data =list(N =length(y), y = dat$y, x = dat[,-1],
                                       iter = 1000, chains = 4, refresh = 0))
```

```
## In file included from D:/R/R-3.4.4/library/BH/include/boost/config.hpp:39:0,
##                  from D:/R/R-3.4.4/library/BH/include/boost/math/tools/config.hpp:13,
##                  from D:/R/R-3.4.4/library/StanHeaders/include/stan/math/rev/core/var.hpp:7,
##                  from D:/R/R-3.4.4/library/StanHeaders/include/stan/math/rev/core/gevv_vvv_vari.hpp:
##                  from D:/R/R-3.4.4/library/StanHeaders/include/stan/math/rev/core.hpp:12,
##                  from D:/R/R-3.4.4/library/StanHeaders/include/stan/math/rev/mat.hpp:4,
##                  from D:/R/R-3.4.4/library/StanHeaders/include/stan/math.hpp:4,
##                  from D:/R/R-3.4.4/library/StanHeaders/include/src/stan/model/model_header.hpp:4,
##                  from file23506f2eea5.cpp:8:
## D:/R/R-3.4.4/library/BH/include/boost/config/compiler/gcc.hpp:186:0: warning: "BOOST_NO_CXX11_RVALUE_
##  #  define BOOST_NO_CXX11_RVALUE_REFERENCES
##  ^
## <command-line>:0:0: note: this is the location of the previous definition
##
## SAMPLING FOR MODEL 't4lr' NOW (CHAIN 1).
##
## Gradient evaluation took 0 seconds
## 1000 transitions using 10 leapfrog steps per transition would take 0 seconds.
## Adjust your expectations accordingly!
##
##
## Iteration:    1 / 2000 [  0%]  (Warmup)
## Iteration:  200 / 2000 [ 10%]  (Warmup)
## Iteration:  400 / 2000 [ 20%]  (Warmup)
## Iteration:  600 / 2000 [ 30%]  (Warmup)
## Iteration:  800 / 2000 [ 40%]  (Warmup)
## Iteration: 1000 / 2000 [ 50%]  (Warmup)
## Iteration: 1001 / 2000 [ 50%]  (Sampling)
## Iteration: 1200 / 2000 [ 60%]  (Sampling)
## Iteration: 1400 / 2000 [ 70%]  (Sampling)
## Iteration: 1600 / 2000 [ 80%]  (Sampling)
## Iteration: 1800 / 2000 [ 90%]  (Sampling)
## Iteration: 2000 / 2000 [100%]  (Sampling)
##
##  Elapsed Time: 0.182 seconds (Warm-up)
##                0.134 seconds (Sampling)
##                0.316 seconds (Total)
##
##
## SAMPLING FOR MODEL 't4lr' NOW (CHAIN 2).
```

```
##
## Gradient evaluation took 0 seconds
## 1000 transitions using 10 leapfrog steps per transition would take 0 seconds.
## Adjust your expectations accordingly!
##
##
## Iteration:    1 / 2000 [  0%]  (Warmup)
## Iteration:  200 / 2000 [ 10%]  (Warmup)
## Iteration:  400 / 2000 [ 20%]  (Warmup)
## Iteration:  600 / 2000 [ 30%]  (Warmup)
## Iteration:  800 / 2000 [ 40%]  (Warmup)
## Iteration: 1000 / 2000 [ 50%]  (Warmup)
## Iteration: 1001 / 2000 [ 50%]  (Sampling)
## Iteration: 1200 / 2000 [ 60%]  (Sampling)
## Iteration: 1400 / 2000 [ 70%]  (Sampling)
## Iteration: 1600 / 2000 [ 80%]  (Sampling)
## Iteration: 1800 / 2000 [ 90%]  (Sampling)
## Iteration: 2000 / 2000 [100%]  (Sampling)
##
##  Elapsed Time: 0.174 seconds (Warm-up)
##                0.131 seconds (Sampling)
##                0.305 seconds (Total)
##
##
## SAMPLING FOR MODEL 't4lr' NOW (CHAIN 3).
##
## Gradient evaluation took 0 seconds
## 1000 transitions using 10 leapfrog steps per transition would take 0 seconds.
## Adjust your expectations accordingly!
##
##
## Iteration:    1 / 2000 [  0%]  (Warmup)
## Iteration:  200 / 2000 [ 10%]  (Warmup)
## Iteration:  400 / 2000 [ 20%]  (Warmup)
## Iteration:  600 / 2000 [ 30%]  (Warmup)
## Iteration:  800 / 2000 [ 40%]  (Warmup)
## Iteration: 1000 / 2000 [ 50%]  (Warmup)
## Iteration: 1001 / 2000 [ 50%]  (Sampling)
## Iteration: 1200 / 2000 [ 60%]  (Sampling)
## Iteration: 1400 / 2000 [ 70%]  (Sampling)
## Iteration: 1600 / 2000 [ 80%]  (Sampling)
## Iteration: 1800 / 2000 [ 90%]  (Sampling)
## Iteration: 2000 / 2000 [100%]  (Sampling)
##
##  Elapsed Time: 0.172 seconds (Warm-up)
##                0.126 seconds (Sampling)
##                0.298 seconds (Total)
##
##
## SAMPLING FOR MODEL 't4lr' NOW (CHAIN 4).
##
## Gradient evaluation took 0 seconds
## 1000 transitions using 10 leapfrog steps per transition would take 0 seconds.
## Adjust your expectations accordingly!
```

```
##
##
## Iteration:    1 / 2000 [  0%]  (Warmup)
## Iteration:  200 / 2000 [ 10%]  (Warmup)
## Iteration:  400 / 2000 [ 20%]  (Warmup)
## Iteration:  600 / 2000 [ 30%]  (Warmup)
## Iteration:  800 / 2000 [ 40%]  (Warmup)
## Iteration: 1000 / 2000 [ 50%]  (Warmup)
## Iteration: 1001 / 2000 [ 50%]  (Sampling)
## Iteration: 1200 / 2000 [ 60%]  (Sampling)
## Iteration: 1400 / 2000 [ 70%]  (Sampling)
## Iteration: 1600 / 2000 [ 80%]  (Sampling)
## Iteration: 1800 / 2000 [ 90%]  (Sampling)
## Iteration: 2000 / 2000 [100%]  (Sampling)
##
##  Elapsed Time: 0.189 seconds (Warm-up)
##                0.13 seconds (Sampling)
##                0.319 seconds (Total)
```

```
traceplot(t4lr, pars =names(t4lr)[1:12])
```



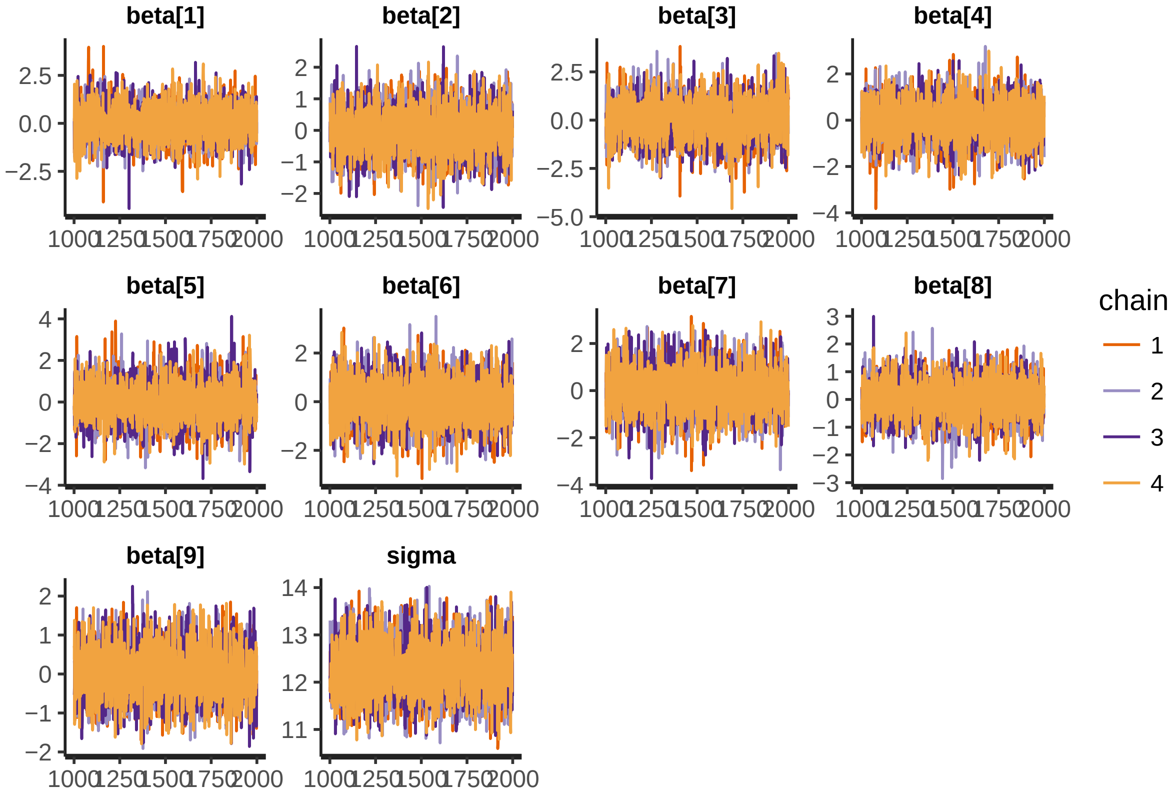
5.

Here I use another heavy tailed distribution, student_t(10). This result is similar with question 4.

```
tlr =stan(file = "tlr.stan", data =list(N =length(y), y = dat$y, x = dat[,-1],
                                        iter = 1000, chains = 4, refresh = 0))
```

```
## In file included from D:/R/R-3.4.4/library/BH/include/boost/config.hpp:39:0,
##                  from D:/R/R-3.4.4/library/BH/include/boost/math/tools/config.hpp:13,
##                  from D:/R/R-3.4.4/library/StanHeaders/include/stan/math/rev/core/var.hpp:7,
##                  from D:/R/R-3.4.4/library/StanHeaders/include/stan/math/rev/core/gevv_vvv_vari.hpp:
##                  from D:/R/R-3.4.4/library/StanHeaders/include/stan/math/rev/core.hpp:12,
##                  from D:/R/R-3.4.4/library/StanHeaders/include/stan/math/rev/mat.hpp:4,
##                  from D:/R/R-3.4.4/library/StanHeaders/include/stan/math.hpp:4,
##                  from D:/R/R-3.4.4/library/StanHeaders/include/src/stan/model/model_header.hpp:4,
##                  from file23503053571.cpp:8:
## D:/R/R-3.4.4/library/BH/include/boost/config/compiler/gcc.hpp:186:0: warning: "BOOST_NO_CXX11_RVALUE_
##  #  define BOOST_NO_CXX11_RVALUE_REFERENCES
##  ^
## <command-line>:0:0: note: this is the location of the previous definition
##
## SAMPLING FOR MODEL 'tlr' NOW (CHAIN 1).
##
## Gradient evaluation took 0 seconds
## 1000 transitions using 10 leapfrog steps per transition would take 0 seconds.
## Adjust your expectations accordingly!
##
##
## Iteration:    1 / 2000 [  0%]  (Warmup)
## Iteration:  200 / 2000 [ 10%]  (Warmup)
## Iteration:  400 / 2000 [ 20%]  (Warmup)
## Iteration:  600 / 2000 [ 30%]  (Warmup)
## Iteration:  800 / 2000 [ 40%]  (Warmup)
## Iteration: 1000 / 2000 [ 50%]  (Warmup)
## Iteration: 1001 / 2000 [ 50%]  (Sampling)
## Iteration: 1200 / 2000 [ 60%]  (Sampling)
## Iteration: 1400 / 2000 [ 70%]  (Sampling)
## Iteration: 1600 / 2000 [ 80%]  (Sampling)
## Iteration: 1800 / 2000 [ 90%]  (Sampling)
## Iteration: 2000 / 2000 [100%]  (Sampling)
##
##  Elapsed Time: 0.172 seconds (Warm-up)
##                0.122 seconds (Sampling)
##                0.294 seconds (Total)
##
##
## SAMPLING FOR MODEL 'tlr' NOW (CHAIN 2).
##
## Gradient evaluation took 0 seconds
## 1000 transitions using 10 leapfrog steps per transition would take 0 seconds.
## Adjust your expectations accordingly!
##
##
## Iteration:    1 / 2000 [  0%]  (Warmup)
## Iteration:  200 / 2000 [ 10%]  (Warmup)
## Iteration:  400 / 2000 [ 20%]  (Warmup)
## Iteration:  600 / 2000 [ 30%]  (Warmup)
## Iteration:  800 / 2000 [ 40%]  (Warmup)
## Iteration: 1000 / 2000 [ 50%]  (Warmup)
## Iteration: 1001 / 2000 [ 50%]  (Sampling)
## Iteration: 1200 / 2000 [ 60%]  (Sampling)
## Iteration: 1400 / 2000 [ 70%]  (Sampling)
## Iteration: 1600 / 2000 [ 80%]  (Sampling)
## Iteration: 1800 / 2000 [ 90%]  (Sampling)
## Iteration: 2000 / 2000 [100%]  (Sampling)
##
##  Elapsed Time: 0.166 seconds (Warm-up)
##                0.117 seconds (Sampling)
##                0.283 seconds (Total)
##
##
## SAMPLING FOR MODEL 'tlr' NOW (CHAIN 3).
##
## Gradient evaluation took 0 seconds
## 1000 transitions using 10 leapfrog steps per transition would take 0 seconds.
## Adjust your expectations accordingly!
##
##
## Iteration:    1 / 2000 [  0%]  (Warmup)
## Iteration:  200 / 2000 [ 10%]  (Warmup)
## Iteration:  400 / 2000 [ 20%]  (Warmup)
## Iteration:  600 / 2000 [ 30%]  (Warmup)
## Iteration:  800 / 2000 [ 40%]  (Warmup)
## Iteration: 1000 / 2000 [ 50%]  (Warmup)
## Iteration: 1001 / 2000 [ 50%]  (Sampling)
## Iteration: 1200 / 2000 [ 60%]  (Sampling)
## Iteration: 1400 / 2000 [ 70%]  (Sampling)
## Iteration: 1600 / 2000 [ 80%]  (Sampling)
## Iteration: 1800 / 2000 [ 90%]  (Sampling)
## Iteration: 2000 / 2000 [100%]  (Sampling)
##
##  Elapsed Time: 0.165 seconds (Warm-up)
##                0.119 seconds (Sampling)
##                0.284 seconds (Total)
##
##
## SAMPLING FOR MODEL 'tlr' NOW (CHAIN 4).
##
## Gradient evaluation took 0 seconds
## 1000 transitions using 10 leapfrog steps per transition would take 0 seconds.
## Adjust your expectations accordingly!
##
##
## Iteration:    1 / 2000 [  0%]  (Warmup)
## Iteration:  200 / 2000 [ 10%]  (Warmup)
## Iteration:  400 / 2000 [ 20%]  (Warmup)
## Iteration:  600 / 2000 [ 30%]  (Warmup)
## Iteration:  800 / 2000 [ 40%]  (Warmup)
## Iteration: 1000 / 2000 [ 50%]  (Warmup)
## Iteration: 1001 / 2000 [ 50%]  (Sampling)
## Iteration: 1200 / 2000 [ 60%]  (Sampling)
## Iteration: 1400 / 2000 [ 70%]  (Sampling)
## Iteration: 1600 / 2000 [ 80%]  (Sampling)
## Iteration: 1800 / 2000 [ 90%]  (Sampling)
## Iteration: 2000 / 2000 [100%]  (Sampling)
##
##  Elapsed Time: 0.186 seconds (Warm-up)
##                0.147 seconds (Sampling)
##                0.333 seconds (Total)
```

```
traceplot(tlr, pars =names(tlr)[1:10])
```



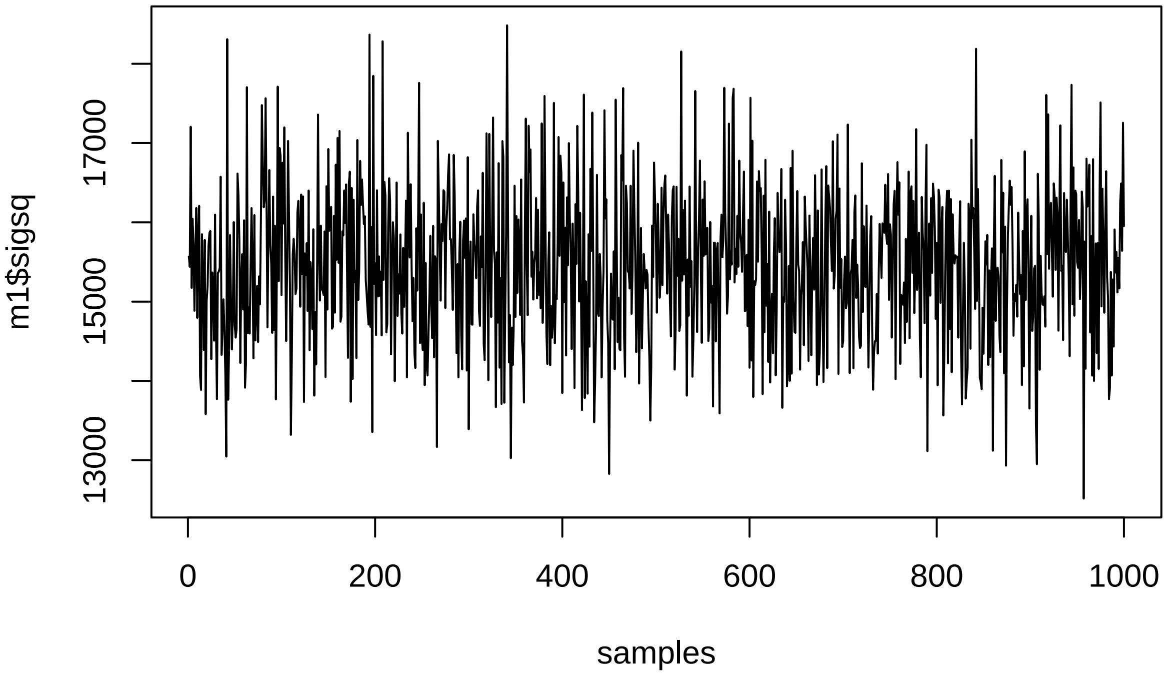
2.

1.

Fit a regression model using the g-prior. Posterior confidence intervals are shown below.

```
az = read.table('azdiabetes.dat',header = T)
az = az[,-8]
az[,-2] = scale(az[,-2])
X = as.matrix(az[,-2])
xtx = X%*%solve(t(X)%*%X)%*%t(X)
```

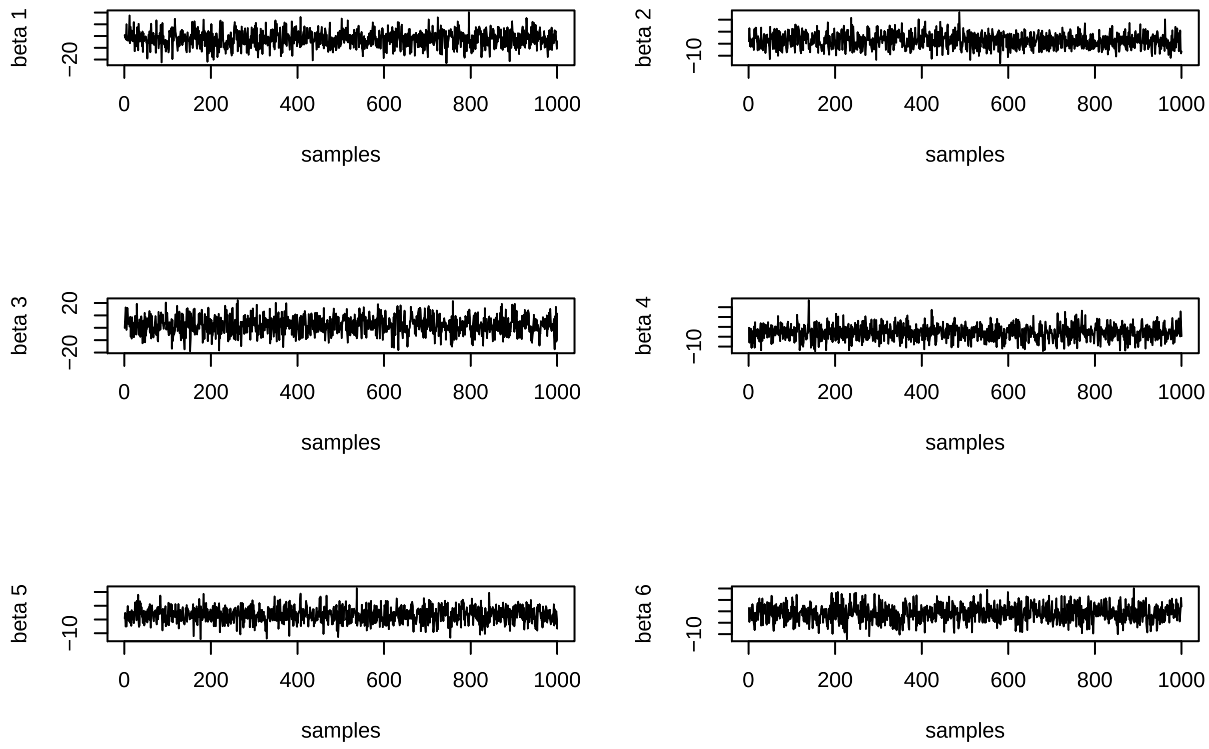
```
library(MASS)
pos = function(y,x,g,nu_0,sig_0,S)
{
  n = dim(x)[1]; p = dim(X)[2]
  Hg = (g/(g+1))*(x%*%solve(t(x)%*%x)%*%t(x))
  SSRg = t(y)%*%(diag(1,nrow = n) - Hg)%*%y
  sig = 1/rgamma(S,(nu_0+n)/2,(nu_0*sig_0 + SSRg)/2)
  Vb <-  g*solve(t(X)%*%X)/(g+1)
  Eb <- Vb%*%t(X)%*%y
  E <- matrix (rnorm(S*p , 0 , sqrt(sig)),S,p)
  beta <- t(t(E%*%chol(Vb))+c(Eb))
  return(list(beta = beta, sigsq = sig))

}
y = as.vector(az[,2]); X = as.matrix(az[,-2]);S = 1000
m1 = pos(y,X,532,2,1,S)
ts.plot(m1$sigsq,xlab = "samples")
```



```
mean(m1$sigsq)
```

```
## [1] 15451.31
```

```
par(mfrow = c(3,2))
for(i in 1:6)
{
  ts.plot(m1$beta[,i], xlab = "samples", ylab = paste("beta",i))
}
```



```
colMeans(m1$beta)
```

```
##     npreg        bp      skin       bmi       ped       age
## -2.290226  2.595054  2.170446  4.242215  3.424044  8.280674
```

```
std = apply(m1$beta,2,sd)
posin = apply(m1$beta,2,function(x)quantile(x,c(0.025,.975)))
posin
```

```
##           npreg        bp      skin       bmi       ped       age
## 2.5%  -15.87626 -8.255967 -11.94173 -10.77122 -7.847565 -6.399887
## 97.5%  11.50143 14.379379  16.11428  18.67705 13.494391 23.008683
```

2.

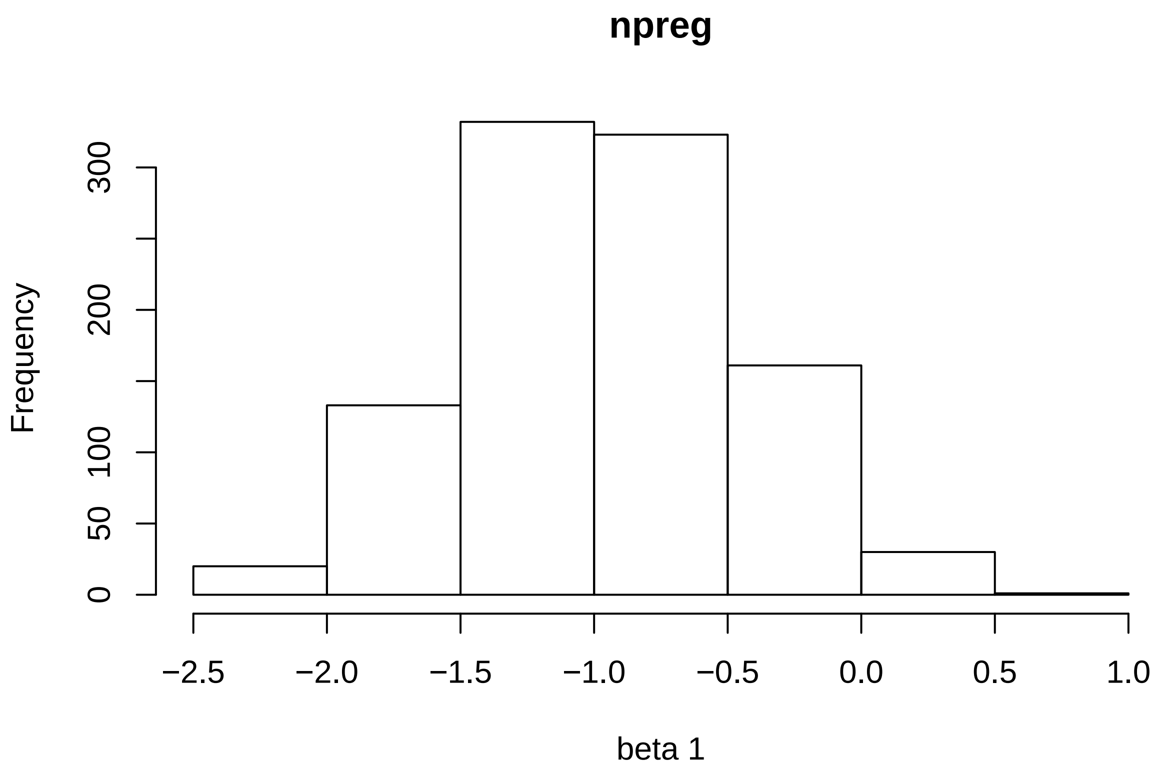
```
lpy.X <- function( y ,X, g=length( y ),
                          nu_0=2, sig_0=1)
{
  n <- dim(X)[1] ; p <- dim(X)[2]
  if ( p==0) { Hg <- 0 }
  if(p>0){
    Hg = (g /( g+1))*X%*%solve(t(X)%*%X)%*%t(X)
  }
  SSRg <-  t(y)%*%( diag(1,nrow=n )- Hg)%*%y
  ans = -.5*( n* log(pi)+p*log(1+g)+(nu_0+n)*log(nu_0* sig_0+SSRg)-
        nu_0*log(nu_0* sig_0)) +  lgamma (( nu_0+n ) / 2 ) - lgamma ( nu_0 / 2 )
  return(ans)
}
#####
##### starting values and MCMC setup
z <- rep ( 1 , dim(X)[2])
lpy.c <- lpy.X(y,X[ , z==1,drop=FALSE] )
S <- 1000
Z <- matrix (NA, S , dim(X)[2])
B = matrix (NA, S , dim(X)[2])
sigsq = rep(NA,S)
#####
##### Gibbs s ample r
for ( s in 1:S )
{
  for ( j in sample ( 1:dim(X)[2]))
  {
    zp <- z ; zp [j] <- 1-zp[j]
    lpy.p <- lpy.X( y ,X[, zp==1,drop=FALSE] )
    r <-  ( lpy.p - lpy.c )*( -1)^( zp[j]==0)
    z [ j ] <- rbinom ( 1 , 1 , 1/( 1+ exp(-r ) ) )
    if ( z [j]==zp [ j ] ) { lpy.c <- lpy.p}
  }
  Z [ s ,] <- z
  m = pos(y,X[,z==1],length(y),nu_0 = 2,sig_0 = 1,S = 1)
  B[s,] = m$beta; sigsq[s] = m$sigsq

}
```

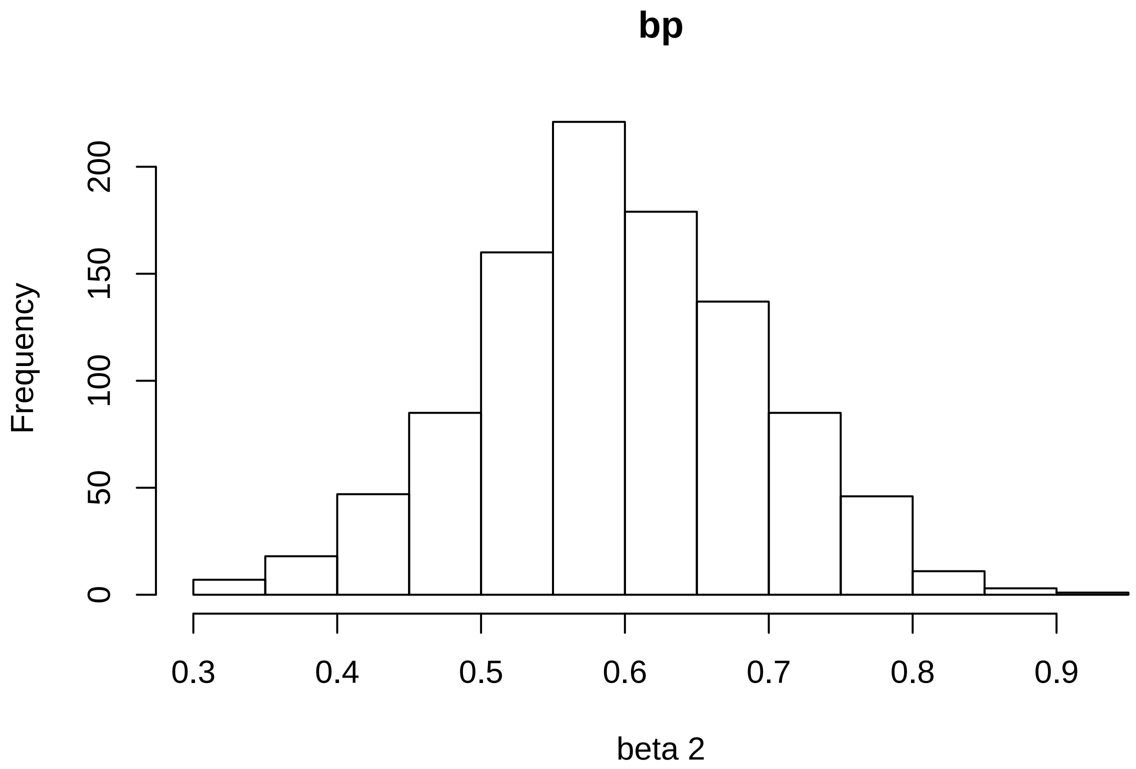
I compare the results with averaging procedures. It is clear that by P. Hoff's method the variance of npreg increases and others are very similar.

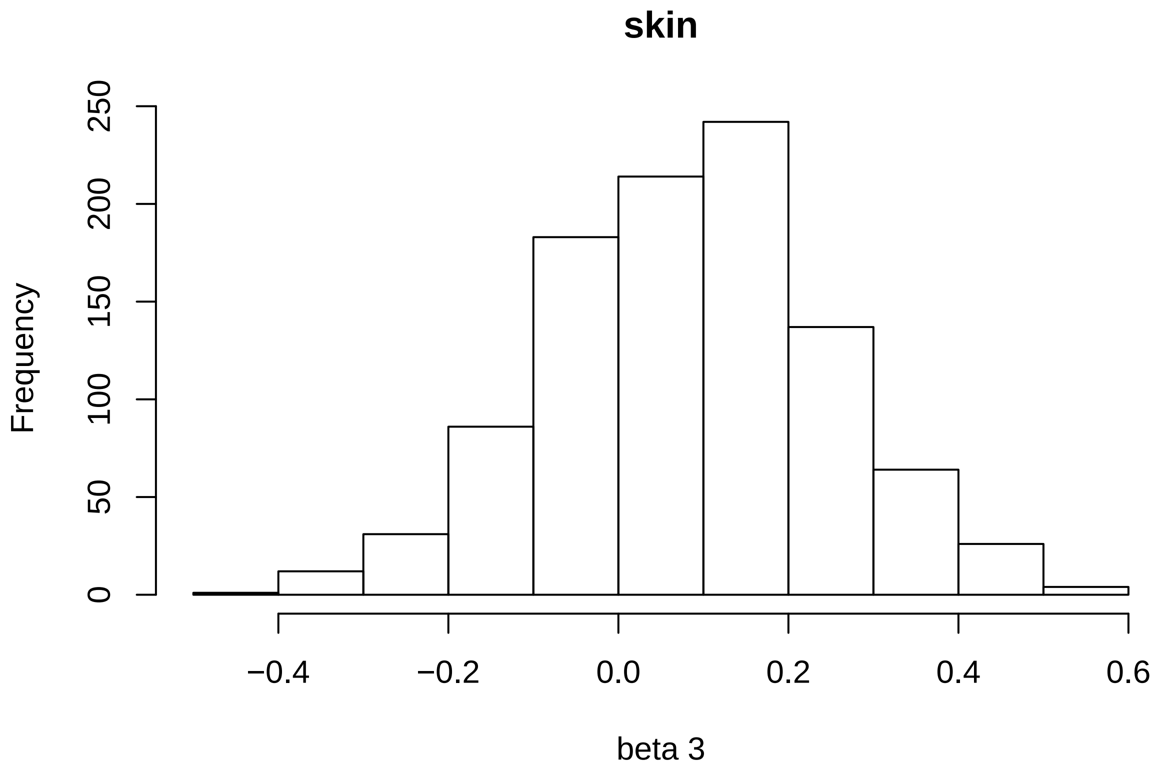
```
load('.RData')
prob = colMeans(Z); nm = colnames(az); nm = nm[-2]; cbind(nm,prob)
```

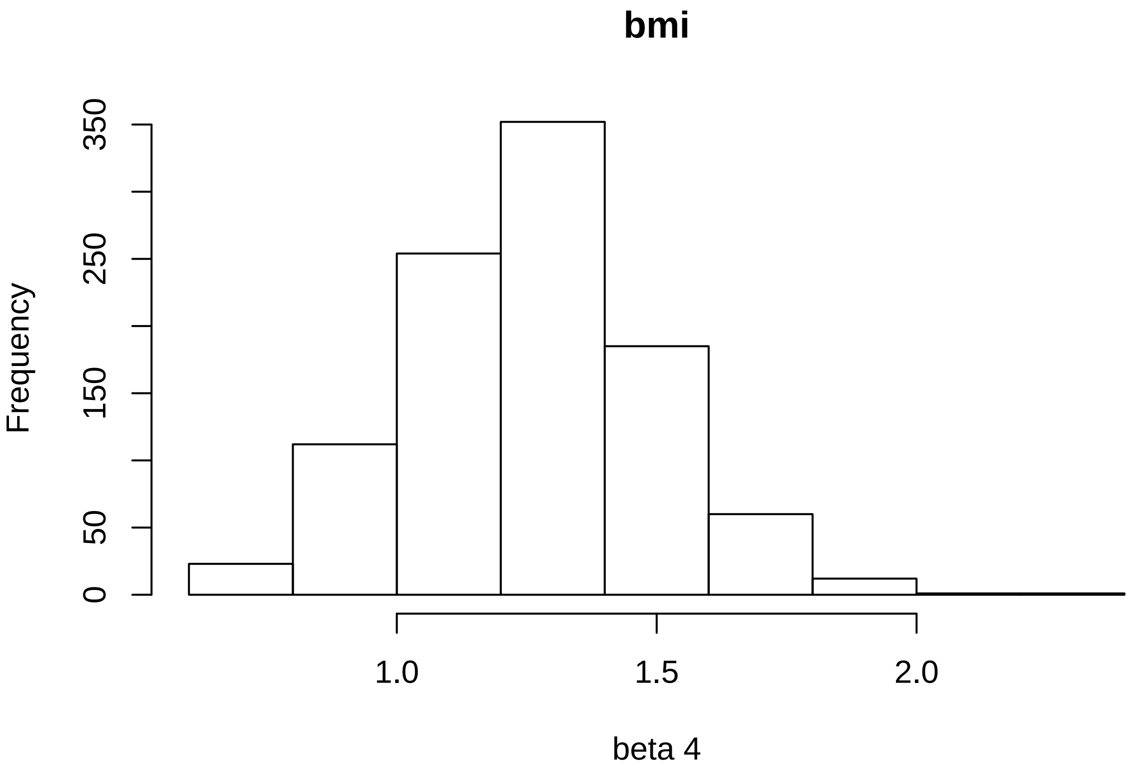
```
##      nm      prob   
## [1,] "npreg" "0.202"
## [2,] "bp"    "1"    
## [3,] "skin"  "0.036"
## [4,] "bmi"   "1"    
## [5,] "ped"   "0.958"
## [6,] "age"   "1"    
```

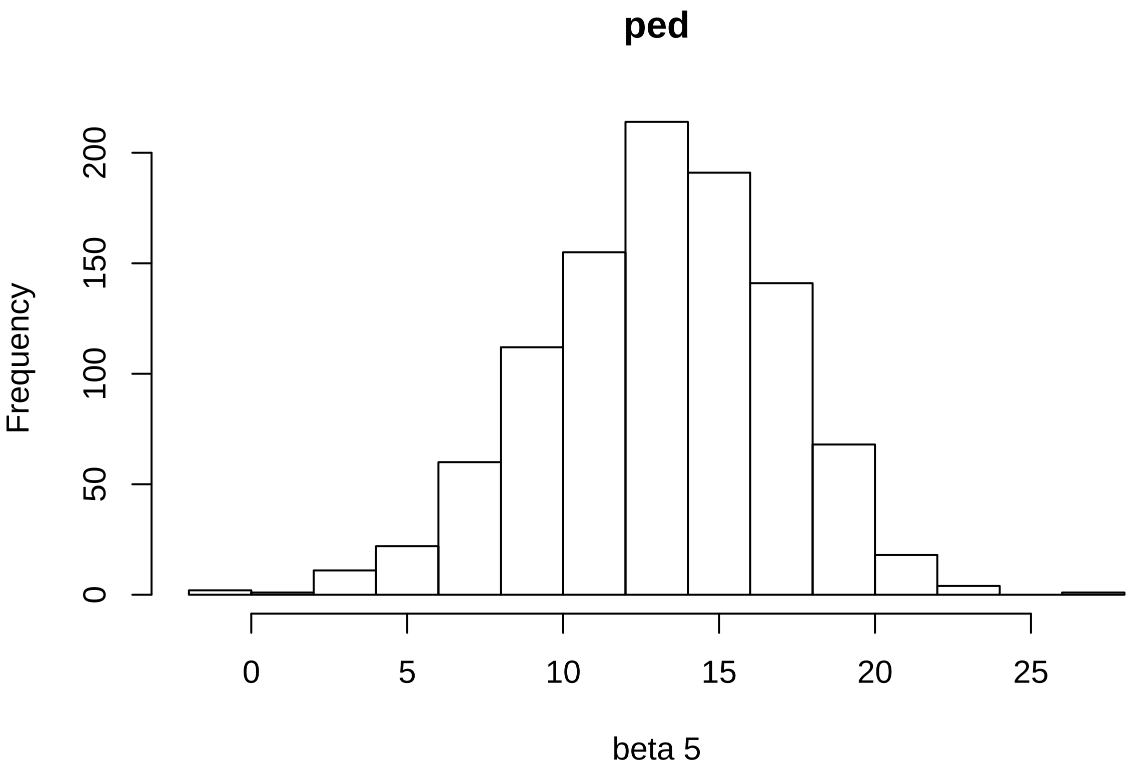
```
for(i in 1:6)
{
  hist(B[,i], xlab = paste("beta",i),main = nm[i])
}
```

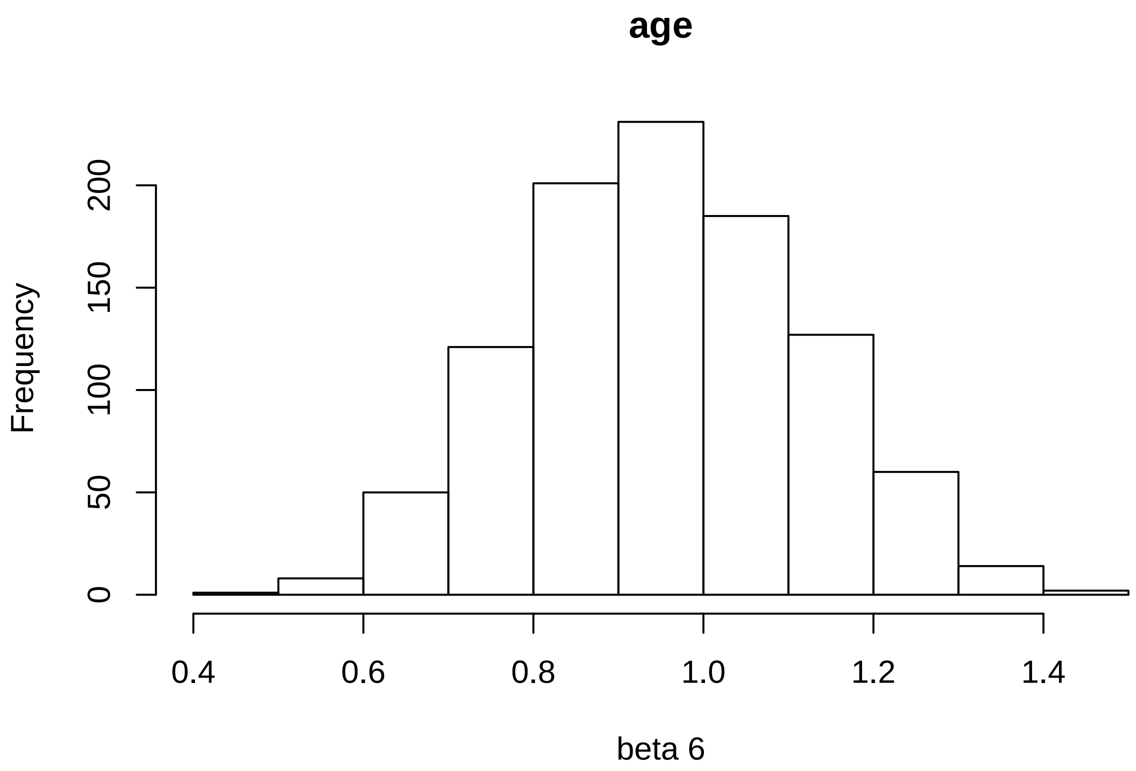

npreg
Frequency
0
50
100
200
300
−2.5
−2.0
−1.5
−1.0
−0.5
0.0
0.5
1.0
beta 1


bp
Frequency
0
50
100
150
200
0.3
0.4
0.5
0.6
0.7
0.8
0.9
beta 2


skin
Frequency
0
50
100
150
200
250
−0.4
−0.2
0.0
0.2
0.4
0.6
beta 3


bmi
Frequency
0
50
150
250
350
1.0
1.5
2.0
beta 4


ped
Frequency
0
50
100
150
200
0
5
10
15
20
25
beta 5



```
posin2 = apply(B,2,function(x)quantile(x,c(0.025,.975)))
colnames(posin2) = nm
colnames(posin) = nm
posin2
```

```
##              npreg        bp       skin       bmi      ped       age
## 2.5%  -1.95418228 0.4027476 -0.2303870 0.8023952  5.19179 0.6328707
## 97.5%  0.03012726 0.7818540  0.4108407 1.7228908 19.78169 1.2789291
```