RStan: linear model example

Heejung Shim

14/10/2019

Install rstan package

```
setwd("/Users/hjshim/Dropbox/MAS2017/lectures/Bayes/Scripts/RStan/")
#install.packages("rstan", repos = "https://cloud.r-project.org/", dependencies=TRUE)
library(rstan) # load the library
```

```
## Loading required package: StanHeaders
```

```
## Loading required package: ggplot2
```

```
## rstan (Version 2.19.2, 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)
```

```
#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)
rstan_options(auto_write = TRUE)
options(mc.cores = parallel::detectCores())
```

Prepare data for rstan

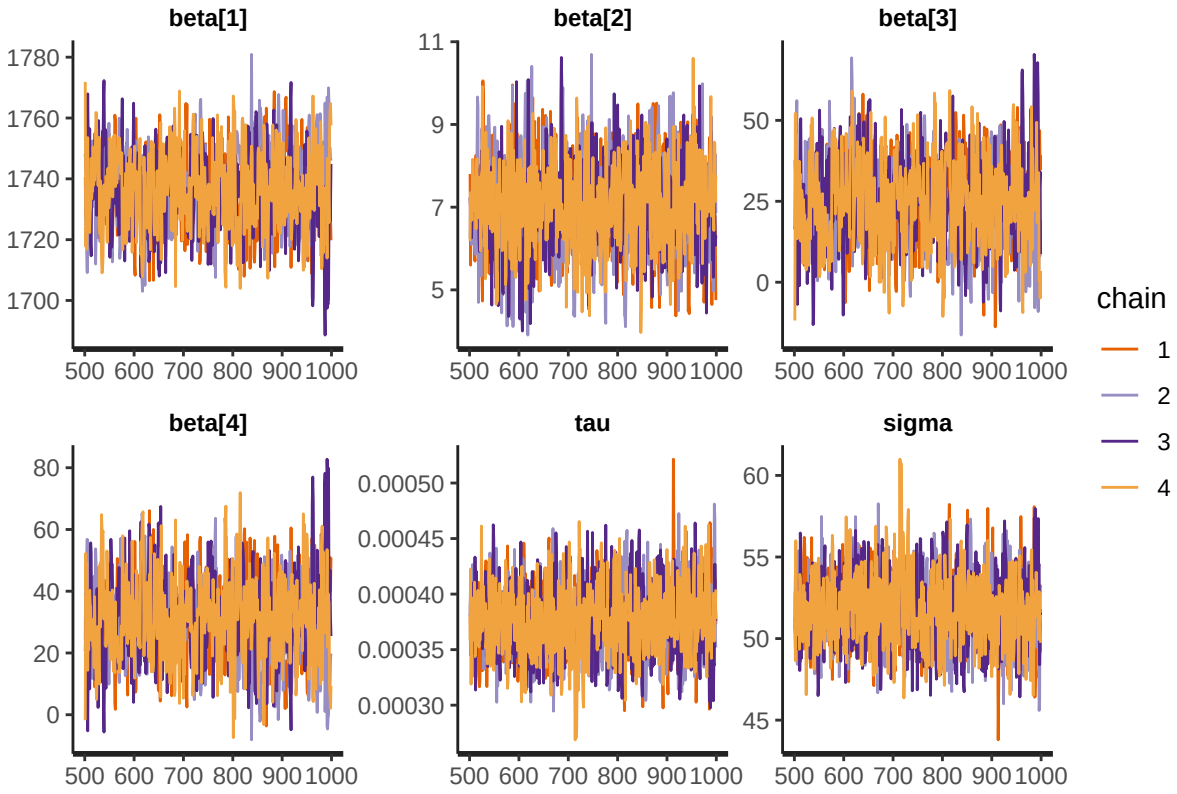
```
# Load the data
data = read.table(file="income-lm-data.txt", header = TRUE)
str(data)
```

```
## 'data.frame':    284 obs. of  3 variables:
##  $ INCOME: num  1673 1743 1747 1735 1749 ...
##  $ AVAGE : num  37.4 39.4 40.6 40.7 41.4 ...
##  $ RURAL : int  2 2 2 2 2 2 2 2 2 2 ...
```

```
# Center the AVAGE varaible
data$centered_AVAGE = data$AVAGE - mean(data$AVAGE)

# Convert the data to a design matrix in R before we pass it to stan
X <- model.matrix(~ data$centered_AVAGE + factor(data$RURAL))

# Create input data to stan as a list
income_data <- list(N = length(data$INCOME), K = ncol(X), X = X, y = data$INCOME)
```

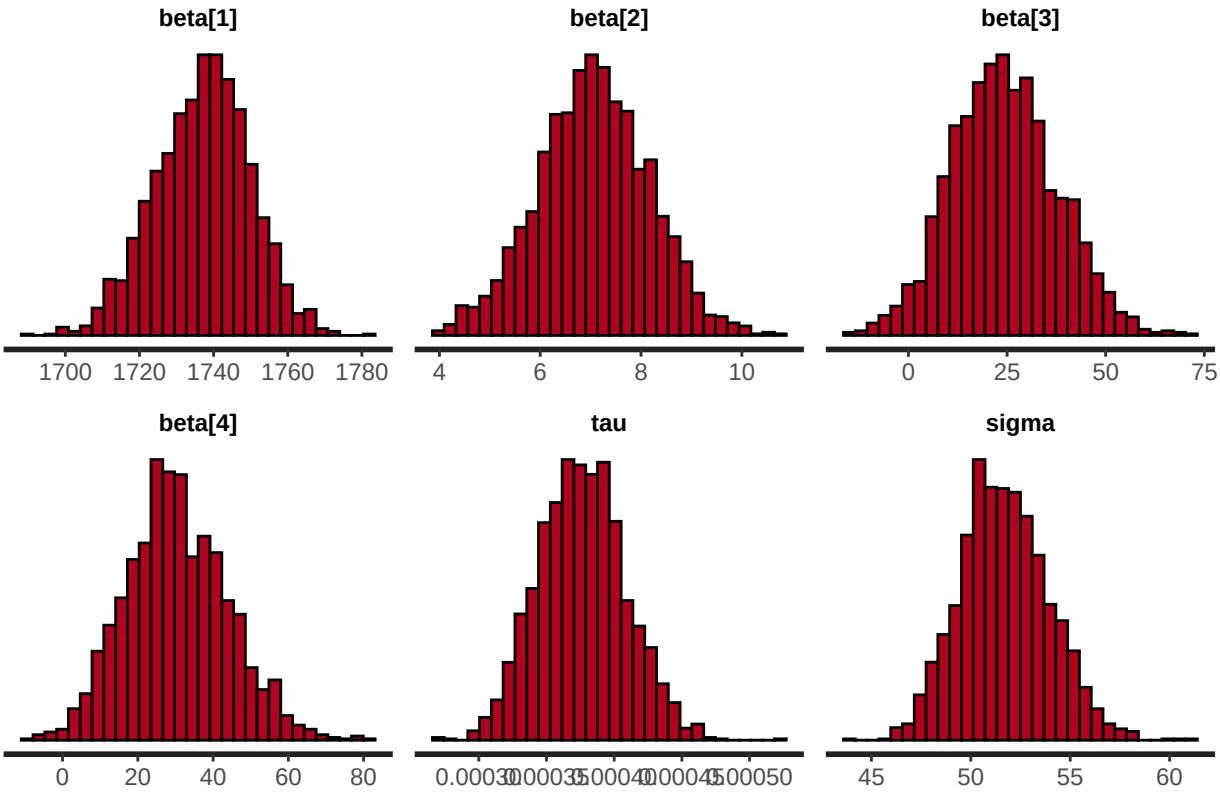
Run rstan and visualise results

```
# run stan
fit <- stan(file = "income-lm.stan", data = income_data, iter = 1000, chains = 4)

# trace plot
plot(fit, plotfun = "trace")
```

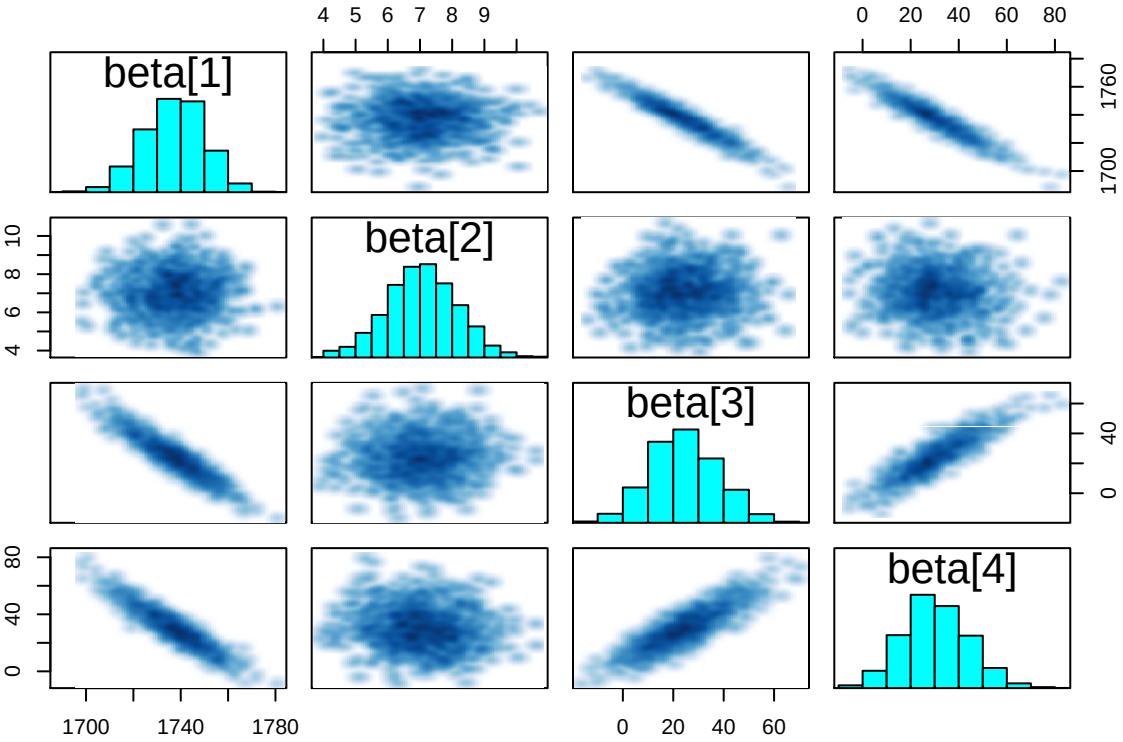



```
# empirical posterior dist
plot(fit, plotfun = "hist")
```

```
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```

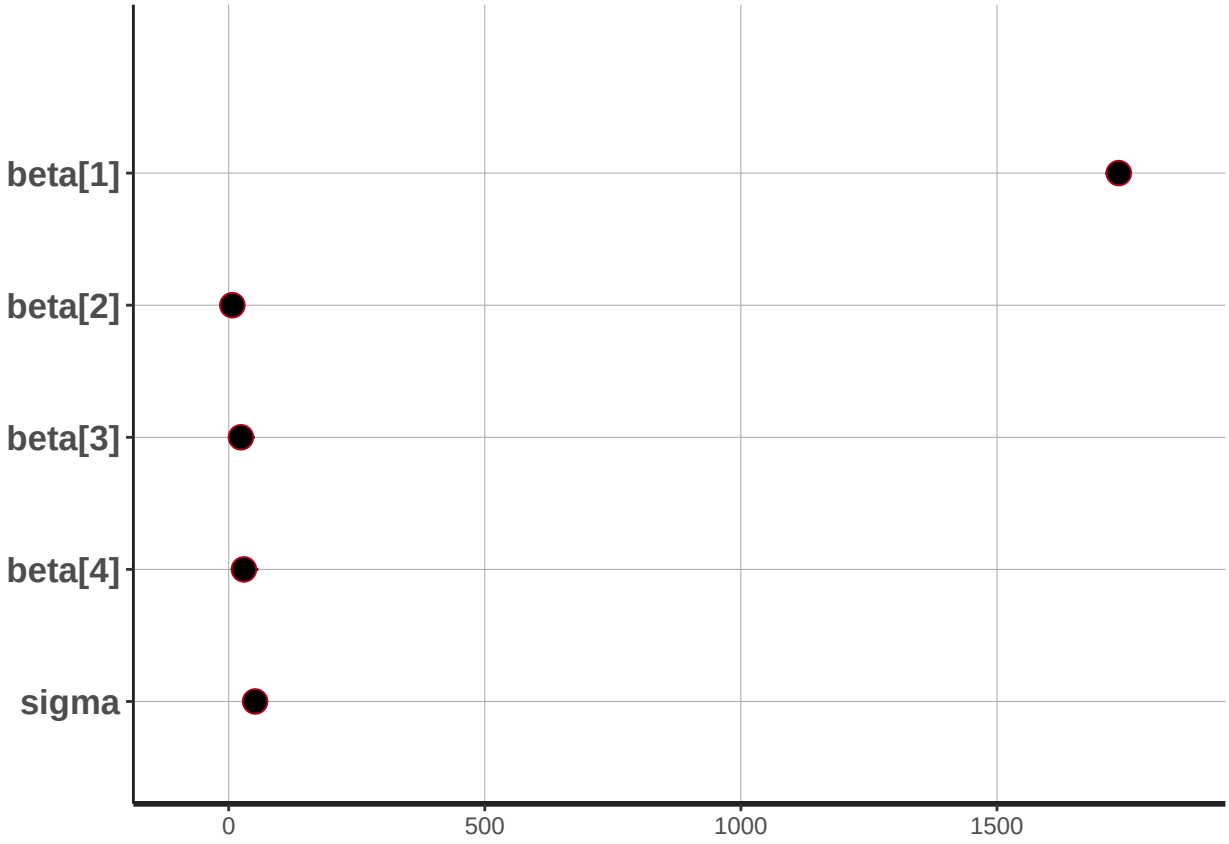



```
# Plot the correlation between the parameters
pairs(fit, pars="beta")
```




```
# plotting credible intervals
plot(fit,pars=c("beta", "sigma"))
```

```
## ci_level: 0.8 (80% intervals)
## outer_level: 0.95 (95% intervals)
```



Want to have an access to samples

```
post_beta<-As.mcmc.list(fit,pars="beta")
length(post_beta)
```

```
## [1] 4
```

```
str(post_beta[[1]])
```

```
##  'mcmc' num [1:500, 1:4] 1741 1727 1731 1718 1733 ...
##  - attr(*, "dimnames")=List of 2
##   ..$ : NULL
##   ..$ : chr [1:4] "beta[1]" "beta[2]" "beta[3]" "beta[4]"
##  - attr(*, "mcpar")= num [1:3] 501 1000 1
```

```
dim(post_beta[[1]])
```

```
## [1] 500   4
```

```
hist(post_beta[[1]][,3], breaks = 100)
```

Want to have an access to summary

```
fit_summary <- summary(fit)
print(fit_summary$summary)
```

```
##                  mean      se_mean           sd          2.5%
## beta[1]  1.737076e+03 5.295447e-01 1.272632e+01  1.711376e+03
## beta[2]  7.060912e+00 3.072301e-02 1.104496e+00  4.824384e+00
## beta[3]  2.433352e+01 5.451900e-01 1.333333e+01 -1.052169e+00
## beta[4]  3.043412e+01 5.664064e-01 1.356572e+01  5.612671e+00
## tau      3.761758e-04 8.868827e-07 3.214463e-05  3.167703e-04
## sigma    5.170095e+01 6.344756e-02 2.222143e+00  4.765700e+01
## lp__    -1.418238e+03 5.690184e-02 1.499884e+00 -1.421858e+03
##                  25%           50%           75%         97.5%     n_eff
## beta[1]  1.728712e+03  1.737822e+03  1.745804e+03  1.761002e+03  577.5649
## beta[2]  6.318879e+00  7.055822e+00  7.801302e+00  9.219903e+00 1292.4113
## beta[3]  1.522655e+01  2.390175e+01  3.298561e+01  5.083332e+01  598.1107
## beta[4]  2.118859e+01  2.965262e+01  3.956404e+01  5.793530e+01  573.6271
## tau      3.538495e-04  3.761346e-04  3.966924e-04  4.402978e-04 1313.6648
## sigma    5.020802e+01  5.156183e+01  5.316070e+01  5.618595e+01 1226.6317
## lp__    -1.419054e+03 -1.417924e+03 -1.417130e+03 -1.416160e+03  694.8049
##             Rhat
## beta[1] 1.005787
## beta[2] 1.002314
## beta[3] 1.004825
## beta[4] 1.006078
## tau     1.003887
```

```
## sigma   1.004057
## lp__    1.002819
```