Lab 3. Probabilistic Programming

Roi Naveiro

Stan in Linear Regression

Libraries

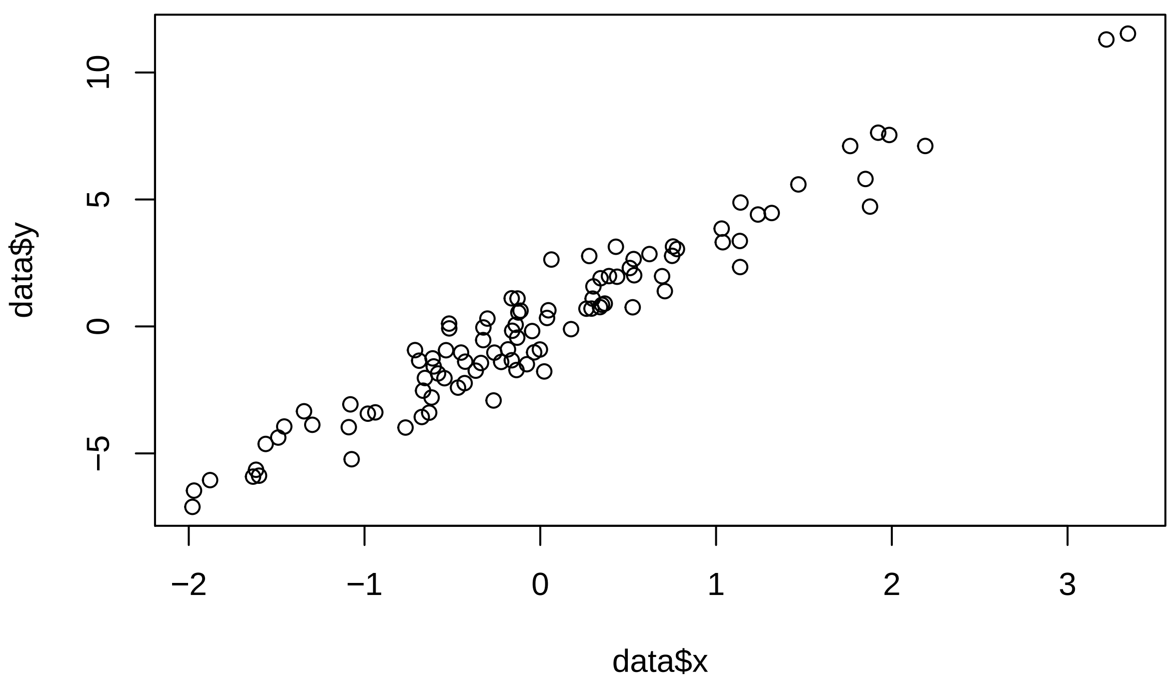
```
library(cmdstanr)
library(posterior)
library(bayesplot)
color_scheme_set("brightblue")
```

We first generate data from easy linear regression model

```
gen_dat <- function(n, beta, sigma) {
  x <- rnorm(n)
  y <- 0 + beta*x + rnorm(n, 0, sigma)
  data.frame(x = x, y = y)
}
```

We visualize them

```
data = gen_dat(100, 3.5, 0.85)
plot(data$x, data$y)
```



Posterior Inference using MCMC

Compile model

```
mod <- cmdstan_model("lr.stan")
mod$print()
```

```
## // Linear Model with Normal Errors
## data {
##   // number of observations
##   int N;
##   // response
##   vector[N] y;
##   // covariate
##   vector[N] x;
## }
## parameters {
##   // regression coefficient vector
##   real beta;
##   real<lower=0> sigma;
## }
## transformed parameters {
##   vector[N] mu;
##
##   mu = x * beta;
## }
## model {
##   // priors
##   beta ~ normal(0., 2.0);
##   sigma ~ exponential(0.01);
##   // likelihood
##   y ~ normal(mu, sigma);
## }
```

Run model using MCMC

```
data_l <- list(N=100, y=data$y, x=data$x )
fit_mcmc <- mod$sample(
  data = data_l,
  seed = 123,
  chains = 4,
  parallel_chains = 4,
  refresh = 500
)
```

```
## Running MCMC with 4 parallel chains...
##
## Chain 1 Iteration:    1 / 2000 [  0%]  (Warmup)
## Chain 1 Iteration:  500 / 2000 [ 25%]  (Warmup)
## Chain 1 Iteration: 1000 / 2000 [ 50%]  (Warmup)
## Chain 1 Iteration: 1001 / 2000 [ 50%]  (Sampling)
## Chain 1 Iteration: 1500 / 2000 [ 75%]  (Sampling)
## Chain 2 Iteration:    1 / 2000 [  0%]  (Warmup)
## Chain 2 Iteration:  500 / 2000 [ 25%]  (Warmup)
## Chain 2 Iteration: 1000 / 2000 [ 50%]  (Warmup)
## Chain 2 Iteration: 1001 / 2000 [ 50%]  (Sampling)
```

```
## Chain 3 Iteration:    1 / 2000 [  0%]  (Warmup)
## Chain 3 Iteration:  500 / 2000 [ 25%]  (Warmup)
## Chain 3 Iteration: 1000 / 2000 [ 50%]  (Warmup)
## Chain 3 Iteration: 1001 / 2000 [ 50%]  (Sampling)
## Chain 3 Iteration: 1500 / 2000 [ 75%]  (Sampling)
## Chain 4 Iteration:    1 / 2000 [  0%]  (Warmup)
## Chain 4 Iteration:  500 / 2000 [ 25%]  (Warmup)
## Chain 4 Iteration: 1000 / 2000 [ 50%]  (Warmup)
## Chain 4 Iteration: 1001 / 2000 [ 50%]  (Sampling)
## Chain 4 Iteration: 1500 / 2000 [ 75%]  (Sampling)
## Chain 1 Iteration: 2000 / 2000 [100%]  (Sampling)
## Chain 1 finished in 0.2 seconds.
## Chain 2 Iteration: 1500 / 2000 [ 75%]  (Sampling)
## Chain 2 Iteration: 2000 / 2000 [100%]  (Sampling)
## Chain 2 finished in 0.3 seconds.
## Chain 3 Iteration: 2000 / 2000 [100%]  (Sampling)
## Chain 3 finished in 0.3 seconds.
## Chain 4 Iteration: 2000 / 2000 [100%]  (Sampling)
## Chain 4 finished in 0.3 seconds.
##
## All 4 chains finished successfully.
## Mean chain execution time: 0.3 seconds.
## Total execution time: 0.5 seconds.
```

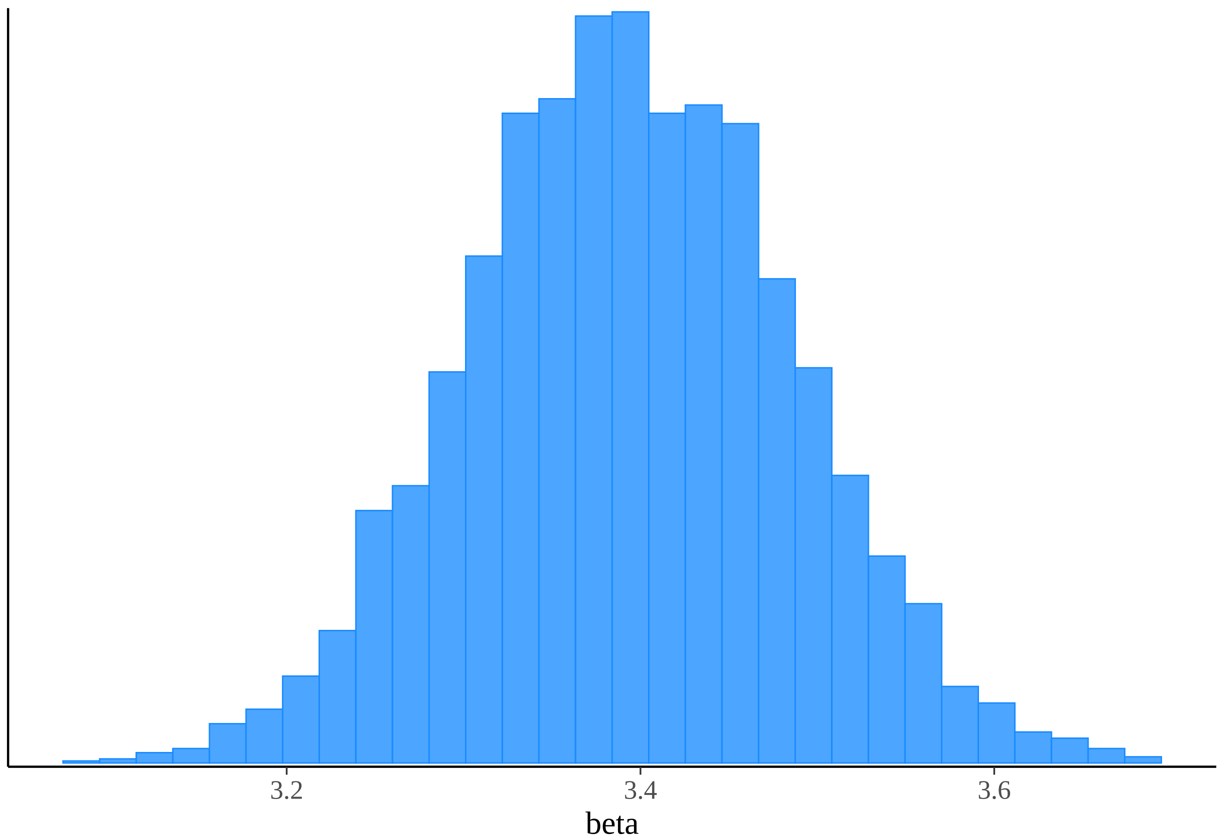
Summary of results

```
fit_mcmc$summary()
```

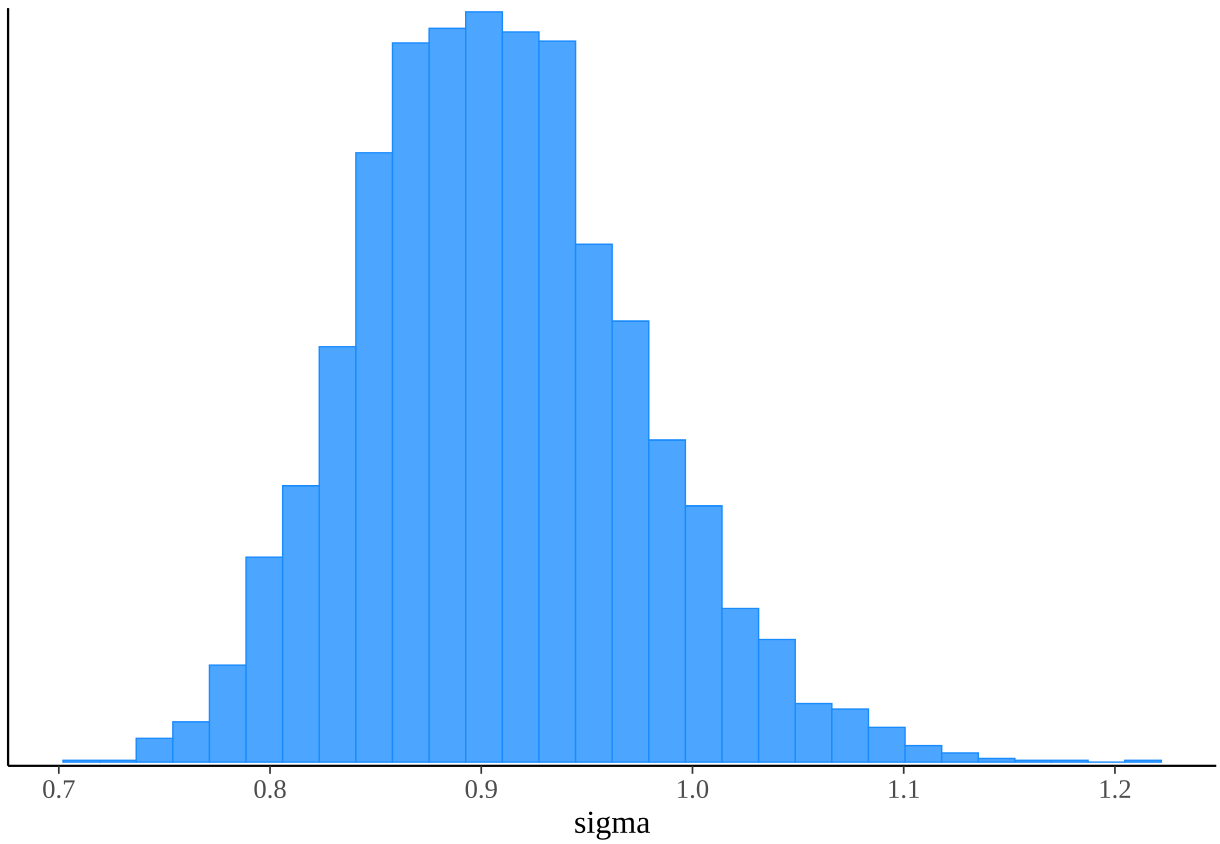
```
## # A tibble: 103 x 10
##    variable      mean    median      sd     mad       q5      q95  rhat ess_bulk
##    <chr>        <dbl>     <dbl>   <dbl>   <dbl>    <dbl>    <dbl> <dbl>    <dbl>
##  1 lp__     -41.4     -41.1     9.90e-1 7.34e-1 -4.34e+1 -4.04e+1  1.00    1749.
##  2 beta       3.39      3.39    9.10e-2 9.21e-2  3.25e+0  3.54e+0  1.00    3264.
##  3 sigma      0.909     0.905   6.61e-2 6.49e-2  8.06e-1  1.02e+0  1.00    3093.
##  4 mu[1]     -1.97     -1.97    5.29e-2 5.36e-2 -2.06e+0 -1.89e+0  1.00    3265.
##  5 mu[2]      3.86      3.86    1.03e-1 1.05e-1  3.69e+0  4.03e+0  1.00    3265.
##  6 mu[3]     -0.00725  -0.00725 1.94e-4 1.97e-4 -7.57e-3 -6.93e-3  1.00    3264.
##  7 mu[4]     -6.72     -6.72    1.80e-1 1.82e-1 -7.01e+0 -6.42e+0  1.00    3264.
##  8 mu[5]      1.46      1.46    3.91e-2 3.96e-2  1.40e+0  1.52e+0  1.00    3265.
##  9 mu[6]     -2.60     -2.60    6.97e-2 7.06e-2 -2.72e+0 -2.49e+0  1.00    3265.
## 10 mu[7]     -0.476    -0.476   1.27e-2 1.29e-2 -4.97e-1 -4.55e-1  1.00    3264.
## # ... with 93 more rows, and 1 more variable: ess_tail <dbl>
```

Posterior samples

```
mcmc_hist(fit_mcmc$draws("beta"))
```




```
mcmc_hist(fit_mcmc$draws("sigma"))
```



Posterior Inference using VI

```
fit_vi <- mod$variational(
  data = data_l,
  seed = 123,
  refresh = 500
)
```

```
## ------------------------------------------------------------
## EXPERIMENTAL ALGORITHM:
##   This procedure has not been thoroughly tested and may be unstable
##   or buggy. The interface is subject to change.
## ------------------------------------------------------------
## Gradient evaluation took 9e-06 seconds
## 1000 transitions using 10 leapfrog steps per transition would take 0.09 seconds.
## Adjust your expectations accordingly!
## Begin eta adaptation.
## Iteration:   1 / 250 [  0%]  (Adaptation)
## Iteration:  50 / 250 [ 20%]  (Adaptation)
## Iteration: 100 / 250 [ 40%]  (Adaptation)
## Iteration: 150 / 250 [ 60%]  (Adaptation)
## Iteration: 200 / 250 [ 80%]  (Adaptation)
## Success! Found best value [eta = 1] earlier than expected.
## Begin stochastic gradient ascent.
##   iter             ELBO   delta_ELBO_mean   delta_ELBO_med   notes
```

```
##    100         -141.771            1.000            1.000
##    200         -141.933            0.501            1.000
##    300         -141.749            0.334            0.001   MEDIAN ELBO CONVERGED
## Drawing a sample of size 1000 from the approximate posterior...
## COMPLETED.
## Finished in  0.1 seconds.
```

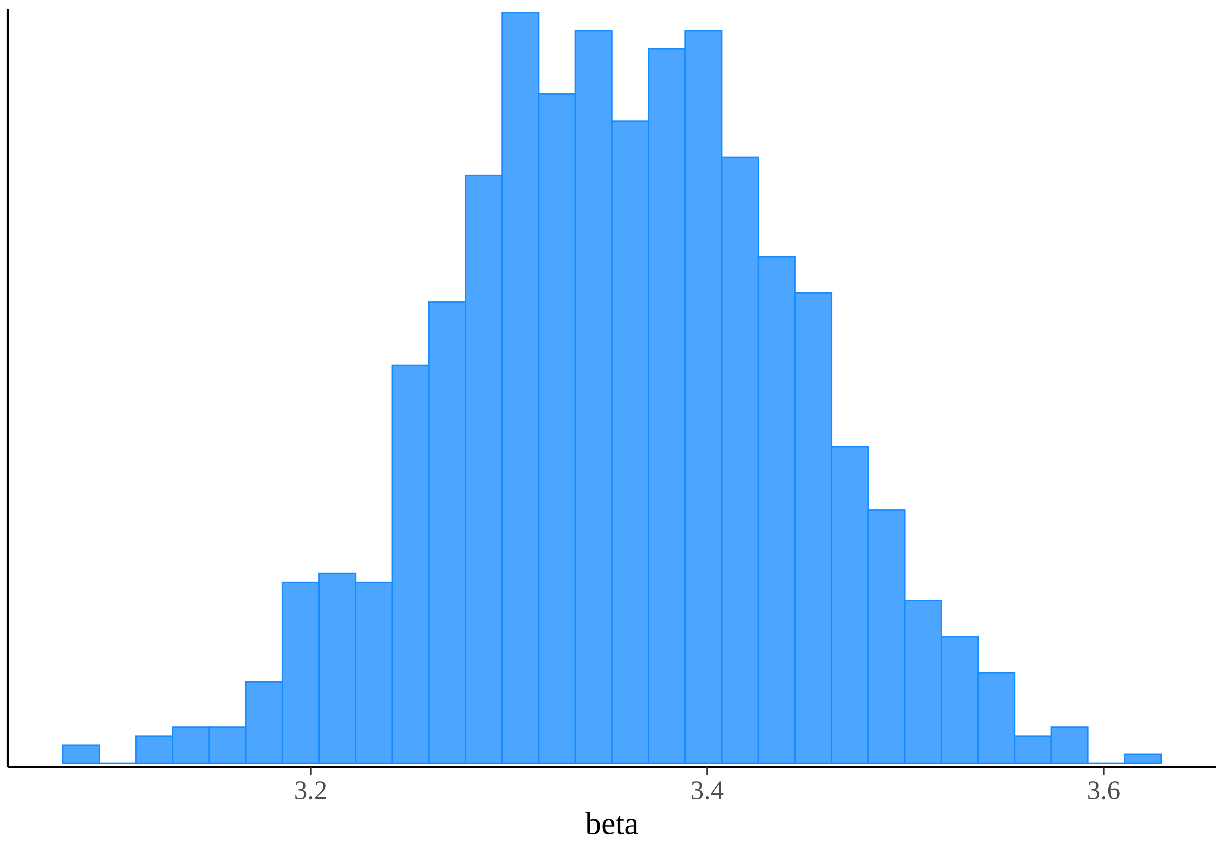
Summary of results

```
fit_vi$summary()
```

```
## # A tibble: 104 x 7
##    variable         mean     median       sd      mad         q5        q95
##    <chr>           <dbl>      <dbl>    <dbl>    <dbl>      <dbl>      <dbl>
##  1 lp__         -140.      -139.       1.15     0.792    -142.      -138.
##  2 lp_approx__    -1.07      -0.736    1.06     0.754      -3.17      -0.0576
##  3 beta            3.36       3.36     0.0868   0.0887      3.21       3.50
##  4 sigma           0.911      0.907    0.0689   0.0679      0.802      1.03
##  5 mu[1]          -1.95      -1.95     0.0505   0.0516     -2.04      -1.87
##  6 mu[2]           3.82       3.82     0.0987   0.101       3.65       3.98
##  7 mu[3]          -0.00717   -0.00717  0.000186 0.000189   -0.00748   -0.00686
##  8 mu[4]          -6.64      -6.64     0.172    0.175      -6.93      -6.36
##  9 mu[5]           1.44       1.44     0.0373   0.0381      1.38       1.50
## 10 mu[6]          -2.57      -2.57     0.0666   0.0680     -2.68      -2.46
## # ... with 94 more rows
```

Posterior samples

```
mcmc_hist(fit_vi$draws("beta"))
```



```r
mcmc_hist(fit_vi$draws("sigma"))
```