```r
# Load the lars package and the diabetes dataset
library(reshape2)
library(lars)
```

```
## Loaded lars 1.3
```

```r
data(diabetes)
library(GGally)
```

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

```
## Registered S3 method overwritten by 'GGally':
##   method from
##   +.gg   ggplot2
```

```r
library(ggplot2)
library(gridExtra)

library("rstan") # observe startup messages
```

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

```
##
## rstan version 2.32.3 (Stan version 2.26.1)
```

```
## 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 within-chain threading using `reduce_sum()` or `map_rect()` Stan functions,
## change `threads_per_chain` option:
## rstan_options(threads_per_chain = 1)
```

```r
options(mc.cores = parallel::detectCores())
rstan_options(auto_write = TRUE)
#data
X_matrix <- diabetes$x
class(X_matrix) <- "matrix"
y_vector <- diabetes$y

X_design <- cbind(1, X_matrix)
```

```r
# x matrix has been standardized to have unit L2 norm in each column and zero mean

#There are 10 explanatory variables, including age (age), sex (sex), body mass index (bmi) and mean art


#https://garthtarr.github.io/mplot/reference/diabetes.html

#tc
#Total cholesterol (mg/dL)? Desirable range: below 200 mg/dL

#ldl
#Low-density lipoprotein ("bad" cholesterol)? Desirable range: below 130 mg/dL

#hdl
#High-density lipoprotein ("good" cholesterol)? Desirable range: above 40 mg/dL
```

```r
#serum concentration of lamorigine (LTG),
#glucose (GLU)


#https://hastie.su.domains/Papers/LARS/

data_xy <- data.frame(X=X_matrix, y=y_vector)
ggpairs(rawx[c(1,3,4,11)])
ggpairs(rawx[c(5:10,11)])
```

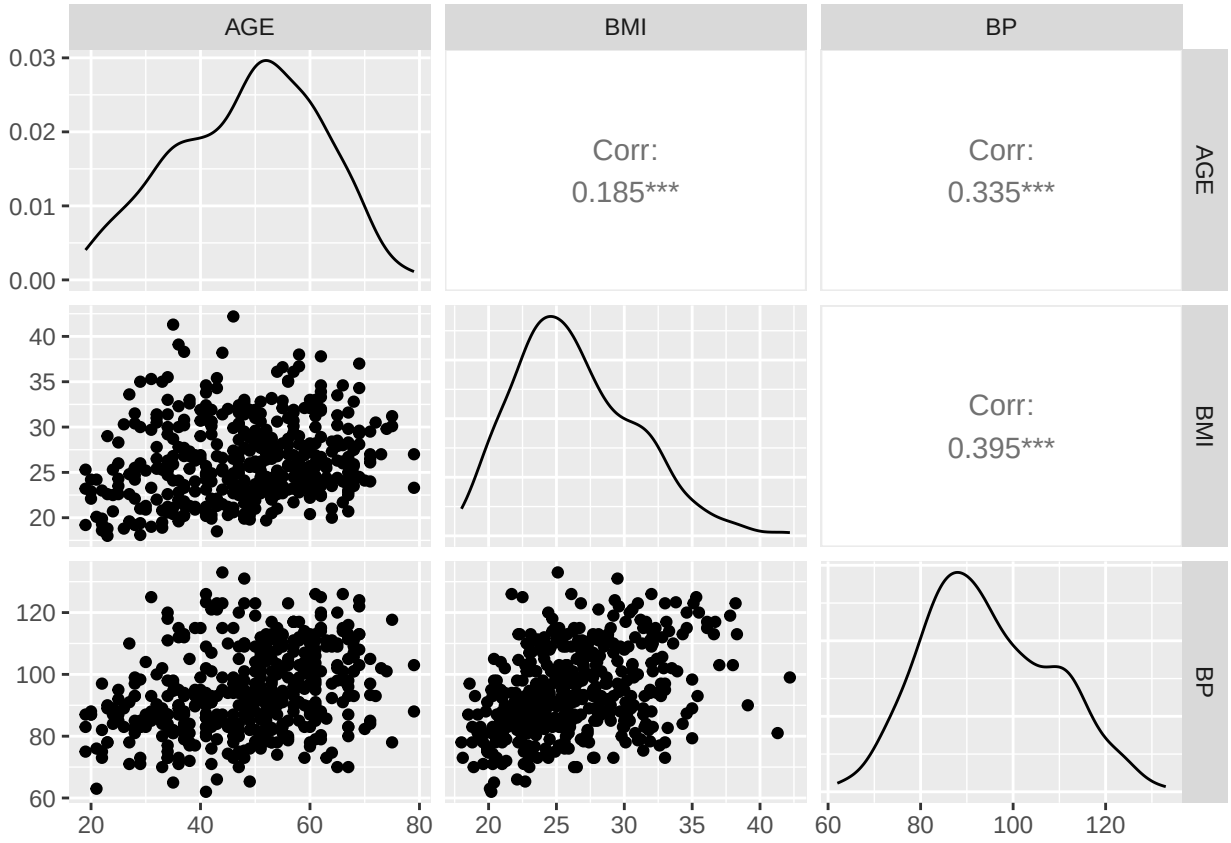
```r
#ggpairs(data_xy, columns=c(1,3,4,11),
#       ggplot2::aes(color=as.factor(X.sex>0)))
data_xy$X.tch.new <- rawx$S4.new
ggpairs(data_xy, columns=1:4,
    ggplot2::aes(color=factor(X.tch.new),alpha = 0.7))
```

```r
rawx$S4.new <- 1
rawx$S4.new[which(rawx$S4<3)] <- 2
rawx$S4.new[which((rawx$S4>=3) & (rawx$S4<4))] <- 3
rawx$S4.new[which((rawx$S4>=4) & (rawx$S4<5))] <- 4
rawx$S4.new[which((rawx$S4>=5) & (rawx$S4<6))] <- 5
rawx$S4.new[which(rawx$S4>=6)] <- 6
```

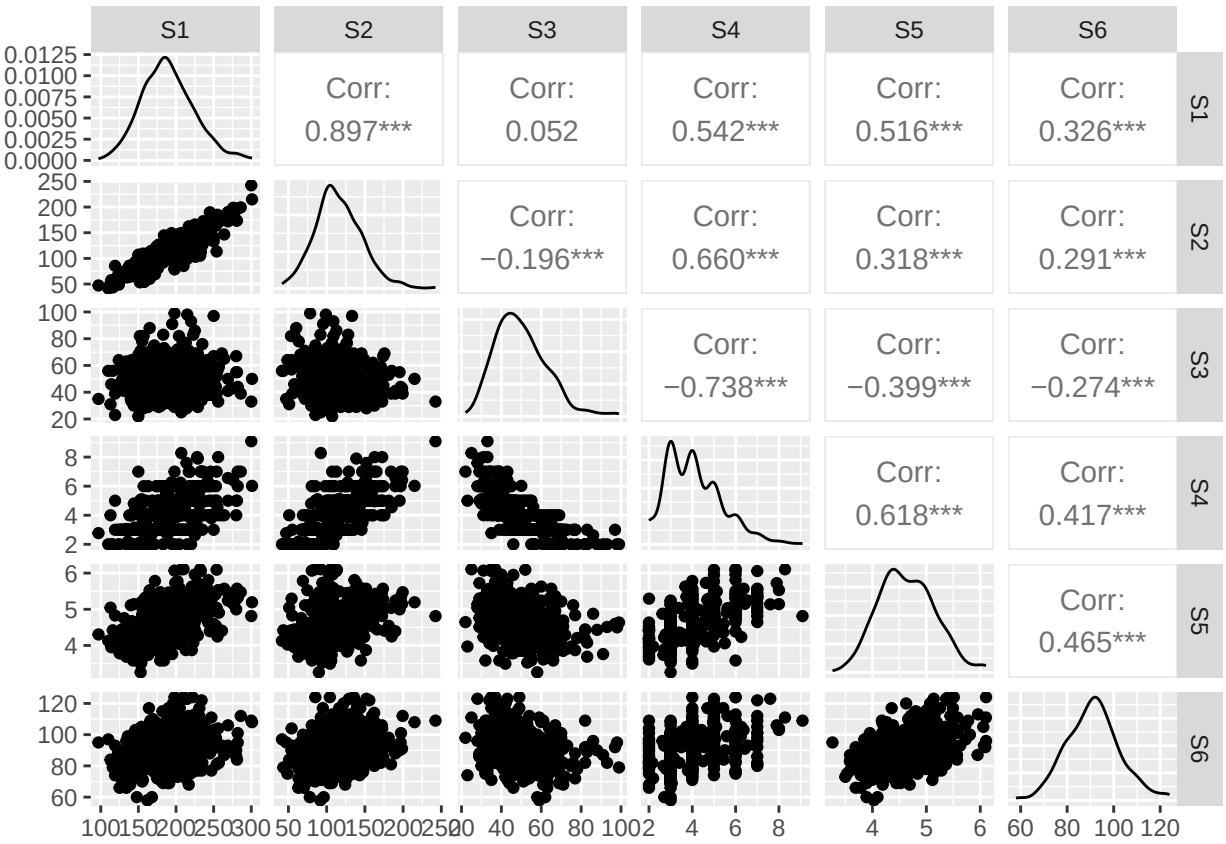
```r
# First, boxplot and density plot of tch
plot1 <- ggplot(X_temp, aes(x = factor(1), y = tch)) +
  geom_boxplot(fill = "skyblue", color = "black", alpha = 0.7) +
  ggtitle("Boxplot of tch") +
  theme_minimal()

plot2 <- ggplot(X_temp, aes(x = tch)) +
  geom_density(fill = "skyblue", color = "black") +
  ggtitle("Density Plot of tch") +
  theme_minimal()

# Second, boxplot and density plot of tch, classified by sex
plot3 <- ggplot(X_temp, aes(x = factor(as.integer(sex>0)), y = tch,
                            fill = factor(as.integer(sex>0)))) +
  geom_boxplot(color = "black", alpha = 0.7) +
  labs(x = "Sex", y = "tch", fill = "Sex",
       # Set axis and fill labels
       title = "Boxplot of tch by Sex")

plot4 <- ggplot(data_xy, aes(x = y,
                            color = factor(X.tch.new)))+
  geom_density()+
  labs(x = "tch", y = "Density", color = "Sex",
       # Set axis and fill labels
       title = "Density Plot of tch by Sex")

# Display the plots
plot4
```

```r
rawx <- read.table(file='https://hastie.su.domains/Papers/LARS/diabetes.data', header=T)
```

```
ggpairs(rawx[c(1,3,4)])
```




```
ggpairs(rawx[5:10])
```




```r
x <- diabetes$x
y <- diabetes$y
# Pairwise scatter plots for the first six variables in x
pairs_data <- as.data.frame(x[, 1:6])
pairs_plot <- ggplot(pairs_data, aes(color = y)) +
  geom_point() +
  ggtitle("Pairwise Scatter Plots for x")

# Boxplot of the target variable "y" (disease progression)
boxplot_plot <- ggplot() +
  geom_boxplot(aes(y = y)) +
  ggtitle("Boxplot of Disease Progression (y)") +
  ylab("Disease Progression")

# Correlation matrix heatmap for all variables in x
cor_matrix_x <- cor(x)
cor_matrix_plot <- ggplot(data = as.data.frame(cor_matrix_x), aes(x = Var1, y = Var2, fill = value)) +
  geom_tile() +
  ggtitle("Correlation Matrix Heatmap for x")

# Histogram of the target variable "y"
hist_plot <- ggplot() +
  geom_histogram(aes(x = y), fill = "lightblue", color = "black") +
  ggtitle("Histogram of Disease Progression (y)") +
  xlab("Disease Progression")

# Scatter plot of the first column in x against "y"
```

```r
scatter_plot_x1 <- ggplot(data = as.data.frame(cbind(x[, 1], y)), aes(x = V1, y = y)) +
  geom_point(color = "blue") +
  ggtitle("Scatter Plot: x1 vs. Disease Progression") +
  xlab("x1") +
  ylab("Disease Progression")
```

The x matrix has been standardized to have unit L2 norm in each column and zero mean.

We can directly use x and y.

(In extended models, we may consider x2.)

```r
#ols

ols <- lm(y_vector~X_matrix)
summary(ols)
```

```
##
## Call:
## lm(formula = y_vector ~ X_matrix)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -155.829  -38.534   -0.227   37.806  151.355
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)  152.133      2.576  59.061  < 2e-16 ***
## X_matrixage  -10.012     59.749  -0.168 0.867000
## X_matrixsex -239.819     61.222  -3.917 0.000104 ***
## X_matrixbmi  519.840     66.534   7.813 4.30e-14 ***
## X_matrixmap  324.390     65.422   4.958 1.02e-06 ***
## X_matrixtc  -792.184    416.684  -1.901 0.057947 .
## X_matrixldl  476.746    339.035   1.406 0.160389
## X_matrixhdl  101.045    212.533   0.475 0.634721
## X_matrixtch  177.064    161.476   1.097 0.273456
## X_matrixltg  751.279    171.902   4.370 1.56e-05 ***
## X_matrixglu   67.625     65.984   1.025 0.305998
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 54.15 on 431 degrees of freedom
## Multiple R-squared:  0.5177, Adjusted R-squared:  0.5066
## F-statistic: 46.27 on 10 and 431 DF,  p-value: < 2.2e-16
```

we use rstan.

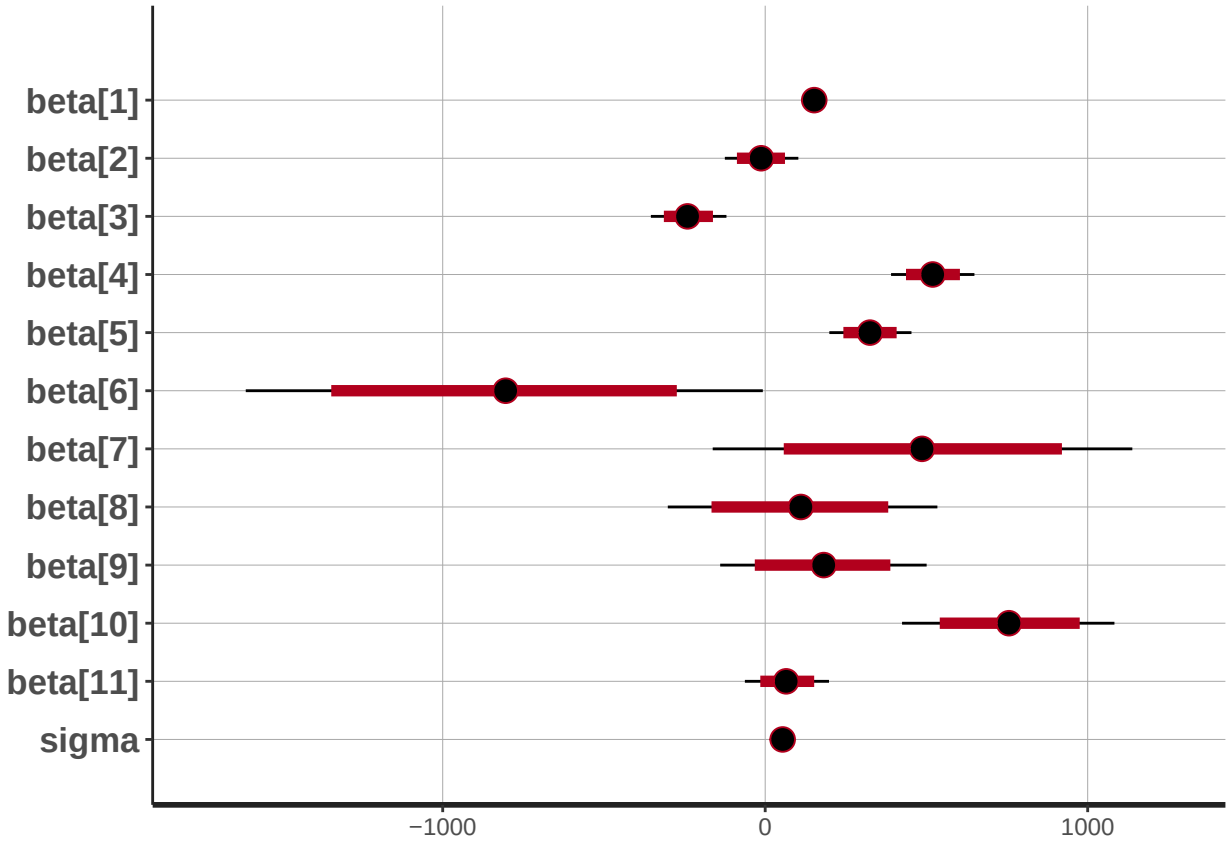
```r
set.seed(123)
# Create a data list for Stan
data_list_m1 <- list(
  N = length(y_vector),  # Number of observations
  K = dim(X_design)[2], # Number of predictors
  x = X_design,  # Predictor variable
  y = y_vector  # Response variable
)
```

```r
# Compile the Stan model
stan_m1 <- stan_model(file='./priors/prior_M1_v2.stan')

# Fit the model to the data
stan_fit_m1 <- sampling(stan_m1,
                    data = data_list_m1,
                    chains = 4,
                    iter = 2000)

# Print a summary of the results
print(stan_fit_m1)
```

```
## Inference for Stan model: anon_model.
## 4 chains, each with iter=2000; warmup=1000; thin=1;
## post-warmup draws per chain=1000, total post-warmup draws=4000.
##
##              mean se_mean     sd     2.5%      25%      50%      75%    97.5%
## beta[1]    152.10    0.04   2.59   147.03   150.33   152.10   153.89   157.06
## beta[2]    -12.23    0.94  58.47  -125.00   -51.64   -12.45    26.45   103.42
## beta[3]   -239.03    1.03  60.04  -354.46  -280.22  -240.75  -197.65  -120.17
## beta[4]    519.34    1.14  65.94   390.38   475.32   519.34   563.84   648.85
## beta[5]    325.18    1.16  64.53   198.94   283.15   324.89   367.65   453.62
## beta[6]   -805.25   12.25 412.33 -1610.34 -1076.34  -804.81  -528.15    -7.79
## beta[7]    486.16    9.59 334.74  -162.53   257.81   486.36   707.34  1138.39
## beta[8]    108.47    5.98 213.64  -301.89   -38.95   110.58   247.76   533.85
## beta[9]    180.17    3.32 164.95  -139.57    68.90   181.51   291.06   500.57
## beta[10]   757.34    4.82 169.90   424.47   638.40   755.93   873.20  1082.96
## beta[11]    66.81    1.06  65.62   -62.97    22.83    65.10   111.41   198.11
## sigma       54.18    0.03   1.86    50.73    52.91    54.16    55.40    57.97
## lp__     -1989.81    0.06   2.46 -1995.50 -1991.26 -1989.53 -1988.02 -1985.98
##          n_eff Rhat
## beta[1]   4543    1
## beta[2]   3876    1
## beta[3]   3405    1
## beta[4]   3332    1
## beta[5]   3069    1
## beta[6]   1132    1
## beta[7]   1219    1
## beta[8]   1275    1
## beta[9]   2473    1
## beta[10]  1240    1
## beta[11]  3826    1
## sigma     3948    1
## lp__      1496    1
##
## Samples were drawn using NUTS(diag_e) at Mon Dec 11 20:52:58 2023.
## For each parameter, n_eff is a crude measure of effective sample size,
## and Rhat is the potential scale reduction factor on split chains (at
## convergence, Rhat=1).
```

```r
# Plot the posterior distributions
plot(stan_fit_m1,
     pars=c('beta', 'sigma'))
```

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



M2

```r
# Create a data list for Stan
set.seed(123)
data_list_m2 <- list(
  N = length(y_vector),  # Number of observations
  K = dim(X_design)[2], # Number of predictors, contain intercept
  x = X_design,  # Predictor variable
  y = y_vector,  # Response variable

  m0 = rep(1, dim(X_design)[2]),
  C0 = diag(1, dim(X_design)[2]),
  v0 = 1,
  s0 = 1
)


# Compile the Stan model
stan_m2 <- stan_model(file='./priors/prior_M2.stan')

# Fit the model to the data
stan_fit_m2 <- sampling(stan_m2,
                        data = data_list_m2,
                        chains = 4,
                        iter = 2000)
```

```r
# Print a summary of the results
print(stan_fit_m2)
```

```
## Inference for Stan model: anon_model.
## 4 chains, each with iter=2000; warmup=1000; thin=1;
## post-warmup draws per chain=1000, total post-warmup draws=4000.
##
##              mean se_mean     sd     2.5%      25%      50%      75%    97.5%
## beta[1]    151.81    0.03   2.99   145.92   149.82   151.85   153.80   157.59
## beta[2]     29.54    0.55  46.52   -61.07    -1.62    30.84    60.69   121.86
## beta[3]    -83.38    0.53  45.14  -173.08  -114.10   -83.68   -52.99     4.38
## beta[4]    306.85    0.61  46.13   216.25   275.93   306.41   337.45   398.21
## beta[5]    201.36    0.59  47.07   111.46   169.22   201.12   232.92   294.82
## beta[6]      6.03    0.73  52.47   -97.92   -28.36     7.37    41.57   105.90
## beta[7]    -29.59    0.69  51.56  -130.39   -63.98   -29.94     5.97    71.78
## beta[8]   -150.49    0.65  49.64  -249.23  -184.27  -150.44  -117.94   -52.64
## beta[9]    118.21    0.75  52.50    15.57    82.38   117.47   154.11   219.27
## beta[10]   263.47    0.62  49.58   167.43   230.51   263.24   295.60   360.79
## beta[11]   111.78    0.55  46.07    21.79    81.52   111.75   143.32   200.59
## sigma2    3898.91    2.99 267.41  3414.87  3712.84  3886.06  4067.51  4483.70
## sigma       62.40    0.02   2.13    58.44    60.93    62.34    63.78    66.96
## lp__     -2107.83    0.06   2.51 -2113.63 -2109.28 -2107.46 -2106.02 -2103.98
##          n_eff Rhat
## beta[1]   7649    1
## beta[2]   7206    1
## beta[3]   7129    1
## beta[4]   5745    1
## beta[5]   6269    1
## beta[6]   5113    1
## beta[7]   5592    1
## beta[8]   5836    1
## beta[9]   4881    1
## beta[10]  6388    1
## beta[11]  6925    1
## sigma2    8010    1
## sigma     8103    1
## lp__      1558    1
##
## Samples were drawn using NUTS(diag_e) at Mon Dec 11 20:52:59 2023.
## For each parameter, n_eff is a crude measure of effective sample size,
## and Rhat is the potential scale reduction factor on split chains (at
## convergence, Rhat=1).
```

```r
# Plot the posterior distributions
plot(stan_fit_m2,
     pars=c('beta', 'sigma'))
```

```
## ci_level: 0.8 (80% intervals)
```

```
## outer_level: 0.95 (95% intervals)
```

M3

```
# Create a data list for Stan
set.seed(123)
data_list_m3 <- list(
  N = length(y_vector),  # Number of observations
  K = dim(X_design)[2], # Number of predictors, contain intercept
  x = X_design,  # Predictor variable
  y = y_vector,  # Response variable

  b0 = c(5,7,10,9,29,10,39,2,48,10,23),
  g = 1
)


# Compile the Stan model
stan_m3 <- stan_model(file='./priors/prior_M3.stan')

# Fit the model to the data
stan_fit_m3 <- sampling(stan_m3,
                    data = data_list_m3,
                    chains = 4,
                    iter = 2000)

# Print a summary of the results
print(stan_fit_m3)
```

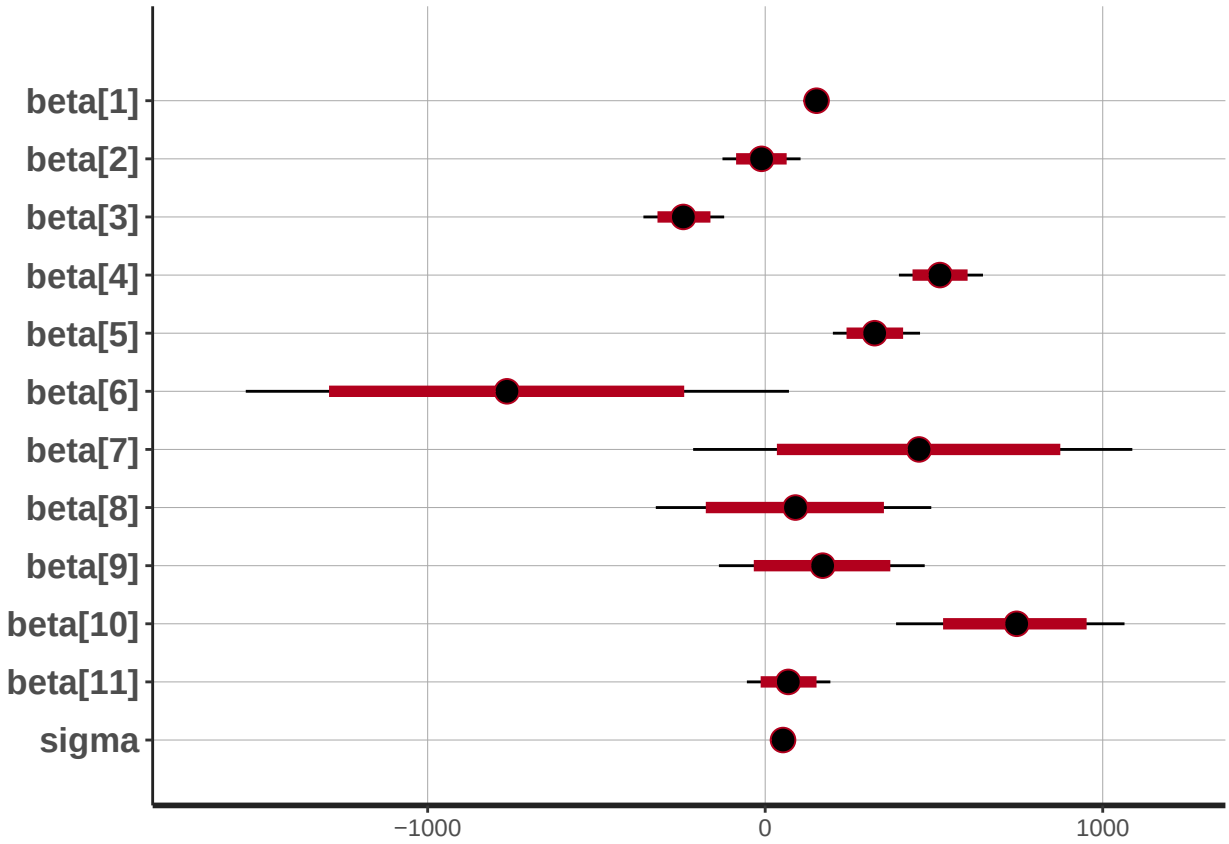
```
## Inference for Stan model: anon_model.
```

```
## 4 chains, each with iter=2000; warmup=1000; thin=1;
## post-warmup draws per chain=1000, total post-warmup draws=4000.
##
##              mean se_mean     sd     2.5%      25%      50%      75%    97.5%
## beta[1]    152.11    0.04   2.53   146.97   150.42   152.11   153.78   157.15
## beta[2]    -10.82    0.91  58.60  -126.56   -49.76   -10.76    28.34   104.79
## beta[3]   -241.02    1.01  60.80  -360.84  -281.62  -242.53  -200.46  -121.53
## beta[4]    517.90    1.01  64.15   396.18   473.79   517.58   559.98   645.54
## beta[5]    325.08    1.01  65.97   200.58   280.57   324.37   370.20   458.72
## beta[6]   -763.13   12.30 409.83 -1538.68 -1048.08  -765.07  -494.92    70.14
## beta[7]    454.15    9.85 331.04  -213.28   236.10   455.76   679.49  1087.67
## beta[8]     86.13    5.84 209.80  -324.95   -60.06    89.71   232.82   492.01
## beta[9]    170.06    3.01 156.77  -138.23    63.88   170.25   278.38   472.62
## beta[10]   741.66    4.65 167.56   387.78   632.25   744.95   855.00  1064.50
## beta[11]    69.01    0.99  64.14   -54.26    25.20    68.47   112.25   193.08
## sigma       52.89    0.03   1.79    49.49    51.68    52.83    54.10    56.55
## sigma2    2800.24    3.03 189.91  2448.91  2671.11  2791.33  2926.28  3197.81
## lp__     -2077.32    0.06   2.56 -2083.25 -2078.79 -2076.96 -2075.46 -2073.38
##          n_eff Rhat
## beta[1]   4177    1
## beta[2]   4180    1
## beta[3]   3643    1
## beta[4]   4047    1
## beta[5]   4296    1
## beta[6]   1111    1
## beta[7]   1129    1
## beta[8]   1292    1
## beta[9]   2712    1
## beta[10]  1300    1
## beta[11]  4201    1
## sigma     3929    1
## sigma2    3918    1
## lp__      1843    1
##
## Samples were drawn using NUTS(diag_e) at Mon Dec 11 20:53:01 2023.
## For each parameter, n_eff is a crude measure of effective sample size,
## and Rhat is the potential scale reduction factor on split chains (at
## convergence, Rhat=1).
```

```
# Plot the posterior distributions
plot(stan_fit_m3,
     pars=c('beta', 'sigma'))
```

```
## ci_level: 0.8 (80% intervals)
```

```
## outer_level: 0.95 (95% intervals)
```




```
# Create a data list for Stan
set.seed(123)
data_list_m6 <- list(
  N = length(y_vector),  # Number of observations
  K = dim(X_design)[2], # Number of predictors, contain intercept
  x = X_design,  # Predictor variable
  y = y_vector,  # Response variable
  b0 = c(5,7,10,9,29,10,39,2,48,10,23)
)


# Compile the Stan model
stan_m6 <- stan_model(file='./priors/prior_M6.stan')

# Fit the model to the data
stan_fit_m6 <- sampling(stan_m6,
                    data = data_list_m6,
                    chains = 4,
                    iter = 2000)
```

```
## Warning: There were 620 divergent transitions after warmup. See
## https://mc-stan.org/misc/warnings.html#divergent-transitions-after-warmup
## to find out why this is a problem and how to eliminate them.
```

```
## Warning: There were 975 transitions after warmup that exceeded the maximum treedepth. Increase max_t
## https://mc-stan.org/misc/warnings.html#maximum-treedepth-exceeded
```

```
## Warning: Examine the pairs() plot to diagnose sampling problems
```

```
## Warning: The largest R-hat is 1.53, indicating chains have not mixed.
## Running the chains for more iterations may help. See
## https://mc-stan.org/misc/warnings.html#r-hat
```

```
## Warning: Bulk Effective Samples Size (ESS) is too low, indicating posterior means and medians may be
## Running the chains for more iterations may help. See
## https://mc-stan.org/misc/warnings.html#bulk-ess
```

```
# Print a summary of the results
print(stan_fit_m6)
```

```
## Inference for Stan model: anon_model.
## 4 chains, each with iter=2000; warmup=1000; thin=1;
## post-warmup draws per chain=1000, total post-warmup draws=4000.
##
##              mean se_mean     sd     2.5%      25%      50%      75%    97.5%
## g          442.58  101.58 176.14   174.54   291.51   407.76   684.23   684.23
## beta[1]    151.29    0.34   2.34   146.88   150.17   150.55   152.81   156.50
## beta[2]    -15.47    1.25  50.90  -116.64   -36.04   -29.77    14.07    99.45
## beta[3]   -235.40    0.94  52.56  -348.76  -262.38  -231.84  -209.68  -128.18
## beta[4]    525.69    7.44  57.05   400.59   490.67   544.43   549.86   633.01
## beta[5]    314.29    7.40  56.18   206.41   290.21   294.48   348.94   436.54
## beta[6]   -622.53  219.44 468.55 -1511.47  -991.99  -632.54   -97.02   -31.65
## beta[7]    364.25  145.78 355.48  -147.35    10.80   336.96   639.04  1075.20
## beta[8]     49.95   65.26 206.16  -298.81  -113.36    14.99   202.64   476.68
## beta[9]    164.92    3.68 138.88   -99.59   103.31   128.35   249.42   473.36
## beta[10]   693.09   72.50 178.78   431.66   517.90   682.12   829.02  1050.67
## beta[11]    51.11   21.83  63.91   -50.85    -0.49    40.13    96.87   186.71
## sigma       54.05    0.03   1.64    50.79    53.26    53.81    54.82    57.75
## lp__     -2075.70    0.06   2.18 -2080.88 -2076.72 -2075.10 -2074.64 -2072.11
##          n_eff Rhat
## g            3 1.71
## beta[1]     46 1.05
## beta[2]   1649 1.02
## beta[3]   3126 1.00
## beta[4]     59 1.04
## beta[5]     58 1.04
## beta[6]      5 1.35
## beta[7]      6 1.25
## beta[8]     10 1.14
## beta[9]   1426 1.01
## beta[10]     6 1.24
## beta[11]     9 1.15
## sigma     2503 1.00
## lp__      1448 1.01
##
## Samples were drawn using NUTS(diag_e) at Mon Dec 11 20:53:34 2023.
## For each parameter, n_eff is a crude measure of effective sample size,
## and Rhat is the potential scale reduction factor on split chains (at
## convergence, Rhat=1).
```

```
# Plot the posterior distributions
plot(stan_fit_m6,
     pars=c('beta', 'sigma', 'g'))
```

```
## ci_level: 0.8 (80% intervals)
```

```
## outer_level: 0.95 (95% intervals)
```