# Lab 14: Robust Linear Regression and MCMC

*Wayne Stewart*

## MCMC problem

We have learned a lot about GLM regression. One MCMC problem we may encounter is strong covariance between parameters. This will cause the sampler to move slowly through the parameter space and cause strong autocorrelation for a parameter.

### Fix

One method to fix this is to transform the x and y variables. This may also help for situations where the size of the x and y metrics are large for the computer numerics.

### Standardize the variables

Note that $Z$ has mean 0 and standard deviation 1.

$$Z = \frac{X - \bar{X}}{S_X}$$

## Task 1

Prove sample mean of $Z$ is 0 and standard deviation is 1.

## Outliers problem

Outliers can be a problem because they may overly influence an analysis - meaning they might be most responsible for parameter estimates and dominate the rest of the data in terms of their impact on estimates.

One method to lessen the impact of outliers is to use a distribution on Y that has large tails.

### Fix

The t-distribution would be a suitable replacement to the normal.

Read pages 479-487 (Section 17.2)

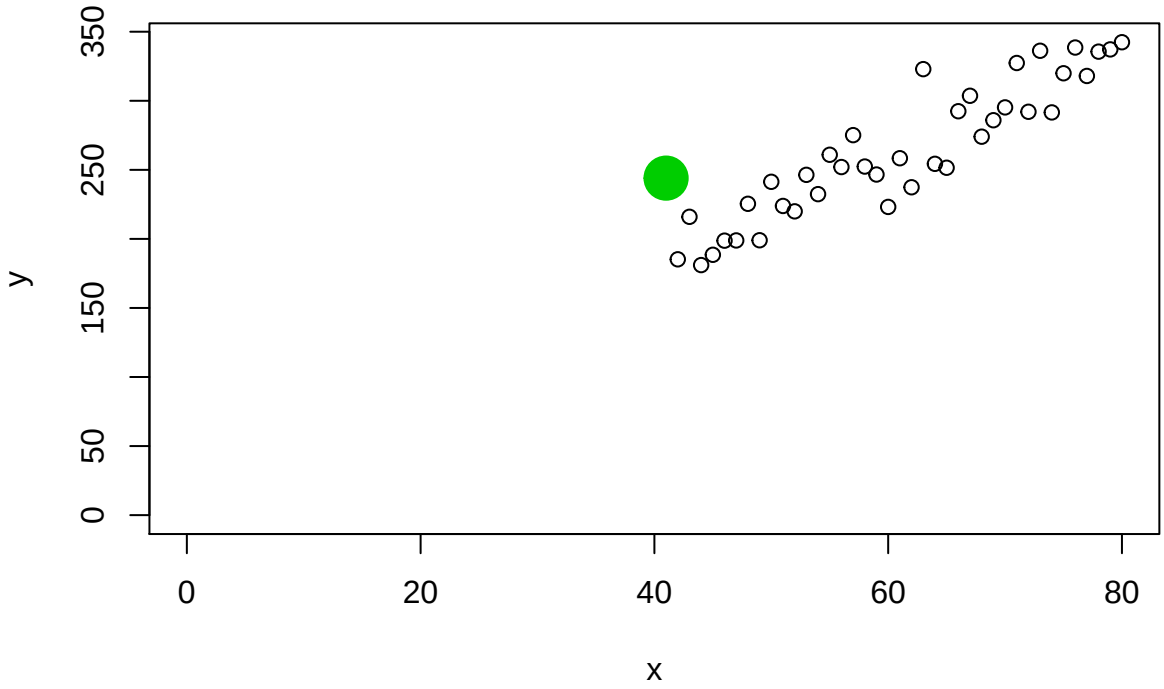
## Task 2

After reading the above sections do the following:

## Make a Jags model that will analyze the following simulated data

Do this by NOT using a t distribution and NOT using ceneterd variables. Make sure you diagnose the MCMC

```
x = 42:80
set.seed(34) # we will all have the same data
y = 20 + 4*x + rnorm(39,0,20)
xx = 41
yy = 20+4*xx + 60
x = c(xx,x)
y = c(yy, y)
plot(y~x, xlim=range(c(0, x)),ylim=range(c(0,y)))
points(xx,yy,pch=19,cex=3,col="green3")
```




## Make a jags model that will analyze the data.

See pages 485-486. This time center the data and use a t distribution. Make sure you diagnose the MCMC. Back transform (transform to original scale) and then:

## Compare results!!

## Summarize what you have learnt!!

**Now read the QR document on the next page and answer the questions!**

# QR decomposition

Dr Wayne Stewart

2022-01-14

## Contents



## Introduction

When conducting Bayesian modeling there is a distinction between the theoretical truth of the expressions used and the ability, given the tools available, to sample from the posterior.

In the case of MLR we have

$$E(Y) = \eta = X\beta$$

If the sampler is moving through the space of the $\beta$ parameters $(\beta_0, \beta_1 \ldots, \beta_k)$ and these are highly correlated then the sampler will jump in small steps and not quickly visit a representative subset of the parameter space.

To get over this we can factor the design matrix $X$ by using the thin QR factorization theorem (as opposed to the "fat" one).

$$X_{n\times(k+1)} = Q_{n\times(k+1)}R_{(k+1)\times(k+1)}$$

Where Q contains orthogonal columns and R is an upper triangular matrix.

To help in the calculation we can define

$$X = Q^*R^*$$

where $Q^* = Q\sqrt{n-1}$ and $R^* = R\frac{1}{\sqrt{n-1}}$.

This means that

$$\eta = X\beta = Q^*R^*\beta = Q^*\theta$$

where $\theta = R^*\beta$. To estimate $\theta$, however, we do not need to know $\beta$, that is, we will not find $\theta$ from $\beta$ but the reverse. Our regression will be using $\theta$ as the unknown parameters. This will give better sampling characteristics and then we will apply the transformation

$$\beta = (R^*)^{-1}\theta$$

to find the wanted posterior $\beta$ parameters.

## R example

To show you how to the QR factorization works we will use the ddt data set.

First lets make X

```
ddt <- Intro2R::ddt
X <- model.matrix(LENGTH ~ WEIGHT + DDT, ddt)
head(X,2)
```

```
##   (Intercept) WEIGHT DDT
## 1           1    732  10
## 2           1    795  16
```

Now we will use the `qr()` function

```
QR <- qr(X)
Q <- qr.Q(QR)
R <- qr.R(QR)
head(Q)
```

```
##             [,1]        [,2]         [,3]
## [1,] -0.08333333 -0.07055892 -0.013043077
## [2,] -0.08333333 -0.05656774 -0.007775982
## [3,] -0.08333333 -0.11164413 -0.002481158
## [4,] -0.08333333 -0.12985487 -0.004398154
## [5,] -0.08333333  0.04492384  0.022335308
## [6,] -0.08333333  0.04559009  0.107351709
```

```
R
```

```
##   (Intercept)     WEIGHT        DDT
## 1         -12 -12596.583 -292.26000
## 2           0   4502.837  -14.00628
## 3           0      0.000 1176.35340
```

Notice that Q has the same dimension as X and has orthogonal columns.

```
dim(X)
```

```
## [1] 144   3
```

```
dim(Q)
```

```
## [1] 144   3
```

```
round(t(Q)%*%Q,4)
```

```
##      [,1] [,2] [,3]
## [1,]    1    0    0
## [2,]    0    1    0
## [3,]    0    0    1
```

Once we have Q and R we can make $Q^*$ and $R^*$.

We can verify that the factorization is correct

```
head(round(Q%*%R-X,5))
```

```
##      (Intercept) WEIGHT DDT
## [1,]           0      0   0
## [2,]           0      0   0
## [3,]           0      0   0
## [4,]           0      0   0
## [5,]           0      0   0
## [6,]           0      0   0
```

## More documentation

Please note that you **MUST** read the following items:

- https://mc-stan.org/docs/2_28/stan-users-guide/QR-reparameterization.html
- https://en.wikipedia.org/wiki/QR_decomposition

Since this Stan program generates equivalent predictions for $y$ and the same posterior distribution for $\alpha$, $\beta$, and $\sigma$ as the previous Stan program, many wonder why the version with this QR reparameterization performs so much better in practice, often both in terms of wall time and in terms of effective sample size. The reasoning is threefold:

1. The columns of $Q^*$ are orthogonal whereas the columns of $x$ generally are not. Thus, it is easier for a Markov Chain to move around in $\theta$-space than in $\beta$-space.
2. The columns of $Q^*$ have the same scale whereas the columns of $x$ generally do not. Thus, a Hamiltonian Monte Carlo algorithm can move around the parameter space with a smaller number of larger steps
3. Since the covariance matrix for the columns of $Q^*$ is an identity matrix, $\theta$ typically has a reasonable scale if the units of $y$ are also reasonable. This also helps HMC move efficiently without compromising numerical accuracy.

Figure 1: Taken from the STAN documentation

## A STAN model

Because we wish to carry out a regression in STAN we must use STAN functions.

The basic STAN blocks are **data**, **parameters**, and **model**. We will need extra blocks to carry out our thin QR on the data (**transformed data**) and another block for the transformation of $\theta$ to $\beta$ (**generated quantities**).

## STAN code with extra blocks

Please look at the code below and see how the QR factorization plays out in practice. These tools will really help with making the ampler work more efficiently.

```
library(rstan)
library(Intro2R)


with(ddt, lm(LENGTH ~ SPECIES + DDT))->ylm
X <- model.matrix(ylm)# make a design matrix
y <- ddt[,"LENGTH"]


dataslr5 <- list(X = X, y = y, N = length(y),nbetas=dim(X)[2])
```

```
data {
  int<lower=0> N;
  int<lower=0> nbetas;
  matrix[N,nbetas] X; // Design matrix
  vector[N] y; // Vector of response values
}

transformed data {
matrix[N, nbetas] Q_ast;
matrix[nbetas, nbetas] R_ast;
matrix[nbetas, nbetas] R_ast_inverse;
// thin and scale the QR decomposition
Q_ast = qr_thin_Q(X) * sqrt(N - 1);
R_ast = qr_thin_R(X) / sqrt(N - 1);
R_ast_inverse = inverse(R_ast);
}


parameters {
  vector[nbetas] theta; // vector of real theta
  real<lower=0> sigma; // STD
}
model {
  y ~ normal( Q_ast * theta, sigma ); // mvn mean Xbeta
  theta ~ normal(0,100);// mvn
  sigma ~ gamma(1,1); // univariate
}

generated quantities {
```

```
  vector[nbetas] beta;
  beta = R_ast_inverse * theta; // coefficients on x
}
```

```
options(mc.cores = parallel::detectCores())

fit5 <- stan(file = "D:/2021-MATH4753/Bayesian/SLR2-Q.stan",
             model_name = "Simple Linear Regression matrix Q",
             data = dataslr5,
             chains = 3,
             warmup = 1000,
             iter = 3000,
             pars = c("beta", "sigma")
)

library(shinystan)

afit5 <- as.shinystan(fit5)
shinystan::launch_shinystan(afit5)
```

## Questions

- Make the above code run and make a screen shot of the posterior histograms of the Diagnostic page, NUTS(Plots) Tab, Parameter: beta[3] all else default. Include the picture in your write up.
- Make a screen shot of the Explore page, beta[1], Multiview. Add the picture to your write up
- Now change the STAN code so that you do not transform the data using the QR transformation. That is the model will use $\beta$ directly. Make copies of the above two pages for the new model.