Lasso

John Tipton

May 4, 2015

Put model statement here

then we define the model parameters

```
N <- 1000
n <- 100
beta <- -3:3
s2_epsilon <- 0.25
tau <- length(beta)
```

Given these model parameters, we simulate some data where there is no multicollinearity.

```
make.lm.data <- function(N, n, beta, sigma.sqaured_epsilon){
  tau <- length(beta)
  X <- matrix(rnorm(N * tau), nrow = N, ncol = tau)
  Y <- X %*% beta + rnorm(N, 0, s2_epsilon)
  data.frame(Y, X)
}

data <- make.lm.data(N, n, beta, s2_epsilon)
```

To examine this model further, we subsample from the truth and attempt to estimate model parameters.

```
H <- sample(1:N, n)
data.samp <- data[H, ]
```

For comparison, we examine a simple linear regression model.

```
summary(mod <- lm(Y ~ ., data = data.samp))
```

```
##
## Call:
## lm(formula = Y ~ ., data = data.samp)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -0.54907 -0.15561 -0.00661  0.16529  0.72432
##
## Coefficients:
##             Estimate Std. Error  t value Pr(>|t|)
## (Intercept)  0.05612    0.02553    2.198   0.0304 *
## X1          -3.00026    0.02699 -111.165   <2e-16 ***
## X2          -1.99575    0.02344  -85.139   <2e-16 ***
## X3          -0.97352    0.02803  -34.731   <2e-16 ***
```

```
## X4          -0.01235    0.02458   -0.502   0.6166
## X5           1.01652    0.02525   40.263   <2e-16 ***
## X6           1.98679    0.02160   92.001   <2e-16 ***
## X7           3.04999    0.02801  108.896   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.2427 on 92 degrees of freedom
## Multiple R-squared:  0.9978, Adjusted R-squared:  0.9976
## F-statistic:  5889 on 7 and 92 DF,  p-value: < 2.2e-16
```

Since we are using a Bayesian approach, we specify our prior parameters as

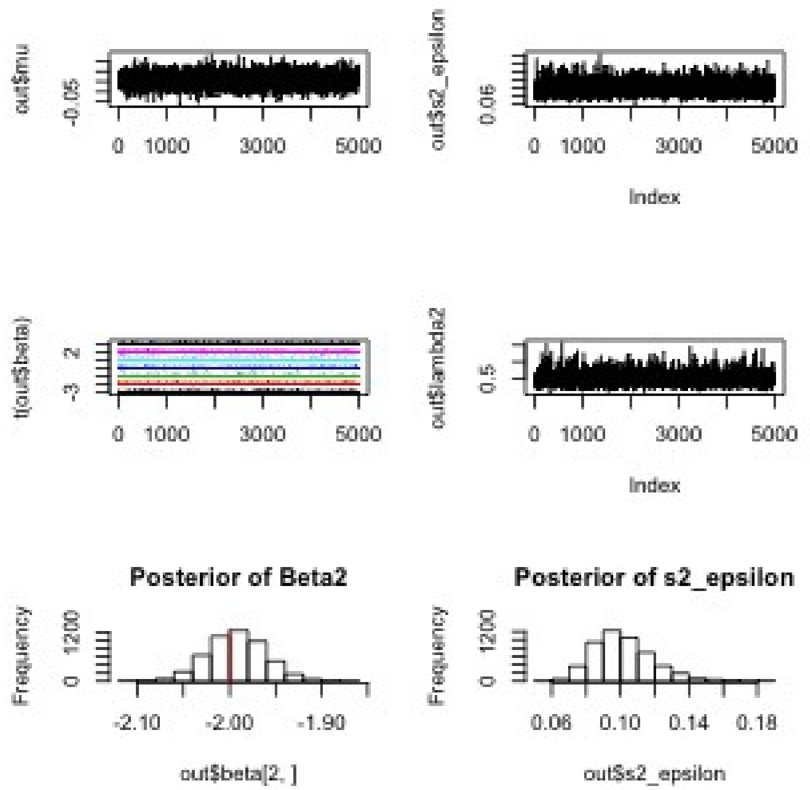
```
alpha_epsilon <- 1
beta_epsilon <- 1
alpha_lambda <- 10
beta_lambda <- 1

n_mcmc <- 10000

Y <- data.samp[, 1]
X <- as.matrix(data[, 2:(tau + 1)], ncol = tau)

out <- mcmc(Y, X, H, n_mcmc, alpha_epsilon, beta_epsilon, alpha_lambda, beta_lambda)
```

```
##
## Attaching package: 'myFunctions'
##
## The following objects are masked from 'package:base':
##
##     colSums, rowMeans, rowSums
```

```
##   100  200  300  400  500  600  700  800  900  1000  1100  1200  1300  1400  1500  1600  1700  1800
```

Examine model output

```
make_model_plot(out)
```



Examine estimates $\hat{\beta}$

```
library(pander)
results=data.frame(rbind(c(0, beta), c(mean(out$mu), rowMeans(out$beta))), row.names=c("Truth", "Estima
names(results)=c("mu", "Beta1", "Beta2", "Beta3", "Beta4", "Beta5", "Beta6", "Beta7")
pandoc.table(results, style="rmarkdown")
```

```
##
##
## |            |   mu    |  Beta1  |  Beta2  |  Beta3  |  Beta4  |
## |:---------------:|:-------:|:-------:|:-------:|:-------:|:-------:|
## |    **Truth**    |    0    |   -3    |   -2    |   -1    |    0    |
## |  **Estimate**   | 0.05699 | -2.998  | -1.993  | -0.9691 | -0.0121 |
##
## Table: Table continues below
##
##
##
## |            |  Beta5  |  Beta6  |  Beta7  |
## |:---------------:|:-------:|:-------:|:-------:|
## |    **Truth**    |    1    |    2    |    3    |
## |  **Estimate**   |  1.013  |  1.984  |  3.047  |
```

Examine MSPE

```
## linear model
preds <- mod$coefficients[1] + out$X %*% mod$coefficients[2:8]
mean((data$Y - preds)^2)
```

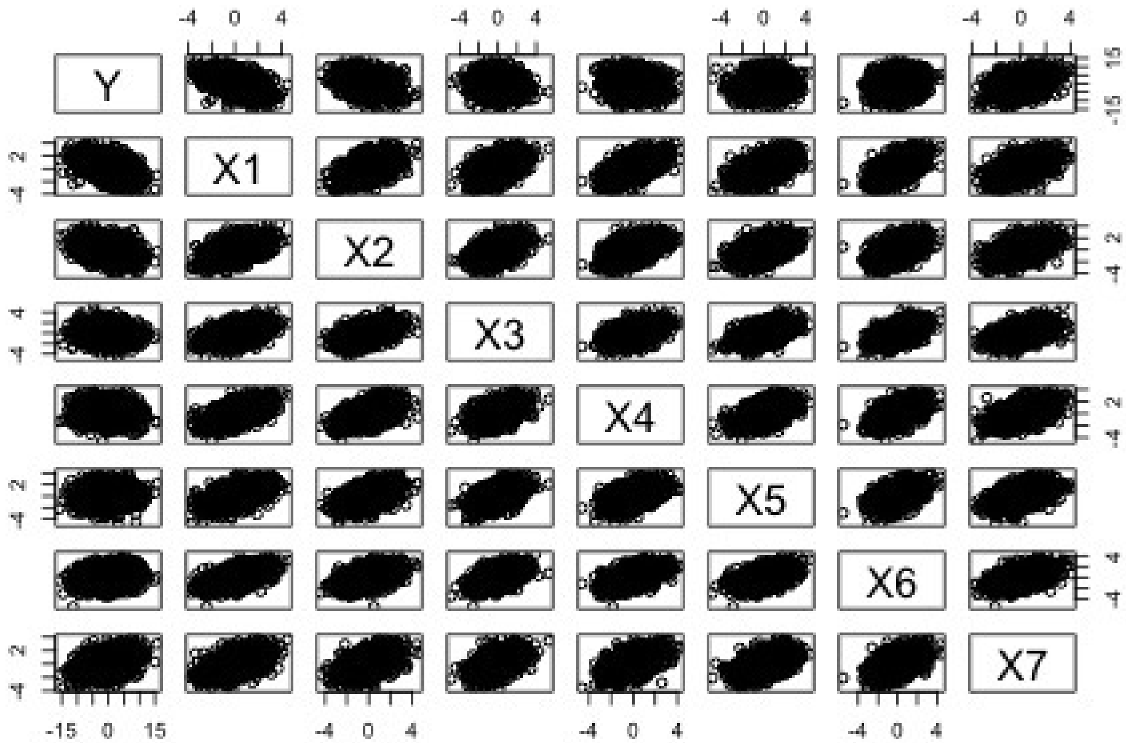
```
## [1] 0.06678272
```

```
## mcmc model
preds_mcmc=rowMeans(out$mu + as.matrix(data[, -1]) %*% out$beta)
mean((data$Y - preds_mcmc)^2)
```

```
## [1] 0.06750132
```

Next a model with multicollinearity

```
##
## Simulate some data
##

library(myFunctions)
make.lm.data <- function(N, n, beta, sigma.sqaured_epsilon){
  tau <- length(beta)
  D = as.matrix(dist(1:N))
  X <- matrix(t(mvrnormArma(tau, rnorm(N), 0.75 * exp(- D * 0.5) + 0.25 * diag(N))), nrow = N, ncol = ta
  Y <- X %*% beta + rnorm(N, 0, s2_epsilon)
  data.frame(Y, X)
}

data <- make.lm.data(N, n, beta, s2_epsilon)
pairs(data)
```



Subsample the data

```
H <- sample(1:N, n)
data.samp <- data[H, ]
```

Examine a linear regression model

```
summary(mod2 <- lm(Y ~ ., data = data.samp))
```

```
##
## Call:
## lm(formula = Y ~ ., data = data.samp)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -0.54788 -0.16343  0.01062  0.13610  0.57142
##
## Coefficients:
##             Estimate Std. Error  t value Pr(>|t|)
## (Intercept)  0.01772    0.02412    0.734    0.465
## X1          -2.98275    0.02283 -130.666   <2e-16 ***
## X2          -1.99034    0.01917 -103.824   <2e-16 ***
## X3          -1.02779    0.02457  -41.838   <2e-16 ***
## X4          -0.01071    0.02313   -0.463    0.645
## X5           1.02228    0.02147   47.618   <2e-16 ***
```

```
## X6           1.98628    0.02297   86.481   <2e-16 ***
## X7           2.98854    0.02219  134.678   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.2321 on 92 degrees of freedom
## Multiple R-squared:  0.9979, Adjusted R-squared:  0.9978
## F-statistic:  6379 on 7 and 92 DF,  p-value: < 2.2e-16
```

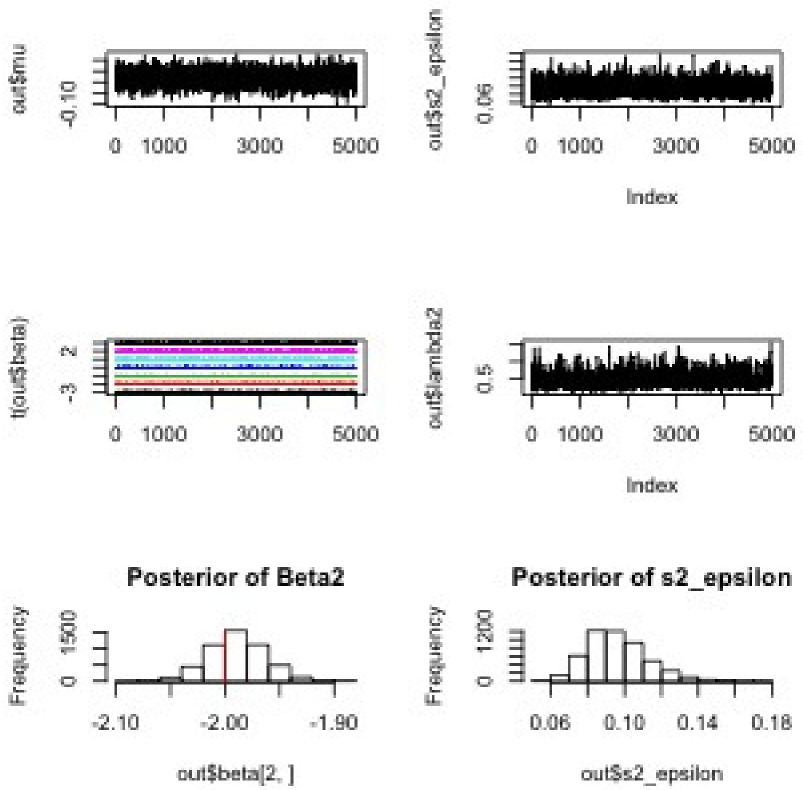
Specify priors for a Bayesian model

```
##
## Setup priors
##
# hyperparameters for mu.beta and s2.beta
alpha_epsilon <- 1
beta_epsilon <- 1
alpha_lambda <- 10
beta_lambda <- 1

n_mcmc <- 10000

##
## Fit mcmc
##

Y <- data.samp[, 1]
X <- as.matrix(data[, 2:(tau + 1)], ncol = tau)

out <- mcmc(Y, X, H, n_mcmc, alpha_epsilon, beta_epsilon, alpha_lambda, beta_lambda)
```

```
##   100  200  300  400  500  600  700  800  900  1000  1100  1200  1300  1400  1500  1600  1700  1800
```

Examine model output

```
make_model_plot(out)
```



Examine estimates $\hat{\beta}$

```
library(pander)
results=data.frame(rbind(c(0, beta), c(mean(out$mu), rowMeans(out$beta))), row.names=c("Truth", "Estima
names(results)=c("mu", "Beta1", "Beta2", "Beta3", "Beta4", "Beta5", "Beta6", "Beta7")
pandoc.table(results, style="rmarkdown")
```

```
##
##
## |            |   mu    |  Beta1  |  Beta2  |  Beta3  |  Beta4   |
## |:---------------:|:-------:|:-------:|:-------:|:-------:|:--------:|
## |   **Truth**     |    0    |   -3    |   -2    |   -1    |    0     |
## |  **Estimate**   | 0.01789 | -2.979  | -1.989  | -1.024  | -0.01099 |
##
## Table: Table continues below
##
##
##
## |            |  Beta5  |  Beta6  |  Beta7  |
## |:---------------:|:-------:|:-------:|:-------:|
## |   **Truth**     |    1    |    2    |    3    |
## |  **Estimate**   |  1.019  |  1.985  |  2.986  |
```

Examine MSPE

```
## linear model
preds <- mod2$coefficients[1] +  as.matrix(data[, - 1]) %*% mod2$coefficients[2:8]
mean((data$Y - preds)^2)
```

```
## [1] 0.06644458
```

```
## mcmc model
preds_mcmc=rowMeans(out$mu + as.matrix(data[, -1]) %*% out$beta)
mean((data$Y - preds_mcmc)^2)
```

```
## [1] 0.06692067
```

Now let's examine a principle components model

Examine a linear regression model

```
summary(mod3 <- lm(Y ~ makePCA(as.matrix(data[, -1]))$X_pca[H, ], data = data.samp))
```

```
##
## Call:
## lm(formula = Y ~ makePCA(as.matrix(data[, -1]))$X_pca[H, ], data = data.samp)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -0.54788 -0.16343  0.01062  0.13610  0.57142
##
## Coefficients:
##                                             Estimate Std. Error t value
## (Intercept)                                -0.216084   0.023598  -9.157
## makePCA(as.matrix(data[, -1]))$X_pca[H, ]1 -0.126189   0.007976 -15.821
## makePCA(as.matrix(data[, -1]))$X_pca[H, ]2 -0.551926   0.021818 -25.297
## makePCA(as.matrix(data[, -1]))$X_pca[H, ]3  1.309408   0.023739  55.158
## makePCA(as.matrix(data[, -1]))$X_pca[H, ]4 -1.666675   0.022575 -73.828
## makePCA(as.matrix(data[, -1]))$X_pca[H, ]5  2.836758   0.022639 125.303
## makePCA(as.matrix(data[, -1]))$X_pca[H, ]6  1.749824   0.024948  70.138
## makePCA(as.matrix(data[, -1]))$X_pca[H, ]7  3.451731   0.027568 125.208
##                                            Pr(>|t|)
## (Intercept)                                1.36e-14 ***
## makePCA(as.matrix(data[, -1]))$X_pca[H, ]1  < 2e-16 ***
## makePCA(as.matrix(data[, -1]))$X_pca[H, ]2  < 2e-16 ***
## makePCA(as.matrix(data[, -1]))$X_pca[H, ]3  < 2e-16 ***
## makePCA(as.matrix(data[, -1]))$X_pca[H, ]4  < 2e-16 ***
## makePCA(as.matrix(data[, -1]))$X_pca[H, ]5  < 2e-16 ***
## makePCA(as.matrix(data[, -1]))$X_pca[H, ]6  < 2e-16 ***
## makePCA(as.matrix(data[, -1]))$X_pca[H, ]7  < 2e-16 ***
## ---
```

```
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.2321 on 92 degrees of freedom
## Multiple R-squared:  0.9979, Adjusted R-squared:  0.9978
## F-statistic:  6379 on 7 and 92 DF,  p-value: < 2.2e-16
```

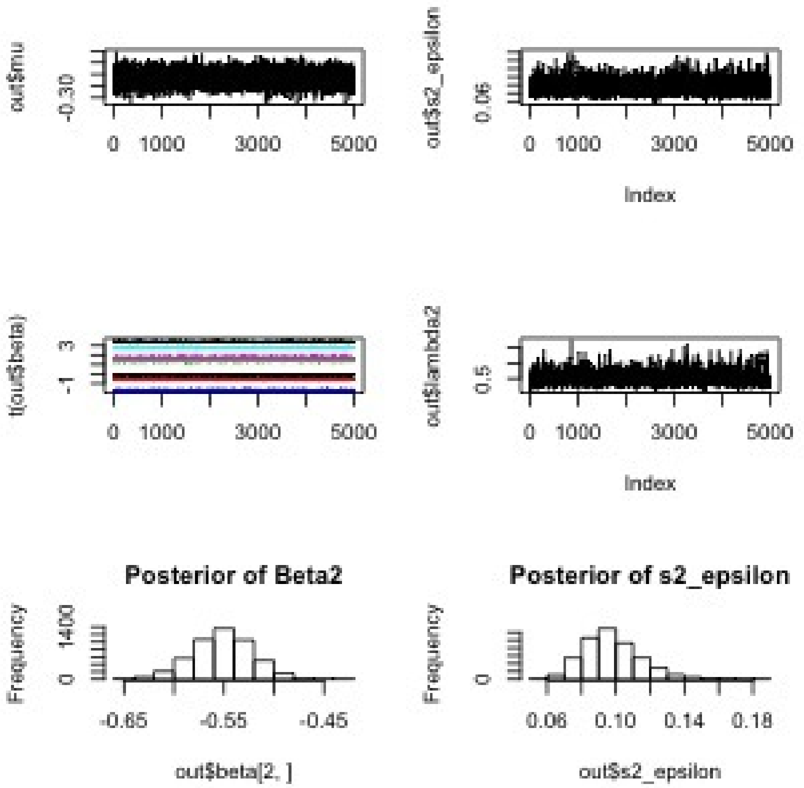
Specify priors for a Bayesian model

```r
##
## Setup priors
##
# hyperparameters for mu.beta and s2.beta
alpha_epsilon <- 1
beta_epsilon <- 1
alpha_lambda <- 10
beta_lambda <- 1

n_mcmc <- 10000

##
## Fit mcmc
##

Y <- data.samp[, 1]
X <- as.matrix(data[, 2:(tau + 1)], ncol = tau)

out <- mcmc(Y, X, H, n_mcmc, alpha_epsilon, beta_epsilon, alpha_lambda, beta_lambda, pca = TRUE)
```

```
##   100  200  300  400  500  600  700  800  900  1000  1100  1200  1300  1400  1500  1600  1700  1800
```

Examine model output

```r
make_model_plot(out)
```



Examine estimates $\hat{\beta}$

```
library(pander)
results=data.frame(rbind(c(0, beta), c(mean(out$mu), rowMeans(out$beta))), row.names=c("Truth", "Estima
names(results)=c("mu", "Beta1", "Beta2", "Beta3", "Beta4", "Beta5", "Beta6", "Beta7")
pandoc.table(results, style="rmarkdown")
```

```
##
##
## |            |   mu   |  Beta1  |  Beta2  |  Beta3  |  Beta4  |
## |:---------------:|:------:|:-------:|:-------:|:-------:|:-------:|
## |   **Truth**     |   0    |   -3    |   -2    |   -1    |    0    |
## |  **Estimate**   | -0.216 | -0.1262 | -0.5515 |  1.307  | -1.665  |
##
## Table: Table continues below
##
##
##
## |            |  Beta5  |  Beta6  |  Beta7  |
## |:---------------:|:-------:|:-------:|:-------:|
## |   **Truth**     |    1    |    2    |    3    |
## |  **Estimate**   |  2.835  |  1.747  |  3.448  |
```

Examine MSPE

```
## linear model
preds <- mod3$coefficients[1] + out$X %*% mod3$coefficients[2:8]
mean((data$Y - preds)^2)
```

```
## [1] 0.06644458
```

```
## mcmc model
preds_mcmc=rowMeans(out$mu + out$X %*% out$beta)
mean((data$Y - preds_mcmc)^2)
```

```
## [1] 0.06644551
```