Applied Bayesian Analysis : NCSU ST 540

Homework 7

Bruce Campbell

In this assignment we perform Bayesian linear regression for the microbiome data on the course website

`https://www4.stat.ncsu.edu/~reich/ABA/assignments/homes.RData`

Let Y_i be the precipitation for observation i and X_{ij} equal one if OTU j is present in sample i.

First, extract the 50 OTU with the largest absolute correlation between X_{ij} and Y_i. Then fit a Bayesian linear regression model precipitation as the response and with these 50 covariates (and an intercept term) using two priors:

(1) Uninformative normal priors: $\beta_j \sim Normal(0, 100^2)$

(2) Hierarchical normal priors: $\beta_j|\tau \sim Normal(0, \tau^2)$ where $\tau^2 \sim InvGamma(0:01, 0:01)$

(3) Bayesian LASSO: $\beta_j|\tau^2 \sim DE(0, \tau^2)$ where $\tau^2 \sim InvGamma(0:01, 0:01)$

Compare convergence and the posterior distribution of the regression coeffcients under these three priors. In particular, are the same OTU's significant in all three fits?

Load data and select 50 most ocrrelated OUT variables.

```r
library(rjags)
library(coda)
library(modeest)
load("homes.RData")

X <- OTU != 0
Y <- homes$MeanAnnualPrecipitation

C_xy <- cor(X, Y)

top <- function(x, n) {
    tail(order(x), n)
}
# One of the X is all 1's -
# resulting in an NA for the
# correlation.
indices <- top(C_xy, 51)
# Remove the NA - I'm sure there's
# a more elegant way...
indices <- indices[1:50]
```

```
X <- X[, indices]

predictor.names <- names(OTU)[indices]
predictor.names[51] <- "intercept"

top.corr <- C_xy[indices]

# Y <- scale(Y) X <- scale(X)

DEBUG <- FALSE
if (DEBUG) {
    nSamples <- 100
    n.chains <- 1
} else {
    nSamples <- 100
    n.chains <- 1
}
```

We sample from our model after burn in. Not all of the diagnostic plots are not presented. See the diagnostic plots in `https://github.com/brucebcampbell/bayesian-learning-with-R.git` we assesed convergence by; - viewing the time sereies for the intercept and each of the predictors. For this we utilized the coda package. - ran multiple chains and viewed evaluated the autocorrelation plots. - calculated the posterior means for the intercept and the $beta_j$ - utilized the mlv funtions in the modeest to calculate the MAP estimated of the posterior modes - compared the 95% prediction intervals for the intercepts against the p-values from the logistic regression maximum likelihood model - Gelman plots are optionally produced whem the nuymber of MCMC chains is greater than one.

Some of the code is run conditionally through the DEBUG flag. We ran under debug mode and noted that all models convereged. All but one of the predictors were the same for all the modesl. Depending on the run OTU_624 was swapped with another pridictor in the uninformative model.

This was an interesting project. I iterated several versions and had to debug working jags models to get things running well. Also we encountrered a NaN in the correlation due to one of the predictors being all 1's. If this was not accounted for the run times went up and convergence was bad. The width of the credible intervals could be investigated. We'll be running a longer simulation as a follow on task for fun. We set the precision of the model error to be 0.01 and 0.1 and got similar results.

Normal Uniformative

It's not specified what the prior variance is for $E[Y_j|X_j]$. We wull assume $Y|\beta \sim N(y \cdot \beta, \sigma^2)$ where $\sigma^2 \sim InvGamma(0.1, 0.1)$

```
n <- nrow(X)

sigma.beta  <- 100
inv.gamma.param  <- 0.1
p <- ncol(X)
```

```
model_string.normal_uniformative <- "model{
  # Likelihood
  for(i in 1:n){
    Y[i]   ~ dnorm(mu[i],inv.var)
    mu[i] <- intercept +inprod(X[i,],beta[])
  }

  # Prior for beta
  for(j in 1:p){
    beta[j] ~ dnorm(0,1/sigma.beta^2)
  }
  intercept ~ dnorm(0,1/sigma.beta^2)

  # Prior for the inverse variance
  inv.var   ~ dgamma(inv.gamma.param, inv.gamma.param)
  sigma     <- 1/sqrt(inv.var)
}"

model.normal_uniformative <- jags.model(textConnection(model_string.normal_uniformative), data
```

```
## Compiling model graph
##    Resolving undeclared variables
##    Allocating nodes
## Graph information:
##    Observed stochastic nodes: 1133
##    Unobserved stochastic nodes: 52
##    Total graph size: 61100
##
## Initializing model
```

```
update(model.normal_uniformative, nSamples, progress.bar="none"); # Burnin
samp.coeff.normal_uniformative <- coda.samples(model.normal_uniformative, variable.names=c("in

sum.normal_uniformative <-  summary(samp.coeff.normal_uniformative)
quantiles<-sum.normal_uniformative$quantiles
left.05.quantile.sign  <- sign(quantiles[,1])==-1
right.95.quantile.sign <- sign(quantiles[,5])==1
significant <- xor(left.05.quantile.sign ,right.95.quantile.sign)
beta.significant <- quantiles[significant,]


pander(data.frame(beta.significant), caption = "significant normal uninformative ")
```

Table 1: significant normal uninformative

	X2.5.	X25.	X50.	X75.	X97.5.
beta[7]	-7.827	-5.457	-4.224	-3.005	-0.6621

	X2.5.	X25.	X50.	X75.	X97.5.
beta[20]	-8.088	-5.673	-4.357	-3.024	-0.6061
beta[25]	0.1149	2.069	3.094	4.126	6.123
beta[29]	1.499	3.934	5.168	6.43	8.858
beta[37]	1.112	3.114	4.146	5.174	7.108
beta[38]	0.2462	2.753	4.052	5.345	7.773
beta[41]	1.753	4.26	5.582	6.873	9.375
beta[45]	1.86	4.175	5.394	6.63	8.951
beta[46]	2.866	5.031	6.182	7.346	9.545
beta[49]	1.848	3.855	4.933	5.981	8.05
beta[50]	3.523	5.752	6.923	8.105	10.33
intercept	51.54	53.23	54.13	55.02	56.69

```
credible.widths <- beta.significant[,5]-beta.significant[,1]

predictor.names.significant <- predictor.names[significant]

pander(data.frame(predictor.names.significant,credible.widths), caption = "credible widths norm
```

Table 2: credible widths normal uninformative

	predictor.names.significant	credible.widths
beta[7]	OTU_54646	7.165
beta[20]	OTU_9405	7.482
beta[25]	OTU_624	6.008
beta[29]	OTU_999	7.36
beta[37]	OTU_66	5.996
beta[38]	OTU_51578	7.527
beta[41]	OTU_8086	7.622
beta[45]	OTU_72918	7.092
beta[46]	OTU_97	6.679
beta[49]	OTU_277	6.201
beta[50]	OTU_18758	6.811
intercept	intercept	5.155

```
if (DEBUG)
  {
  autocorr.plot(samp.coeff.normal_uniformative)

  plot(samp.coeff.normal_uniformative)

  #Sample again and estimate posterior means and MAP posterior modes.
  samp.coeff.normal_uniformative.jags <- jags.samples(model.normal_uniformative, variable.names
  posterior_means.normal_uniformative <- lapply(samp.coeff.normal_uniformative.jags, apply, 1,
  pander(posterior_means.normal_uniformative, caption = "posterior means second sample")
  posterior_modes.normal_uniformative <- lapply(samp.coeff.normal_uniformative.jags, apply, 1,
  posterior_modes.normal_uniformative

  if(n.chains>1)
  {
  gelman.plot(samp.coeff)
  }
}
```

Hierarchical Normal Priors

$\beta_j|\tau \sim Normal(0, \tau^2)$ where $\tau^2 \sim InvGamma(0:01, 0:01)$

```
beta.inv.gamma.param  <- 0.01
variance.inv.gamma.param  <- 0.1
p <- ncol(X)

model_string.normal_hierarchical <- "model{
  # Likelihood
  for(i in 1:n){
    Y[i]   ~ dnorm(mu[i],inv.var)
    mu[i] <- intercept +inprod(X[i,],beta[])
  }

  # Prior for beta
  for(j in 1:p){
    beta[j] ~ dnorm(0,beta.inv.gamma.param)
  }
  intercept ~ dnorm(0,beta.inv.gamma.param)

  # Prior for the inverse variance
  inv.var   ~ dgamma(variance.inv.gamma.param, variance.inv.gamma.param)
  sigma     <- 1/sqrt(inv.var)

  #Beta Prior for the inverse variance
  inv.var.beta   ~ dgamma(beta.inv.gamma.param, beta.inv.gamma.param)
}"

model.normal_hierarchical <- jags.model(textConnection(model_string.normal_hierarchical), data
```

```
## Compiling model graph
##    Resolving undeclared variables
##    Allocating nodes
## Graph information:
##    Observed stochastic nodes: 1133
##    Unobserved stochastic nodes: 53
##    Total graph size: 61098
```

```
##
## Initializing model
```

```
update(model.normal_hierarchical, nSamples, progress.bar="none"); # Burnin

samp.coeff.normal_hierarchical <-coda.samples(model.normal_hierarchical,variable.names=c("inte

sum.normal_hierarchical <-  summary(samp.coeff.normal_hierarchical)
quantiles<-sum.normal_hierarchical$quantiles
left.05.quantile.sign  <- sign(quantiles[,1])==-1
right.95.quantile.sign <- sign(quantiles[,5])==1
significant <- xor(left.05.quantile.sign ,right.95.quantile.sign)
beta.significant <- quantiles[significant,]

pander(data.frame(beta.significant), caption = "significant normal hierarchical ")
```

Table 3: significant normal hierarchical

	X2.5.	X25.	X50.	X75.	X97.5.
beta[7]	-7.61	-5.329	-4.149	-2.971	-0.6566
beta[20]	-7.457	-5.11	-3.872	-2.649	-0.3519
beta[25]	0.04946	1.967	2.994	4.037	5.979
beta[29]	1.425	3.765	4.974	6.232	8.588
beta[37]	1.273	3.195	4.174	5.171	7.08
beta[38]	0.2197	2.598	3.854	5.106	7.561
beta[41]	1.704	4.207	5.477	6.782	9.215
beta[45]	1.883	4.207	5.401	6.601	8.842
beta[46]	2.955	5.128	6.242	7.388	9.49
beta[49]	1.723	3.72	4.769	5.834	7.889
beta[50]	3.331	5.541	6.686	7.858	10.08
intercept	50.8	52.41	53.29	54.18	55.81

```
credible.widths <- beta.significant[,5]-beta.significant[,1]

predictor.names.significant <- predictor.names[significant]

pander(data.frame(predictor.names.significant,credible.widths), caption = "credible widths norm
```

Table 4: credible widths normal hierarchical

	predictor.names.significant	credible.widths
beta[7]	OTU_54646	6.953
beta[20]	OTU_9405	7.105
beta[25]	OTU_624	5.929
beta[29]	OTU_999	7.163
beta[37]	OTU_66	5.807
beta[38]	OTU_51578	7.341

	predictor.names.significant	credible.widths
beta[41]	OTU_8086	7.511
beta[45]	OTU_72918	6.959
beta[46]	OTU_97	6.535
beta[49]	OTU_277	6.167
beta[50]	OTU_18758	6.75
intercept	intercept	5.011

```
if (DEBUG)
{
  autocorr.plot(samp.coeff.normal_hierarchical)

  plot(samp.coeff.normal_hierarchical)

  #Sample again and estimate posterior means and MAP posterior modes.
  samp.coeff.normal_hierarchical.jags <- jags.samples(model.normal_hierarchical, variable.names
  posterior_means.normal_hierarchical <- lapply(samp.coeff.normal_hierarchical.jags, apply, 1,
  pander(posterior_means.normal_hierarchical, caption = "posterior means second sample")
  posterior_modes.normal_hierarchical <- lapply(samp.coeff.normal_hierarchical.jags, apply, 1,
  posterior_modes.normal_hierarchical
  if(n.chains>1)
  {
  gelman.plot(samp.coeff)
  }
}
```

BLASSO

```
beta.inv.gamma.param  <- 0.01
variance.inv.gamma.param  <- 0.1
p <- ncol(X)

model_string.normal_blasso <- "model{
  # Likelihood
  for(i in 1:n){
    Y[i]   ~ dnorm(mu[i],inv.var)
    mu[i] <- intercept +inprod(X[i,],beta[])
  }

  # Prior for beta
  for(j in 1:p){
    beta[j] ~ ddexp(0,beta.inv.gamma.param)
  }
  intercept ~ ddexp(0,beta.inv.gamma.param)
  # Prior for the inverse variance
  inv.var   ~ dgamma(variance.inv.gamma.param, variance.inv.gamma.param)
  sigma     <- 1/sqrt(inv.var)

  #Beta Prior for the inverse variance
  inv.var.beta   ~ dgamma(beta.inv.gamma.param, beta.inv.gamma.param)
}"

model.normal_blasso <- jags.model(textConnection(model_string.normal_blasso), data = list(Y=Y,
```

```
## Compiling model graph
##    Resolving undeclared variables
##    Allocating nodes
## Graph information:
##    Observed stochastic nodes: 1133
##    Unobserved stochastic nodes: 53
##    Total graph size: 61098
##
## Initializing model
```

```
update(model.normal_blasso, nSamples, progress.bar="none"); # Burnin

samp.coeff.normal_blasso <- coda.samples(model.normal_blasso,variable.names=c("intercept","beta

sum.normal_blasso <-  summary(samp.coeff.normal_blasso)
quantiles<-sum.normal_blasso$quantiles
left.05.quantile.sign  <- sign(quantiles[,1])==-1
right.95.quantile.sign <- sign(quantiles[,5])==1
significant <- xor(left.05.quantile.sign ,right.95.quantile.sign)
beta.significant <- quantiles[significant,]

pander(data.frame(beta.significant), caption = "significant normal BLASSO ")
```

Table 5: significant normal BLASSO

	X2.5.	X25.	X50.	X75.	X97.5.
beta[3]	-6.674	-4.211	-3.01	-1.924	-0.1404
beta[7]	-7.822	-5.008	-4.062	-2.941	-0.889
beta[14]	0.03215	1.478	2.234	3.076	4.432
beta[20]	-7.226	-5.517	-4.526	-3.501	-1.09
beta[29]	1.267	2.816	3.943	5.993	9.438
beta[37]	1.697	3.26	4.181	5.08	6.759
beta[38]	0.606	3.031	4.266	5.291	7.576
beta[41]	3.209	4.949	6.462	7.557	9.844
beta[45]	1.957	3.834	4.832	5.891	7.838
beta[46]	2.805	4.244	5.507	6.542	8.556
beta[49]	2.423	4.116	5.142	6.158	8.033
beta[50]	3.699	5.236	6.424	7.735	9.765

	X2.5.	X25.	X50.	X75.	X97.5.
intercept	52.12	53.48	54.25	55.33	58.21

```
credible.widths <- beta.significant[,5]-beta.significant[,1]

predictor.names.significant <- predictor.names[significant]

pander(data.frame(predictor.names.significant,credible.widths), caption = "credible widths norm
```

Table 6: credible widths normal BLASSO

	predictor.names.significant	credible.widths
beta[3]	OTU_64123	6.533
beta[7]	OTU_54646	6.933
beta[14]	OTU_80	4.399
beta[20]	OTU_9405	6.137
beta[29]	OTU_999	8.17
beta[37]	OTU_66	5.062
beta[38]	OTU_51578	6.97
beta[41]	OTU_8086	6.635
beta[45]	OTU_72918	5.881
beta[46]	OTU_97	5.751
beta[49]	OTU_277	5.609
beta[50]	OTU_18758	6.066
intercept	intercept	6.089

```
if (DEBUG)
{
  autocorr.plot(samp.coeff.normal_blasso)

  plot(samp.coeff.normal_blasso)

  #Sample again and estimate posterior means and MAP posterior modes.
  samp.coeff.normal_blasso.jags <- jags.samples(model.normal_blasso, variable.names = c("interc
  posterior_means.normal_blasso <- lapply(samp.coeff.normal_blasso.jags, apply, 1, "mean")
  pander(posterior_means.normal_blasso, caption = "posterior means second sample")
  posterior_modes.normal_blasso <- lapply(samp.coeff.normal_blasso.jags, apply, 1, "mlv")
  posterior_modes.normal_blasso
  if(n.chains>1)
  {
  gelman.plot(samp.coeff)
  }
}
```