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.

```
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]
```

```r
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 <- 20000
    n.chains <- 1
} else {
    nSamples <- 20000
    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 $\sigma^2 \sim InvGamma(0.1, 0.1)$

```r
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]	-7.739	-5.44	-4.222	-2.997	-0.6958

	X2.5.	X25.	X50.	X75.	X97.5.
beta[20]	-7.958	-5.59	-4.345	-3.099	-0.772
beta[25]	0.13	2.076	3.09	4.127	6.063
beta[29]	1.564	3.955	5.224	6.485	8.891
beta[37]	1.19	3.136	4.132	5.141	7.062
beta[38]	0.2196	2.694	3.993	5.269	7.673
beta[41]	1.807	4.29	5.582	6.848	9.3
beta[45]	1.808	4.137	5.381	6.635	8.986
beta[46]	2.818	5.028	6.171	7.315	9.509
beta[49]	1.837	3.881	4.927	6.004	8.04
beta[50]	3.476	5.733	6.886	8.045	10.38
intercept	51.55	53.28	54.15	55.03	56.68

```
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 norn
```

Table 2: credible widths normal uninformative

	predictor.names.significant	credible.widths
beta[7]	OTU_54646	7.043
beta[20]	OTU_9405	7.186
beta[25]	OTU_624	5.933
beta[29]	OTU_999	7.326
beta[37]	OTU_66	5.872
beta[38]	OTU_51578	7.453
beta[41]	OTU_8086	7.493
beta[45]	OTU_72918	7.179
beta[46]	OTU_97	6.691
beta[49]	OTU_277	6.203
beta[50]	OTU_18758	6.906
intercept	intercept	5.123

```
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]	-7.596	-5.331	-4.136	-2.927	-0.6075
beta[20]	-7.475	-5.115	-3.891	-2.634	-0.1955
beta[25]	0.04628	2.007	3.017	4.042	5.99
beta[29]	1.383	3.783	5.038	6.291	8.641
beta[37]	1.269	3.163	4.16	5.152	7.05
beta[38]	0.1728	2.547	3.824	5.123	7.563
beta[41]	1.866	4.238	5.481	6.751	9.218
beta[45]	1.821	4.134	5.337	6.556	8.874
beta[46]	2.879	5.077	6.217	7.34	9.52
beta[49]	1.724	3.728	4.782	5.824	7.866
beta[50]	3.358	5.578	6.742	7.893	10.05
intercept	50.81	52.46	53.33	54.21	55.82

```
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	6.989
beta[20]	OTU_9405	7.28
beta[25]	OTU_624	5.944
beta[29]	OTU_999	7.259
beta[37]	OTU_66	5.781
beta[38]	OTU_51578	7.39

	predictor.names.significant	credible.widths
beta[41]	OTU_8086	7.352
beta[45]	OTU_72918	7.053
beta[46]	OTU_97	6.641
beta[49]	OTU_277	6.142
beta[50]	OTU_18758	6.696
intercept	intercept	5.009

```
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,X
```

```
## 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[7]	-7.679	-5.347	-4.12	-2.887	-0.6366
beta[20]	-7.997	-5.617	-4.344	-3.062	-0.6887
beta[25]	0.09942	2.068	3.086	4.111	6.067
beta[29]	1.491	3.855	5.13	6.409	8.788
beta[37]	1.16	3.093	4.092	5.12	7.096
beta[38]	0.174	2.708	4.018	5.33	7.873
beta[41]	1.798	4.261	5.554	6.848	9.343
beta[45]	1.712	4.155	5.381	6.596	8.981
beta[46]	2.843	5.007	6.162	7.315	9.516
beta[49]	1.823	3.876	4.937	5.991	8.019
beta[50]	3.49	5.701	6.866	8.042	10.26
intercept	51.54	53.27	54.15	55.06	56.71

	X2.5.	X25.	X50.	X75.	X97.5.

```
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 norr
```

Table 6: credible widths normal BLASSO

	predictor.names.significant	credible.widths
beta[7]	OTU_54646	7.042
beta[20]	OTU_9405	7.308
beta[25]	OTU_624	5.968
beta[29]	OTU_999	7.297
beta[37]	OTU_66	5.935
beta[38]	OTU_51578	7.699
beta[41]	OTU_8086	7.545
beta[45]	OTU_72918	7.269
beta[46]	OTU_97	6.674
beta[49]	OTU_277	6.196
beta[50]	OTU_18758	6.766
intercept	intercept	5.166

```
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)
  }
}
```