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]
```

```
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 <- 1000
    n.chains <- 1
} else {
    nSamples <- 1000
    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 note that all of the models converged. There was some minor disagreement on the predictors among them. The normal uninformative had predictor

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  <- 0.01
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]	-7.988	-5.512	-4.194	-2.979	-0.5415
beta[20]	-8.148	-5.744	-4.374	-3.028	-0.5424
beta[25]	0.05089	2.059	3.096	4.207	6.319
beta[29]	1.602	3.978	5.263	6.52	8.644
beta[36]	0.0127	2.376	3.751	5.038	7.425
beta[37]	1.308	3.058	4.116	5.222	7.419
beta[38]	0.4849	2.688	4.043	5.414	8.038

	X2.5.	X25.	X50.	X75.	X97.5.
beta[41]	1.474	4.157	5.624	6.948	9.313
beta[45]	1.577	4.003	5.241	6.498	8.991
beta[46]	2.713	4.962	6.094	7.213	9.418
beta[49]	1.477	3.772	4.896	6.072	8.13
beta[50]	3.308	5.794	7	8.134	10.38
intercept	51.65	53.22	54.17	55.11	56.81

```
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 2: credible widths normal uninformative

	predictor.names.significant	credible.widths
beta[7]	OTU_54646	7.446
beta[20]	OTU_9405	7.605
beta[25]	OTU_624	6.268
beta[29]	OTU_999	7.043
beta[36]	OTU_43955	7.412
beta[37]	OTU_66	6.11
beta[38]	OTU_51578	7.553
beta[41]	OTU_8086	7.839
beta[45]	OTU_72918	7.413
beta[46]	OTU_97	6.705
beta[49]	OTU_277	6.654
beta[50]	OTU_18758	7.075
intercept	intercept	5.159

```
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  <- 0.01
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]	-7.965	-5.384	-4.132	-2.925	-0.679
beta[25]	0.05159	2.003	3	4.02	5.812
beta[29]	1.386	3.764	5.196	6.43	8.688
beta[37]	1.269	3.238	4.331	5.364	7.184
beta[38]	0.01352	2.523	3.837	5.225	7.74
beta[41]	1.32	3.989	5.313	6.622	9.445
beta[45]	1.802	4.134	5.34	6.535	8.702
beta[46]	3.002	5.178	6.183	7.297	9.345
beta[49]	1.727	3.721	4.701	5.814	7.711
beta[50]	3.321	5.563	6.707	7.882	10.29
intercept	50.91	52.46	53.34	54.16	55.75

```
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 nor
```

Table 4: credible widths normal hierarchical

	predictor.names.significant	credible.widths
beta[7]	OTU_54646	7.286
beta[25]	OTU_624	5.76
beta[29]	OTU_999	7.302
beta[37]	OTU_66	5.915
beta[38]	OTU_51578	7.727
beta[41]	OTU_8086	8.125
beta[45]	OTU_72918	6.9
beta[46]	OTU_97	6.343
beta[49]	OTU_277	5.984

	predictor.names.significant	credible.widths
beta[50]	OTU_18758	6.97
intercept	intercept	4.845

```
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  <- 0.01
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[7]	-7.902	-5.477	-4.294	-2.987	-0.7535
beta[20]	-7.628	-5.419	-4.264	-2.977	-1.165
beta[25]	0.24	2.024	3.068	4.128	6.068
beta[29]	1.731	4.086	5.162	6.271	8.55
beta[37]	1.365	3.191	4.328	5.302	7.246
beta[38]	0.7214	2.89	4.149	5.316	7.604
beta[41]	1.935	4.175	5.361	6.569	9
beta[45]	1.389	3.863	5.151	6.447	8.623
beta[46]	3.039	5.288	6.405	7.476	9.379
beta[49]	1.899	3.9	4.896	5.874	8.101
beta[50]	3.741	5.859	6.969	8.112	10.51
intercept	51.78	53.24	54.22	55.1	56.61

```
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[7]	OTU_54646	7.149
beta[20]	OTU_9405	6.462
beta[25]	OTU_624	5.828
beta[29]	OTU_999	6.819
beta[37]	OTU_66	5.882
beta[38]	OTU_51578	6.883
beta[41]	OTU_8086	7.064
beta[45]	OTU_72918	7.235
beta[46]	OTU_97	6.34
beta[49]	OTU_277	6.202
beta[50]	OTU_18758	6.766
intercept	intercept	4.824

```
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)
  }
}
```