STA 601/360 Homework 9

Yifei Wang

15 November, 2019

1. Hoff problem 9.2 Model selection: As described in Example 6 of Chapter 7, The file `azdiabetes.dat` contains data on health-related variables of a population of 532 women. In this exercise we will be modeling the conditional distribution of glucose level (`glu`) as a linear combination of the other variables, excluding the variable `diabetes`.

```
az = read.table("azdiabetes.dat", header = TRUE)[, -8]
```

Part (a) Fit a regression model using the g-prior with $g = n$, $\nu_0 = 2$ and $\sigma_0^2 = 1$. Obtain posterior confidence intervals for all of the parameters.

```
# glu ~ 1 + npreg + bp + skin + bmi + ped + age

### data and priors
n = nrow(az)
intercept = as.matrix(rep(1, n), ncol = 1)
colnames(intercept) = c("intercept")
rownames(intercept) = 1:nrow(az)
X = cbind(intercept, as.matrix(az[, -2]))
XT = t(X)
y = as.matrix(az[, 2], ncol = 1)
yT = t(y)
p = ncol(X)
nu0 = 2
s20 = 1
g = n
S = 10000

### MC sampling
SSRg = yT %*% (diag(1, nrow = n) - (g/(g+1)) * X %*% solve(XT %*% X) %*% XT) %*% y
Vb = (g/(g+1)) * solve(XT %*% X)
Eb = Vb %*% XT %*% y

sigma2.mc = as.matrix(1/rgamma(S, (nu0+n)/2, (nu0*s20 + SSRg)/2), nrow=S)
BETA_Z = matrix(rnorm(S*p, 0, sqrt(sigma2.mc)), S, p)
beta.mc = t(t(BETA_Z %*% chol(Vb)) + c(Eb))
```

```
### posterior confidence interval
colnames(sigma2.mc) = "sigma^2"
(sigma2_CIa = apply(sigma2.mc, 2, quantile, c(0.025, 0.975)))
```

```
##        sigma^2
## 2.5%  745.8112
## 97.5% 948.5228
```

```
(beta_CIa = apply(beta.mc, 2, quantile, c(0.025, 0.975)))
```

```
##       intercept      npreg          bp       skin       bmi       ped
```

```
## 2.5%   35.18012 -1.6302199 -0.01744267 -0.1145368 0.1462393  3.255444
## 97.5%  69.15315  0.3210557  0.42816520  0.4979025 1.1415990 17.608222
##             age
## 2.5%  0.4533627
## 97.5% 1.0755089
```

Part (b) Perform the model selection and averaging procedure described in Section 9.3. Obtain $\Pr(\beta_j \neq 0|y)$, as well as posterior confidence intervals for all of the parameters. Compare to the results in part a).

```
lpy.X = function(y, X, g, nu0, s20) {
  n = nrow(X)
  p = ncol(X)
  XT = t(X)
  yT = t(y)
  # at least have intercept: p > 0
  Hg = (g/(g+1)) * X %*% solve(XT %*% X) %*% XT
  SSRg = yT %*% (diag(1, nrow = n) - Hg) %*% y
  # assume constant prior on z
  lpy = -0.5 * (n*log(pi) + p*log(1+g) + (nu0+n)*log(nu0*s20+SSRg) - nu0*log(nu0*s20)) +
    lgamma((nu0+n)/2) - lgamma(nu0/2)
  return(lpy)
}

### initiate
S = 1000
Z = rep(1, p)
lpy.Z = lpy.X(y, X[, Z == 1, drop = FALSE], g, nu0, s20)
z.mcmc = array(dim = c(S, p))
sigma2.mcmc = array(dim = c(S, 1))
b.mcmc = array(dim = c(S, p))

### gibbs sampling
for (s in 1:S) {
  ### update Z
  for (i in 1:p) {
    Zp = Z; Zp[i] = 1 - Zp[i]
    lpy.Zp = lpy.X(y, X[, Zp == 1, drop = FALSE], g, nu0, s20)
    r = (lpy.Zp - lpy.Z) * (-1)^(Zp[i] == 0)
    Z[i] = rbinom(1, 1, 1/(1 + exp(-r)))
    if(Z[i] == Zp[i]) {lpy.Z = lpy.Zp}
  }
  z.mcmc[s, ] = Z

  ### update SIGMA2 and BETA
  XZ = X[, Z == 1, drop = FALSE]
  Hgz = (g/(g+1)) * XZ %*% solve(t(XZ) %*% XZ) %*% t(XZ)
  SSRgz = yT %*% (diag(1, nrow = n) - Hgz) %*% y
  Vbz = (g/(g+1)) * solve(t(XZ) %*% XZ)
  Ebz = Vbz %*% t(XZ) %*% y
  SIGMA2 = 1/rgamma(1, (nu0+n)/2, (nu0*s20 + SSRgz)/2)
  BETA_Z = matrix(rnorm(1*sum(Z), 0, sqrt(SIGMA2)), 1, sum(Z))

  ### store drawn samples
```

```
  sigma2.mcmc[s, 1] = SIGMA2
  b.mcmc[s, as.logical(Z)] = t(t(BETA_Z %*% chol(Vbz)) + c(Ebz))
}
```

```
### Pr(beta_j != 0 | y)
colnames(z.mcmc) = colnames(X)
beta.mcmc = z.mcmc * b.mcmc
colnames(sigma2.mcmc) = "sigma^2"
apply(z.mcmc != 0, 2, mean)
```

```
## intercept     npreg        bp      skin       bmi       ped       age
##     1.000     0.100     0.170     0.099     0.982     0.692     1.000
```

```
### confidence interval for samples of SELECTED parameters
(sigma2_CIb = apply(sigma2.mcmc, 2, quantile, c(0.025, 0.975), na.rm = TRUE))
```

```
##        sigma^2
## 2.5%  761.4030
## 97.5% 965.6683
```

```
(z_CIb = apply(z.mcmc, 2, quantile, c(0.025, 0.975), na.rm = TRUE))
```

```
##       intercept npreg bp skin bmi ped age
## 2.5%          1     0  0    0   1   0   1
## 97.5%         1     1  1    1   1   1   1
```

```
(beta_CIb = apply(beta.mcmc, 2, quantile, c(0.025, 0.975), na.rm = TRUE))
```

```
##       intercept      npreg          bp       skin       bmi       ped
## 2.5%   42.61064 -1.5049497 -0.02446156 -0.1289222 0.5200586  2.785541
## 97.5%  75.92928  0.2680316  0.40158922  0.6396591 1.3196147 17.733843
##             age
## 2.5%  0.4808481
## 97.5% 1.0101671
```

The confidence intervals in part (a) are similar to the results in part (b), but in part (b) we have explicit parameter z indicating the posterior belief of whether should we include the corresponding β into our regression model. In addition, the distance between the boundary (center) of confidence intervals to 0 do not tell much about the probability of whether should we include this parameter (CI of `bmi` is closer to 0 than `ped`, but the posterior mean of z for `bmi` is smaller than `ped`).

2. NOTE THAT THERE IS AN UPDATE TO THE STATEMENT HERE (Monday Nov 11). YOU CAN DO EITHER VERSION OF IT. In addition to the implementation above, consider the George and McCullagh 1993 paper I mentioned in class.

In particular, fit the following specification using the data in 9.2:

$$\text{ORIGINAL VERSION: } Y_i|X_i, z, \beta, \sigma^2 \sim N((z \cdot \beta)^t X_i, \sigma^2), \text{ where } z \cdot \beta = (z_1\beta_1, \ldots, z_p\beta_p)$$

OR the version from the PAPER:

$$Y_i|X_i, \beta, \sigma^2 \sim N(\beta^t X_i, \sigma^2)$$

the prior is,

$$\beta_j | z_j \sim (1 - z_j) N(0, \tau_j^2) + z_j N(0, c_j^2 \tau_j^2)$$

$$p(z_j = 1) = 1/2$$

and

$$\sigma^2 \sim \text{inverseGamma}$$

Perform a sensitivity analysis for different values of

$$c_j > 1$$
$$\tau_j^2 > 0$$

Find the posterior 95% credible intervals for the β_js

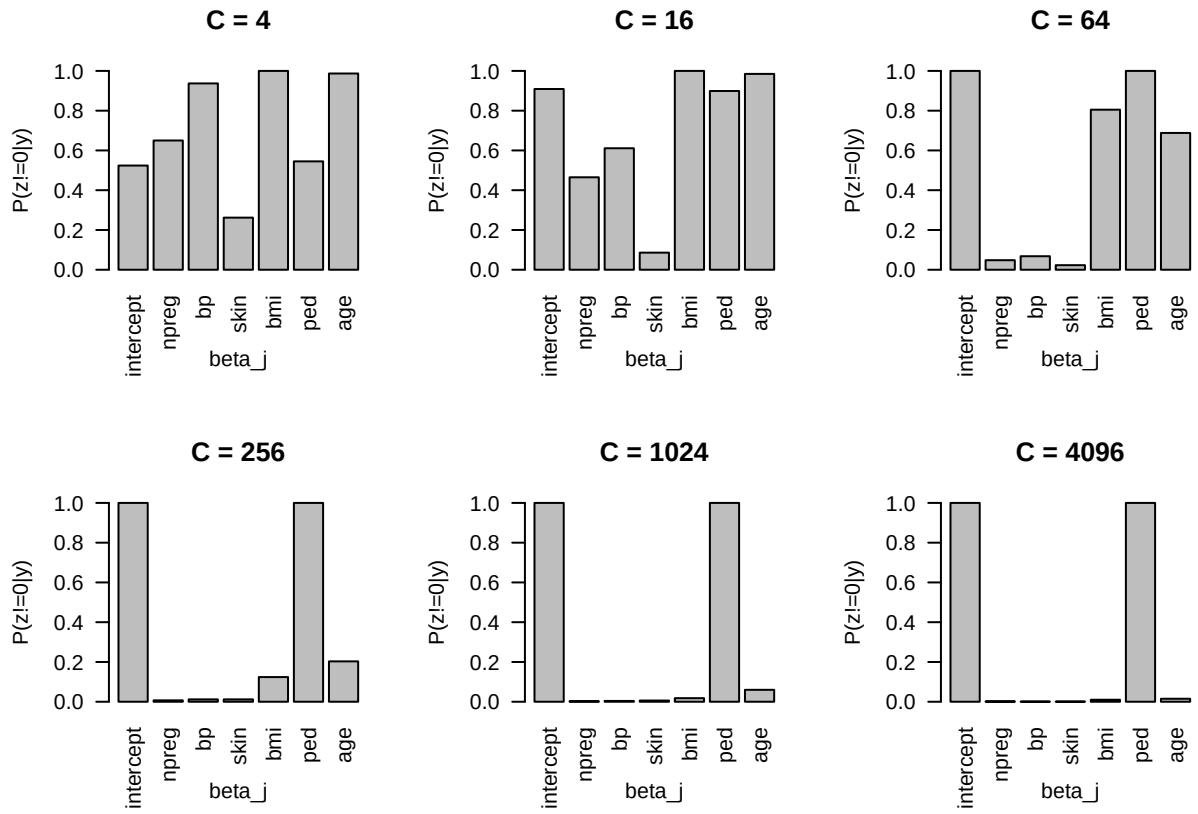
Derivation of the full conditional could be found in the attached scanned PDF file.

```
### data and priors
n = nrow(X)
p = ncol(X)
nu0 = 2
s20 = 1
g = n
c = rep(10, p)
tau = rep(1, p)

betaOLS = solve(t(X) %*% X) %*% t(X) %*% y
sigma2OLS = t(y - X %*% betaOLS) %*% (y - X %*% betaOLS) / (n - p)
XTX = t(X) %*% X

gibbs_sampler = function(c, tau) {
  ### intemediate
  D = diag(c * tau)
  Dm2 = diag(1 / (c^2*tau^2))

  ### initiation
  S = 1000
  beta.gibbs = array(dim = c(S, p))
  sigma2.gibbs = array(dim = c(S, 1))
  z.gibbs = array(dim = c(S, p))
  BETA = betaOLS
  SIGMA2 = as.numeric(sigma2OLS)
  Z = rep(1, p)

  ### gibbs sampling
  for (s in 1:S) {
    AZ = Z * (c-1) + 1
    Dm2 = diag(1/(AZ*tau)^2)
    ### sample BETA
    Vb = solve(XTX / SIGMA2 + Dm2)
    Eb = Vb %*% XTX %*% betaOLS / SIGMA2
    BETA = MASS::mvrnorm(1, mu = Eb, Sigma = Vb)
```

```
    ### sample SIGMA2
    nun = n + nu0
    s2n = nu0*s20 + sum((y - X %*% BETA)^2)
    SIGMA2 = 1/rgamma(1, nun, s2n)

    ### sample Z
    for (i in 1:p) {
      a = dnorm(BETA[i], 0, c[i]*tau[i])
      b = dnorm(BETA[i], 0, tau[i])
      pz1 = a / (a+b)
      Z[i] = rbinom(1, 1, pz1)
    }

    ### store samples
    beta.gibbs[s, ] = BETA
    sigma2.gibbs[s, 1] = SIGMA2
    z.gibbs[s, ] = Z
  }
  return(list(beta = beta.gibbs, sigma2 = sigma2.gibbs, z = z.gibbs))
}
```

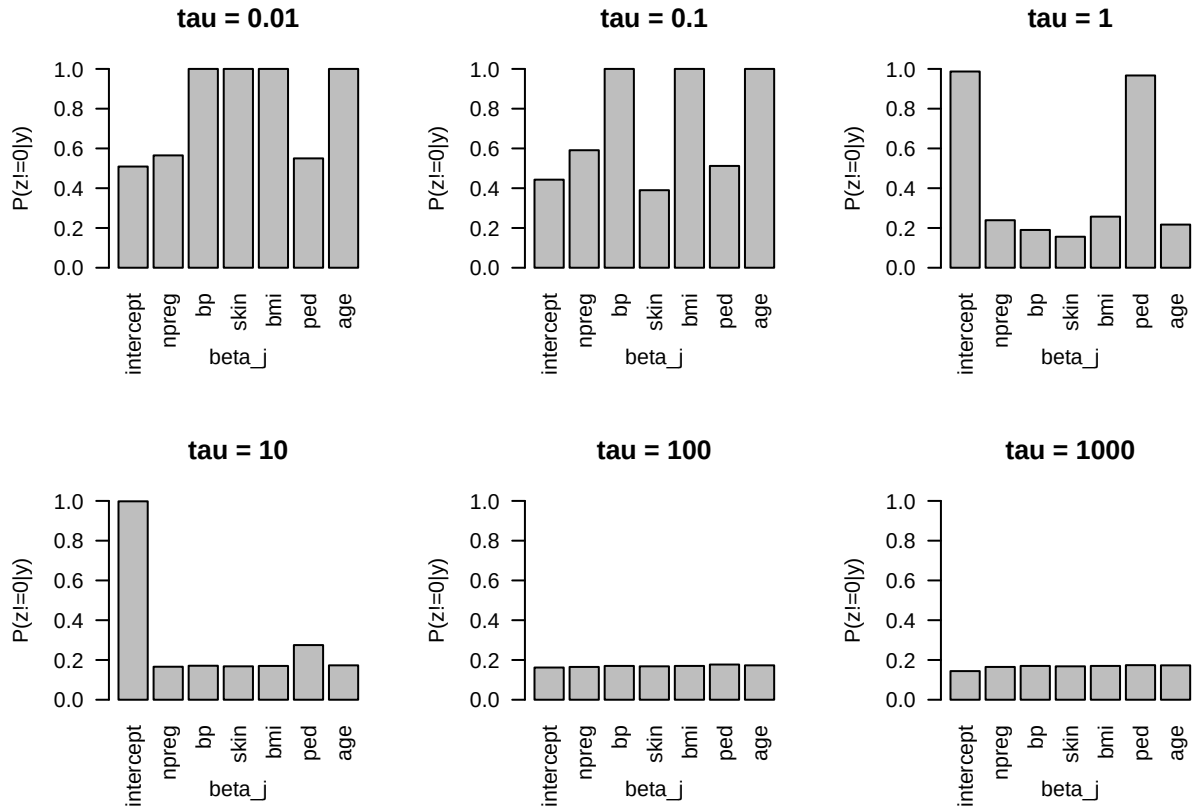
Due to the independence among β and z in our prior settings, we could simply set all c and τ identical to explore the effect of c and τ, respectively.

```
### sensitivity plot of c
par(mfrow=c(2, 3))
for (i in 1:6) {
  c = rep(4^i, p)
  tau = rep(0.2, p)
  set.seed(10)
  params = gibbs_sampler(c, tau)

  barplot(colMeans(params$z),
        xlab = "beta_j",
        ylab = "P(z!=0|y)",
        ylim = c(0, 1),
        names.arg=colnames(X),
        las=2,
        main=paste("C =", 4^i))
}
```



Larger c means the included parameters will have a more diffused prior, which means the prior of dropped parameters will be more relatively centered. This will make the "significant" parameters more significant and "insignificant" parameters less significant.

```
### sensitivity plot of tau
par(mfrow=c(2, 3))
for (i in 1:6) {
  c = rep(5, p)
  tau = rep(10^(i-3), p)
  set.seed(10)
  params = gibbs_sampler(c, tau)

  barplot(colMeans(params$z),
        xlab = "beta_j",
        ylab = "P(z!=0|y)",
        ylim = c(0, 1),
        names.arg=colnames(X),
        las=2,
        main=paste("tau =", 10^(i-3)))
}
```



Larger τ means a more diffused baseline prior, for both included and dropped parameters. When c is constant and τ increases, the differences of spread between priors of dropped and included parameters shrinkage towards 0. In the extreme case there will be little difference between the priors of included and dropped parameters.

```
c = rep(4, p)
tau = rep(1, p)
params = gibbs_sampler(c = c, tau = tau)
z.gibbs = params$z
beta.gibbs = params$beta
sigma2.gibbs = params$sigma2

### Pr(z_i == 0 | Y)
colnames(z.gibbs) = colnames(X)
apply(z.gibbs, 2, mean)
```

```
## intercept     npreg        bp      skin       bmi       ped       age
##     0.930     0.273     0.218     0.196     0.338     0.953     0.256
```

```
### posterior confidence interval
colnames(beta.gibbs) = colnames(X)
apply(beta.gibbs, 2, quantile, c(0.025, 0.975))
```

```
##         intercept       npreg        bp       skin       bmi        ped
## 2.5%  -0.06999309 -1.67848029 0.3392163 -0.1781909 0.7469106  0.5608831
## 97.5% 16.53410389  0.07345738 0.7463198  0.4234166 1.6451992 12.2120677
##             age
## 2.5%  0.5975914
## 97.5% 1.2038149
```

```r
colnames(sigma2.gibbs) = "sigma^2"
apply(sigma2.gibbs, 2, quantile, c(0.025, 0.975))
```

```
##        sigma^2
## 2.5%  791.2827
## 97.5% 941.2239
```

The results are quite different from the results in previous problem. In this prior setup, both `bmi` and `age` are identified as less informative features, and `ped` is shown much more significant than other features.