STAT 632: Bayesian probit regression

Consider a probit regression model

$$\Pr(y_i = 1 \mid \beta) = \Phi(x_i'\beta), \quad i = 1, \ldots, n,$$

with the covariates $x_i \in \mathbb{R}^d$. Assume a $N(0, \xi^{-1} I_d)$ prior on β. Our goal is to sample from the posterior distribution of $\beta \mid y$:

$$\pi(\beta \mid y) \propto \left[\prod_{i=1}^{n} \{\Phi(x_i'\beta)\}^{y_i} \{1 - \Phi(x_i'\beta)\}^{1-y_i} \right] e^{-\xi\, \beta'\beta/2}.$$

Using the Albert-Chib data-augmentation trick, introduce latent variables $z_i, i = 1, \ldots, n$, with

$$y_i = \mathbf{1}(z_i > 0), \quad z_i \mid \beta \sim N(x_i'\beta, 1).$$

Clearly, this implies $Pr(y_i = 1) = \Phi(x_i'\beta)$. Letting $z = (z_1, \ldots, z_n)'$, the joint posterior of β, z is

$$\pi(\beta, z \mid y) \propto \pi(\beta) \prod_{i=1}^{n} \left[\mathbf{1}(z_i > 0)\mathbf{1}(y_i = 1) + \mathbf{1}(z_i \le 0)\mathbf{1}(y_i = 0) \right] \phi(z_i - x_i'\beta).$$

The Albert–Chib Gibbs sampler cycles through:

- Sample $\beta \mid y, z$ from $N((X'X + \xi I_d)^{-1} X'z, (X'X + \xi I_d)^{-1})$.
- For $i = 1, \ldots, n$, independently sample $z_i \mid y, \beta$ from
  - a $N(x_i'\beta, 1)$ truncated to $(0, \infty)$ if $y_i = 1$.
  - a $N(x_i'\beta, 1)$ truncated to $(-\infty, 0)$ if $y_i = 0$.

Note that the sampling step for β is again of the $N(Q^{-1}b, Q^{-})$ form (no surprise there!) Let us now see this sampler in practice. We start with generating some data.

```
set.seed(1287)
n = 60
d = 9
betastar = rep(0,d)
betastar[1:5] = c(1.25,2,1,1.75,-1.35)
X = matrix(rnorm(n*d),nrow=n)
mustar = X%*%betastar
#pstar = exp(mustar)/(1+exp(mustar))   #for logistic regression
pstar = pnorm(mustar)
y = rbinom(n,1,pstar)
```

Let us first calculate the mle using glm.

```
# compute MLE and its MSE
MLfit = glm.fit(X,y,family=binomial(link="probit"))
```

```
## Warning: glm.fit: algorithm did not converge
```

```
## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred
```

```
mle_beta = MLfit$coefficients
rbind(mle_beta,betastar)
```

```
##              [,1]     [,2]     [,3]     [,4]      [,5]     [,6]      [,7]
## mle_beta 46.75761 68.98461 44.76269 59.90675 -33.85458 10.69883 -11.06321
## betastar  1.25000  2.00000  1.00000  1.75000  -1.35000  0.00000   0.00000
##                [,8]       [,9]
## mle_beta -0.5309015 -0.5059117
## betastar  0.0000000  0.0000000
```

```
mse = sum((mle_beta-betastar)^2)
mse
```

```
## [1] 13149.21
```

Now, fit the Albert–Chib Gibbs sampler. We set $\xi = 1/2$, a reasonable choice for binary regression (you rarely expect signals of magnitude bigger than 5 or 6). I have run the Gibbs sampler for 5000 iterations, discarding the first 1000 as burn-in.

```
# Albert-Chib
val_AC = probit_ridge_AC(y,X,xi=1/2)
pmean_AC = val_AC$postmean_beta
rbind(pmean_AC,betastar)
```

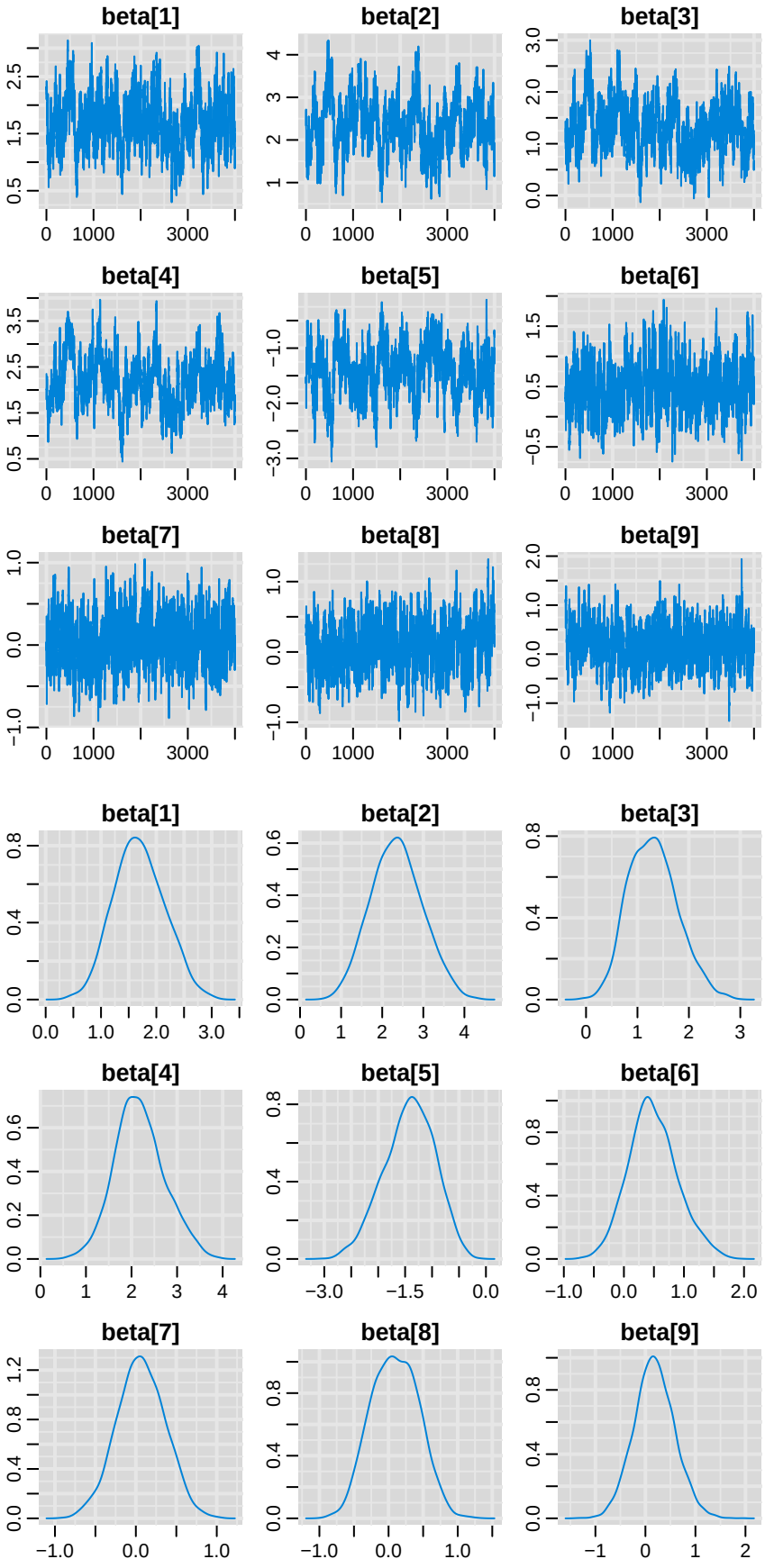
```
##              [,1]     [,2]     [,3]     [,4]      [,5]      [,6]
## pmean_AC 1.691364 2.352949 1.306892 2.177153 -1.421189 0.4876711
## betastar 1.250000 2.000000 1.000000 1.750000 -1.350000 0.0000000
##                [,7]       [,8]      [,9]
## pmean_AC 0.06449173 0.09251187 0.1789179
## betastar 0.00000000 0.00000000 0.0000000
```

```
mse_AC = sum((pmean_AC-betastar)^2)
mse_AC
```

```
## [1] 0.8836379
```

Let us know analyze the output more closely, look at traceplots etc.

```
betaout = val_AC$Betaout
zout = val_AC$Zout
pnames <- c(paste("beta[", 1:9, "]", sep=""))
colnames(betaout) <- pnames
```


beta[1]
beta[2]
beta[3]
beta[4]
beta[5]
beta[6]
beta[7]
beta[8]
beta[9]
beta[1]
beta[2]
beta[3]
beta[4]
beta[5]
beta[6]
beta[7]
beta[8]
beta[9]

We see that although the point estimates look reasonable, the mixing doesn't look that good. The beta chains look sticky, which isn't unusual since β and z are expected to be strongly correlated in the posterior. As an alternative, Holmes and Held (2006) proposed marginalizaing out β when sampling z. The updated sampling steps look as follows:

- Sample $\beta \mid y, z$ from $N((X'X + \xi I_d)^{-1}X'z, (X'X + \xi I_d)^{-1})$.
- Sample $z \mid y$ from a $N(0, I_n + \xi^{-1}XX')$ distribution truncated to the region $\mathcal{C} = \mathcal{C}_1 \otimes \ldots \mathcal{C}_n$, with $\mathcal{C}_i = (0, \infty)$ if $y_i = 1$ and $(-\infty, 0)$ if $y_i = 0$.

To implement this sampler, one needs to draw from a multivariate truncated normal distribution. This is a difficult problem in itself. One can use Gibbs sampling to sample a multivariate truncated normal from univariate full conditionals, which are all univariate truncated normals, but then this again mixes slowly. We shall use a recent Hamiltonian Monte Carlo algorithm due to Packman and Paninski, which is implemented in the R pakcage tmg.

```
# Holmes-Held
val_HH = probit_ridge_HH(y,X,xi=1/2)
pmean_HH = val_HH$postmean_beta
mse_HH = sum((pmean_HH-betastar)^2)
rbind(pmean_HH,betastar)
```

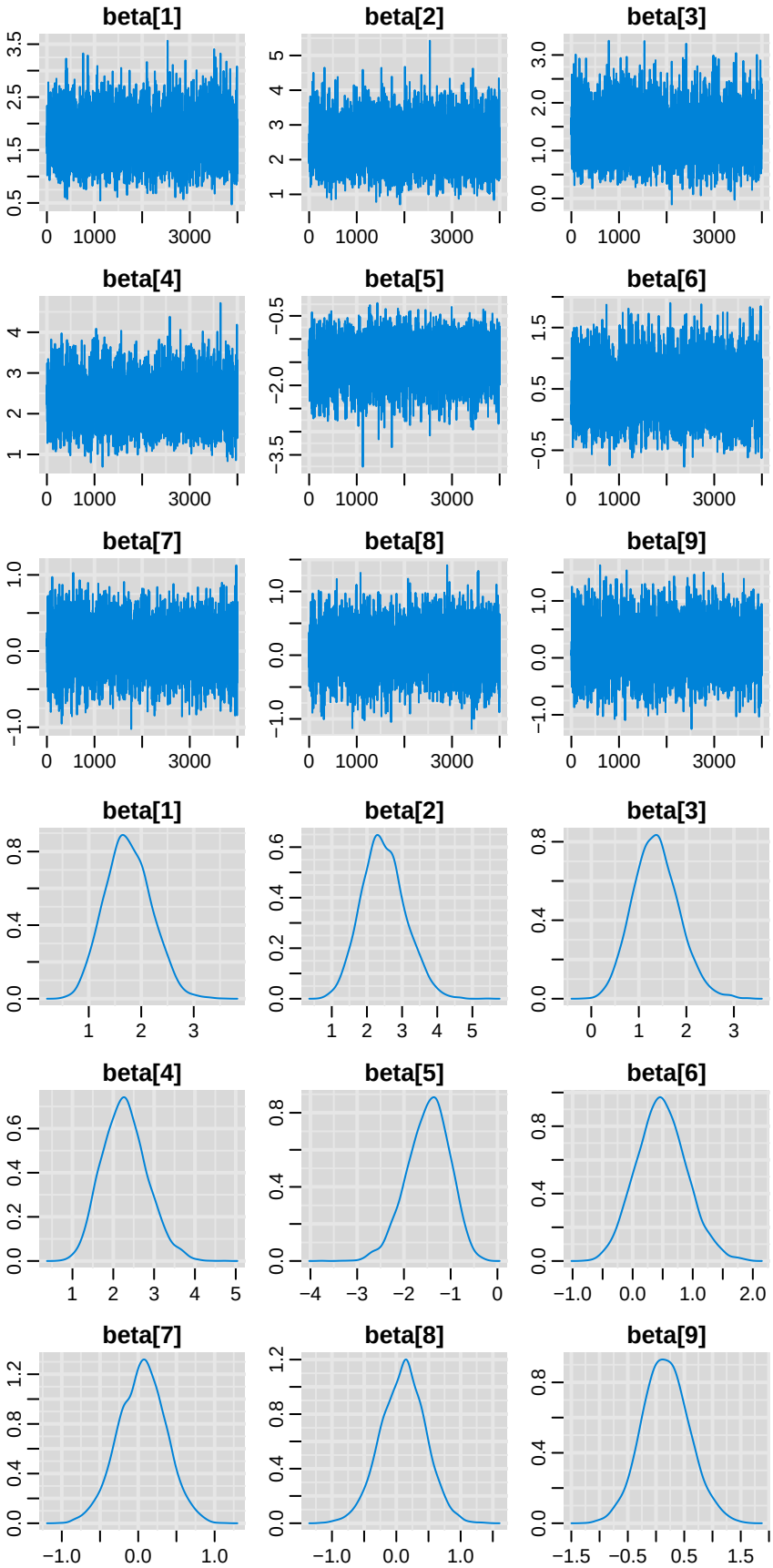
```
##              [,1]     [,2]     [,3]     [,4]     [,5]    [,6]       [,7]
## pmean_HH 1.752216 2.471404 1.363245 2.282229 -1.48166 0.48343 0.04954142
## betastar 1.250000 2.000000 1.000000 1.750000 -1.35000 0.00000 0.00000000
##               [,8]      [,9]
## pmean_HH 0.1038241 0.1720237
## betastar 0.0000000 0.0000000
```

```
mse_HH = sum((pmean_HH-betastar)^2)
mse_HH
```

```
## [1] 1.183522
```

```
betaout = val_HH$Betaout
colnames(betaout) <- pnames
zout = val_HH$Zout
```

Let us know analyze the output more closely, look at traceplots etc.


beta[1]
beta[2]
beta[3]
beta[4]
beta[5]
beta[6]
beta[7]
beta[8]
beta[9]
beta[1]
beta[2]
beta[3]
beta[4]
beta[5]
beta[6]
beta[7]
beta[8]
beta[9]