Bayesian Statistics: Laboratory 3

Vincenzo Gioia

DEAMS

University of Trieste

vincenzo.gioia@units.it

Building D, room 2.13

Office hour: Friday, 15 - 17

28/04/2024

MCMC

- Class of methods for simulating draws that are slightly dependent and are approximately from a posterior distribution $\pi(\theta|y)$
- We then take those draws and calculate quantities of interest for the posterior distribution, such as

$$\mathbb{E}_{\pi(\theta|y)}(h(\theta)) = \int_{\Theta} h(\theta)\pi(\theta|y)d\theta$$

- **Metropolis - Hastings**
- **Gibbs sampling**

Load the following packages

```
library(mvtnorm) # For the multivariate normal distribution
library(truncnorm) # For the truncated normal distribution
library(microbenchmark) # For computational time comparisons
library(gclus) # For the dataset bank

set.seed(123)
```

Outline

Section 1

Random walk Metropolis-Hastings

Metropolis-Hastings (last LAB)

Independent Metropolis-Hastings algorithm

- Initialize $\theta^{(0)}$ on a value of the support of the target $\pi(\theta|y)$
- At s iteration ($s = 1, \ldots, S$)
  - generate a candidate θ^* from the proposal distribution $g(\theta^*)$
  - Compute the acceptance probability as

$$\rho(\theta^{(s-1)}, \theta^*) = \min\left(1, \frac{\pi(\theta^*|y)g(\theta^{(s-1)})}{\pi(\theta^{(s-1)}|y)g(\theta^*)}\right)$$

  - Generate $U \sim U(0,1)$: if $U < \rho(\theta^{(s-1)}, \theta^*)$, then $\theta^{(s)} = \theta^*$, otherwise $\theta^{(s)} = \theta^{(s-1)}$.

Note that the proposal distribution at time s does not depend on the value of the chain at time $s-1$ and that the ratio of proposal distributions in the acceptance ratio does not simplify.

Metropolis-Hastings

- The independent MH ignore local information, while take into account the value previously simulated to generate the next value gives a more local exploration of the neighborhood of the current value

Metropolis-Hastings algorithm

- Initialize $\theta^{(0)}$ on a value of the support of the target $\pi(\theta|y)$
- At s iteration ($s = 1, \ldots, S$)
  - generate a candidate θ^* from the proposal distribution $g(\theta^*|\theta^{(s-1)})$
  - Compute the acceptance probability as

$$\rho(\theta^{(s-1)}, \theta^*) = \min\left(1, \frac{\pi(\theta^*|y)g(\theta^{(s-1)}|\theta^*)}{\pi(\theta^{(s-1)}|y)g(\theta^*|\theta^{(s-1)})}\right)$$

  - Generate $U \sim U(0,1)$: if $U < \rho(\theta^{(s-1)}, \theta^*)$, then $\theta^{(s)} = \theta^*$, otherwise $\theta^{(s)} = \theta^{(s-1)}$

Randow walk

- Let $\{X^{(t)}\}$ a Markov Chain (that is a sequence of random variables such that $p(X^{(t)}|X^{(0)}, X^{(1)}, \ldots, X^{(t-1)}) \equiv p(X^{(t)}|X^{(t-1)})$ or $X^{(t)}|X^{(0)}, X^{(1)}, \ldots, X^{(t-1)} \sim K(X^{(t-1)}, X^{(t)}))$

- A simple random walk Markov chain satisfies

$$X^{(t+1)} = X^{(t)} + \epsilon_t,$$

  where $\epsilon_t \sim \mathcal{N}(0, 1)$, independently of $X^{(t)}$

- $K(X^{(t)}, X^{(t+1)})$ corresponds to a $\mathcal{N}(X^{(t)}, 1)$ density

Randow walk Metropolis-Hastings algorithm

- Idea **Random Walk Metropolis Hastings**: At the s-step we simulate θ^* according to

$$\theta^* = \theta^{(s-1)} + \epsilon_s,$$

  - ϵ_s is a random perturbation with distribution h (independent of $\theta^{(s-1)}$)

- The proposal density $g(\theta^*|\theta^{(s-1)})$ is of the form $h(\theta^* - \theta^{(s-1)})$

- The Markov chain associated with g is a **random walk** when the density of h is symmetric around zero $(h(-t) = h(t))$

- Then, the acceptance probability does not depend on g

$$\rho(\theta^{(s-1)}, \theta^*) = \min\left(1, \frac{\pi(\theta^*|y)}{\pi(\theta^{(s-1)}|y)}\right)$$

  - This not means that the choice of h has no impact

Random walk Metropolis - Hastings: Example

- Consider the Bayesian *probit* regression model defined as

$$
\begin{aligned}
y_i &\sim \textit{Bernoulli}(p_i), \quad i = 1, \ldots, n \\
p_i &= \Phi(\eta_i) \\
\eta_i &= x_i \beta \\
\beta &\sim \pi(\beta)
\end{aligned}
$$

- $y_i \in \{0, 1\}$: binary response variable
- $x_i \in \mathbb{R}^p$: (row) vector of covariates ($x_{i1} = 1$)
- $\beta = (\beta_1, \ldots, \beta_p)^\top$: regression coefficient vector
- η_i: linear predictor
- Φ: cdf of $\mathcal{N}(0, 1)$

Random walk Metropolis - Hastings: Example

- Consider the dataset `bank` from the package `gclus`

```
data(bank)
```

Take a look to the dataset (via the help() function)

- Status: of the banknotes (0 = genuine, 1 = counterfeit) - **Outcome**
- Length: Length of bill (in mm)
- Left: Width of lef edge (in mm)
- Right: Width of right edge (in mm)
- Bottom: Bottom margin width (in mm)
- Top: Top margin width (in mm)
- Diagonal: Length of image diagonal (in mm)

Random walk Metropolis - Hastings: Example

We will focus our analysis only considering the dependent variables *Length*, *Left*, *Right* and *Bottom*. Then:

```
y <- bank$Status
# equivalently
# y <- bank[,1]

X <- model.matrix(~ Length + Left + Right + Bottom,
                  data = bank)
# equivalently
# X <- cbind(1,bank[,2:5])

n <- dim(bank)[1]
```

Random walk Metropolis - Hastings: Example

Likelihood inference: MLE for binary probit regression

$$\hat{\beta} = \arg\max_{\beta} \mathcal{L}(\beta)$$

where $\mathcal{L}(\beta)$ is the likelihood function ($p(y|\beta)$)

- Use the `glm()` (with the suitable arguments) to fit the probit regression model
- Extract the ML Estimate $\hat{\beta}$
- Extract from the summary the estimated covariance matrix of $\hat{\beta}$ ($\hat{\Sigma}$), that is the inverse of the Fisher information matrix evaluated at the MLE

Random walk Metropolis - Hastings: Example

```
fit_glm <- glm(y ~ Length + Left + Right + Bottom,
               family = binomial(link = "probit"), data = bank)
# Alternatively
# fit_glm <- glm(Status ~ X - 1, family = binomial(link = "probit"))
beta_mle <- coef(fit_glm); round(beta_mle, 2)
```

```
## (Intercept)      Length        Left       Right
##     -113.11       -0.81        1.06        1.06
##      Bottom
##        1.11
```

```
sigma.asymp <- summary(fit_glm)$cov.unscaled; round(sigma.asymp, 2)
```

```
##             (Intercept) Length  Left Right Bottom
## (Intercept)     6956.75 -22.04 -7.22 -9.81  -0.39
## Length           -22.04   0.15 -0.06 -0.01   0.00
## Left              -7.22  -0.06  0.36 -0.20   0.01
## Right             -9.81  -0.01 -0.20  0.29  -0.02
## Bottom            -0.39   0.00  0.01 -0.02   0.03
```

Random walk Metropolis - Hastings: Example

Recall a result from the asymptotic theory of the ML estimator

$$\hat{\beta} \dot{\sim} \mathcal{N}(\beta, i(\beta)^{-1})$$

with $i(\beta)$ the Fisher information matrix. Then, the variance covariance matrix Σ is simply given by $i(\beta)^{-1}$, which in GLM takes the form

$$\Sigma = (X^\top W X)^{-1}$$

where W is a $n \times n$ diagonal matrix with entries $W_{ii} = 1/(g'(\mu_i)^2 Var(Y_i))$.

In such a case,

- $Var(Y_i) = p_i(1 - p_i) = \Phi(x_i\beta)(1 - \Phi(x_i\beta))$
- $g'(\mu_i) = 1/\phi(x_i\beta)$, with $\phi(\cdot)$ the pdf of $\mathcal{N}(0, 1)$ (here g is the link function)

Random walk Metropolis - Hastings: Example

- An estimate of Σ is obtained by substituting the MLE $\hat{\beta}$ into the quantities involved into the matrix W

```
#Alternatively sigma.asymp can be obtained as
eta_hat <- predict(fit_glm)
p_hat <- predict(fit_glm, type = "response")
Wii <- dnorm(eta_hat)^2 / (p_hat * (1 - p_hat))
sigma.asymp2 <- solve(t(X) %*% diag(Wii) %*% X)
round(sigma.asymp2, 2)
```

```
##             (Intercept) Length  Left Right Bottom
## (Intercept)     6956.88 -22.04 -7.23 -9.81  -0.39
## Length           -22.04   0.15 -0.06 -0.01   0.00
## Left              -7.23  -0.06  0.36 -0.20   0.01
## Right             -9.81  -0.01 -0.20  0.29  -0.02
## Bottom            -0.39   0.00  0.01 -0.02   0.03
```

Random walk Metropolis - Hastings: Example

The previous implementation of the matrix product $X^\top WX$ is not efficient. Consider the following implementation that

- at first explicitly take into account the diagonal structure of W
- and then we make use of the efficient matrix product via `crossprod`

```
PREC1 <- t(X) %*% diag(Wii) %*% X
PREC2 <- t(X) %*% (X * Wii)
PREC3 <- crossprod(X *  sqrt(Wii))
c(max(abs(PREC1 - PREC2)), max(abs(PREC1 - PREC3)))
```

```
## [1] 1.455192e-11 4.656613e-10
```

Random walk Metropolis - Hastings: Example

Then, we compare the computational performances by means of the `microbenchmark()` function of the **microbenchmark** package

```
microbenchmark(
PREC1 = t(X) %*% diag(Wii) %*% X,
PREC2 = t(X) %*% (X * Wii),
PREC3 = crossprod(X *  sqrt(Wii)))
```

```
## Unit: microseconds
##   expr   min    lq    mean median     uq   max
##  PREC1 262.7 296.4 334.357 327.00 350.45 629.2
##  PREC2  11.7  14.7  17.141  15.90  18.70  52.2
##  PREC3   8.5  10.6  12.401  11.55  12.85  28.9
##  neval cld
##    100  a
##    100   b
##    100   b
```

Random walk Metropolis-Hastings algorithm

Coming back to the Bayesian probit regression model

- Set a flat prior distribution on β ($\pi(\beta) \propto 1$). What is the posterior distribution? Write the R function for the posterior distribution.

Likelihood:

$$\mathcal{L}(\beta) = \prod_{i=1}^{n} p_i^{y_i}(1-p_i)^{1-y_i} = \prod_{i=1}^{n} \Phi(x_i\beta)^{y_i}(1-\Phi(x_i\beta))^{1-y_i}$$

Posterior:

$$\pi(\beta|y) \propto \mathcal{L}(\beta)\pi(\beta) = \prod_{i=1}^{n} \Phi(x_i\beta)^{y_i}(1-\Phi(x_i\beta))^{1-y_i}$$

```
post <- function(b, y, X){
    pi <- pnorm(X %*% as.vector(b))
    return(prod(pi ^ y * (1 - pi) ^ (1 - y)))
}
```

Random walk Metropolis - Hastings: Example

Implement the following random walk Metropolis-Hastings algorithm:

- Initialize $\beta^{(0)} = \hat{\beta}$
- For S-1 iterations repeat the following steps
  - Generate $\beta^* \sim \mathcal{N}_p(\beta^{(s-1)}, \tau^2\hat{\Sigma})$
  - Compute the acceptance probability as

$$\rho(\beta^{(s-1)}, \beta^*) = \min\left(1, \frac{\pi(\beta^*|y)}{\pi(\beta^{(s-1)}|y)}\right)$$

  - Generate $U \sim U(0,1)$: if $U < \rho(\beta^{(s-1)}, \beta^*)$, then $\beta^{(s)} = \beta^*$, otherwise $\beta^{(s)} = \beta^{(s-1)}$

Note: $\hat{\beta}$ is the MLE, while $\hat{\Sigma}$ is the estimated covariance matrix of $\hat{\beta}$

Random walk Metropolis-Hastings algorithm

```
mh <- function(Nsim, tau,y, X, beta_mle, sigma){
    b <- matrix(0, nrow = Nsim, ncol = ncol(X))
    b[1,] <- beta_mle
        for(s in 2:Nsim){
        b_star <- rmvnorm(1, b[s - 1, ], tau ^ 2 * sigma)
        num <- post(b_star, y, X)
        den <- post(b[s - 1, ], y, X)
        rho <- min(1, num / den)
        if ( runif(1) < rho ) {
            b[s, ] <- b_star
        } else {
            b[s, ] <- b[s - 1, ]
        }
    }
    return(b)
}
```

Random walk Metropolis-Hastings algorithm

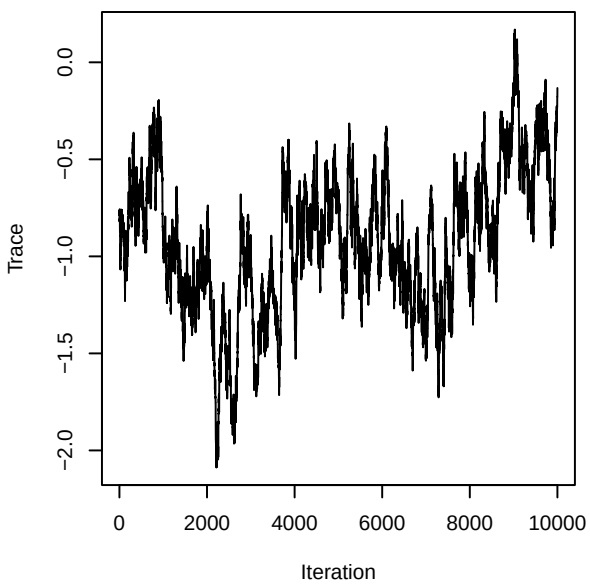
- Simulate 10000 iterations using different values of $\tau = 0.1, 1, 4$
- Plot the trace of the parameter β_2, the autocorrelation and the histogram adding a vertical line for the posterior estimate of the mean.
- Which value of τ seems the best?

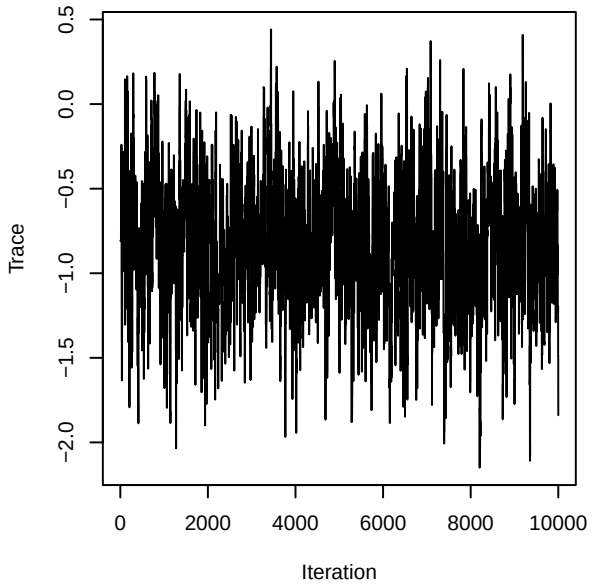
```
tau <- c(0.1, 1, 4)
K <- length(tau)
nsim <- 1e4
b_sim <- lapply(1 : K,
                function(.x) mh(nsim, tau[.x], y, X,
                                beta_mle, sigma.asymp))
```

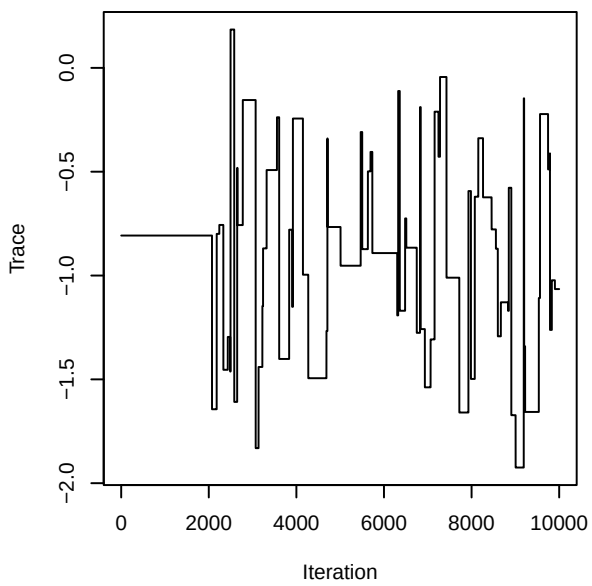
Random walk Metropolis-Hastings algorithm

```
for(j in 1 : K) plot(b_sim[[j]][, 2], type = "l",
                     ylab = "Trace", xlab = "Iteration",
                     main = "")

for(j in 1 : K) acf(b_sim[[j]][, 2], main = "")

for(j in 1 : K) {
 hist(b_sim[[j]][,2], breaks = 50,
      border = "gray40", freq = F, main = "")
 abline(v = mean(b_sim[[j]][, 2]),
        col = "firebrick3", lty = 2)
}
```

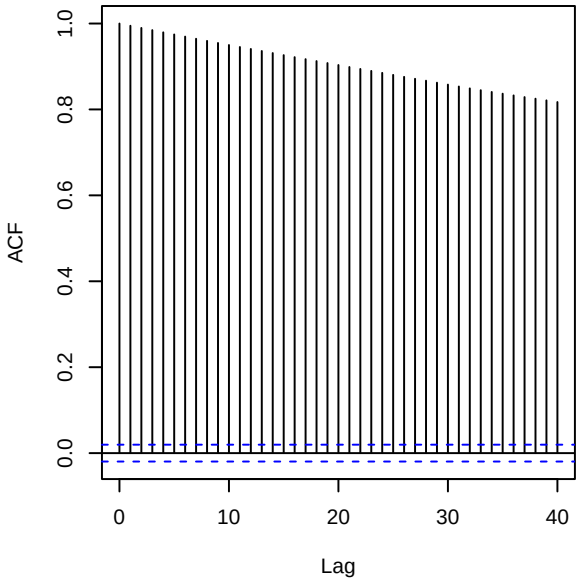
Random walk Metropolis-Hastings algorithm

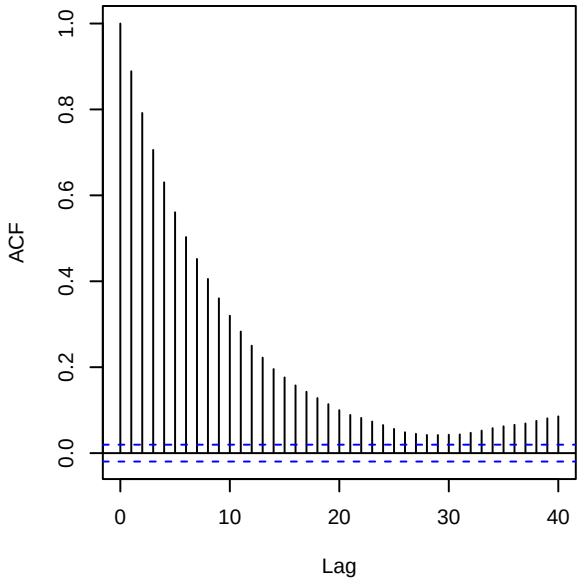


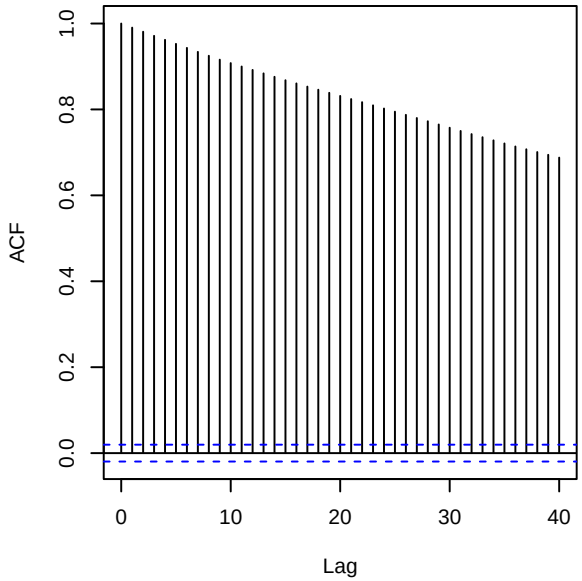




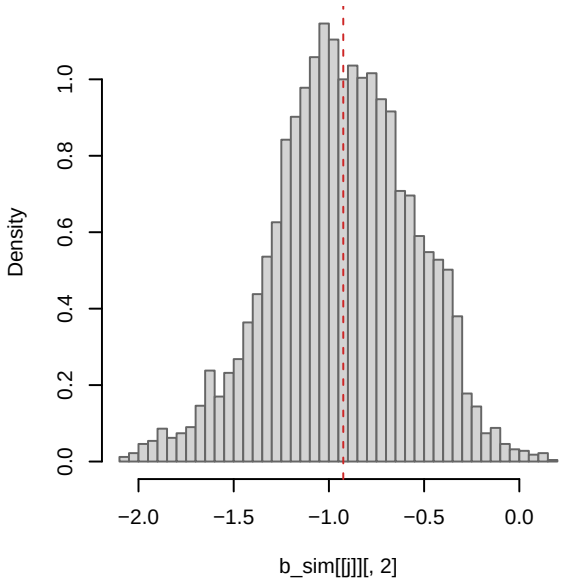
Random walk Metropolis-Hastings algorithm

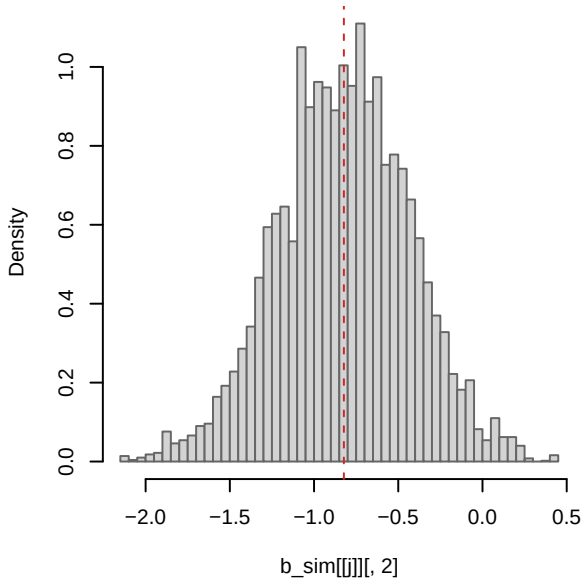


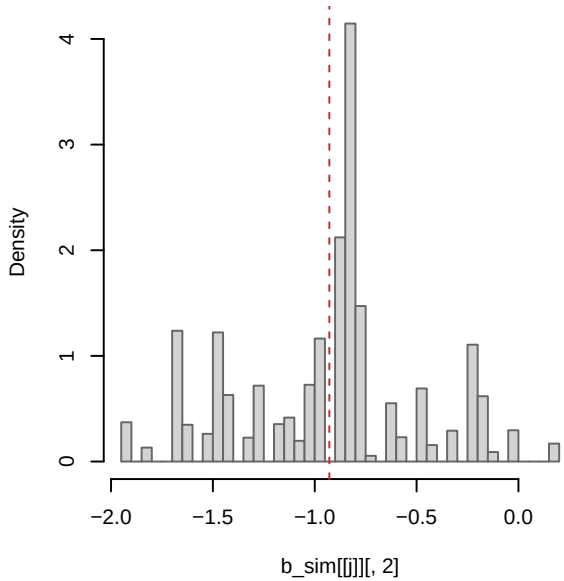




Random walk Metropolis-Hastings algorithm







Random walk Metropolis-Hastings algorithm

- Using the selected simulations, give a Monte Carlo estimate of the posterior mean and variance of β (burn-in=1000 and thin=10).

```
post_sample <- b_sim[[2]][seq(1e3, 1e4, by = 10),]
beta_post_mean <- round(colMeans(post_sample), 2)
beta_post_mean
```

```
## [1] -119.04   -0.84    1.07    1.14    1.15
```

```
beta_post_var <- round(apply(post_sample, 2, var), 2)
beta_post_var
```

```
## [1] 7384.11    0.15    0.38    0.32    0.03
```

Random walk Metropolis-Hastings algorithm

- Compare the posterior mean estimates with the MLE ones.

```
rbind(beta_post_mean, round(beta_mle,2))
```

```
##                (Intercept) Length Left Right
## beta_post_mean     -119.04  -0.84 1.07  1.14
##                    -113.11  -0.81 1.06  1.06
##                Bottom
## beta_post_mean   1.15
##                  1.11
```

```
rbind(beta_post_var, round(diag(sigma.asymp),2))
```

```
##                (Intercept) Length Left Right
## beta_post_var      7384.11   0.15 0.38  0.32
##                    6956.75   0.15 0.36  0.29
##                Bottom
## beta_post_var   0.03
##                 0.03
```

Section 2

Gibbs sampler

Gibbs sampler

Gibbs sampler creates a Markov chain from a joint distribution

- Gather most of their calibration from the target density
- Break complex problems into a series of easier problems, like a sequence of small-dimensional targets
- The algorithm straightforward if simulating from conditionals is feasible
- Converge insured, unless the supports of the conditionals are not connected, despite may take longer time
- Note: the value is always accepted
- Popular since it was the perfect computational complement to hierarchical models

Gibbs sampler

(multistage) Gibbs sampler

Suppose we wish to to sample from $\pi(\theta|y)$, with $\theta = (\theta_1, \ldots, \theta_k)$.

- Initialise $\theta^{(0)}$
- At step s, given $\theta^{(s-1)} = (\theta_1^{(s-1)}, \ldots, \theta_k^{(s-1)})$ we can draw:
  - $\theta_1^{(s)} \sim \pi_1(\theta_1|\theta_2^{(s-1)}, \ldots, \theta_k^{(s-1)}, y)$
  - $\theta_2^{(s)} \sim \pi_2(\theta_2|\theta_1^{(s)}, \theta_3^{(s-1)}, \ldots, \theta_k^{(s-1)}, y)$
  - $\ldots$
  - $\theta_k^{(s)} \sim \pi_k(\theta_k|\theta_1^{(s)}, \theta_2^{(s)}, \ldots, \theta_{k-1}^{(s)}, y)$
- Repeat step 2 until until we get S draws, with each draw being a vector $\theta^{(s)}$, $s = 1, \ldots, S$.

$\pi_1, \ldots, \pi_k$ are the *full conditional distributions* for the components $\theta_1, \ldots, \theta_k$, respectively. The final result is a Markov chain with a bunch of draws θ that can be considered approximately drawn from our posterior.

Data augmentation and Gibbs Sampler

- Missing Data Models (Latent variable, censoring models and mixtures)

$$f(y|\theta) = \int_{\mathcal{Z}} f(y, z|\theta) dz$$

with $f(y|\theta)$ the likelihood and z is a missing data

- $f(y, z|\theta)$ is the complete data likelihood, that can be arbitrary and chosen for convenience, so a Gibbs sampler can be carried out

- Given a prior distribution $\pi(\theta)$ on θ, we can then create a Gibbs sampler that iterates between the conditional distributions

$$\pi(\theta|y, z) \quad \text{and} \quad f(z|y, \theta)$$

and will have stationary distribution $\pi(\theta, z|y)$

For more details see Section 4.2 and 7.4 in Robert and Casella (2010).

Gibbs sampler - Example

- Data augmentation allows to derive a Gibbs sampling scheme to carry out Bayesian inference for the binary probit regression model

$$y_i \sim \mathrm{Be}(p_i = \Phi(\eta_i))$$

$$\eta_i = x_i\beta$$

$$\beta \sim \pi(\beta)$$

- By introducing an auxiliary latent variable we obtain a conditional distributions of the model parameters equivalent to those under a Bayesian normal linear regression model. The original idea was by Albert and Chib (1993) https://www.jstor.org/stable/2290350

Gibbs sampler - Example

- Let y_i^*, $i = 1, \ldots, n$, be independent latent variables, following the normal distribution, then the Bayesian binary probit regression model can be written as

$$y_i = \begin{cases} 0 & \text{if} \quad y_i^* \leq 0 \\ 1 & \text{if} \quad y_i^* > 0 \end{cases}$$

$$y_i^* = x_i\beta + \epsilon_i \quad \epsilon_i \overset{iid}{\sim} \mathcal{N}(0, 1)$$

$$\beta \sim \pi(\beta)$$

- Thus, y_i is deterministic conditional on the sign of the stochastic auxiliary variable y_i^*.

- Exercise: Prove that the probit regression model can be equivalently written leveraging the latent variable class model

Gibbs sampler - Example

- Prove that the probit regression model can be equivalently written leveraging the latent variable class model

- It is easy to show that, with $\epsilon_i \sim \mathcal{N}(0,1)$, $i = 1 \ldots, n$, it holds

$$\begin{aligned} P(y_i = 1) =& P(y_i^* > 0) = P(x_i\beta + \epsilon_i > 0) \\ =& P(\epsilon_i > -x_i\beta) = 1 - \Phi(-x_i\beta) = \Phi(x_i\beta) \end{aligned}$$

Gibbs sampler - Example

- Joint posterior distribution of y^* and β given y and X

$$\pi(y^*, \beta|y, X) \propto \pi(\beta) f(y^*|\beta, X) f(y|y^*) = \pi(\beta) \prod_{i=1}^{n} f(y_i^*|\beta, x_i) f(y_i|y_i^*)$$

  which is difficult to normalize and sample from directly

- However, computation of the marginal posterior of β and y^* using the Gibbs sampling requires only computing the full conditional distribution

  - $f(y^*|\beta, y, X)$
  - $\pi(\beta|y^*, y, X) = \pi(\beta|y^*, X)$

Gibbs sampler - Example

In summary, the Gibbs sampler requires the full conditional

- for β: $\pi(\beta|y^*, X)$

$$\pi(y^*, \beta|y, X) \propto \pi(\beta) \prod_{i=1}^{n} f(y_i^*|\beta, x_i)$$

which, with $\pi(\beta) \propto 1$ and since $y_i^*|\beta, x_i \sim \mathcal{N}(x_i\beta, 1)$, is

$$\beta|X, y^* \sim \mathcal{N}_p((X^TX)^{-1}X^Ty^*, (X^TX)^{-1})$$

- for y^*: $f(y^*|\beta, y, X)$

$$\pi(y^*, \beta|y, X) \propto \prod_{i=1}^{n} f(y_i^*|\beta, x_i) f(y_i|y_i^*)$$

$$y_i^*|\beta, x_i, y_i \sim \begin{cases} \mathcal{TN}(x_i\beta, 1, 0, +\infty) & \text{if } y_i = 1 \\ \mathcal{TN}(x_i\beta, 1, -\infty, 0) & \text{if } y_i = 0 \end{cases}$$

where $\mathcal{TN}$ denoting the left and right truncated normal distribution on 0

Gibbs sampler - Example

Implement a Gibbs' sampler to generate β from the posterior distribution:

- Initialize $\beta^{(1)} = \hat{\beta}$ (to be precise we also should intialise $y^{*(1)}$)
- for s in $2, \ldots, S$
  - generate $y^{*(s)}|\beta^{(s-1)}, X, y \sim \begin{cases} \mathcal{TN}(x_i\beta^{(s-1)}, 1, 0, +\infty) & \text{if } y_i = 1 \\ \mathcal{TN}(x_i\beta^{(s-1)}, 1, -\infty, 0) & \text{if } y_i = 0 \end{cases}$
  - generate $\beta^{(s)}|X, y^{*(s)} \sim N_p((X^TX)^{-1}X^Ty^*, (X^TX)^{-1})$

As above, simulate a sample of 10000 draws of β and visualise the results for β_2 (traceplot, acf, histograms). Then extract the posterior mean and variance of β (by considering burn-in=1000 and thin = 10)

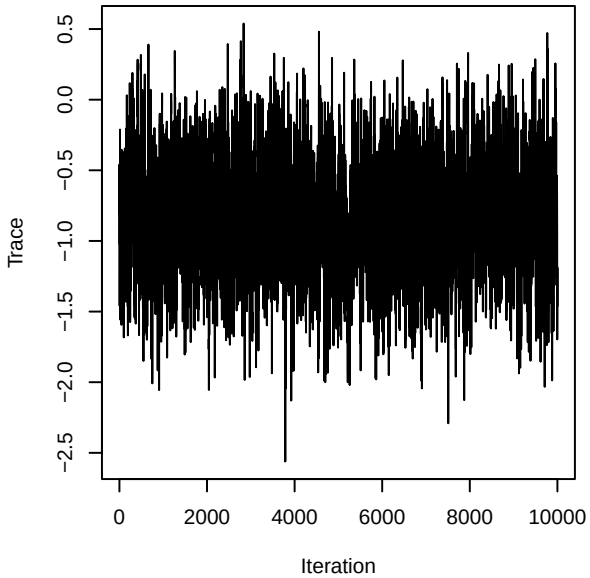
Gibbs sampler - Example

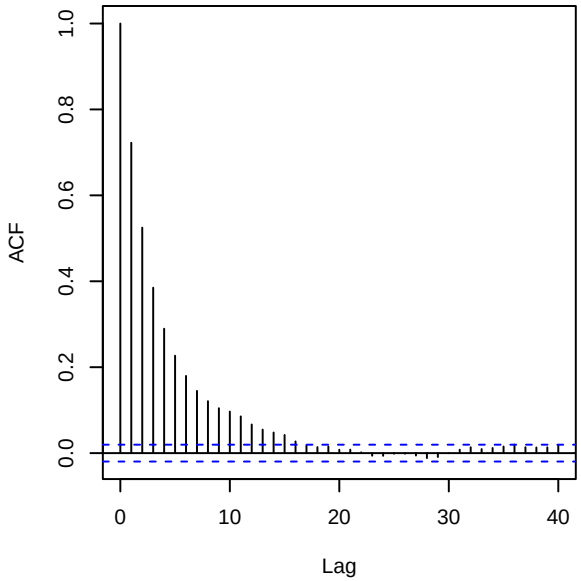
```
gs <- function(Nsim, y, X, beta_mle){
  b <- matrix(0, nrow = Nsim, ncol = ncol(X))
  y_star <- rep(NA, nrow(X))
  b[1,] <- beta_mle
  S <- solve(crossprod(X))
  SX <- S %*% t(X)
  for(i in 2 : Nsim){
    mu <- X %*% b[i - 1,]
    y_star[y == 1] <- rtruncnorm(sum(y), a = 0, b = Inf,
                                 mean = mu[y == 1])
    y_star[y == 0] <- rtruncnorm(n - sum(y), a = -Inf,
                                 b = 0, mean = mu[y == 0])
    b[i,] <- rmvnorm(1, SX %*% y_star, S)
  }
  return(b)
}
```

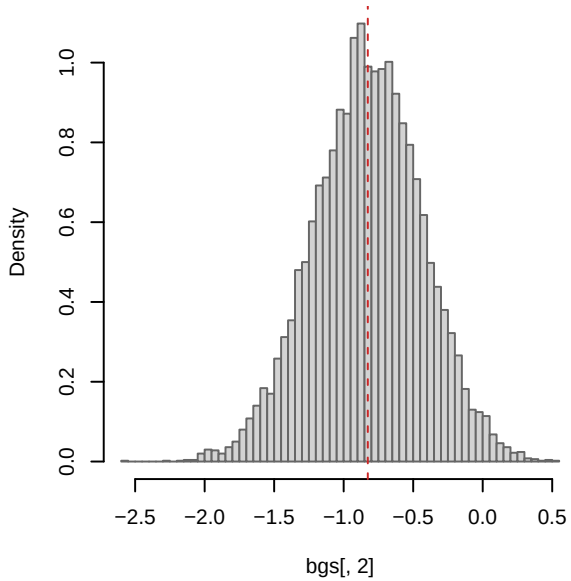
Gibbs sampler - Example

```
bgs <- gs(1e4, y, X, beta_mle)
plot(bgs[, 2], type = "l",
     ylab = "Trace", xlab = "Iteration")
acf(bgs[, 2], main = "")
hist(bgs[, 2], breaks = 50, border = "gray40",
     freq = F, main = "")
abline(v = mean(bgs[, 2]), col = "firebrick3", lty = 2)
```

Gibbs sampler - Example







Gibbs sampler - Example

```
post_sample <- bgs[seq(1e3, 1e4, by = 10),]
beta_post_mean <- round(colMeans(post_sample),  2)
beta_post_mean
```

```
## [1] -121.04   -0.81    1.09    1.10    1.15
```

```
beta_post_var <- round(apply(post_sample, 2, var), 2)
beta_post_var
```

```
## [1] 7897.28    0.15    0.37    0.31    0.03
```

Gibbs sampler - Example

- Compare the posterior mean with the MLE ones, as well the posterior variance with the estimated (asymptotic) variance of the MLE

```
rbind(beta_post_mean, round(beta_mle, 2))
```

```
##                (Intercept) Length Left Right
## beta_post_mean     -121.04  -0.81 1.09  1.10
##                    -113.11  -0.81 1.06  1.06
##                 Bottom
## beta_post_mean   1.15
##                  1.11
```

```
rbind(beta_post_var, round(diag(sigma.asymp), 2))
```

```
##                (Intercept) Length Left Right
## beta_post_var      7897.28   0.15 0.37  0.31
##                    6956.75   0.15 0.36  0.29
##                 Bottom
## beta_post_var   0.03
##                 0.03
```

Gibbs sampler - Example

- Compare the two MH and the GS algorithms via `microbenchmark()`. Consider `times = 10` as argument

```
microbenchmark(
MH = mh(nsim, tau = 1, y, X, beta_mle, sigma.asymp),
GS = gs(nsim, y, X, beta_mle), times = 10L)
```

```
## Unit: seconds
##  expr      min       lq     mean   median
##    MH 3.420271 3.566077 3.675833 3.705095
##    GS 3.227189 3.252301 3.291784 3.262801
##         uq      max neval cld
##  3.777922 3.879564    10  a
##  3.292287 3.526341    10   b
```

Gibbs sampler - Example

Note:

- In this toy example, we limited our prior to be uniform. Of course, with normal prior the full conditional of β is still normal
- Potential problem: there is a strong correlation between β and y^*, and this correlation may result in slow mixing of the chain, hence, it requires a larger number of simulations to reach the target distribution
- Possible solution: proposal in Holmes and Held (2006) https://projecteuclid.org/journals/bayesian-analysis/volume-1/issue-1/Bayesian-auxiliary-variable-models-for-binary-and-multinomial-regression/10.1214/06-BA105.full