# 732A91 - Lab 3

Joris van Doorn || Weng Hang Wong

17 May 2020

## Normal model, mixture of normal model with semi-conjugate prior.

*The data rainfall.dat consist of daily records, from the beginning of 1948 to the end of 1983, of precipitation (rain or snow in units of 1/100 inch, and records of zero precipitation are excluded) at Snoqualmie Falls, Washington. Analyze the data using the following two models.*

### a.

*Assume the daily precipitation $y_1, ..., y_n$ are independent normally distributed, $y_1, ..., y_n|\mu, \sigma^2 \sim N(\mu, \sigma^2)$ where both $\mu$ and $\sigma^2$ are unknown. Let $\mu \sim N(\mu_0, \tau_0^2)$ independently of $\sigma^2 \sim Inv - \chi^2(\nu_0, \sigma_0^2)$.*

### i.

*Implement (code!) a Gibbs sampler that simulates from the joint posterior $p(\mu, \sigma^2|y_1, ..., y_n)$. The full conditional posteriors are given on the slides from Lecture 7.*

We have the following full conditional posteriors:

$$\mu|\sigma^2, x \sim N(\mu_n, \tau_n^2)$$

and

$$\sigma^2|\mu, x \sim Inv - \chi^2(\nu_n, \frac{\nu_0\sigma_0^2 + \sum_{i=1}^n (x_i - \mu)^2}{n + \nu_0})$$

where

$$\mu_n = w\bar{x} + (1 - w)\mu_0$$

$$w = \frac{\frac{n}{\sigma^2}}{\frac{n}{\sigma^2} + \frac{1}{\tau_0^2}}$$

$$\tau_n^2 = \frac{\sigma^2}{n} + \tau_0^2$$

## 2. Metropolis Random Walk for Poisson regression.

*Consider the following Poisson regression model*

$$y_i|\beta \sim Poisson[exp(x_i^T\beta)], i = 1, ..., n$$

*where yi is the count for the ith observation in the sample and $x_i$ is the p-dimensional vector with covariate observations for the ith observation. Use the data set eBayNumberOfBidderData.dat. This dataset contains observations from 1000 eBay auctions of coins. The response variable is nBids and records the number of bids in each auction. The remaining variables are features/covariates (x):*

- **Const** (for the intercept)
- **PowerSeller** (is the seller selling large volumes on Ebay?)
- **VerifyID** (is the seller verified by eBay?)
- **Sealed** (was the coin sold sealed in never opned envelope?)
- **MinBlem** (did the coin have a minor defect?)
- **MajBlem** (a major defect?)
- **LargNeg** (did the seller get a lot of negative feedback from customers?)
- **LogBook** (logarithm of the coins book value according to expert sellers. Standardized)
- **MinBidShare** (a variable that measures ratio of the minimum selling price (starting price) to the book value. Standardized)

### a.

*Obtain the maximum likelihood estimator of $\beta$ in the Poisson regression model for the eBay data [Hint: glm.R, don't forget that glm() adds its own intercept so don't input the covariate Const]. Which covariates are significant?*

```
##
## Call:
## glm(formula = Y ~ X, family = poisson(link = "log"))
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -3.5800  -0.7222  -0.0441   0.5269   2.4605
##
## Coefficients:
##              Estimate Std. Error z value Pr(>|z|)
## (Intercept)   1.07244    0.03077  34.848  < 2e-16 ***
## XPowerSeller -0.02054    0.03678  -0.558   0.5765
## XVerifyID    -0.39452    0.09243  -4.268 1.97e-05 ***
## XSealed       0.44384    0.05056   8.778  < 2e-16 ***
## XMinblem     -0.05220    0.06020  -0.867   0.3859
## XMajBlem     -0.22087    0.09144  -2.416   0.0157 *
## XLargNeg      0.07067    0.05633   1.255   0.2096
## XLogBook     -0.12068    0.02896  -4.166 3.09e-05 ***
## XMinBidShare -1.89410    0.07124 -26.588  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for poisson family taken to be 1)
```

```
##
##     Null deviance: 2151.28  on 999  degrees of freedom
## Residual deviance:  867.47  on 991  degrees of freedom
## AIC: 3610.3
##
## Number of Fisher Scoring iterations: 5
```

The intercept, VerifyID, Sealed, Logbook, and MinBidShare are al significant with $p < 0.0001$. Furthermore is MajBlem significant at $p < 0.01$. PowerSeller, Minblem, and LargNeg do not appear to be significant.

### b.

*Let's now do a Bayesian analysis of the Poisson regression. Let the prior be $\beta \sim N[0, 100(X^T X)^{-1}$ where $X$ is the nxp covariate matrix. This is a commonly used prior which is called Zellner's g-prior. Assume first that the posterior density is approximately multivariate normal:*

$$\beta | y \sim N(\tilde{\beta}, J_y^{-1}(\tilde{\beta}))$$

*where $\tilde{\beta}$ is the posterior mode and $J_y(\tilde{\beta})$ is the negative Hessian at the posterior mode. $\tilde{\beta}$ and $J_y(\tilde{\beta})$ can be obtained by numerical optimization (optim.R) exactly like you already did for the logistic regression in Lab 2 (but with the log posterior function replaced by the corresponding one for the Poisson model, which you have to code up.).*

| | Const | PowerSeller | VerifyID | Sealed | Minblem | MajBlem | LargNeg | LogBook | M |
|---|---|---|---|---|---|---|---|---|---|
| Const | 0.0009455 | -0.0007139 | -0.0002742 | -0.0002709 | -0.0004455 | -0.0002772 | -0.0005128 | 0.0000644 | |
| PowerSeller | -0.0007139 | 0.0013531 | 0.0000402 | -0.0002949 | 0.0001143 | -0.0002083 | 0.0002802 | 0.0001182 | |
| VerifyID | -0.0002742 | 0.0000402 | 0.0085154 | -0.0007825 | -0.0001014 | 0.0002283 | 0.0003314 | -0.0003192 | |
| Sealed | -0.0002709 | -0.0002949 | -0.0007825 | 0.0025578 | 0.0003577 | 0.0004532 | 0.0003376 | -0.0001311 | |
| Minblem | -0.0004455 | 0.0001143 | -0.0001014 | 0.0003577 | 0.0036246 | 0.0003492 | 0.0000584 | 0.0000585 | |
| MajBlem | -0.0002772 | -0.0002083 | 0.0002283 | 0.0004532 | 0.0003492 | 0.0083651 | 0.0004049 | -0.0000898 | |
| LargNeg | -0.0005128 | 0.0002802 | 0.0003314 | 0.0003376 | 0.0000584 | 0.0004049 | 0.0031751 | -0.0002542 | |
| LogBook | 0.0000644 | 0.0001182 | -0.0003192 | -0.0001311 | 0.0000585 | -0.0000898 | -0.0002542 | 0.0008385 | |
| MinBidShare | 0.0011099 | -0.0005686 | -0.0004293 | -0.0000576 | -0.0000644 | 0.0002622 | -0.0001063 | 0.0010374 | |

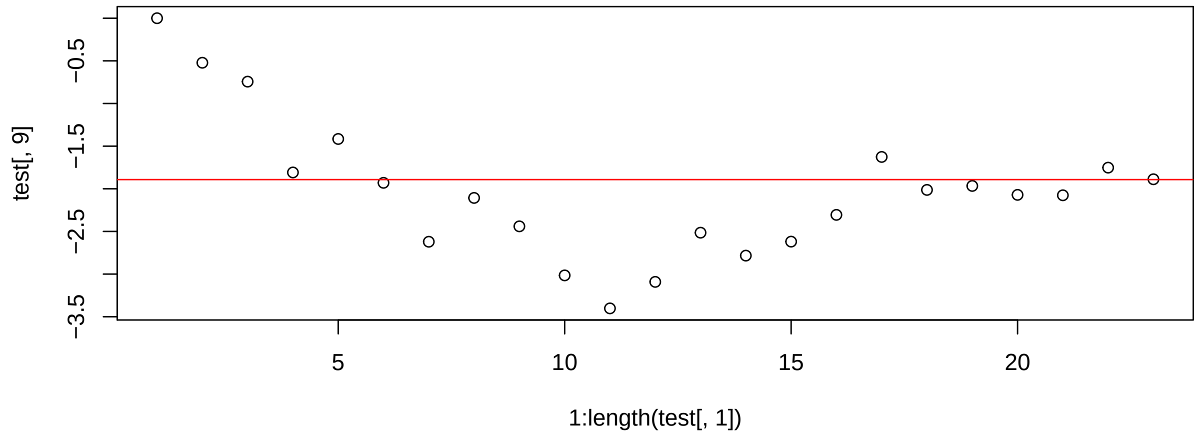
| | Verification | Beta_hat | Beta_std |
|---|---|---|---|
| (Intercept) | 1.0724421 | 1.0698412 | 0.0307484 |
| XPowerSeller | -0.0205408 | -0.0205125 | 0.0367842 |
| XVerifyID | -0.3945165 | -0.3930060 | 0.0922787 |
| XSealed | 0.4438426 | 0.4435555 | 0.0505745 |
| XMinblem | -0.0521983 | -0.0524663 | 0.0602047 |
| XMajBlem | -0.2208712 | -0.2212384 | 0.0914607 |
| XLargNeg | 0.0706725 | 0.0706968 | 0.0563477 |
| XLogBook | -0.1206776 | -0.1202177 | 0.0289564 |
| XMinBidShare | -1.8940966 | -1.8919850 | 0.0710968 |

### c.

*Now, let's simulate from the actual posterior of $\beta$ using the Metropolis algorithm and compare with the approximate results in b). Program a general function that uses the Metropolis algorithm to generate random draws from an arbitrary posterior density. In order to show that it is a general function for any model, I will denote the vector of model parameters by $\theta$. Let the proposal density be the multivariate normal density* *mentioned in Lecture 8 (random walk Metropolis):*

$$\theta_p|\theta^{i-1} \sim N(\theta^{i-1}, c \cdot \sum)$$

*where* $\sum = J_y^{-1}(\hat{\beta})$ *obtained in b). The value c is a tuning parameter and should be an input to your Metropolis function. The user of your Metropolis function should be able to supply her own posterior density function, not necessarily for the Poisson regression, and still be able to use your Metropolis function. This is not so straightforward, unless you have come across function objects in R and the triple dot (...) wildcard argument. I have posted a note (HowToCodeRWM.pdf) on the course web page that describes how to do this in R. Now, use your new Metropolis function to sample from the posterior of* $\beta$ *in the Poisson regression for the eBay dataset. Assess MCMC convergence by graphical methods.*



## d.

*Use the MCMC draws from c) to simulate from the predictive distribution of the number of bidders in a new auction with the characteristics below. Plot the predictive distribution. What is the probability of no bidders in this new auction?*

- **PowerSeller** (=1)
- **VerifyID** (=1)
- **Sealed** (=1)
- **MinBlem** (=0)
- **MajBlem** (=0)
- **LargNeg** (=0)
- **LogBook** (=1)
- **MinBidShare** (=0.5)

## Appendix

```r
knitr::opts_chunk$set(echo = TRUE)
knitr::opts_chunk$set(fig.width=9, fig.height = 4.1)
library(tidyverse)
library(dplyr)
library(knitr)
library(mvtnorm)
library(MASS)
set.seed(12345)
data0 <- read.table("rainfall.dat",header = F)

#I think mu_n and tau_n is defined as if the sigma2 is known, so like the one in the lecture 2
#And he solved a question about tau0 and v_0 (or nu_0)
#And non-informative means having large prior variance, large tau0 and small v_0
#mu_0  = mean(y), sigma_0_sq = var(y), others = 1

#----------------------
# 2a.
data0 <- read.table("eBayNumberOfBidderData.dat",header = T)
Y<-data0$nBids
X<-as.matrix(data0[,3:10])

reg_model <- glm(Y ~ X, family = poisson(link = "log"))
summary(reg_model)
# -------------------------
# Q2b.

# setting initial values
y <- as.vector(data0[,1])
X <- as.matrix(data0[,2:length(data0[1,])])
nCov <- dim(X)[2]
covNames <- names(data0)[2:length(data0[1,])]

# Prior
mu <- as.vector(rep(0,nCov))
sigma <- as.matrix(100*solve((t(X)%*%X)))

set.seed(12345)
# Logistic regression function that returns the regression coefficients
logiPost <- function(betas,y,X,sigma){
  pred <- as.vector(X%*%betas)
  lambda0 <- t(X)*betas
  loglike <- sum(y*pred-exp(pred)-log(factorial(y)))
  logprior <- dmvnorm(betas, mean=rep(0,length(betas)), sigma, log=T)
  return(loglike+logprior)
}

# setting initial values
initVal <- as.vector(rep(0,nCov))
# optimize over the betas
optRes <- optim(initVal,logiPost,gr=NULL,y,X,sigma,method="BFGS",control=list(fnscale=-1),hessian=T)
```

```r
# retrieving betas
beta_hat <- optRes$par
beta_hes <- -solve(optRes$hessian)
beta_std <- as.matrix(sqrt(diag(beta_hes)))

# printing results
colnames(beta_hes) <- covNames
rownames(beta_hes) <- covNames
kable(beta_hes)

kable(data.frame(Verification=reg_model$coefficients,Beta_hat=beta_hat,Beta_std=beta_std))

#---------------------------
# 2c.

# the randomw alk metropolis algorithm in R
RWMSampler <- function(N, burn = 0, c=0.25, sigma, logPostFunc, theta, ...){
  sample <- theta
  for(i in 2:(N + burn)){
    prop <- mvrnorm(n=1, theta, c*as.matrix(sigma))
    proposal <- logPostFunc(prop, ...)
    target <- logPostFunc(theta, ...)
    if(runif(1)<exp(proposal-target)){
      theta <- prop
      sample <- rbind(sample,theta)
    }
  }
  return(sample)
}

test<-RWMSampler(100000, burn=1, c=0.25, sigma = sigma, logPostFunc =logiPost, theta = initVal, y, X, s

plot(1:length(test[,1]),test[,9])
abline(h=beta_hat[9],col="red")
```