

TDDE07 Bayesian Learning - Lab 4

Erik Linder-Norén - erino397

16 Maj, 2017

1. Poisson regression - the MCMC way.

(a)

The β_{MLE} found by fitting a Poisson regression model with glm can be seen in Table 1. Significant covariates are MinBidShare, Sealed, VerifyID and MajBlem.

	Const	PowerSeller	VerifyID	Sealed	Minblem	MajBlem	LargNeg	LogBook	MinBidShare
1	1.072	-0.021	-0.395	0.444	-0.052	-0.221	0.071	-0.121	-1.894

Table 1: MLE of beta by glm model fitting

(b)

By approximating the posterior distribution of beta as a multivariate normal and determining the value for β_{MLE} by numerical optimization I got the values seen in Figure 2. They closely resemble the values I got by fitting a Poisson regression model using glm in (a). My implementation of the log of the poisson model with a normal prior can be found in Appendix A.

	Const	PowerSeller	VerifyID	Sealed	Minblem	MajBlem	LargNeg	LogBook	MinBidShare
1	1.070	-0.021	-0.393	0.444	-0.053	-0.221	0.071	-0.120	-1.892

Table 2: MLE of beta by numerical optimization

(c)

After having implemented the Metropolis Hastings simulation method and simulated 20000 draws from the posterior distribution from (b) I calculated the β coefficients as the mean of the draws, while omitting the first 10% of the draws because of the burn-in phase. These values can be seen in Table 3.

	Const	PowerSeller	VerifyID	Sealed	Minblem	MajBlem	LargNeg	LogBook	MinBidShare
1	1.069	-0.019	-0.389	0.441	-0.051	-0.223	0.070	-0.120	-1.892

Table 3: Mean values of betas drawn during Metropolis Hastings simulation

The convergence of the parameters can be seen in Figure 1, where I have taken the mean value of every two sequential beta drawn from the posterior and plotted it to visualize the auto-correlation between draws, and to see how the draws asymptotically approaches somewhat stationary values. My implementation of the Metropolis Hastings algorithm can be found in Appendix A.

Convergence of beta during Metropolis Hastings

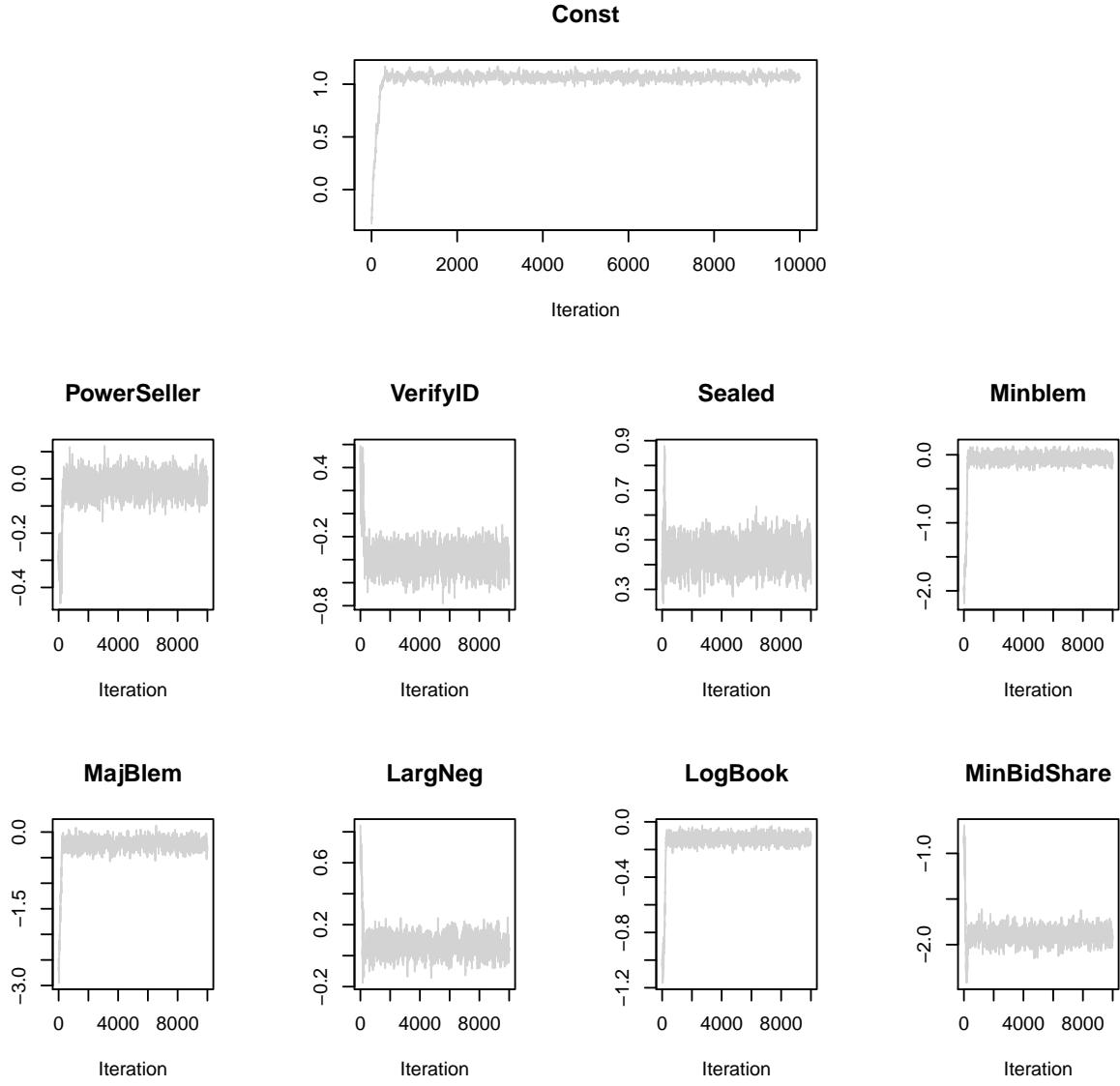


Figure 1: Convergence of betas drawn during Metropolis Hastings simulation

(d)

After having determined the predictive distribution of the sample \hat{x} as $p(\hat{y}|\lambda) \sim \text{Poisson}(\lambda)$, where $\lambda = e^{\hat{x}\beta}$ using the betas drawn during the simulation in (c), I got the distribution seen in Figure 2. The probability that the sample has zero bidders by this distribution was $p(\hat{y} = 0|\lambda) \approx 0.36$.

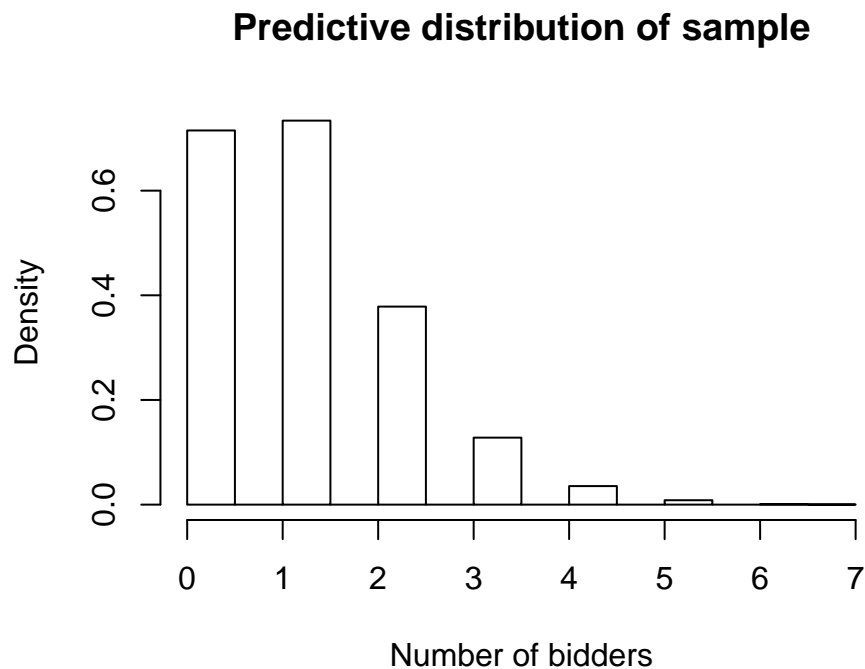


Figure 2: Predictive distribution of sample

Appendix A

Code for Lab 4

```
require(MASS)
require(geoR)
require(mvtnorm)
require(LaplacesDemon)

# -----
# Lab 4
# -----

data = read.table("data/eBayNumberOfBidderData.dat", header=TRUE)

n = length(data)
n_features = ncol(data) - 1 # Except y and const

feature_labels = colnames(data[,2:ncol(data)])

y = data$nBids
X = as.matrix(data[,2:ncol(data)])

X_X = t(X)%*%X

# -----
# (a)
# -----

glm_model = glm(nBids ~ 0 + ., data = data, family = poisson)

pdf("./plots/4_1_1_mle_beta.pdf", width=7, height=7)

par(oma = c(0, 0, 3, 0))
layout(matrix(c(0,1,1,0,2,3,4,5,6,7,8,9), 3, 4, byrow = TRUE))
for (i in 1:ncol(X)){
  mean = glm_model$coefficients[i]
  std_dev = summary(glm_model)[["coefficients"]][,2][i]
  x_grid = seq(mean-4*std_dev, mean+4*std_dev, 0.001)
  plot(x_grid,
       dnorm(x_grid, mean=mean, sd=std_dev),
       type="l",
       ylab="Density",
       xlab=expression(beta),
       main=feature_labels[i])
}
title("Normal approximation of MLE of beta", outer=TRUE, cex=1.5)

dev.off()

# -----
# (b)
```

```

# -----

# Beta prior (Zellner's g-prior)
mu0 = rep(0, n_features)
covar0 = 100 * ginv(X_X)
init_beta = mvrnorm(n=1, mu0, covar0)

# This is the log of the Poisson model
logPostPoiNorm <- function(betas, X, y){

  log_prior = dmvnorm(betas, mu0, covar0, log=TRUE)

  lambda = exp(X%*%betas)

  # Assume independence among samples and take the sum of
  # log(p(y_i/lambda)), where lambda is exp(X.dot(beta)) and p ~ Poisson
  log_lik = sum(dpois(y, lambda, log=TRUE))

  return (log_lik + log_prior)
}

log_post = logPostPoiNorm
opt_results = optim(init_beta,
                    log_post,
                    gr=NULL,
                    X,
                    y,
                    method=c("BFGS"),
                    control=list(fnscale=-1),
                    hessian=TRUE)

# MLE beta
post_mode = opt_results$par
# Covariance ( $J^{-1}(\text{beta hat})$ )
post_cov = -solve(opt_results$hessian)

# -----
# (c)
# -----

Sigma = post_cov
c = .5

n_draws = 20000

metropolisHastings = function(logPostFunc, theta, c, ...){
  theta_draws = matrix(0, n_draws, length(theta))
  # Set initial
  theta_c = mvrnorm(n=1, theta, c*Sigma)
  for(i in 1:n_draws){
    # 1: Draw new proposal theta
    theta_p = mvrnorm(n=1, theta_c, c*Sigma)

```

```

    # 2: Determine the acceptance probability
    p_prev = logPostFunc(theta_c, ...)
    p_new = logPostFunc(theta_p, ...)
    acc_prob = min(c(1, exp(p_new - p_prev)))
    # 3: Set new value with prob = acc_prob
    if(rbern(n=1, p=acc_prob)==1){
        theta_c = theta_p
    }
    theta_draws[i,] = theta_c
}
return (theta_draws)
}

init_beta = mvrnorm(n=1, mu0, covar0)
beta_draws = metropolisHastings(logPostPoiNorm, init_beta, c, X, y)

# Calculate mean of batches of 2 draws to visualize the
# auto correlation between sequential draws
mean_draws = matrix(0, n_draws/2, n_features)
for (i in seq(2,n_draws,2)){
    mean_draws[i/2,] = colMeans(beta_draws[c(i-1,i),])
}

# Avoid first 10% of the draws
burn_in = floor(n_draws / 10)
beta_draws = beta_draws[burn_in:nrow(beta_draws),]

beta_means = colMeans(beta_draws)

pdf("./plots/4_1_2_beta_conv.pdf", width=7, height=7)

par(oma = c(0, 0, 3, 0))
layout(matrix(c(0,1,1,0,2,3,4,5,6,7,8,9), 3, 4, byrow = TRUE))
x_grid = 1:nrow(mean_draws)
for (i in 1:ncol(X)){
    plot(x_grid,
         mean_draws[,i],
         type="l",
         ylab="",
         xlab="Iteration",
         col="lightgray",
         main=feature_labels[i])
}
title("Convergence of beta during Metropolis Hastings", outer=TRUE, cex=1.5)

dev.off()

# -----
# (d)
# -----

sample = c(

```

```

Constant = 1,
PowerSeller = 1,
VerifyID = 1,
Sealed = 1,
MinBlem = 0,
MajBlem = 0,
LargNeg = 0,
LogBook = 1,
MinBidShare = 0.5
)

lambda = exp(beta_draws%%sample)

pred_draws = rpois(10000, lambda)

# Probability that the sample has 0 bidders
prob = length(pred_draws[pred_draws == 0]) / length(pred_draws)

pdf("./plots/4_1_3_pred_distr.pdf", width=5, height=4)

# Plot the predictive distribution
plot(hist(pred_draws, right=FALSE, plot=FALSE),
      freq=FALSE,
      xaxt="n",
      xlab="Number of bidders",
      ylab="Density",
      main="Predictive distribution of sample")

axis(1,
      at=0:max(pred_draws),
      labels=0:max(pred_draws)
)

dev.off()

```