Lab4, Bayesian Learning

Hampus Hjelm Andersson

1)Obtain the maximum likelihood estimator of β in the Poisson regression model for the eBay data.

```
data <- read.table("ebaydata.txt",header=T)
x <- as.matrix(data[,3:10])
y <- as.vector(data[,1])
g <- glm(y~x,family=poisson)
summary(g)
```

```
##
## Call:
## glm(formula = y ~ x, family = poisson)
##
## 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 maximum likelihood estimator of β is thus

```
g$coeff
```

```
##  (Intercept) xPowerSeller    xVerifyID      xSealed     xMinblem
##   1.07244206  -0.02054076  -0.39451647   0.44384257  -0.05219829
##     xMajBlem     xLargNeg     xLogBook xMinBidShare
##  -0.22087119   0.07067246  -0.12067761  -1.89409664
```

and with a significance of 0.05 we can say that the covariates ,Intercept = β_1, xVerifyID = β_3, xSealed = β_4, xMajblem = β_6, xLogBook = β_8, xminiBidShare = β_9 are significant.

1b)Let's now do a Bayesian analysis of the Poisson regression.
First we need the log posterior, which is shown to be

$$\log p(\beta|y) = \log p(y|\beta)p(\beta) =$$
$$= \log\left(\exp(x_1'\beta)^{y_1}\exp(-\exp(x_1'\beta)) \cdot \ldots \exp(x_9'\beta)^{y_9}\exp(-\exp(x_9'\beta))\right) + \log(p(\beta)) =$$
$$= y_1x_1'\beta + \ldots y_9x_9'\beta - \exp(x_1'\beta) - \ldots \exp(x_9'\beta) + log(p(\beta)), \quad where\ \ p(\beta) \quad \text{is}\ \ N(0, 100 \cdot (X'X)^{-1})$$

And if we use this with optim we get

```
data <- read.table("ebaydata.txt",header=T);
x <- as.matrix(data[,2:10]);
y <- as.vector(data[,1]);
library("mvtnorm")
library("MASS")

#This is the Prior options
tau <- 10;
mu <- as.vector(rep(0,length(y)));
Sigma <- 100*ginv(t(x)%*%x);

#Here is the posterior for the poisson model
LogPostPoi <- function(betaVect,y,x,mu, Sigma){
        nPara <- length(betaVect);
        linPred <- x%*%betaVect;

        logLik <- sum( linPred*y -(exp(linPred)));#logposterior for poisson distribution
        if(abs(logLik) == Inf) logLik = -20000; # Likelihood is not finite, stear the optimizer away fr
        if(is.infinite(logLik)) logLik = -20000;
        #print(dim(Sigma))
        logPrior <- dmvnorm(betaVect, matrix(0,nPara,1), Sigma, log=TRUE);
        #print(logLik)
        return(logLik + logPrior)
}

initVal <- as.vector(solve(crossprod(x,x))%*%t(x)%*%y); # Initial values by OLS
OptimResults<-optim(initVal,LogPostPoi,gr=NULL,y,x,mu,Sigma,method=c("BFGS"),control=list(fnscale=-1),h

print("Optimal solution")
```

```
## [1] "Optimal solution"
```

```r
print(OptimResults$par)
```

```
## [1]  1.06984585 -0.02050285 -0.39299372  0.44352659 -0.05254904 -0.22152102
## [7]  0.07070538 -0.12013397 -1.89169910
```

1c)Program a general function that uses the Metropolis algorithm to generate random draws from an arbitrary posterior density. Let the proposal density be the multivariate normal density mentioned in Lecture 8 (random walk Metropolis)

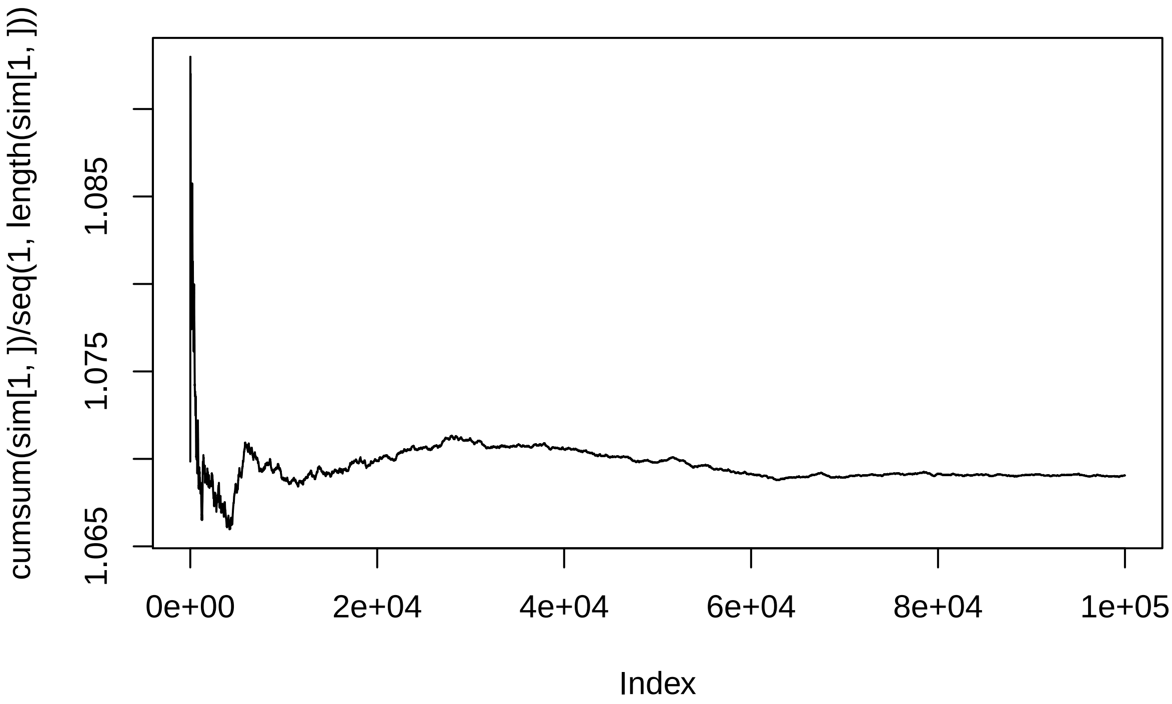
```r
start <- OptimResults$par
NegHesInv <- -solve(OptimResults$hessian)
c <-0.5
#choosen so that the average acceptance probability is around 30 %
Niter <- 100000


RWM <- function(LogPostFunc,start,c,NegHesInv,Niter,...){
        xacc <- start
        sim <- matrix(0,nrow=length(start),ncol = Niter)
        rejects <- 0
        for(i in 1:Niter){
                xprop <- as.vector(t(rmvnorm(1,xacc,sigma= c*NegHesInv)))

                if(runif(1) < min(1,exp(LogPostFunc(xprop,...)-LogPostFunc(xacc,...)))){
                        xacc <- xprop

                }
                else{rejects <- rejects + 1}
                sim[,i] <- xacc
                #print(i)
        }
        print("Rejects")
        print(rejects)
        return(sim)
}
sim <- RWM(LogPostPoi,start,c,NegHesInv,Niter,y,x,mu, Sigma)
```

```
## [1] "Rejects"
## [1] 68344
```

```r
print("Convergence of beta1")
```

```
## [1] "Convergence of beta1"
```

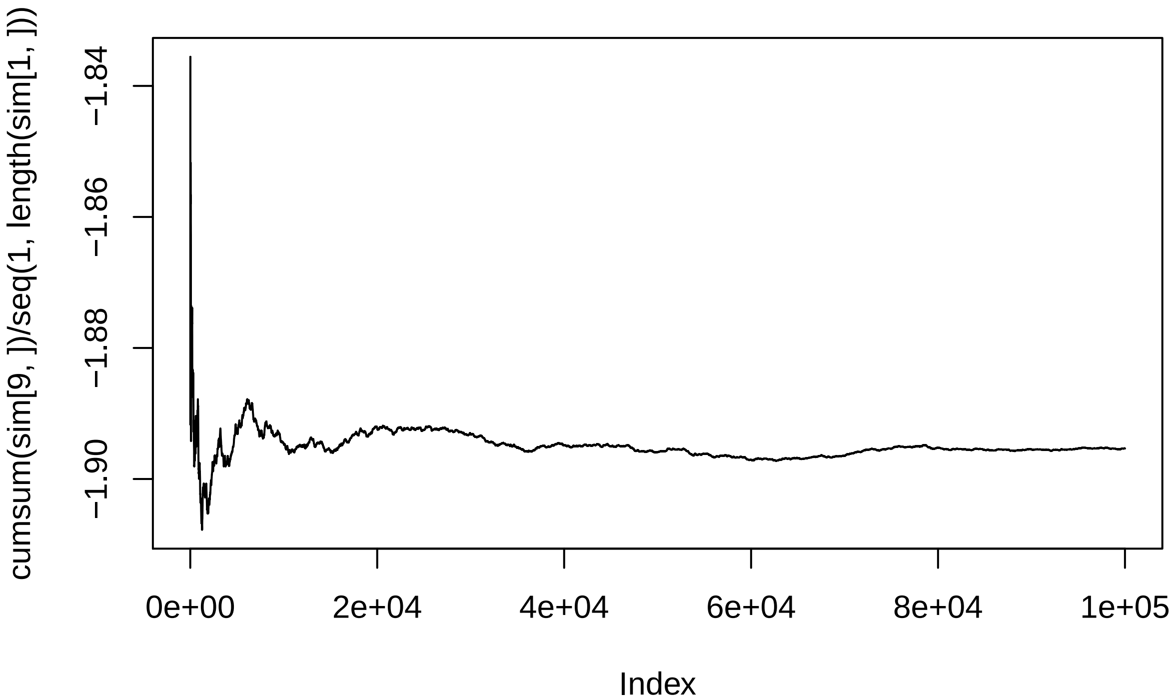
```r
plot(cumsum(sim[1,])/seq(1,length(sim[1,])),xlim=c(0,100000),type = 'l')
```




```r
print("convergence of beta9")
```

```
## [1] "convergence of beta9"
```

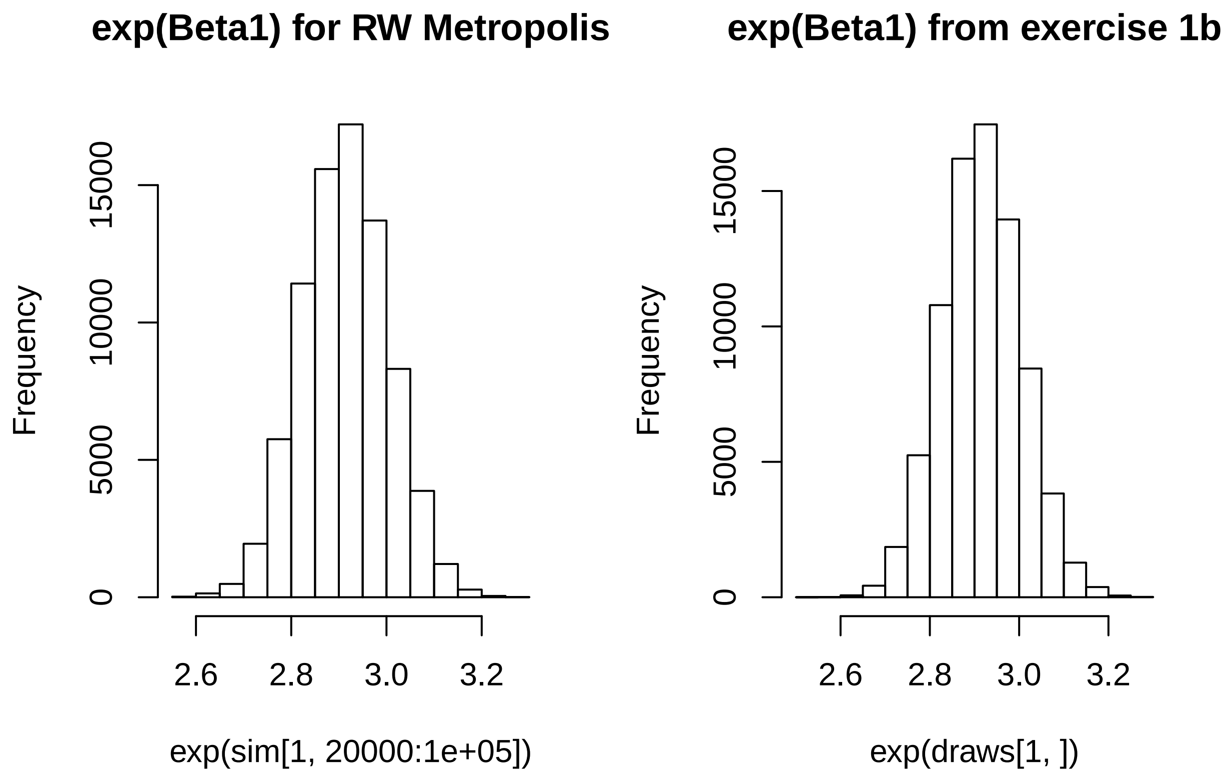
```r
plot(cumsum(sim[9,])/seq(1,length(sim[1,])),xlim=c(0,100000),type = 'l')
```



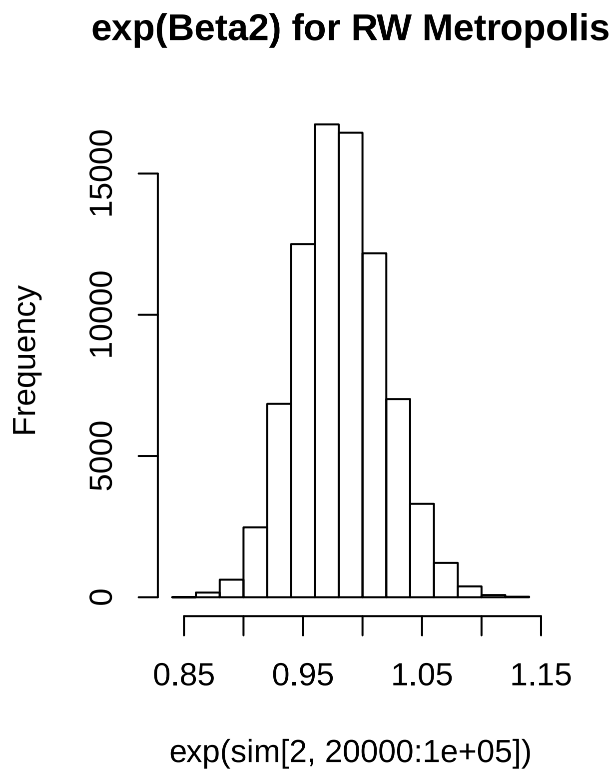
All the beta's appear to have converged after approximately 20000 iterations.

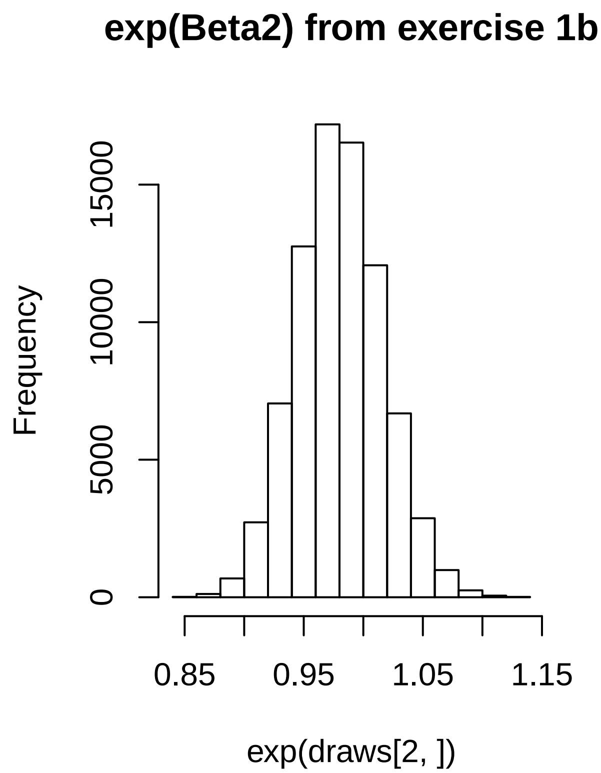
The parameters $\alpha_j = \exp(\beta_j)$ are usually considered more interpretable than the β_j. Compute the posterior distribution of $\alpha_j$for all variables.

```
draws <- matrix(0,nrow=9,ncol=80000)
for(i in 1:80000){draws[,i]<- rmvnorm(1,OptimResults$par,NegHesInv)}
par(mfrow=c(1,2))
hist(exp(sim[1,20000:100000]),main="exp(Beta1) for RW Metropolis")
hist(exp(draws[1,]),main="exp(Beta1) from exercise 1b")
```

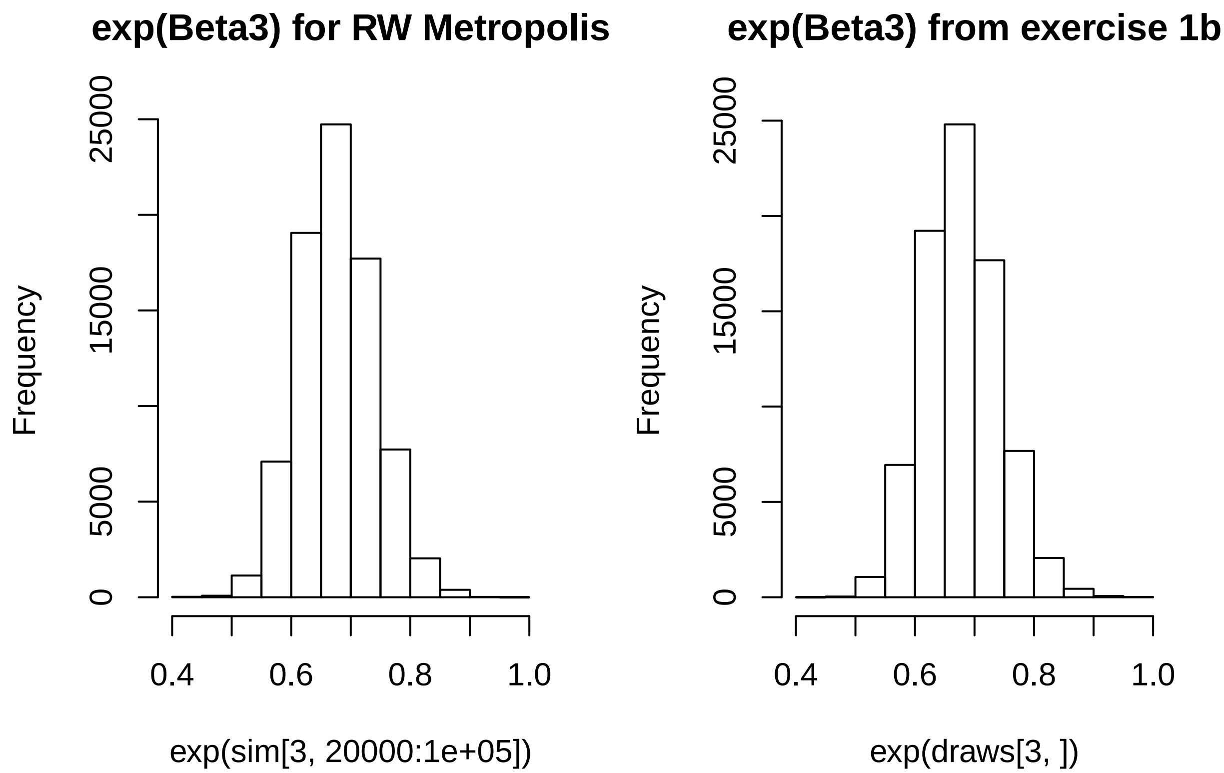



```r
hist(exp(sim[2,20000:100000]),main="exp(Beta2) for RW Metropolis")
hist(exp(draws[2,]),main="exp(Beta2) from exercise 1b")
```

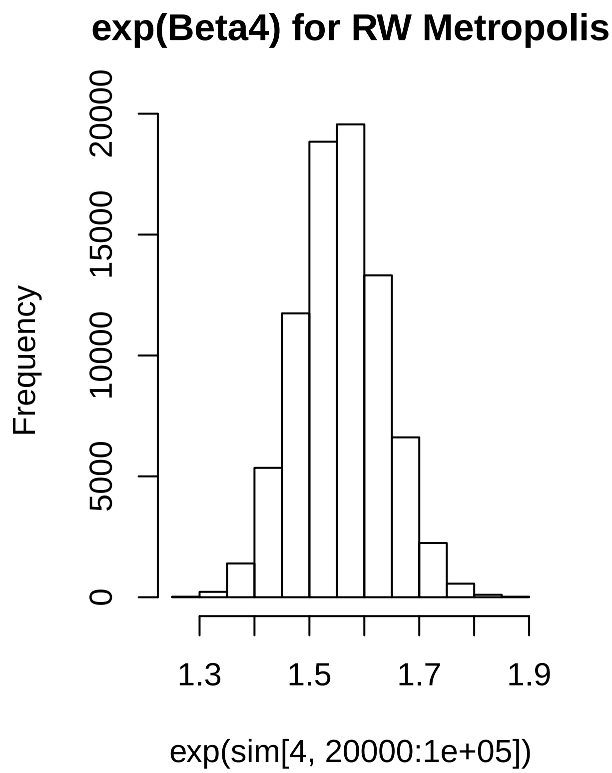


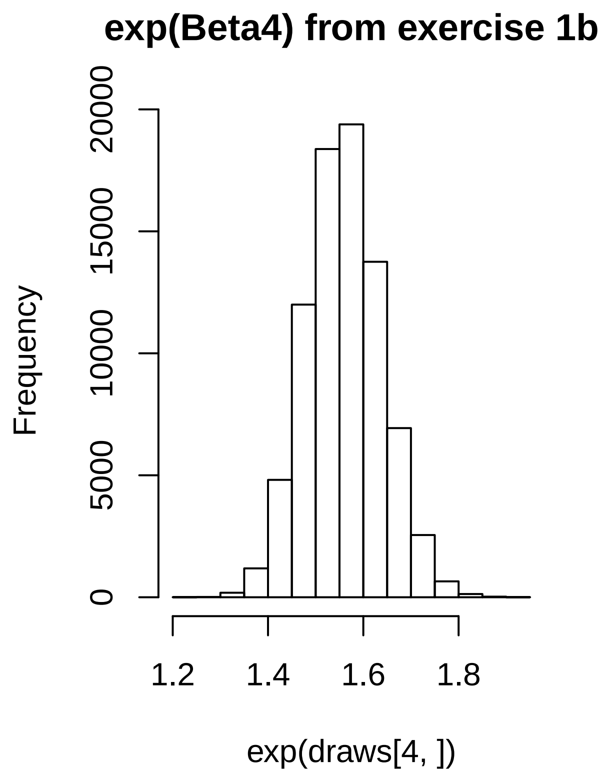



```
par(mfrow=c(1,2))
hist(exp(sim[3,20000:100000]),main="exp(Beta3) for RW Metropolis")
hist(exp(draws[3,]),main="exp(Beta3) from exercise 1b")
```

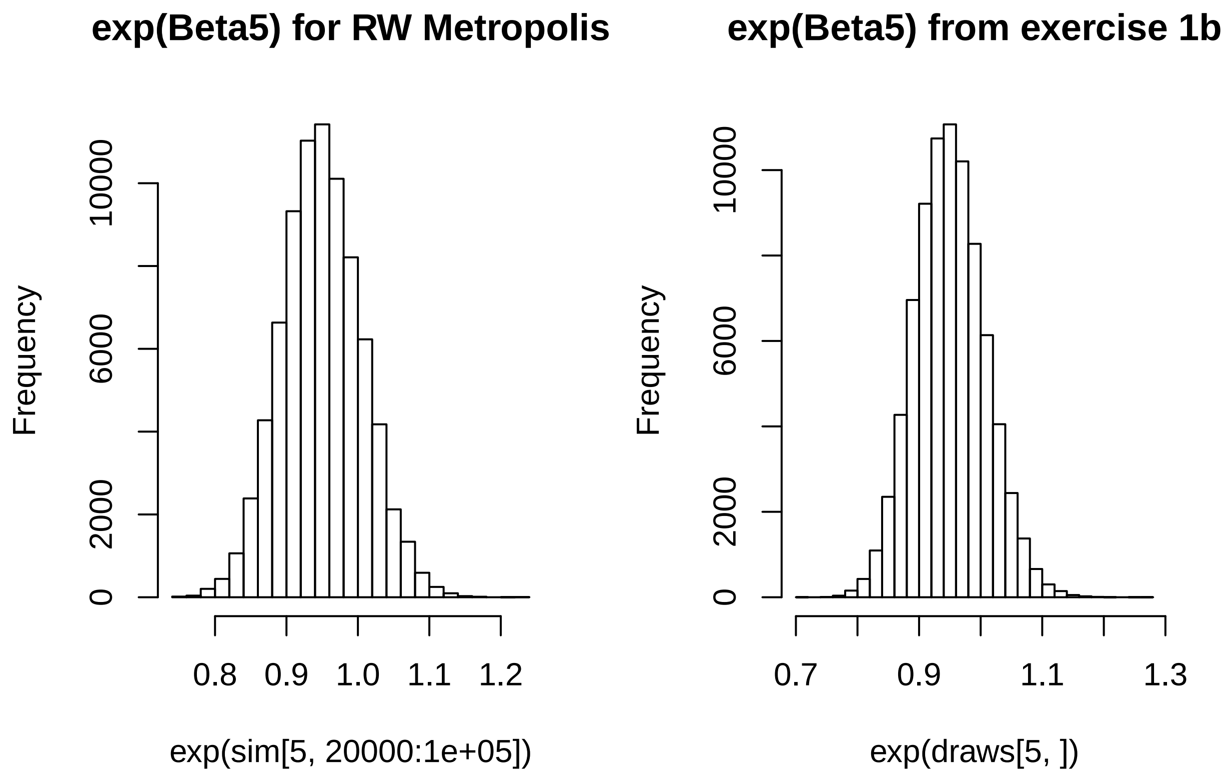



```r
hist(exp(sim[4,20000:100000]),main="exp(Beta4) for RW Metropolis")
hist(exp(draws[4,]),main="exp(Beta4) from exercise 1b")
```

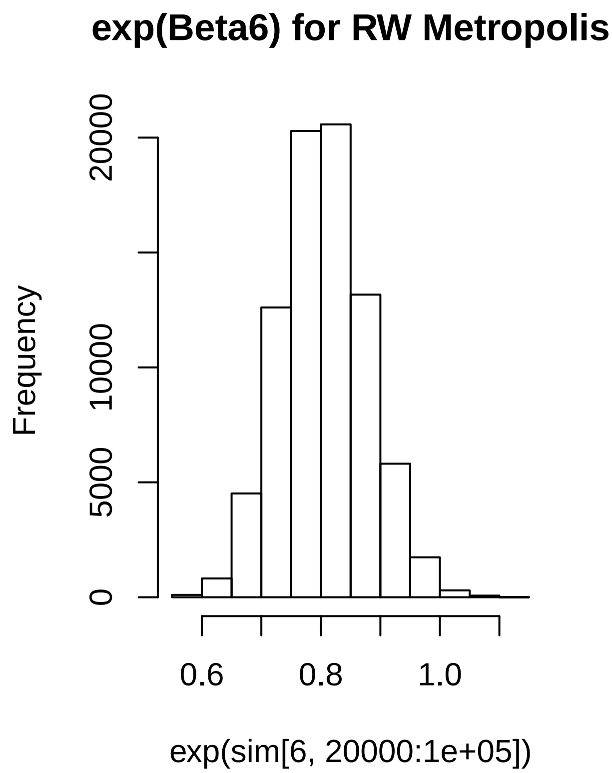


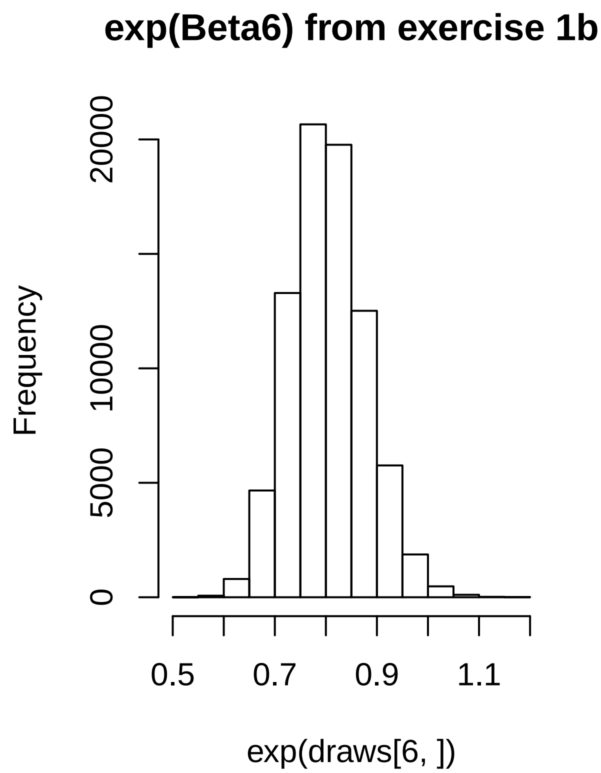



```
hist(exp(sim[5,20000:100000]),main="exp(Beta5) for RW Metropolis",breaks=20)
hist(exp(draws[5,]),main="exp(Beta5) from exercise 1b",breaks=20)
```

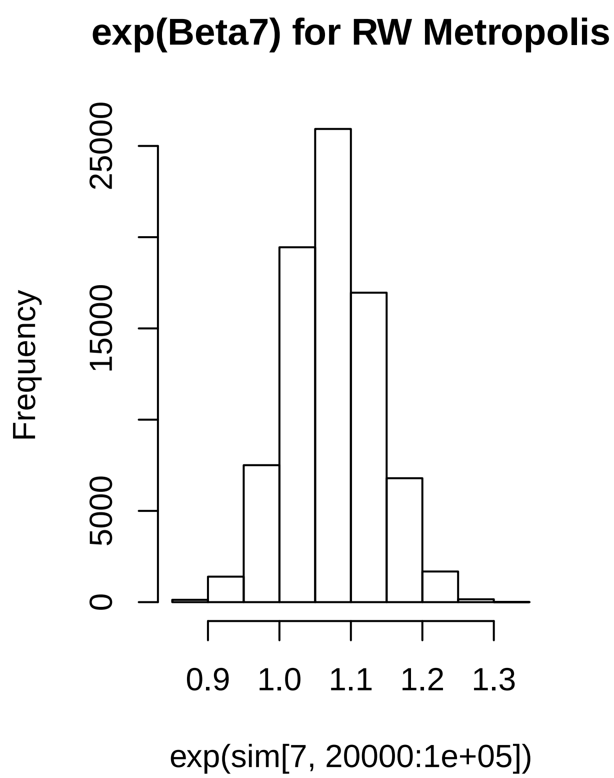


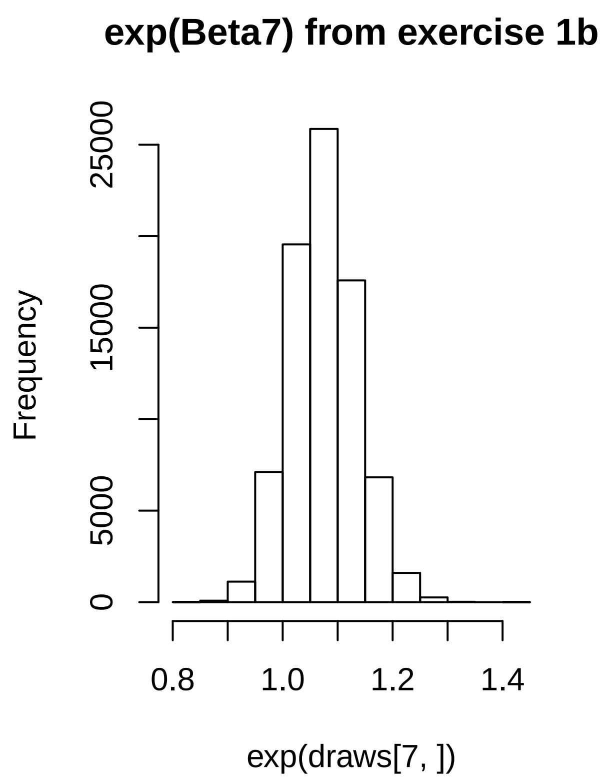

```r
hist(exp(sim[6,20000:100000]),main="exp(Beta6) for RW Metropolis")
hist(exp(draws[6,]),main="exp(Beta6) from exercise 1b")
```



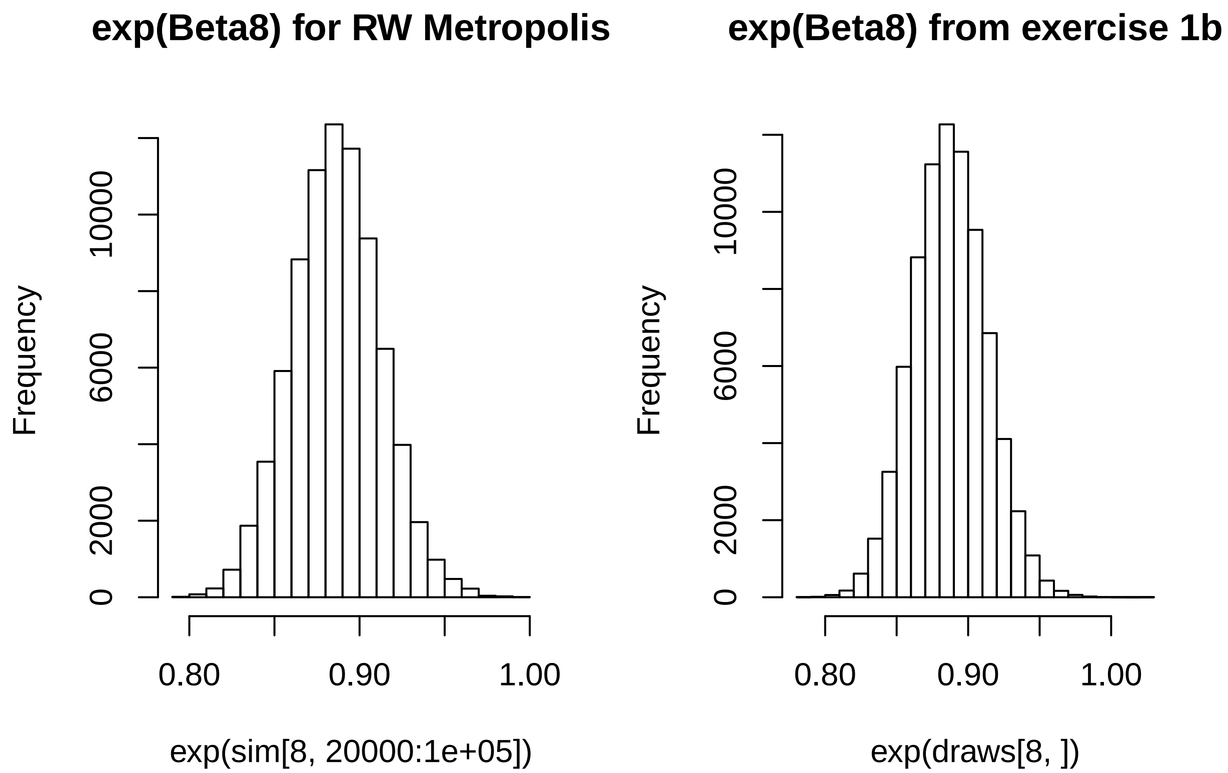



```r
hist(exp(sim[7,20000:100000]),main="exp(Beta7) for RW Metropolis",breaks=15)
hist(exp(draws[7,]),main="exp(Beta7) from exercise 1b",breaks=15)
```

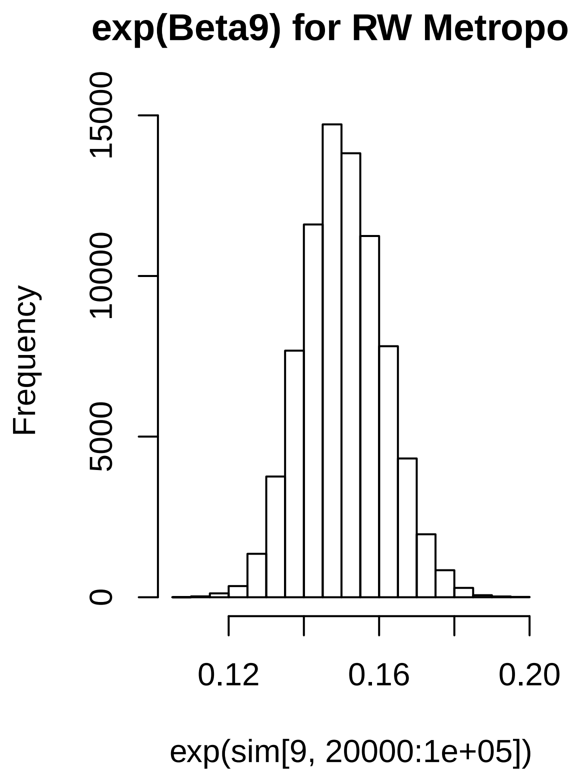


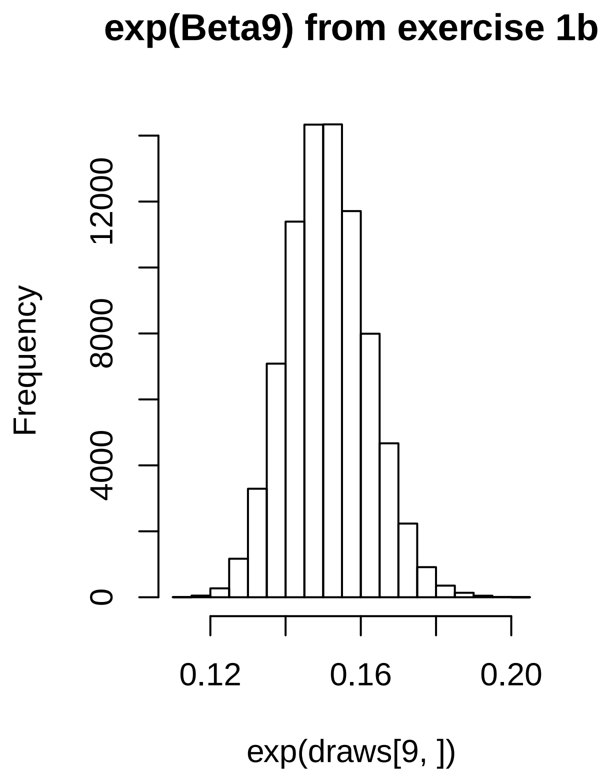



```r
hist(exp(sim[8,20000:100000]),main="exp(Beta8) for RW Metropolis")
hist(exp(draws[8,]),main="exp(Beta8) from exercise 1b")
```




```r
hist(exp(sim[9,20000:100000]),main="exp(Beta9) for RW Metropolis")
hist(exp(draws[9,]),main="exp(Beta9) from exercise 1b")
```





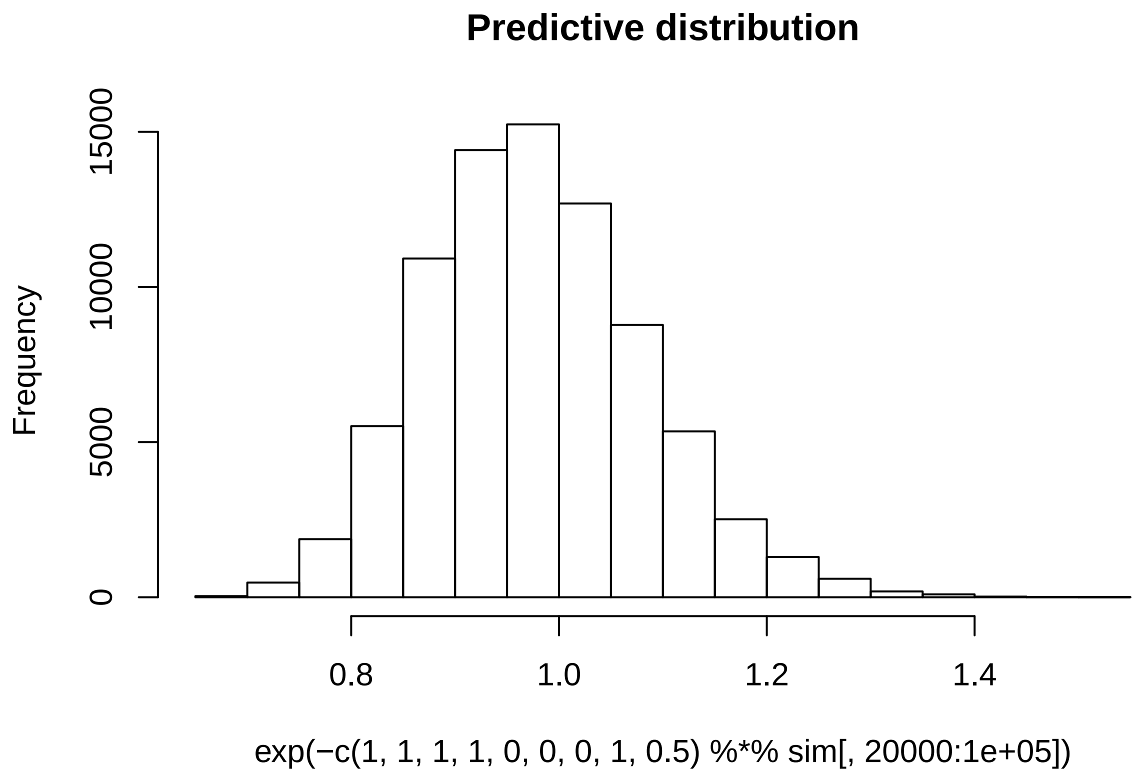
1d)Use the MCMC draws from c) to simulate from the predictive distribution of the number of bidders in a new auction.Plot the predictive distribution. What is the probability of no bidders in this new auction? Probability of no bidders in this auction is

```
mean(ppois(0,exp(-c(1,1,1,1,0,0,0,1,0.5)%*%sim[,20000:100000])))
```

```
## [1] 0.3778378
```

The predictive distribution for this auction is

```
hist(exp(-c(1,1,1,1,0,0,0,1,0.5)%*%sim[,20000:100000]),main="Predictive distribution")
```


Predictive distribution
Frequency
0
5000
10000
15000
0.8
1.0
1.2
1.4
exp(−c(1, 1, 1, 1, 0, 0, 0, 1, 0.5) %*% sim[, 20000:1e+05])