

MATH 598 - PROJECT 2 : SOLUTIONS

1. In this problem, we have a logistic regression model which can be treated in the same way as the Poisson example from earlier analysis. The conditional expectation of a response variable Y_i is modelled as a function of covariates $\mathbf{x}_i = (1, x_{i1})$, specifically

$$\mathbb{E}[Y_i | \mathbf{x}_i; \theta] = n_i g(\mathbf{x}_i \theta) = n_i g(\beta_0 + \beta_1 x_i) = n_i p(x_i, \beta)$$

say inverse link function $g(x) = \text{expit}(x) = e^x / (1 + e^x)$, and where the conditional distribution of Y_i is $\text{Binomial}(n_i, p(x_i, \beta))$. Then the likelihood function is

$$\mathcal{L}_n(\beta) = \prod_{i=1}^K f_{Y|X}(y_i | \mathbf{x}_i; \beta) = \prod_{i=1}^K \binom{n_i}{y_i} p(x_i, \beta)^{y_i} (1 - p(x_i, \beta))^{n_i - y_i}$$

so that the log likelihood is

$$\begin{aligned} \ell_n(\beta) &= \text{const.} + \sum_{i=1}^K \exp \{y_i(\beta_0 + \beta_1 x_i) - n_i \log(1 + \exp\{\beta_0 + \beta_1 x_i\})\} \\ &= \text{const.} + \sum_{i=1}^K \exp \{y_i \mathbf{x}_i \beta - n_i \log(1 + \exp\{\mathbf{x}_i \beta\})\} \end{aligned}$$

where $\mathbf{x}_i = (1, x_i)$. To find the maximum likelihood estimate (mle), we differentiate with respect to β and equate the result to zero: we have

$$\dot{\ell}_n(\beta) = \sum_{i=1}^K (y_i - n_i \text{expit}\{\mathbf{x}_i \beta\}) \mathbf{x}_i^\top = \sum_{i=1}^K (y_i - \mu_i) \mathbf{x}_i^\top$$

say. The second derivative $\ddot{\ell}_n(\beta)$ evaluated at the mle $\hat{\beta}_n$ yields the curvature of the log-likelihood at the mle:

$$\ddot{\ell}_n(\beta) = - \sum_{i=1}^K n_i \text{expit}\{\mathbf{x}_i \beta\} (1 - \text{expit}\{\mathbf{x}_i \beta\}) \mathbf{x}_i^\top \mathbf{x}_i = - \sum_{i=1}^K n_i \mu_i (1 - \mu_i) \mathbf{x}_i^\top \mathbf{x}_i = -\mathbf{X}^\top \mathbf{D}_n \mathbf{X}$$

say, where

$$\mathbf{X} = \begin{pmatrix} \mathbf{x}_1 \\ \mathbf{x}_2 \\ \vdots \\ \mathbf{x}_K \end{pmatrix} \quad \mathbf{D}_n = \mathbf{D}_n(\beta) = \text{diag}(n_1 \mu_1 (1 - \mu_1), \dots, n_K \mu_K (1 - \mu_K)).$$

Using the quadratic approximation to the log-likelihood, we see that for large n the posterior distribution is approximately Normal

$$\pi_n(\beta) \approx \text{Normal} \left(\hat{\beta}_n, (\mathbf{X}^\top \hat{\mathbf{D}}_n \mathbf{X})^{-1} \right)$$

and $\hat{\mathbf{D}}_n = \mathbf{D}_n(\hat{\beta}_n)$. The mle and variance matrices can be computed using `glm` in R, or directly using `optim`.

```
#Data
nvec<-c(20,23,23,23,17,24,19,18)
y<-c(16,17,20,19,4,7,0,3)
x<-c(0.32,1.25,1.62,2.54,4.95,6.46,8.85,8.98)

#Could use GLM
fit1<-glm(cbind(y,nvec-y)~x,family=binomial)

#Could use optim
expit<-function(x){return(1/(1+exp(-x)))}
log.like<-function(be,yv,nv,xv){
```

```

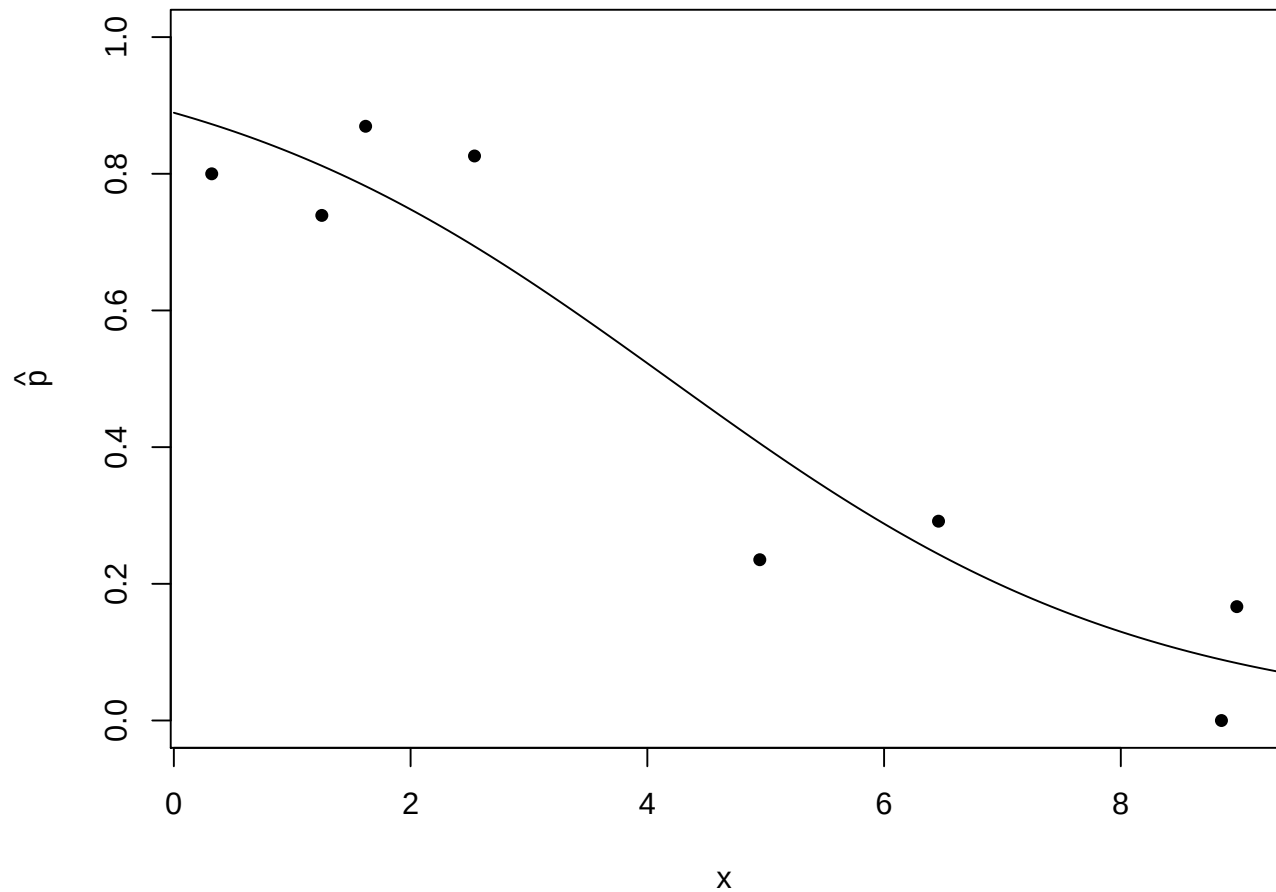
    pv<-expit(be[1]+xv*be[2])
    return(sum(dbinom(yv,nv,pv,log=T)))
}

be.start<-c(0,0)
opt.fit<-optim(be.start,fn=log.like,yv=y,nv=nvec,xv=x,control=list(fnscale=-1),hessian=T)

be.hat<-opt.fit$par
be.var<-solve(-opt.fit$hessian)

xv<-seq(0,10,by=0.01)
etav<-be.hat[1]+be.hat[2]*xv
muv<-expit(etav)
par(mar=c(4,5,1,0))
plot(x,y/nvec,pch=19,cex=0.8,xlab='x',ylim=range(0,1),ylab=expression(hat(p)))
lines(xv,muv)

```



The maximum likelihood fit seems to be reasonable. We may now examine the likelihood surface.

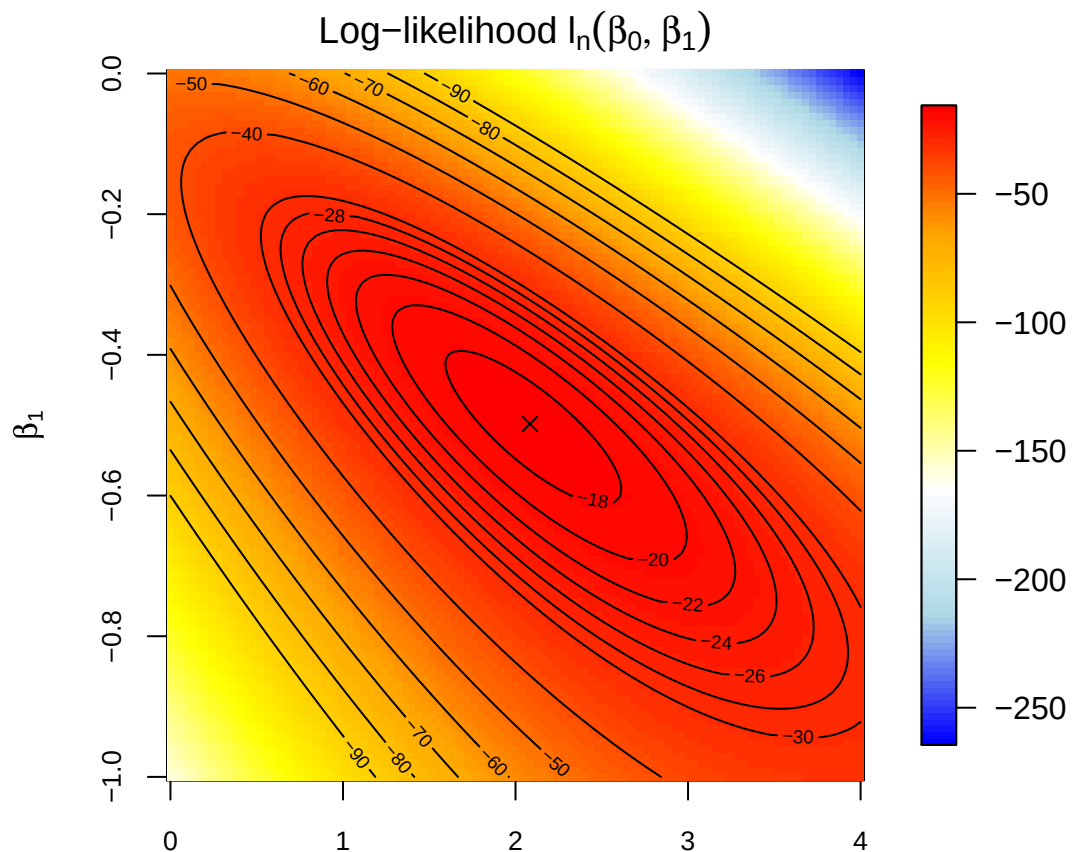
```

#Exact log likelihood
log.like2<-function(b0,b1,yv,nv,xv){
  pv<-expit(b0+xv*b1)
  return(sum(dbinom(yv,nv,pv,log=T)))
}
f <- Vectorize(log.like2,vectorize.args=c("b0","b1"))
be0<-seq(0.0,4.0,length=101)
be1<-seq(-1,0,length=101)
lmat<-outer(be0,be1,f,yv=y,nv=nvec,xv=x)
lmax<-max(lmat)

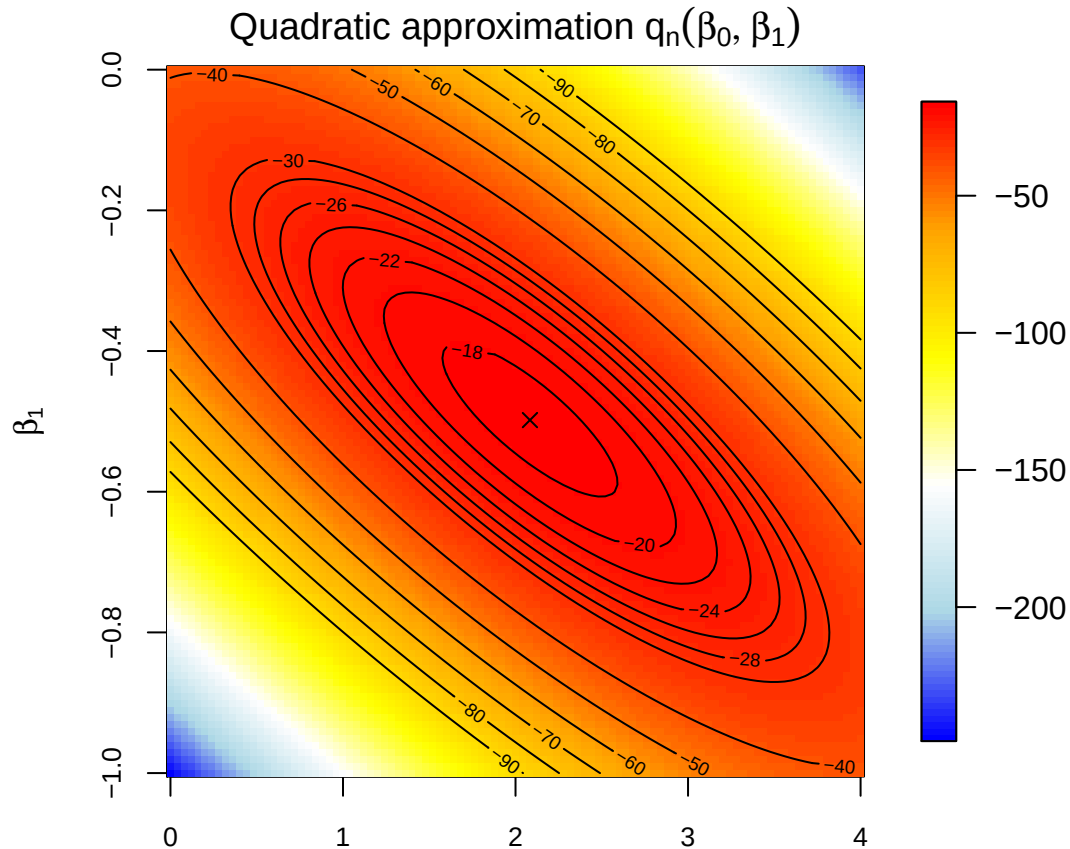
#Quadratic approximation
quad.func<-function(b0,b1,m,M){
  tv<-c(b0,b1)-m
  q<--0.5*t(tv) %*% solve(M) %*% tv
  return(q[1,1])
}
f <- Vectorize(quad.func,vectorize.args=c("b0","b1"))
qmat<-outer(be0,be1,f,m=be.hat,M=be.var)+lmax

#Plot
library(fields,quietly=TRUE)
par(pty='s',mar=c(2,3,2,2))
colfunc <- colorRampPalette(c("blue","lightblue","white","yellow","orange","red"))
image.plot(be0,be1,lmat,col=colfunc(100),
           xlab=expression(beta[0]),ylab=expression(beta[1]),cex.axis=0.8)
contour(be0,be1,lmat,add=T,levels=c(seq(-90,-40,by=10),seq(-30,-10,by=2)))
title(expression(paste('Log-likelihood ',l[n](beta[0],beta[1]))))
points(be.hat[1],be.hat[2],pch=4)

```



```
image.plot(be0,be1,qmat,col=colfunc(100),
           xlab=expression(beta[0]),ylab=expression(beta[1]),cex.axis=0.8)
contour(be0,be1,qmat,add=T,levels=c(seq(-90,-40,by=10),seq(-30,-10,by=2)))
title(expression(paste('Quadratic approximation ',q[n](beta[0],beta[1]))))
points(be.hat[1],be.hat[2],pch=4)
```



The quadratic approximation seems quite good, even though the data are limited in number.

The posterior distribution is obtained in the usual way as

$$\pi_n(\beta) \propto \mathcal{L}_n(\beta)\pi_0(\beta)$$

where we assume

$$\pi_0(\beta) \equiv \text{Normal}_2(\mathbf{0}, \mathbf{L}^{-1})$$

where $\mathbf{L}$ is diagonal with all entries 25. We perform rejection sampling using a *Student*(5) proposal mechanism centered at the posterior mode, and using the inverse Hessian of the posterior evaluated at the mode for the covariance structure. We optimize using `optim`, and then sample using `rmvt` from the `mvnfast` library.

```
#Prior Normal(0,solve(L))
L<-diag(rep(1/25,2))

#Rejection sampling
library(mvnfast)
log.post<-function(be,yv,nv,xv,Lm){
  pv<-expit(be[1]+xv*be[2])
  llike<-sum(dbinom(yv,nv,pv,log=T))
  lprior<-sum(dmvn(be,rep(0,2),solve(Lm),log=T))
  return(llike+lprior)
}
be.start<-c(0,0)
```

```

opt.post<-optim(be.start,fn=log.post,yv=y,nv=nvec,xv=x,Lm=L,control=list(fnscale=-1),hessian=T)

post.mode<-opt.post$par
post.var<-solve(-opt.post$hessian)

rejection.func<-function(be,yv,nv,xv,Lm,muv,Sigv){
  pv<-expit(be[1]+xv*be[2])
  llike<-sum(dbinom(yv,nv,pv,log=T))
  lprior<-sum(dmvn(be,rep(0,2),solve(Lm),log=T))
  lprop<-sum(dmvn(be,muv,Sigv,5,log=T))
  return(llike+lprior-lprop)
}

be.start<-c(0,0)
opt.rej<-optim(be.start,fn=rejection.func,yv=y,nv=nvec,xv=x,Lm=L,
  muv=post.mode,Sigv=post.var,
  control=list(fnscale=-1),hessian=T)

Mval<-opt.rej$value
N<-100000
nsamp<-0
ico<-0
be.rej<-matrix(0,nrow=N,ncol=2)
while(nsamp < N){
  ico<-ico+1
  be<-rmvt(1,post.mode,post.var,5)
  pv<-expit(be[1]+x*be[2])
  llike<-sum(dbinom(y,nvec,pv,log=T))
  lprior<-sum(dmvn(be,rep(0,2),solve(L),log=T))
  lprop<-sum(dmvn(be,post.mode,post.var,5,log=T))

  U<-runif(1)
  if(log(U) < llike+lprior-lprop-Mval){
    nsamp<-nsamp+1
    be.rej[nsamp,]<-be
  }
}

```

We can now compute the acceptance rate, and plot the samples.

```

N/ico  #Acceptance rate

+ [1] 0.7246797

#####
#Exact log likelihood
log.post2<-function(b0,b1,yv,nv,xv,Lm){
  pv<-expit(b0+xv*b1)
  llike<-sum(dbinom(yv,nv,pv,log=T))
  lprior<-sum(dmvn(be,rep(0,2),solve(Lm),log=T))
  return(llike+lprior)
}

f <- Vectorize(log.post2,vectorize.args=c("b0","b1"))
be0<-seq(0.0,4.0,length=101)
be1<-seq(-1,0,length=101)

pmat<-outer(be0,be1,f,yv=y,nv=nvec,xv=x,Lm=L)

par(pty='s',mar=c(2,3,2,2))

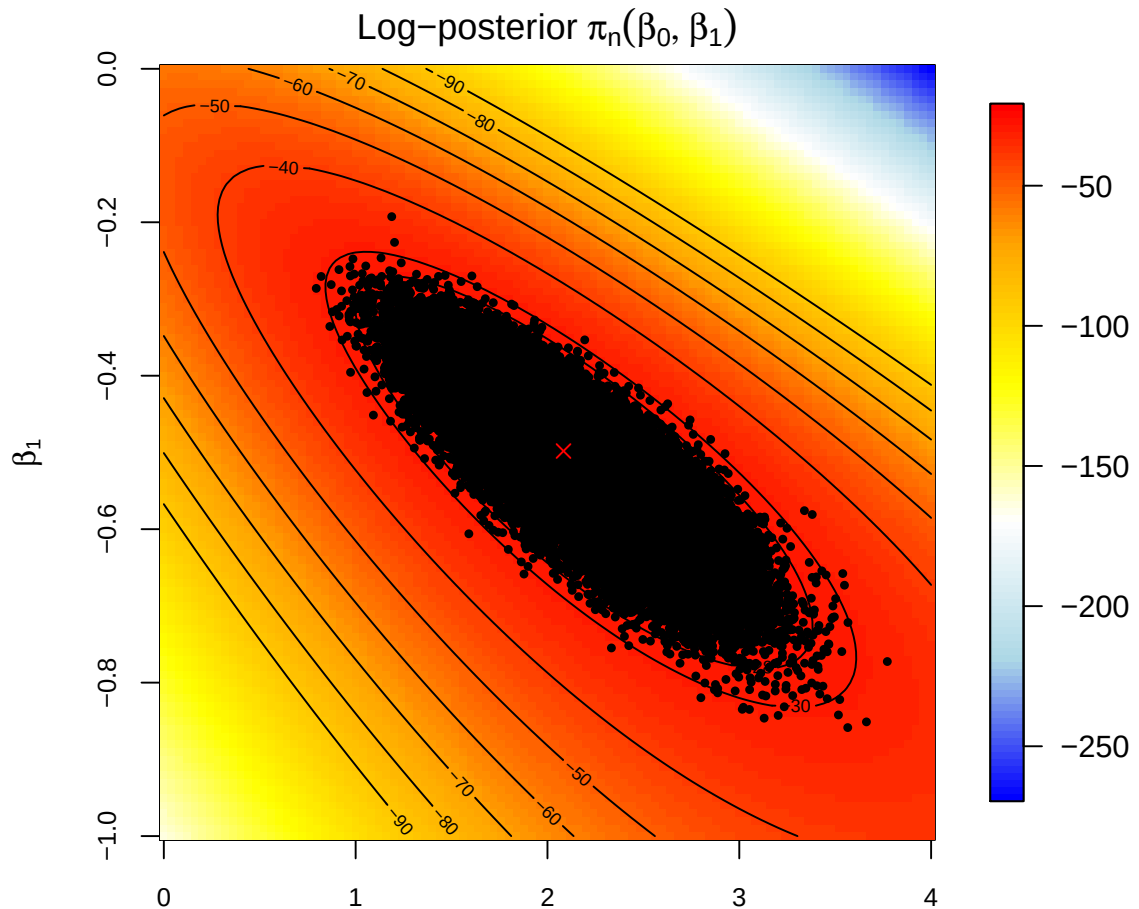
```

```

image.plot(be0,be1,pmat,col=colfunc(100),
           xlab=expression(beta[0]),ylab=expression(beta[1]),cex.axis=0.8)
contour(be0,be1,pmat,add=T,levels=c(seq(-90,-40,by=10),seq(-30,-10,by=2)))
title(expression(paste('Log-posterior ',pi[n](beta[0],beta[1]))))
points(be.rej,pch=19,cex=0.5)

points(be.hat[1],be.hat[2],pch=4,col='red')

```

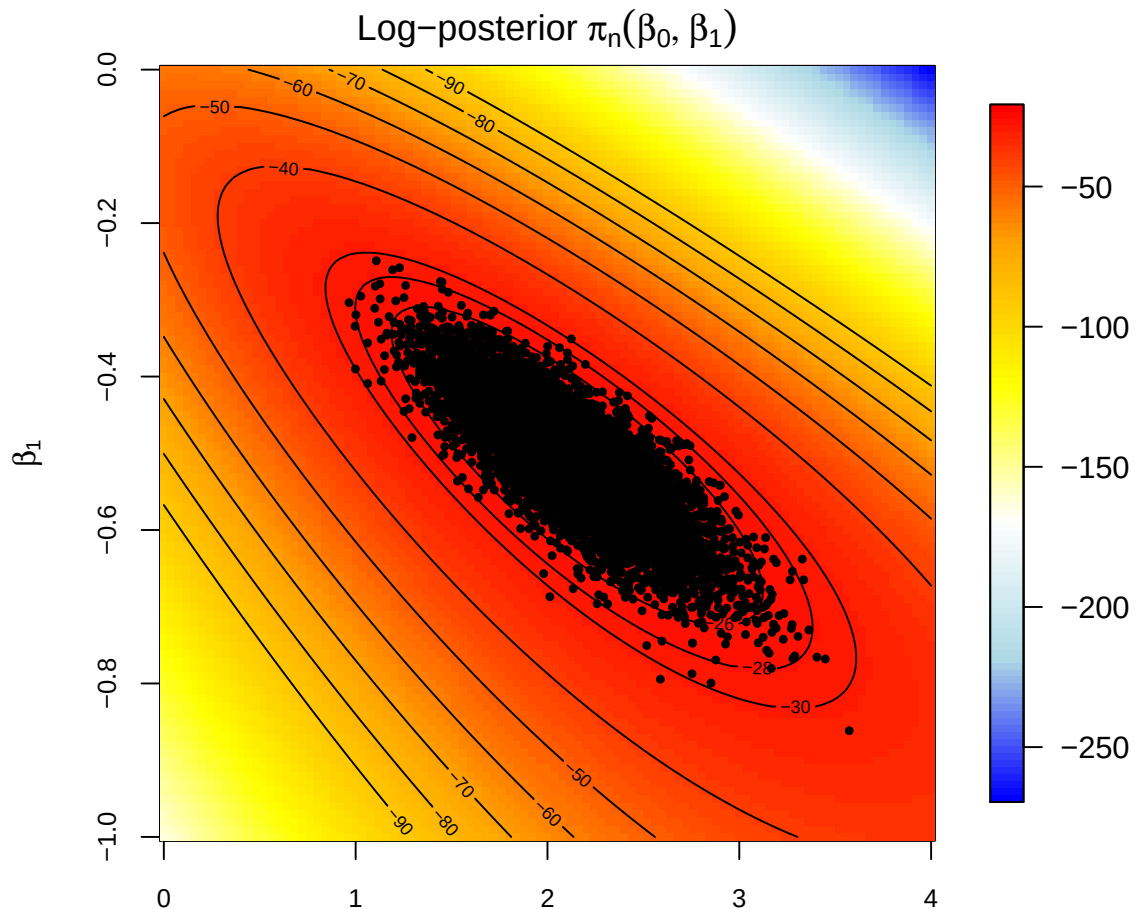


The acceptance rate is 0.724680, which seems quite reasonable. It may be that the acceptance rate can be improved slightly by choosing a different degrees of freedom, or using a different proposal distribution.

20 MARKS

2. The formulation of SIR is as given in the code in the Assessment sheet, using the same *Student*(5) distribution as for the rejection sampling above.

```
set.seed(94)
N0<-50000; N<-10000
be.samp0<-rmvt(N0,post.mode,post.var,5)
lprop<-dmvt(be.samp0,post.mode,post.var,5,log=T)
lpost<-rep(0,N0)
for(i in 1:N0){
  pv<-expit(be.samp0[i,1]+x*be.samp0[i,2])
  llike<-sum(dbinom(y,nvec,pv,log=T))
  lprior<-sum(dmvn(be.samp0[i,],rep(0,2),solve(L),log=T))
  lpost[i]<-llike+lprior
}
w<-exp(lpost-lprop);w<-w/sum(w)
be.sir<-be.samp0[sample(1:N0,prob=w,rep=T,size=N),]
par(pty='s',mar=c(2,3,2,2))
image.plot(be0,be1,pmat,col=colfunc(100),
  xlab=expression(beta[0]),ylab=expression(beta[1]),cex.axis=0.8)
contour(be0,be1,pmat,add=T,levels=c(seq(-90,-40,by=10),seq(-30,-10,by=2)))
title(expression(paste('Log-posterior ',pi[n](beta[0],beta[1]))))
points(be.sir,pch=19,cex=0.5)
```



```
1/sum(w^2) #Effective sample size (out of N=50000)
+ [1] 45868.99
```

10 MARKS