Homework6take2

Robert Winter

Table of Contents

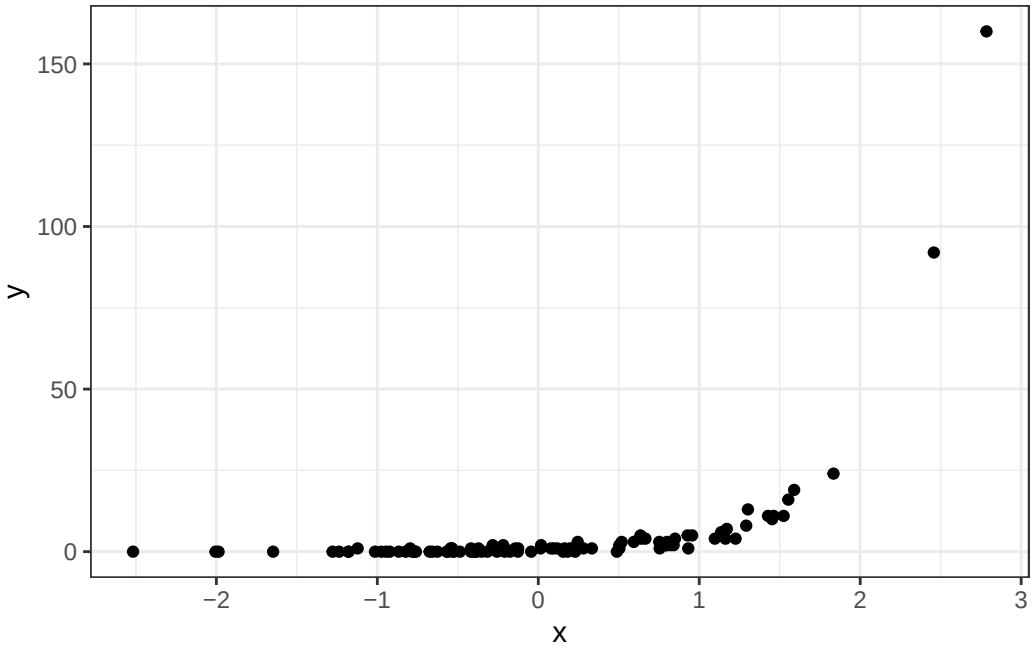
Based on the data `log.txt`, perform a model selection based on RJMCMC. The models under consideration are

$$m = 1 : \log\left(\lambda(x)\right) = \beta_0 + \beta_1 x$$
$$m = 2 : \log\left(\lambda(x)\right) = \beta_0 + \beta_1 x + \beta_2 x^2.$$

In the data file, the first column is x, a continuous covariate, and the second column is y, the Poisson outcome where $y \sim \text{Pois}(\lambda(x))$.

Consider parameter space $\{m = 1, \beta_0, \beta_1\}$ and $\{m = 2, \beta_0, \beta_1, \beta_2\}$. Use RJMCMC to estimate the posterior of m, β_0, β_1, and β_2. Draw the trace plots for them and make a conclusion if $m = 1$ or $m = 2$ is a better model by comparing their marginal posterior probabilities.

We begin by plotting the data in the x-y plane to get our bearings:



Now, we use RJMCMC simulation to estimate the posterior distributions of $m, \beta_0, \beta_1, \beta_2$. I have adapted Prof. Yuan Ji's sample code for this dataset, and to account for the fact that we are running Poisson regression rather than linear regression.

```
## Initialize parameter values
n.mc <- 100000
m <- rep(1, n.mc)

h <- matrix(rep(0.5, 4), ncol=2, nrow=2)  ## the probability of
 ↪  transition between model m=1 and model m=2; with probability 0.5
 ↪  the model can go from 1->1, 1->2, 2->1, and 2->2. Note that it must
 ↪  be true that h(1,1) + h(1,2) = h(2,1) + h(2,2) = 1, since from
 ↪  model 1 (or 2), the algorithm only allows either staying at model 1
 ↪  (or 2) or jump to model 2 (or 1), respectively.

beta01 <- matrix(0, ncol=2, nrow=n.mc)
beta2 <- rep(0, n.mc)

### Set up prior parameters: mu, beta1 ~ N(0, sig.beta), beta2 | m=2 ~
 ↪  N(0, sig.beta),
sig.u <- sqrt(1)
sig.beta <- sqrt(10)

## Assume prior P(m=1) = pi.M = 0.5.
pi.M <- 0.5
```

```r
#### tau is the standard deviation of proposal density for mu, beta1,
 ↪  and beta2
tau <- sqrt(0.05)


m[1] <- 1
beta01[1,] <- c(1,1)
##beta2[1] <- 0.5
```

```r
### Function samp01 is to sample the reduced model with mu and beta1
 ↪  Here, b01 = c(mu, beta1) ###
samp01 <- function(b01){

    curr <- b01
    epi <- rnorm(2, 0, sd=tau)
    prop <- curr + epi


    like.ratio <- - sum(log(dpois(y, lambda=exp(curr[1]+curr[2]*x1))))
 ↪  + sum(log(dpois(y, lambda=exp(prop[1]+prop[2]*x1))))

    prior.ratio <- - sum(log(dnorm(curr, 0, sig.beta))) +
 ↪  sum(log(dnorm(prop, 0, sig.beta)))

    acc <- exp(like.ratio + prior.ratio)

    #cat("samp01 acc", acc, "\n"); #readline()

    ind <- (acc > runif(1))

    res <- prop * ind + curr * (1-ind)
    ##cat(res)

    return(res)
}
```

```r
### Function samp012 is to sample the full model with mu, beta1, beta2.
 ↪  Here, b01 = c(mu, beta1), and b2 = beta2 ###
samp012 <- function(b01, b2){

    curr <- c(b01, b2)
```

```
    epi <- rnorm(3, 0, sd=tau)
    prop <- curr + epi

    like.ratio <- - sum(log(dpois(y,
 ↪  lambda=exp(curr[1]+curr[2]*x1+curr[3]*x2)))) + sum(log(dpois(y,
 ↪  lambda=exp(prop[1]+prop[2]*x1+prop[3]*x2))))

    #cat("curr", curr, "prop", prop, "\n")
    #cat("like-ratio", like.ratio, "\n"); readline()

    prior.ratio <- - sum(log(dnorm(curr, 0, sig.beta))) +
 ↪  sum(log(dnorm(prop, 0, sig.beta)))

    acc <- exp(like.ratio + prior.ratio)
    #cat("samp012 acc", acc, "\n"); #readline()

    ind <- (acc > runif(1))

#    res <- c(res, beta012.tmp * ind + c(beta01,beta2) * (1-ind))
    res <- prop * ind + curr * (1-ind)
    #cat("ind", ind, "res", res[i+1, ], "i", i, "\n"); #readline()
    #cat(res); #readline()
    return(res)
}
```

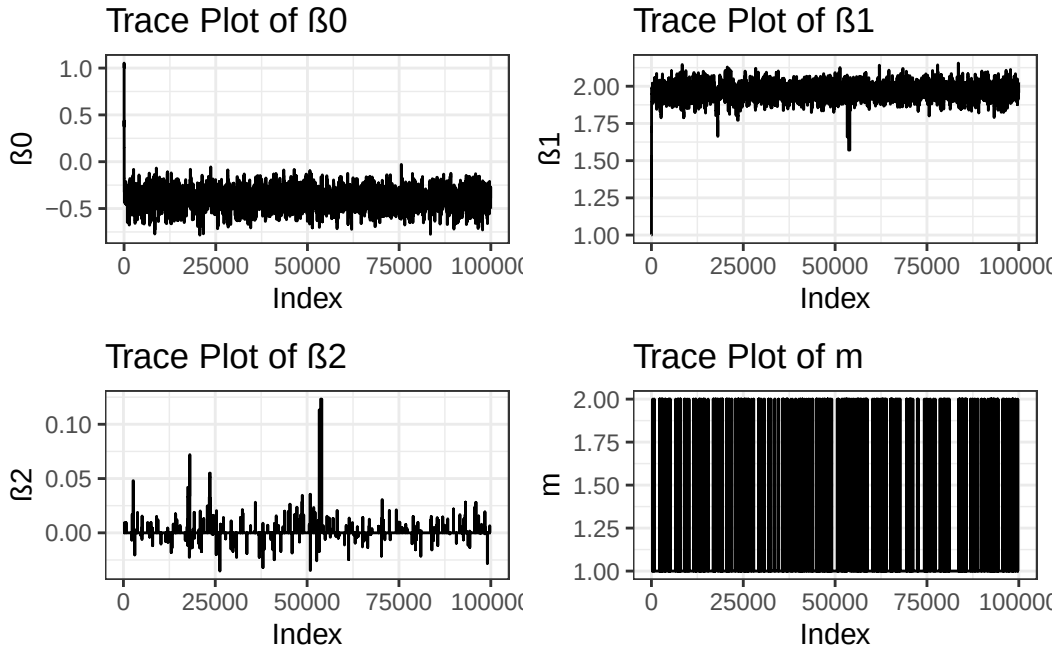
```
set.seed(41)

#### Set up the RJMCMC

for(sim in 1:(n.mc-1)){

    if(m[sim]==1){

        chg.ind <- (runif(1) < h[1,2])  ## Set chg.ind to TRUE with
 ↪  probability h[1,2] (prob from model 1 to 2). If chg.ind is TRUE,
 ↪  propose a move, which will add beta2 to the model. The move needs
 ↪  to be accepted.

        if(chg.ind){

            u <- rnorm(1, 0, sd=sig.u)
            beta2.tmp <- u
            #cat("u", u, "\n"); readline()
```

```
            like.ratio = sum(y*beta2.tmp*x2
                             + exp(beta01[sim,1]+beta01[sim,2]*x1)
                             -
                              ↪  exp(beta01[sim,1]+beta01[sim,2]*x1+beta2.tmp*x2))
            # Sanity check
            LR_sancheck = sum(dpois(y,
↪  lambda=exp(beta01[sim,1]+beta01[sim,2]*x1+beta2.tmp*x2), log=T)) -
↪  sum(dpois(y, lambda=exp(beta01[sim,1]+beta01[sim,2]*x1), log=T))
            # like.ratio <- - sum(-2 * beta2.tmp * x2 * (y -
             ↪  beta01[sim,1]- beta01[sim,2] * x1) + beta2.tmp^2 *
             ↪  x2^2) / 2 / sig^2 ## This is Yuan's example's
             ↪  log-likelihood ratio of m2 (quadratic) over m1 (linear)
            prior.ratio <- - log(sqrt(2*pi) * sig.beta) - beta2.tmp^2 /
↪  2 / sig.beta^2 ## + log((1-pi.M)) - log(pi.M)
            proposal <- log(sqrt(2*pi) * sig.u) + u^2 / 2 / sig.u^2

            acc <- exp(like.ratio + prior.ratio + proposal)

            ind <- (acc > runif(1))  ## If ind is TRUE, accept the move
↪  from model 1 --> 2. If ind is FALSE, do not move.

            if(ind){
                m[sim+1] <- 2
                beta2[sim+1] <- beta2.tmp
                beta01[sim+1, ] <- beta01[sim, ]
                #beta01[sim+1, ] <- c(1,1)
            }
            if(!ind){
                m[sim+1] <- m[sim]
                beta2[sim+1] <-  0
                beta01[sim+1, ] <- beta01[sim, ]
                #beta01[sim+1, ] <- c(1,1)
            }
        }

        if(!chg.ind){  ## if chg.ind is FALSE, do not propose model 2;
         ↪  resample mu and beta1.

            rest <- samp01(beta01[sim,])
            beta01[sim+1, ] <- rest
            beta2[sim+1] <- 0 #beta2[sim]
            m[sim+1] <- m[sim]
```

```
            # beta01[sim+1, ] <- c(1,1)
        }
    }

#    cat(sim, c(beta01[sim+1, ], beta2[sim+1]), "\n")

  ### The reversible move from model 2 to model 1.

    if(m[sim]==2){

        chg.ind <- (runif(1) < h[2,1]) ## Set chg.ind to TRUE with
↪  probability h[2,1] (prob from model 2 to 1). If chg.ind is TRUE,
↪  propose a move, which will eliminate beta2 from the model. The move
↪  needs to be accepted.

        if(chg.ind){

            u <- beta2[sim]
            beta2.tmp <- beta2[sim]

            like.ratio = sum(y*beta2.tmp*x2
                             + exp(beta01[sim,1]+beta01[sim,2]*x1)
                             -
                              ↪  exp(beta01[sim,1]+beta01[sim,2]*x1+beta2.tmp*x2))
            # Sanity check
            LR_sancheck = sum(dpois(y,
↪  lambda=exp(beta01[sim,1]+beta01[sim,2]*x1+beta2.tmp*x2), log=T)) -
↪  sum(dpois(y, lambda=exp(beta01[sim,1]+beta01[sim,2]*x1), log=T))
            # like.ratio <- - sum(-2 * beta2.tmp * x2 * (y -
             ↪  beta01[sim,1]- beta01[sim,2] * x1) + beta2.tmp^2 *
             ↪  x2^2) / 2 / sig^2




            prior.ratio <- - log(sqrt(2*pi) * sig.beta) - beta2.tmp^2 /
↪  2 / sig.beta^2
            proposal <- log(sqrt(2*pi) * sig.u) + u^2 / 2 / sig.u^2

            acc <- exp(-like.ratio - prior.ratio - proposal)

            ind <- (acc > runif(1))
```

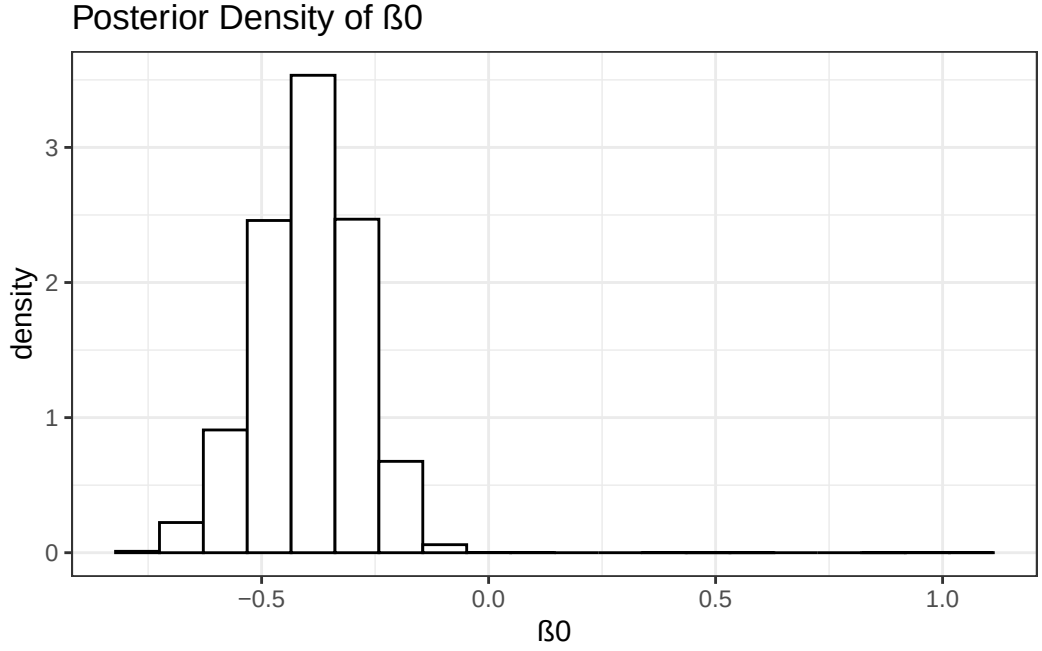
```
            if(ind){
                m[sim+1] <- 1
                beta2[sim+1] <- 0
                beta01[sim+1, ] <- beta01[sim, ]
                #beta01[sim+1, ] <- c(1,1)
            }

            if(!ind){
                m[sim+1] <- m[sim]
                beta01[sim+1, ] <-beta01[sim, ]
                beta2[sim+1] <- beta2[sim]
                #beta01[sim+1, ] <- c(1,1)
            }
        }

        if(!chg.ind){
           rest <- samp012(beta01[sim,], beta2[sim])
           beta01[sim+1, ] <- rest[1:2]
           beta2[sim+1] <- rest[3]
           m[sim+1] <- m[sim]
                #beta01[sim+1, ] <- c(1,1)
        }
    }
#   cat(sim, c(beta01[sim+1, ], beta2[sim+1]), "\n")

    ##readline()

}
```

With our simulation complete, we now check its performance. All four parameters—$\beta_0, \beta_1, \beta_2,$ and m—have trace plots with nice thick bands, indicating that our RJMCMC sampler is mixing well: draws of each parameter never stay at the same level for too long, nor do they ever take too many consecutive steps in the same direction.



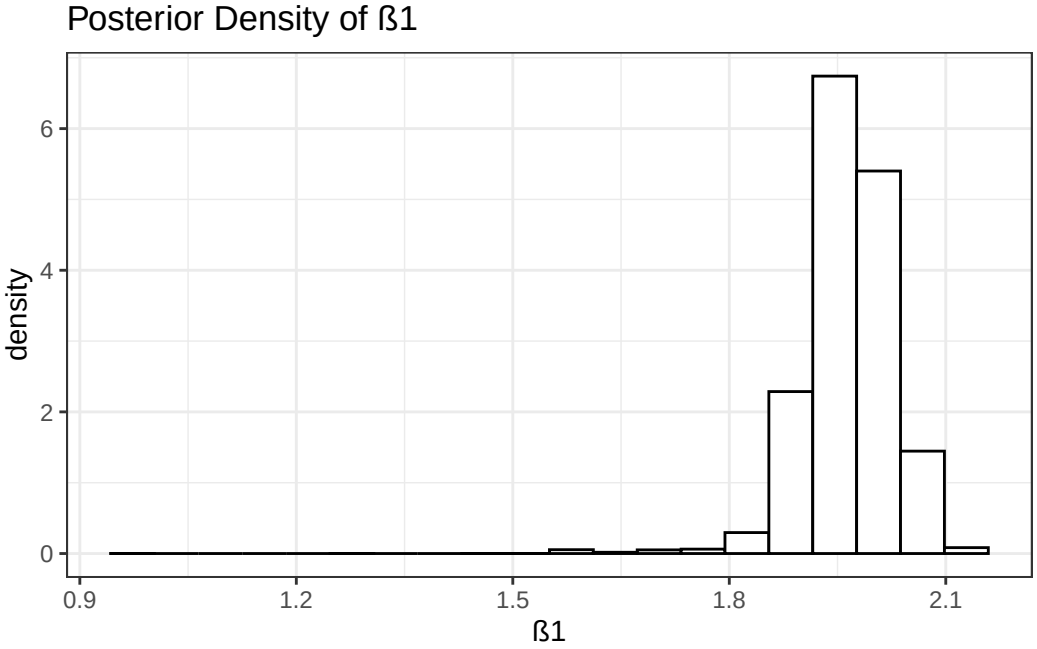
Since our simulation worked well, we move forward interpreting its results.

The posterior density of β_0 is plotted below. The posterior mean of β_0 is approximately -0.395.



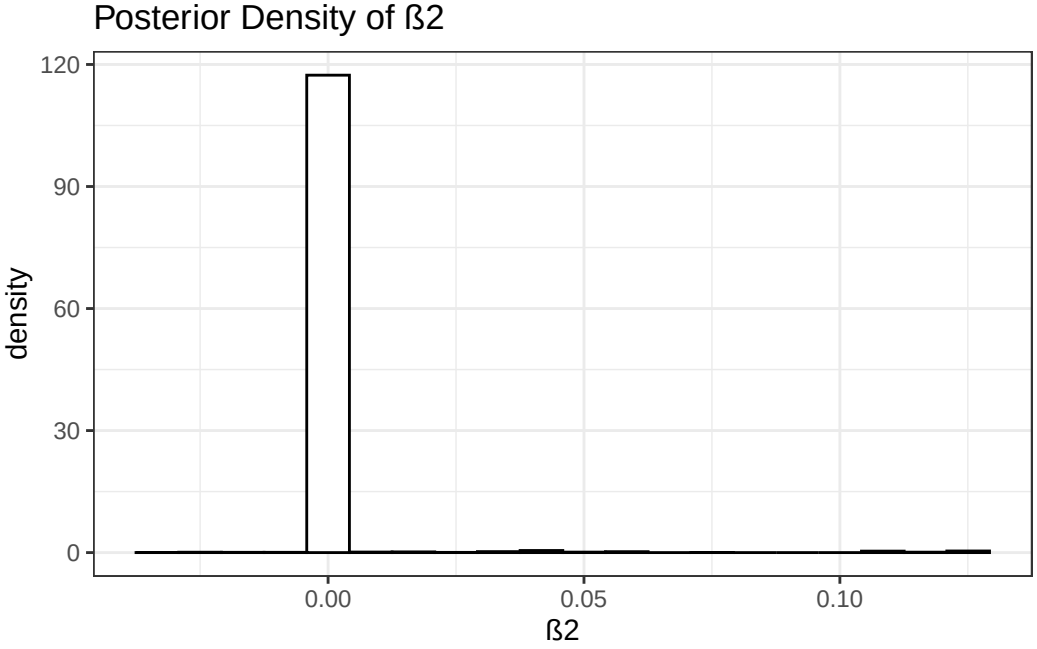

```
[1] -0.3946437
```

The posterior density of β_1 is plotted below. The posterior mean of β_1 is approximately -1.964.



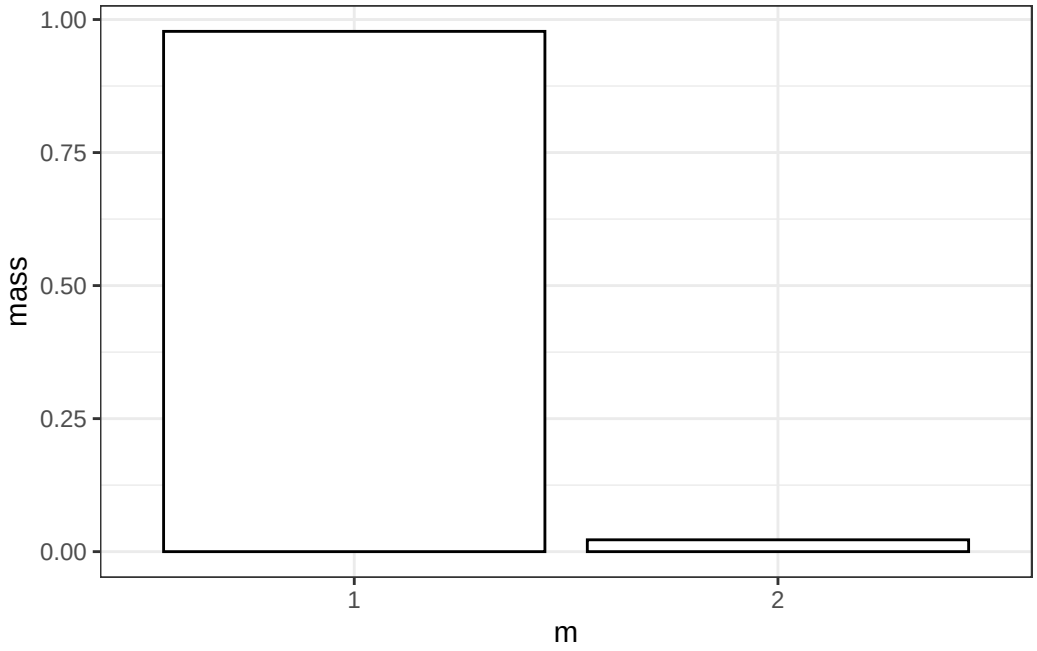

```
[1] 1.964225
```

The posterior density of β_2 is plotted below. The posterior mean of β_2 is approximately 0.001.



```
[1] 0.001253568
```

The posterior mass of m is plotted below. The posterior mode of m is clearly 1.

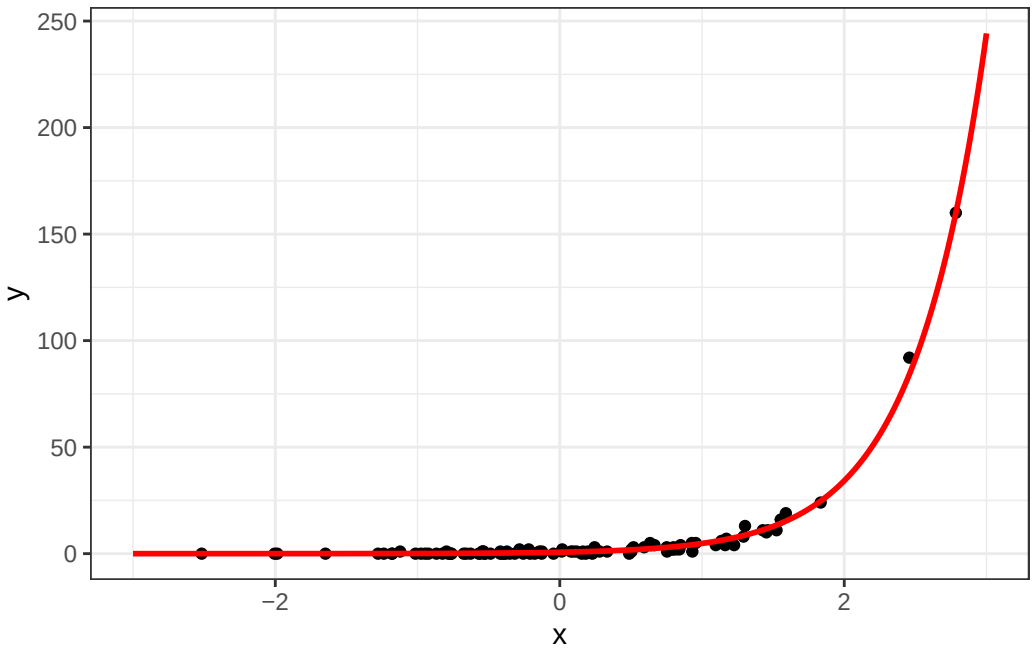



```
[1] 1
```

Model $m = 1$ clearly has a higher posterior mass than Model $m = 2$, so we conclude that $m = 1 : \log(\lambda(x)) = \beta_0 + \beta_1 x$ is the better model for these data. Thus, using the posterior means of β_0 and β_1, our estimate of the data-generating process is

$$\log(\hat{\lambda}(x)) = -0.395 + 1.964x.$$

As shown in the plot below, this estimate fits the data nicely.


250
200
150
100
50
0
y
−2
0
2
x