lecture 4.2 part 2

preliminaries

```
#clear workspace
rm(list=ls())

# load data
data = read.csv("~/Desktop/wages1.csv")

# build variables
wage = data$wage
school = data$school
exper = data$exper
```

build model objects

```
#build parameter grid
boundBeta0Grid = 2
boundBeta1Grid = 1
stepBetaGrid = 0.05
boundSigmaGrid = 1
stepSigmaGrid = 0.05
beta0Grid = seq(-boundBeta0Grid,boundBeta0Grid, by = stepBetaGrid)
beta1Grid = seq(-boundBeta0Grid,boundBeta0Grid, by = stepBetaGrid)
sigmaGrid = seq(stepSigmaGrid,boundSigmaGrid, by = stepSigmaGrid)
nBeta0Grid = length(beta0Grid)
nBeta1Grid = length(beta1Grid)
nSigmaGrid = length(sigmaGrid)

# priors
buildPrior = function() {
  prior = array( rep(1, nBeta0Grid * nBeta1Grid * nSigmaGrid ),
                 dim = c(nBeta0Grid, nBeta1Grid, nSigmaGrid ))
  for (nB0 in 1:nBeta0Grid) {
    for (nB1 in 1:nBeta1Grid) {
      for (nSig in 1:nSigmaGrid) {
        #prior[nB0,nB1,nSig] = dnorm(beta0Grid[nB0]) * dnorm(beta1Grid[nB1])
        prior[nB0,nB1,nSig] = 1
      }
    }
  }
  return(prior)
}


#likelihood
likelihood = function(y,x, b0L, b1L, sL){
  loglike = sum(log(dnorm(y-b0L-b1L*x, mean = 0, sd = sL)))
  like = exp(loglike)
```

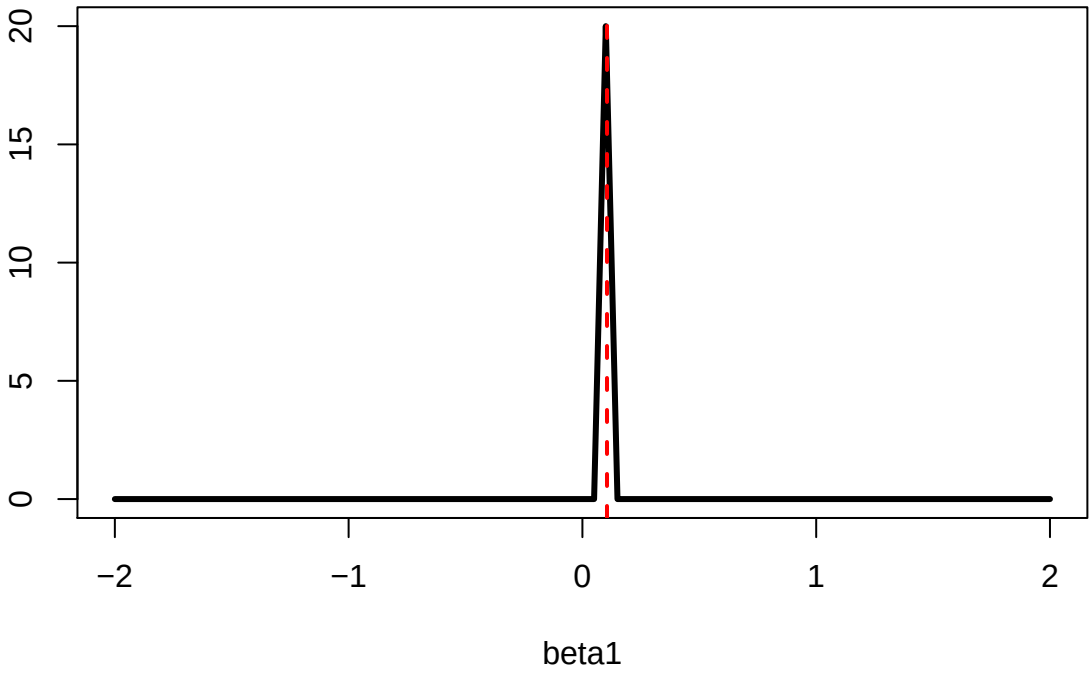
```
  return(like)
}
```

compute posterior

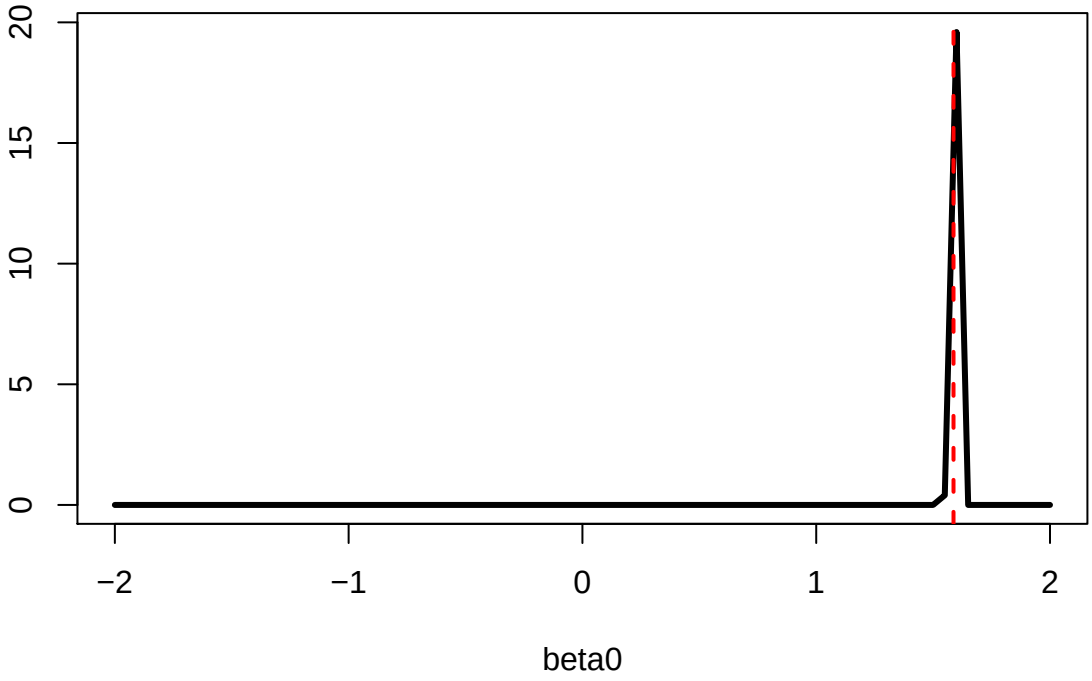
```
prior = buildPrior()
#compute posterior function
compPost = function(y,x, prior){
  #initialize local posterior
  post = array( rep(-1, nBeta0Grid * nBeta1Grid * nSigmaGrid ),
                dim = c(nBeta0Grid, nBeta1Grid, nSigmaGrid ))
  # compute posterior
  for (nBeta0 in 1:nBeta0Grid) {
    b0 = beta0Grid[nBeta0]
    for (nBeta1 in 1:nBeta1Grid) {
      b1 = beta1Grid[nBeta1]
      for (nSigma in 1:nSigmaGrid) {
        s = sigmaGrid[nSigma]
        post[nBeta0,nBeta1,nSigma] = likelihood(y,x, b0, b1, s) * prior[nBeta0,nBeta1,nSigma]
      }
    }
  }
  # normalize posterior
  post = post / ( sum(post) *stepBetaGrid^2 * stepSigmaGrid )
  # return
  return(post)
}

#compute posterior function iteratively using batches of 100 observations
for (k in 1:floor(length(wage)/100)) {
  y = log(wage[(1+(k-1)*100):(k*100)])
  x = school[(1+(k-1)*100):(k*100)] - mean(school)
  post = compPost(y,x,prior)
  prior = post
}

#compute marginal posteriors
margPostBeta0 = apply(post,c(1),sum)
margPostBeta0 = margPostBeta0 / ( sum(margPostBeta0) * stepBetaGrid )

margPostBeta1 = apply(post,c(2),sum)
margPostBeta1 = margPostBeta1 / ( sum(margPostBeta1) * stepBetaGrid )

margPostSigma = apply(post,c(3),sum)
margPostSigma = margPostSigma / ( sum(margPostSigma) * stepSigmaGrid )


# compute beta-hat estimates using classical linear regression
xC = school - mean(school)
m = lm(log(wage) ~ xC)
betaHat0 = coef(m)[1]
betaHat1 = coef(m)[2]
```

summarize and visualize posterior

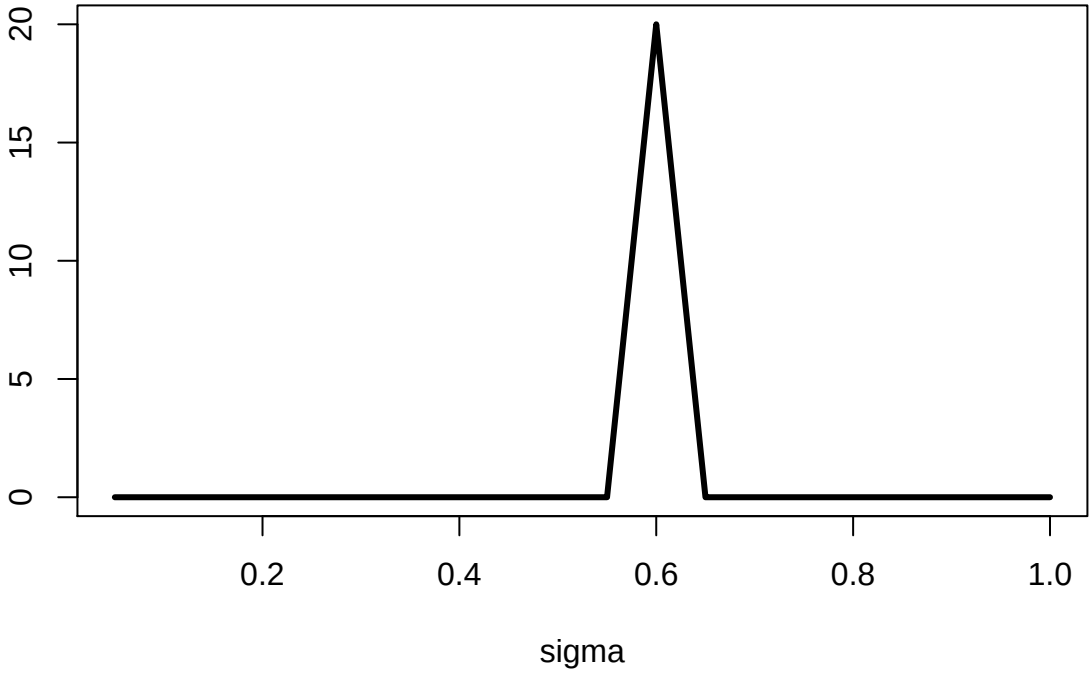
```
#posteriors beta1
plot(beta1Grid, margPostBeta1,
     xlab = "beta1", ylab="",
     type = "l", lwd = 3)
abline(v=betaHat1, lty=2,lwd=2, col="red")
```




```
#posteriors beta0
plot(beta0Grid, margPostBeta0,
     xlab = "beta0", ylab="",
     type = "l", lwd = 3)
abline(v=betaHat0, lty=2, lwd=2, col="red")
```




```
#posteriors sigma
plot(sigmaGrid, margPostSigma,
     xlab = "sigma", ylab="",
     type = "l", lwd = 3)
```

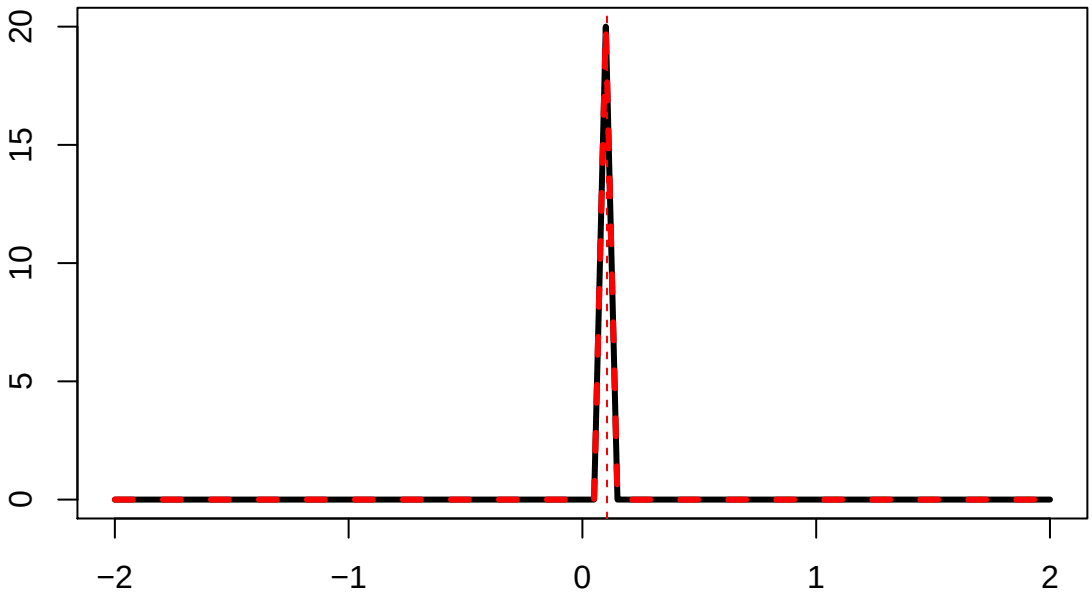


compare posteriors in even and odd trials

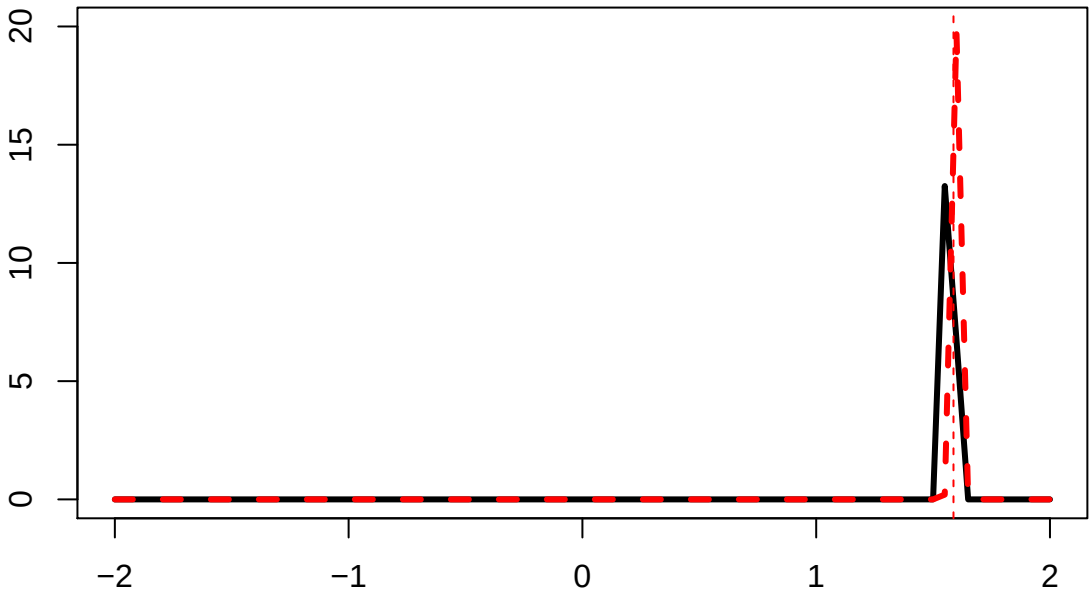
```
#compute posterior even
index = 1:length(data$wage)
```

```
wageE = wage[index%%2==0]
schoolE = school[index%%2==0]

prior = buildPrior()
for (k in 1:floor(length(wageE)/100)) {
  y = log(wageE[(1+(k-1)*100):(k*100)])
  x = schoolE[(1+(k-1)*100):(k*100)] - mean(schoolE)
  postEven = compPost(y,x,prior)
  prior = postEven
}

margPostBeta0E = apply(postEven,c(1),sum)
margPostBeta0E = margPostBeta0E / ( sum(margPostBeta0E) * stepBetaGrid )

margPostBeta1E = apply(postEven,c(2),sum)
margPostBeta1E = margPostBeta1E / ( sum(margPostBeta1E) * stepBetaGrid )

margPostSigmaE = apply(postEven,c(3),sum)
margPostSigmaE = margPostSigmaE / (sum(margPostSigmaE) * stepSigmaGrid )


#compute posterior odd
wage0 = wage[index%%2==1]
school0 = school[index%%2==1]

prior = buildPrior()
for (k in 1:floor(length(wage0)/100)) {
  y = log(wage0[(1+(k-1)*100):(k*100)])
  x = school0[(1+(k-1)*100):(k*100)] - mean(school0)
  post0dd = compPost(y,x,prior)
  prior = post0dd
}

margPostBeta00 = apply(post0dd,c(1),sum)
margPostBeta00 = margPostBeta00 / ( sum(margPostBeta00) * stepBetaGrid )

margPostBeta10 = apply(post0dd,c(2),sum)
margPostBeta10 = margPostBeta10 / ( sum(margPostBeta10) * stepBetaGrid )

margPostSigma0= apply(post0dd,c(3),sum)
margPostSigma0 = margPostSigma0 / ( sum(margPostSigma0) * stepSigmaGrid )
```

```
#posteriors beta1
plot(beta1Grid, margPostBeta1E,
     xlab = "beta1", ylab="",
     type = "l", lwd = 3)
points(beta1Grid, margPostBeta10,type = "l", lwd = 3, lty=2,
     xlab = "beta1", ylab="",
     pch=16, col="red")
abline(v=betaHat1, lty=2, col="red")
```




```
#posteriors beta0
plot(beta0Grid, margPostBeta0E,
     xlab = "beta0", ylab="",
     type = "l", lwd = 3, ylim=c(0,20))
points(beta0Grid, margPostBeta00,
       xlab = "beta0", ylab="", type = "l", lwd = 3, lty=2,
       pch = 16, col="red")
abline(v=betaHat0, lty=2, col="red")
```




```
#posteriors sigma
plot(sigmaGrid, margPostSigmaE,
```

```
     xlab = "sigma", ylab="",
     type = "l", lwd = 3)
points(sigmaGrid, margPostSigma0,type = "l", lwd = 3, lty=2,
     xlab = "sigma", ylab="",
     col='red', pch=16)
```