lecture 4.1 example 4

preliminaries

```
#clear workspace
rm(list=ls())
#initalize random seed
set.seed(123)
```

set simulation parameters and priors

```
nObs = c(25,100)
xGrid = seq(-1,1, by=0.1)

beta0 = 0
beta1 = 1
sigma = 1

boundBetaGrid = 2
sizeBetaGrid = 0.025
boundSigmaGrid = 2
sizeSigmaGrid = 0.025

beta0Grid = seq(-boundBetaGrid,boundBetaGrid, by = sizeBetaGrid)
beta1Grid = seq(-boundBetaGrid,boundBetaGrid, by = sizeBetaGrid)
sigmaGrid = seq(sizeSigmaGrid,boundSigmaGrid, by = sizeSigmaGrid)
nBeta0Grid = length(beta0Grid)
nBeta1Grid = length(beta1Grid)
nSigmaGrid = length(sigmaGrid)

# prior uninf
priorUninf = 1 / sigmaGrid^2

# priorAlt
priorAlt = rep(1, length(sigmaGrid))
```

define key functions

```
#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
compPost = function(y,x,priorSigma){
  #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) * priorSigma[nSigma]
      }
    }
  }
  # normalize posterior
  post = post / ( sum(post) * sizeBetaGrid^2 * sizeSigmaGrid )
  # return
  return(post)
}
```

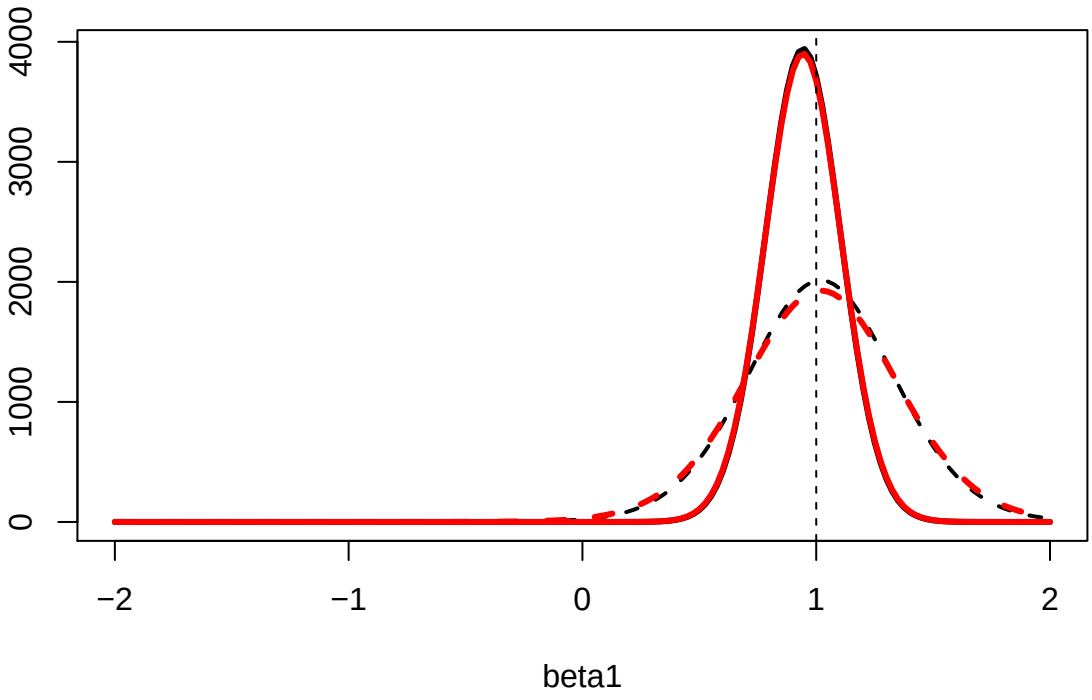
run simulatoions

```
#initialize arrays
postUninf = array( rep(-1, length(nObs) * nBeta0Grid * nBeta1Grid * nSigmaGrid ),
              dim = c(length(nObs), nBeta0Grid, nBeta1Grid, nSigmaGrid ))
postAlt = array( rep(-1, length(nObs) * nBeta0Grid * nBeta1Grid * nSigmaGrid ),
              dim = c(length(nObs), nBeta0Grid, nBeta1Grid, nSigmaGrid ))

# draw 100 sets of observations to be used in both cases
x = sample(xGrid, 100, replace = TRUE )
y = rnorm(100, mean = beta0 + beta1 * x, sd = sigma)

#main loop
for (n in 1:length(nObs)) {
  # compute and store posteriors
  postUninf[n,,,] = compPost(y[1:nObs[n]], x[1:nObs[n]], priorUninf)
  postAlt[n,,,]   = compPost(y[1:nObs[n]], x[1:nObs[n]], priorAlt)
}

#compute marginal posteriors
margPostUninfBeta0 = apply(postUninf,c(1,2),sum)
margPostUninfBeta1 = apply(postUninf,c(1,3),sum)
margPostUninfSigma = apply(postUninf,c(1,4),sum)
margPostAltBeta0 = apply(postAlt,c(1,2),sum)
margPostAltBeta1 = apply(postAlt,c(1,3),sum)
margPostAltSigma = apply(postAlt,c(1,4),sum)
```

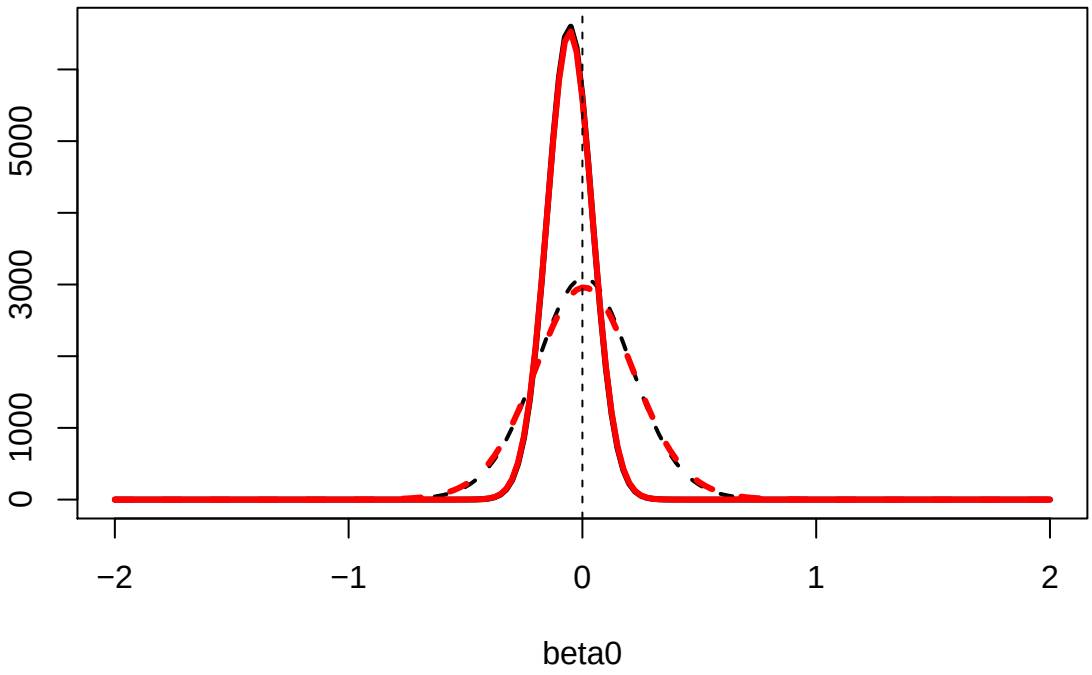
visualize marginal posteriors

```
#posteriors beta1
plot(beta1Grid, margPostUninfBeta1[2,],
```

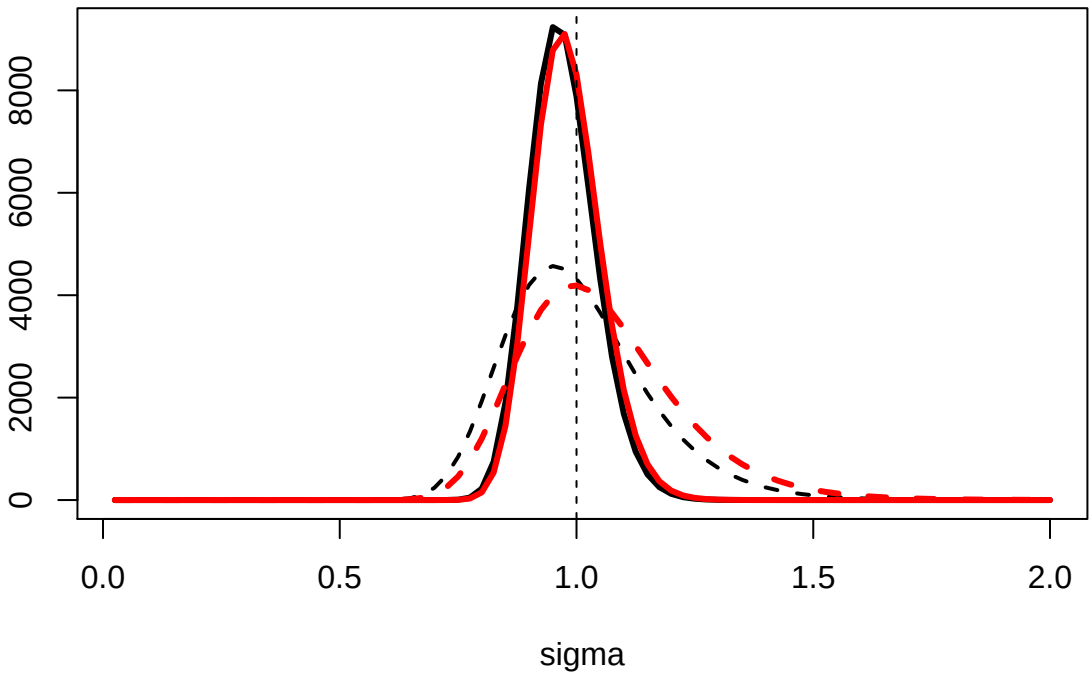
```
         xlab = "beta1", ylab="",
         type = "l", lwd = 3, col=1)
points(beta1Grid, margPostUninfBeta1[1,],
          type = "l", lwd = 2, col=1, lty=2)
points(beta1Grid, margPostAltBeta1[2,],
       type = "l", lwd = 3, col=2, lty=1)
points(beta1Grid, margPostAltBeta1[1,],
       type = "l", lwd = 3, col=2, lty=2)
abline(v=beta1, lty=2)
```




```
#posteriors beta0
plot(beta0Grid, margPostUninfBeta0[2,],
         xlab = "beta0", ylab="",
         type = "l", lwd = 3, col=1)
points(beta0Grid, margPostUninfBeta0[1,],
          type = "l", lwd = 2, col=1, lty=2)
points(beta0Grid, margPostAltBeta0[2,],
       type = "l", lwd = 3, col=2, lty=1)
points(beta0Grid, margPostAltBeta0[1,],
       type = "l", lwd = 3, col=2, lty=2)
abline(v=beta0, lty=2)
```



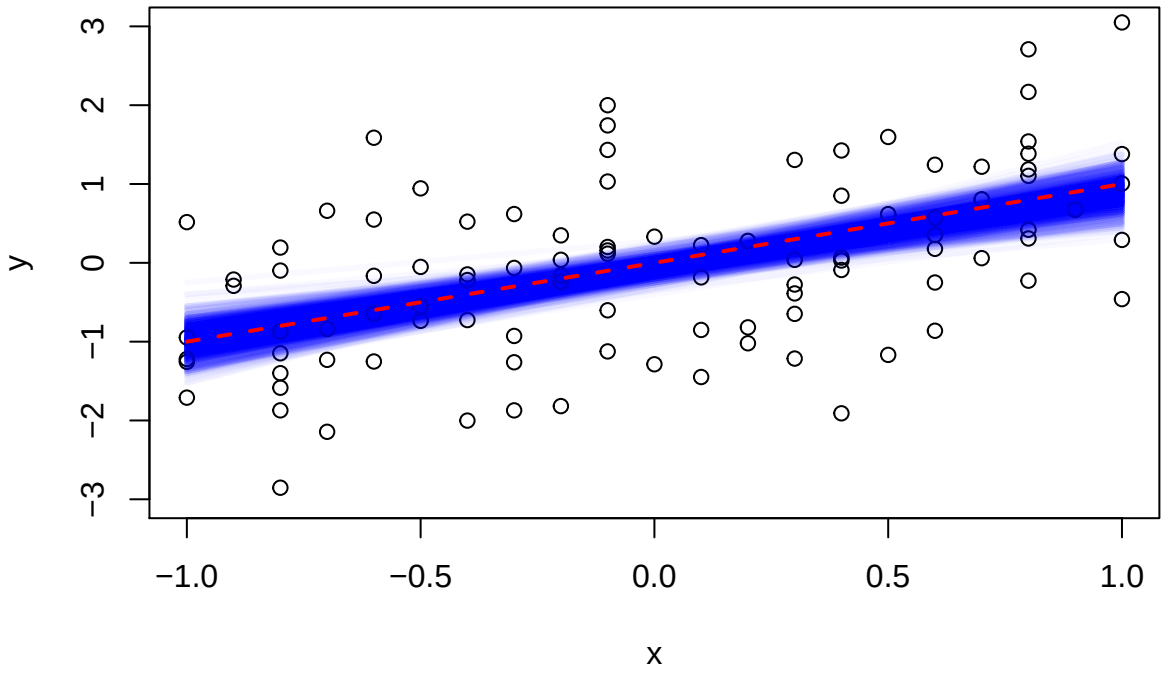

```
#posteriors sigma
plot(sigmaGrid, margPostUninfSigma[2,],
         xlab = "sigma", ylab="",
         type = "l", lwd = 3, col=1)
points(sigmaGrid, margPostUninfSigma[1,],
           type = "l", lwd = 2, col=1, lty=2)
points(sigmaGrid, margPostAltSigma[2,],
       type = "l", lwd = 3, col=2, lty=1)
points(sigmaGrid, margPostAltSigma[1,],
       type = "l", lwd = 3, col=2, lty=2)
abline(v=sigma, lty=2)
```



visualize uncertainty in posterior regression lines: n= 100 case

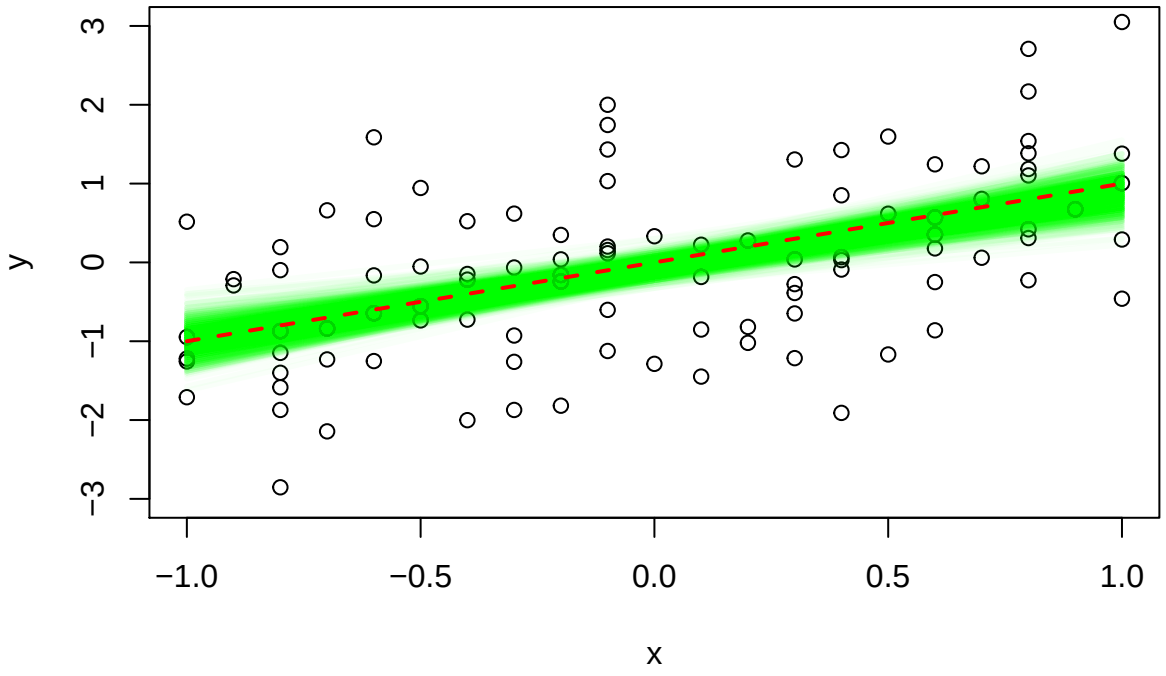
```
#build conditional posteriors
margBeta1GivenBeta0Uninf100 = array(rep(-1,nBeta0Grid * nBeta1Grid), dim= c(nBeta0Grid, nBeta1Grid))
for (nBeta0 in 1:nBeta0Grid) {
  margBeta1GivenBeta0Uninf100[nBeta0,] = apply(postUninf[2,nBeta0,,],c(1),sum)
}

margBeta1GivenBeta0Alt100 = array(rep(-1,nBeta0Grid * nBeta1Grid), dim= c(nBeta0Grid, nBeta1Grid))
for (nBeta0 in 1:nBeta0Grid) {
  margBeta1GivenBeta0Alt100[nBeta0,] = apply(postAlt[2,nBeta0,,],c(1),sum)
}
```

```
# plot uniformative priors case
plot(x,y,xlim=c(-1,1),ylim=c(-3,3), xlab = "x", ylab = "y")
for (sim in 1:1000) {
  b0Index = sample(1:nBeta0Grid, 1, prob=margPostUninfBeta0[2,])
  b1Index = sample(1:nBeta1Grid, 1, prob=margBeta1GivenBeta0Uninf100[b0Index,])
  b0Sample = beta0Grid[b0Index]
  b1Sample = beta1Grid[b1Index]
  points(xGrid, b0Sample + b1Sample*xGrid, type="l",lwd=3,
         col=rgb(red=0.0, green=0.0, blue=1.0, alpha=0.025))
}
points(xGrid, beta0 + beta1*xGrid, lwd=2, col=2, lty=2, type="l")
```




```
# plot alternative priors case
plot(x,y,xlim=c(-1,1),ylim=c(-3,3), xlab = "x", ylab = "y")
for (sim in 1:1000) {
  b0Index = sample(1:nBeta0Grid, 1, prob=margPostAltBeta0[2,])
  b1Index = sample(1:nBeta1Grid, 1, prob=margBeta1GivenBeta0Alt100[b0Index,])
  b0Sample = beta0Grid[b0Index]
  b1Sample = beta1Grid[b1Index]
  points(xGrid, b0Sample + b1Sample*xGrid, type="l",lwd=3,
```

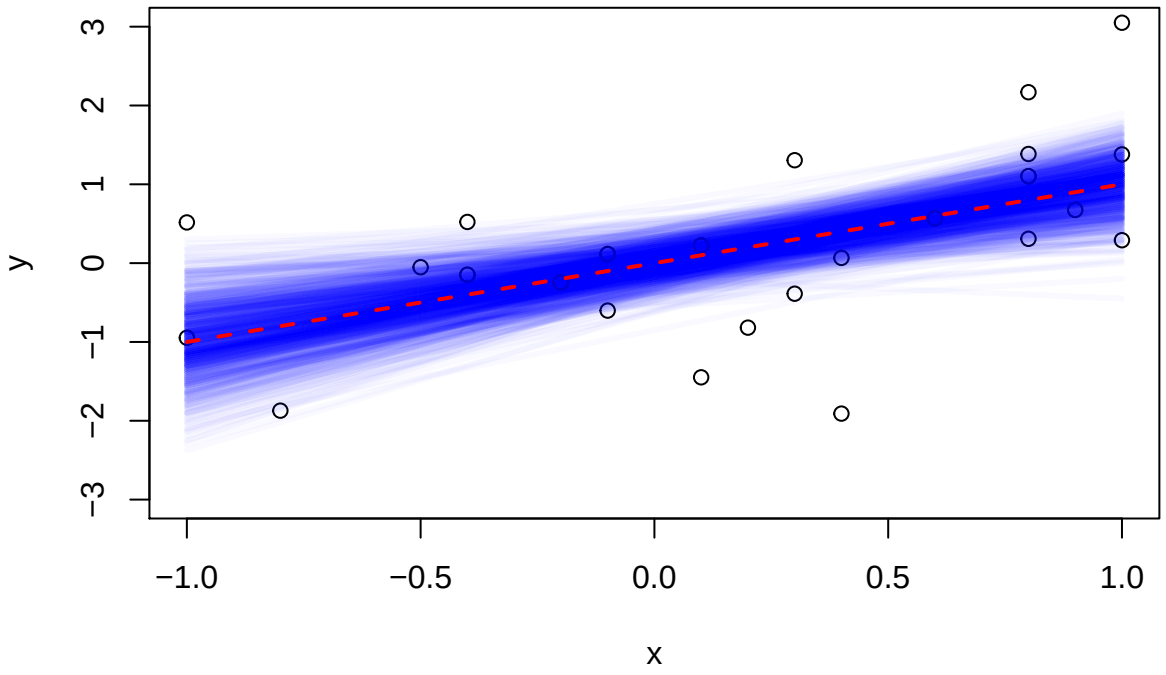
```
        col=rgb(red=0.0, green=1.0, blue=0.0, alpha=0.025))
}
points(xGrid, beta0 + beta1*xGrid, lwd=2, col=2, lty=2, type="l")
```



visualize uncertainty in posterior regression lines: n=25 case

```
#build conditional posteriors
margBeta1GivenBeta0Uninf25 = array(rep(-1,nBeta0Grid * nBeta1Grid), dim= c(nBeta0Grid, nBeta1Grid))
for (nBeta0 in 1:nBeta0Grid) {
  margBeta1GivenBeta0Uninf25[nBeta0,] = apply(postUninf[1,nBeta0,,],c(1),sum)
}

margBeta1GivenBeta0Alt25 = array(rep(-1,nBeta0Grid * nBeta1Grid), dim= c(nBeta0Grid, nBeta1Grid))
for (nBeta0 in 1:nBeta0Grid) {
  margBeta1GivenBeta0Alt25[nBeta0,] = apply(postAlt[1,nBeta0,,],c(1),sum)
}
```

```
# plot uniformative priors case
plot(x[1:25],y[1:25],xlim=c(-1,1),ylim=c(-3,3), xlab = "x", ylab = "y")
for (sim in 1:1000) {
  b0Index = sample(1:nBeta0Grid, 1, prob=margPostUninfBeta0[1,])
  b1Index = sample(1:nBeta1Grid, 1, prob=margBeta1GivenBeta0Uninf25[b0Index,])
  b0Sample = beta0Grid[b0Index]
  b1Sample = beta1Grid[b1Index]
  points(xGrid, b0Sample + b1Sample*xGrid, type="l",lwd=3,
         col=rgb(red=0.0, green=0.0, blue=1.0, alpha=0.025))
}
points(xGrid, beta0 + beta1*xGrid, lwd=2, col=2, lty=2, type="l")
```




```
# plot alternative priors case
plot(x[1:25],y[1:25],xlim=c(-1,1),ylim=c(-3,3), xlab = "x", ylab = "y")
for (sim in 1:1000) {
  b0Index = sample(1:nBeta0Grid, 1, prob=margPostAltBeta0[1,])
  b1Index = sample(1:nBeta1Grid, 1, prob=margBeta1GivenBeta0Alt25[b0Index,])
  b0Sample = beta0Grid[b0Index]
  b1Sample = beta1Grid[b1Index]
  points(xGrid, b0Sample + b1Sample*xGrid, type="l",lwd=3,
         col=rgb(red=0.0, green=1.0, blue=0.0, alpha=0.025))
}
points(xGrid, beta0 + beta1*xGrid, lwd=2, col=2, lty=2, type="l")
```