lecture 4.5 example 2

preliminaries

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

#set seet
set.seed(123)

# 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(-boundBeta1Grid,boundBeta1Grid, by = stepBetaGrid)
sigmaGrid = seq(stepSigmaGrid,boundSigmaGrid, by = stepSigmaGrid)
nBeta0Grid = length(beta0Grid)
nBeta1Grid = length(beta1Grid)
nSigmaGrid = length(sigmaGrid)

# uninformed 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
}
```

Sample from joint posterior and predict at every level of schooling

```
nPredictSamples = 1000

# build conditional marginal posteriors for sampling
#build conditional posteriors
margBeta0 = apply(post,c(1),sum)
magBeta0 = margBeta0 / (sum(margBeta0) * stepBetaGrid)

margBeta1GivenBeta0 = array(rep(-1,nBeta0Grid * nBeta1Grid),
                            dim= c(nBeta0Grid, nBeta1Grid))
for (nBeta0 in 1:nBeta0Grid) {
```

```
  margBeta1GivenBeta0[nBeta0,] = apply(post[nBeta0,,],c(1),sum)
}
margBeta1GivenBeta0 = margBeta1GivenBeta0 / (sum(margBeta1GivenBeta0) *stepBetaGrid * stepSigmaGrid)

#initialize storage arrays
predicted = array(rep(-1, length(unique(school)) * nPredictSamples),
                  dim = c(length(unique(school)), nPredictSamples))

#predictions
for (t in 1:length(unique(school))) {
  xNew = sort(unique(school - mean(school)))[t]
  for (sim in 1:nPredictSamples) {
    b0Index = sample(1:nBeta0Grid, 1, prob=margBeta0)
    b1Index = sample(1:nBeta1Grid, 1, prob=margBeta1GivenBeta0[b0Index,])
    sigIndex = sample(1:nSigmaGrid, 1, prob=post[b0Index,b1Index,])
    b0Sample = beta0Grid[b0Index]
    b1Sample = beta1Grid[b1Index]
    sigSample = sigmaGrid[sigIndex]
    predicted[t,sim] = rnorm(1, mean=b0Sample + xNew * b1Sample, sd=sigSample)
  }
}
```

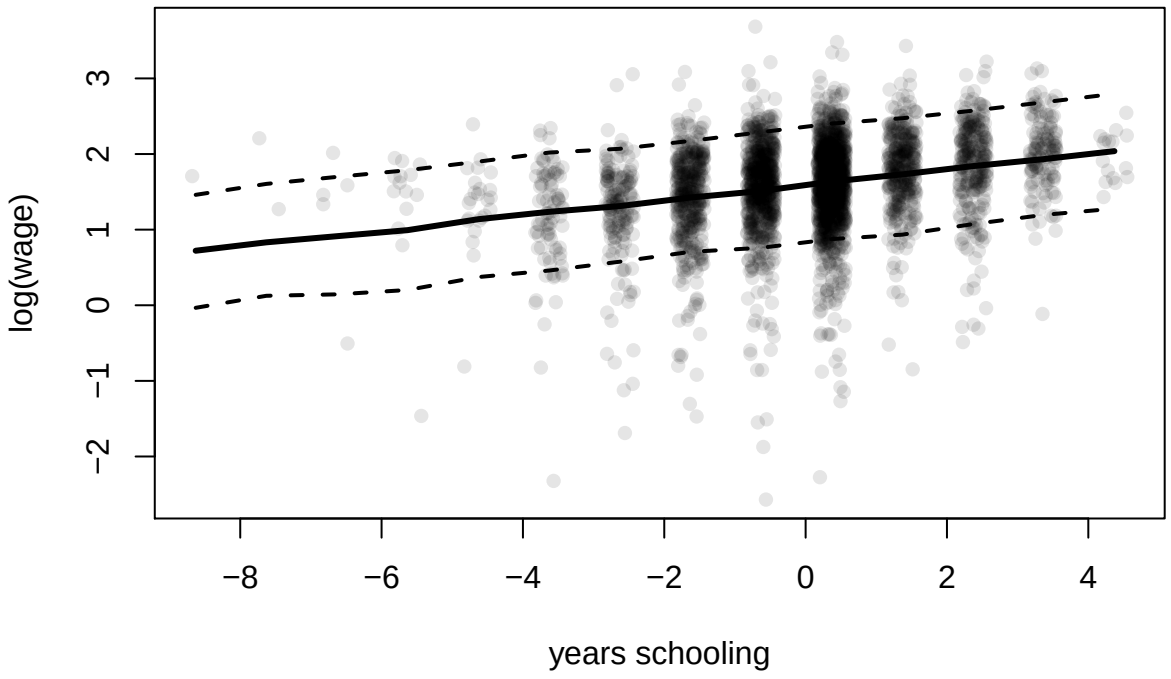
visualize and compare with actual data

```
#summary stats of predictions
predictMean = apply(predicted,c(1),mean)

predict10 = rep(0, length(unique(school)))
for (t in 1:length(unique(school))) {
  predict10[t] = quantile(predicted[t,],probs=0.1)
}

predict90 = rep(0, length(unique(school)))
for (t in 1:length(unique(school))) {
  predict90[t] = quantile(predicted[t,],probs=0.9)
}

# plot predictions vs actual
plot(jitter(school-mean(school),1), log(wage),
     col=rgb(red=0.0, green=0.0, blue=0.0, alpha=0.1),
     pch = 16,
     xlab = "years schooling", ylab = "log(wage)")
points(sort(unique(school-mean(school))), predictMean,
       type="l", lwd = 3)
points(sort(unique(school))-mean(school), predict10,
       type="l", lwd=2, lty=2)
points(sort(unique(school))-mean(school), predict90,
      type="l", lwd=2, lty=2)
```


log(wage)
3
2
1
0
−1
−2
−8
−6
−4
−2
0
2
4
years schooling