lecture 3.5 example

preliminaries

```
# clear work space
rm(list = ls())
#set random seed
set.seed(123)
```

define simulations parameters

```
nObs = 20

# define priors for mu
muMean = 0
muSd = 5

#define priors for sigma
sigMin = 0
sigMax = 10

# choose sim true parameters
muTrue = -2.8
sigTrue = 5
```

build parameter grid

```
nGridPoints = 200
muGridMin = -20
muGridMax = 20
sigGridMin = 0
sigGridMax = 10
muGrid = seq(muGridMin, muGridMax,length.out = nGridPoints)
sigGrid = seq(sigGridMin, sigGridMax,length.out = nGridPoints)
muGridSize = (muGridMax - muGridMin) / nGridPoints
sigGridSize = (sigGridMax - sigGridMin) / nGridPoints
```

define key functions

```
# generate prior matrix
buildPriors = function(meanMean, meanSD, sdMin, sdMax) {
  #initialize prior matrix
  priorM = matrix(rep(0, nGridPoints ^ 2 ),
                  nrow = nGridPoints,
                  ncol = nGridPoints,
                  byrow = TRUE)
  #fill out the prior matrix
```

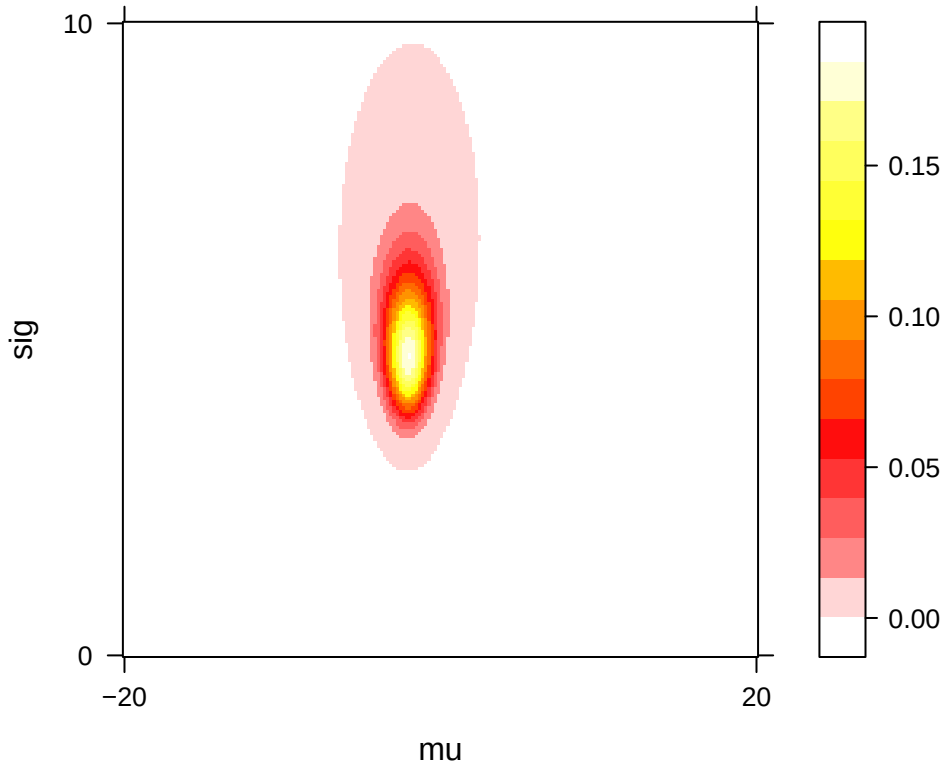
```
  for (row in 1:nGridPoints) {
    for (col in 1:nGridPoints) {
      priorM[row,col] = muPrior[row] * sigPrior[col]
    }
  }
  #normalize the prior matrix
  priorM = priorM / sum(priorM)
  priorM = priorM / (muGridSize & sigGridSize)
  #return the prior matrix
  return(priorM)
}

# compute posterior
computePost = function(data, priorM){
  #initialize posterior matrix
  postM = matrix(rep(-1, nGridPoints ^ 2 ),
                 nrow = nGridPoints,
                 ncol = nGridPoints,
                 byrow = TRUE)
  #fill out the posterior matrix
  for (row in 1:nGridPoints) {
    for (col in 1:nGridPoints) {
      muVal = muGrid[row]
      sigVal = sigGrid[col]
      #compute data likelihood
      loglike = sum(log(dnorm(data, muVal, sigVal)))
      # update posterior matrix cell
      postM[row,col] =  exp(loglike) * priorM[row,col]
    }
  }
  # normalize the posterior & return
  postM = postM / sum(postM * muGridSize * sigGridSize)
  return(postM)
}
```

compute posteriors

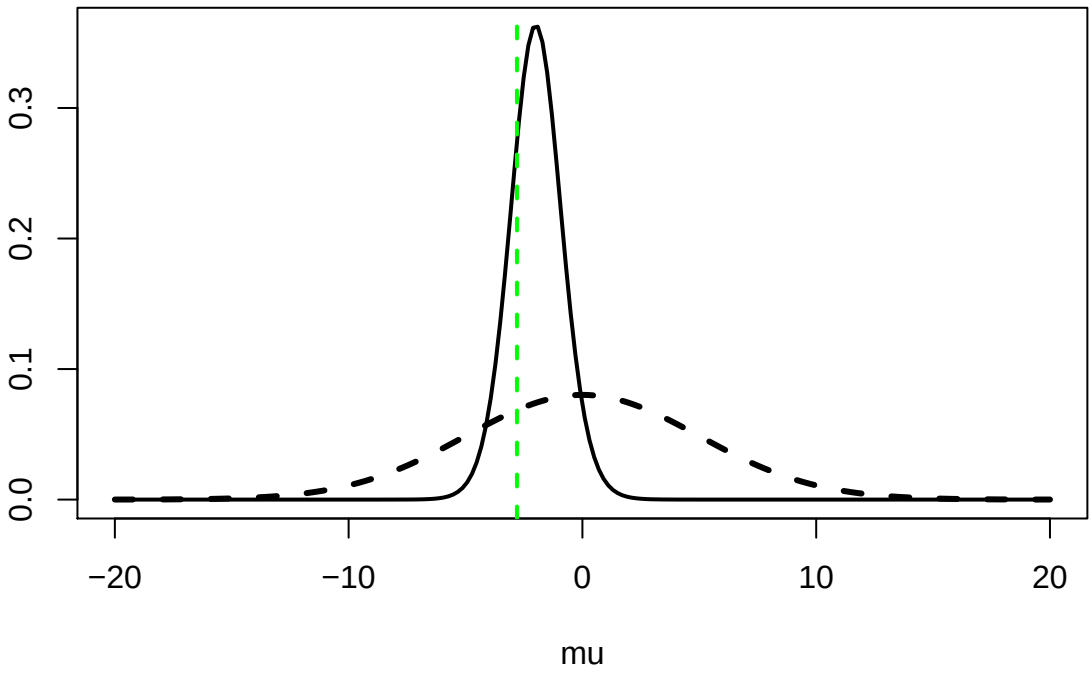
```
# build priors
muPrior = dnorm(muGrid,muMean,muSd)
muPrior = muPrior / ( sum(muPrior) * muGridSize)
sigPrior = dunif(sigGrid,sigMin,sigMax)
sigPrior = sigPrior / ( sum(sigPrior) * sigGridSize )
priorM = buildPriors(muMean, muSd, sigMin, sigMax)

# simulate data
data = rnorm(nObs, muTrue, sigTrue)
# compute posterior
postM = computePost(data, priorM)

# computer marginal posterior distributions
margMu = rowSums(postM * sigGridSize)
margSig = colSums(postM * muGridSize)
```

visualize posteriors

```
# visualize joint posterior
library(lattice)
new.palette=colorRampPalette(c("white","red","yellow","white"),space="rgb")
levelplot(postM, col.regions=new.palette(20),
          xlab = "mu", ylab = "sig",
          #main = paste("muTrue = ", muTrue, ", sigTrue = ", sigTrue),
          scales=list(x=list(at=c(1,nGridPoints), labels=c(muGridMin,muGridMax)),
                      y=list(at=c(1,nGridPoints), labels=c(sigGridMin,sigGridMax))))
```




```
# visualize marginal posterior for mu
plot(muGrid, margMu, type="l", lwd=2,
     xlab = "mu", ylab = "")
points(muGrid,muPrior, type="l", lwd=3, lty=2)
abline(v=muTrue, lwd=2, lty=2,col="green")
```




```
# look at marginal posterior for Sigma
plot(sigGrid, margSig, type="l", lwd=2,
     xlab = "sigma", ylab = "")
points(sigGrid,sigPrior, type="l", lwd=3, lty=2)
abline(v=sigTrue, lwd=2, lty=2,col="green")
```