Introducing probability and statistics to students who have had calculus: a Bayesian approach with computing

Joe Chang
Yale University Statistics Dept

JSM 2015, Seattle
Stat 238 goal: for students who have had calculus and would like to use it, a good semester course in probability and statistics that

- makes sense to students and is interesting all the way through,
- gives a good understanding of everything while not proving everything,
- gets to a good place in practice.
THE BAYESIAN REVOLUTION IN GENETICS

Mark A. Beaumont* and Bruce Rannala‡

Bayesian statistics allow scientists to easily incorporate prior knowledge into their data analysis. Nonetheless, the sheer amount of computational power that is required for Bayesian statistical analyses has previously limited their use in genetics. These computational constraints have now largely been overcome and the underlying advantages of Bayesian approaches are putting them at the forefront of genetic data analysis in an increasing number of areas.

The Bayesian within you.
Simulating the birthday problem: first do it once

```r
k <- 40
birthdays <- sample(1:365, size = k, replace = TRUE)
birthdays

[1]  74  251  335  104   39  256  193  295  350  41 100  180  117  205   96   74  142
[18]  325  203  308  325  264   78   83   52  176  160  353   52  349  163  22 101  12
[35]   6  178  218  219  146  145

table(birthdays)
birthdays

6   12  22  39  41  52  74  78  83  96 100 101 104 117 142 145 146 160
1   1   1   1   1   1   2   2   1   1   1   1   1   1   1   1   1
163 176 178 180 193 203 205 218 219 251 256 264 295 308 325 335 349 350
1   1   1   1   1   1   1   1   1   1   1   1   1   2   1   1   1
353
1

max(table(birthdays))
[1] 2
```
Simulating the birthday problem: repeating many times

```r
nrep <- 10000
maxs <- rep(0, nrep)
for(i in 1:nrep){
  birthdays <- sample(1:365, size = k, replace = TRUE)
  maxs[i] <- max(table(birthdays))
}

sum(maxs >= 2)
[1] 8961
mean(maxs >= 2)
[1] 0.8961
```
Bayes’ rule

A probability problem:

A frog starts at “start,” makes two randomly chosen hops, and ends up at $B$.

What is the probability that the frog went through $A_1$, $A_2$, and $A_3$?

Answer:

\[ P(B) = (.1)(.6) + (.4)(.1) + (.5)(.2) \]

and

\[ P(A_1|B) = \frac{(.1)(.6)}{P(B)}, \quad P(A_2|B) = \frac{(.4)(.1)}{P(B)}, \quad P(A_1|B) = \frac{(.5)(.2)}{P(B)}. \]
Bayes’ rule in R

```
p.hop1 <- c(.1, .4, .5)
p.hop2 <- c(.6, .1, .2)

(paths <- p.hop1 * p.hop2)
[1] 0.06 0.04 0.10

(paths / sum(paths))
[1] 0.3 0.2 0.5
```

![Bayes' rule diagram](image)
Spin $n$ times, observe data $X_1, X_2, \ldots, X_n$.

Parameter of interest is the location $\theta$. 
Froggy essence of Bayesian inference

\[ p(A_i | B) \propto p(A_i) p(B | A_i) \]

\[ p(\theta | x) \propto p(\theta) p(x | \theta) \]

**posterior \propto prior \cdot likelihood**
Likelihood function uses calculus

$$F_X(x) = P\{X \leq x\} = P\{X - \theta \leq x - \theta\} = P(\tan \alpha \leq x - \theta)$$

$$= P\{\alpha \leq \tan^{-1}(x - \theta)\} = \frac{\tan^{-1}(x - \theta) + \pi/2}{\pi}$$

pdf

$$f_X(x) = F_X'(x) = \frac{1}{\pi(1 + (x - \theta)^2)}$$
Simulating some data and doing Bayesian inference

def lik(th):
    return prod(1/(1 + (spins - th)**2))

thetas <- seq(0,5,by=.05)
liks <- sapply(thetas, lik)
prior <- rep(1,length(liks))/length(liks)
post <- prior*liks / sum(prior*liks)

plot(thetas, post, type='h')
points(thetas, post, pch=20)

conf <- 0.95
ord <- order(post, decreasing = TRUE)
cs <- cumsum(post[ord])
include <- min(which(cs >= conf))
conf.set <- thetas[ord[1:include]]
range(conf.set)
[1] 2.35 3.70
MCMC: Metropolis sampling

Generate a random sample from the pdf

\[ f(\theta) \propto \frac{(10 \theta - 3\theta^2) \sin^2(\theta - 2)}{\sqrt{1 + e^{\tan(\theta)}}} \]

for \(0 < \theta < 3\) and \(f(\theta) = 0\) otherwise.
MCMC: Metropolis sampling
MCMC: Metropolis sampling
MCMC: Metropolis sampling
MCMC: Metropolis sampling

\[ \text{ratio} = \frac{f(1.2)}{f(0.9)} = 0.43 \]

runif(1): 0.6063
Generate a random sample from the pdf \( f(\theta) \propto \frac{(10\theta - 3\theta^2) \sin^2(\theta - 2)}{\sqrt{1 + e^{\tan(\theta)}}} I(0 < \theta < 3). \)

\[
f(\theta) = (10\theta - 3\theta^2) \sin(\theta - 2)^2 \times (\theta > 0) \times (\theta < 3) / \sqrt{1 + e^{\tan(\theta)}}
\]

```r
f <- function(th){
  (10*th - 3*th^2) * sin(th-2)^2 * (th>0) * (th<3) / sqrt(1+exp(tan(th)))
}

nit <- 200000
path <- numeric(nit)
state <- 2
path[1] <- state
for(i in 2:nit){
  candidate <- runif(1, state-1, state+1)
  ratio <- f(candidate)/f(state)
  if(runif(1) < ratio) state <- candidate
  path[i] <- state
}
A two-sample problem

\[ X_1, \ldots, X_m \sim N(\mu_x, \sigma_x) \]

\[ Y_1, \ldots, Y_n \sim N(\mu_y, \sigma_y) \]

Parameter vector \( \Theta = (\mu_x, \sigma_x, \mu_y, \sigma_y) \).

Want:

- \( P\{\mu_x > \mu_y \mid X, Y\} \)
- Confidence interval ("credible interval" (CI)) for \( \mu_x - \mu_y \)
Metropolis from scratch for two-sample problem

lik = function(th){
    mux=th[1]; sigx=th[2]; muy=th[3]; sigy=th[4]
    return(prod(dnorm(x,mean=mux,sd=sigx)) * prod(dnorm(y,mean=muy,sd=sigy)))
}

prior = function(th){
a <- -10; b <- 10
    mux=th[1]; sigx=th[2]; muy=th[3]; sigy=th[4]
    if(min(sigx,sigy) <= exp(a) | max(sigx,sigy) >= exp(b))return(0)
    return(dnorm(mux,0,1000) * dnorm(muy,0,1000) * 1/((b-a)*sigx) * 1/((b-a)*sigy))
}

post = function(th){
    mux=th[1]; sigx=th[2]; muy=th[3]; sigy=th[4]
    if(sigx <= 0 | sigy <= 0) return(0)
    return(prior(th) * lik(th))
}

mux <- 10; sigx <- 3; muy <- 10; sigy <- 3
nit <- 10000
path <- matrix(0, nrow=nit, ncol=4)
path[1,] <- th <- c(mux, sigx, muy, sigy)
for(it in 2:nit){
    cand <- th + rnorm(4)
    ratio <- post(cand)/post(th)
    if(runif(1) < ratio) {th <- cand}
    path[it,] <- th
}
Metropolis from scratch for two-sample problem

```r
p <- data.frame(path); colnames(p) <- c('mux', 'sigx', 'muy', 'sigy')
par(mfrow=c(4,2))
plot(p$mux, type="l"); hist(p$mux)
plot(p$sigx, type="l"); hist(p$sigx)
plot(p$muy, type="l"); hist(p$muy)
plot(p$sigy, type="l", xlab='MC iteration'); hist(p$sigy)
```
Metropolis from scratch for two-sample problem

```r
hist(p$mux - p$muy)

mean(p$mux - p$muy > 0)
[1] 0.9573

quantile(p$mux - p$muy, c(.025,.975))
    2.5%   97.5%
-0.640502  7.042552
```
MCMC from scratch in R

Steps we have learned to use to solve problems with Metropolis sampling from scratch:

1. Make a model to describe the situation.
2. Write a likelihood function \[ \text{lik} = \text{function}(\text{th})\{...\} \]
3. Write a prior \[ \text{prior} = \text{function}(\text{th})\{...\} \]
4. Multiply to get posterior \[ \text{post} = \text{function}(\text{th})\{\text{prior}(\text{th}) \ast \text{lik}(\text{th})\} \]
5. Do MCMC to sample from the posterior:
   a) Choose a way to propose random changes to \text{th}
   b) Accept or reject the proposals in a Metropolis way
6. Use the MCMC results to answer the questions of interest.
MCMC from scratch in R using JAGS

Steps we have learned to use to solve problems with Metropolis sampling from scratch:

1. Make a model to describe the situation. ← (write a JAGS model)
2. Write a likelihood function [ lik = function(th){...} ]
3. Write a prior [ prior = function(th){...} ]
4. Multiply to get posterior [ post = function(th){prior(th) * lik(th)} ]
5. Do MCMC to sample from the posterior:
   a) Choose a way to propose random changes to th
   b) Accept or reject the proposals in a Metropolis way
6. Use the MCMC results to answer the questions of interest. ← (get rich and famous)

Next we learn to use JAGS, which takes care of steps 2-5!
JAGS for two-sample problem

```r
library(rjags)

jm <- jags.model(
  textConnection(jags.model),
  data=list(x=x, y=y, nx=length(x), ny=length(y))
)

cs <- coda.samples(jm, c("mux","muy","sigx","sigy"), 10000)

plot(cs)
```

```r
jags.model <- "
  model{
    for(i in 1:nx){
      x[i] ~ dnorm(mux, taux)
    }
    for(i in 1:ny){
      y[i] ~ dnorm(muy, tauy)
    }

    mux ~ dnorm(0, .000001)
    muy ~ dnorm(0, .000001)
    logsigx ~ dunif(-10, 10)
    logsigy ~ dunif(-10, 10)

    sigx <- exp(logsigx)
    sigy <- exp(logsigy)
    taux <- 1/sigx^2
    tauy <- 1/sigy^2
  }
"
```

JAGS for two-sample problem
JAGS for two-sample problem

```r
s <- as.data.frame(cs[[1]])
hist(s$mux - s$muy)

mean(s$mux - s$muy > 0)
[1] 0.9545

quantile(s$mux - s$muy, c(.025, .975))
  2.5%     97.5%
-0.6377312  7.0547779
```
Now lots of things can be done quite easily

E.g. here’s a JAGS model for linear regression:

```jags
model{
  for(i in 1:n){
    y[i] ~ dnorm(alpha + beta*x[i], tau)
  }
  alpha ~ dnorm(0, 1.0E-6)
  beta ~ dnorm(0, 1.0E-6)
  tau ~ dgamma(.001,.001)
}
```
Hidden Markov models

Hidden Markov models revolution(s) in...
  – cryptography
  – speech recognition
  – biological sequence analysis
Hidden Markov models

A frog hops around on two lily pads (states).

E.g. the frog could produce this path:

111222111111111122211111111222222222222222222222
222222222211111111112222222222111111111111112222222
2221111122111111111112222222222222222111122222222222
2222222222222222222222222221112222222111112222222

1112222111111111122211111111222222222222222222222
222222222211111111112222222222111111111111112222222
2221111122111111111112222222222222222111122222222222
2222222222222222222222222221112222222111112222222

With dice...

E.g. we could have $p = .02$ and $q = .02$, and the dice could have these probabilities for $\{1,2,3,4,5,6\}$:

$$B = \begin{pmatrix} .3 & .03 & .03 & .3 & .04 & .3 \\ .03 & .3 & .3 & .03 & .3 & .04 \end{pmatrix} \quad \left\{ \begin{array}{c} \text{die } 1 \\ \text{die } 2 \end{array} \right. $$

...and all happening hidden behind a curtain.
Hidden Markov models

Data: We are told the sequence of die rolls obtained by the frog.

Things we would like to guess, given the data:

\[ p, q, \quad B = \begin{pmatrix} b_{11} & b_{12} & b_{13} & b_{14} & b_{15} & b_{16} \\ b_{21} & b_{22} & b_{23} & b_{24} & b_{25} & b_{26} \end{pmatrix}, \]

and the hidden path of 400 states.
JAGS model for HMM

model{
    x[1] ~ dcat(pi1)
    for(i in 2:n){x[i] ~ dcat(A[x[i-1],])}
    for(i in 1:n){y[i] ~ dcat(B[x[i],])}

    for(i in 1:2){pi1[i] <- 0.5}
    p ~ dunif(0,1)
    q ~ dunif(0,1)
    A[1,1] <- 1-p
    A[1,2] <- p
    A[2,1] <- q
    A[2,2] <- 1-q

    for(j in 1:6){
        b1[j] ~ dexp(1)
        b2[j] ~ dexp(1)
        B[1,j] <- b1[j] / sum(b1)
        B[2,j] <- b2[j] / sum(b2)
    }
}

A = \begin{pmatrix}
    1-p & p \\
    q & 1-q
\end{pmatrix}

B = \begin{pmatrix}
    b_{11} & b_{12} & b_{13} & b_{14} & b_{15} & b_{16} \\
    b_{21} & b_{22} & b_{23} & b_{24} & b_{25} & b_{26}
\end{pmatrix}
Some HMM estimation results

```
> Bmeans = apply(r[,1:12], 2, mean)
> Bmeans = matrix(Bmeans, nrow=2, byrow=T)
> round(Bmeans,2)

[1,] 0.27 0.04 0.02 0.29 0.06 0.32
[2,] 0.02 0.36 0.29 0.03 0.27 0.04
```
Outlook

Parts that seem to be going well...
- Mixture of math, computing, theory, and practice.
- By the end of the class students have learned a lot and many seem to like it; some even feel inspired and empowered.

...and parts I am still working on
- Time management.
- Role and goals of the class, relationship with other classes. Is it OK if they don’t know what a standard P value is?
- Pedagogy: more interactive class, flipping, ...?

Thank you!