SIMPLE MCMC EXERCISE

Generate 200 observations on the logit model

$$P[y_i = 1 \mid x_i, \beta, \gamma] = \frac{e^{x_i\beta+\gamma}}{1+e^{x_i\beta+\gamma}}\,, \tag{1}$$

by first generating a 200 by 1 vector x of variables distributed as i.i.d. draws from a uniform distribution on $(0, 20)$ and setting $\beta = .1$, $\gamma = -1$. Then with the data you have generated on y and x, generate random walk Metropolis draws from the posterior density under a flat prior for β and γ. Try using 1000, 10000, and 100000 draws, and for each case generate trace plots and effective sample size measures. Comment on how many draws seem needed for good convergence.

R can be downloaded at no cost from the internet. Whether you use R, matlab, or octave, you can use the `coda` package, which is also downloadable at no cost. With that package, the plots and effective size calculations are one-line commands. The coda package or something similar may well be available for other environments you might be using, like Python or Julia, but you're on your own for locating them.

R code that does the assignment is below. It is for the sample size 100000 case and would need modification for the smaller sample sizes. You could of course just run one large sequence of draws and then look also at truncated versions, but you might want to start with a smaller size to be sure your code is right and your computer can doe the large MCMC run in reasonable time.

There is no need to use R if you are used to another environment. The code below may be useful as a guide even if you are programming in another language.

Note that in this 2-dimensional example, it would be perfectly possible to evaluate the likelihood at, say, a 100 by 100 grid of points in α, γ space and construct a contour plot to see what the likelihood looks like. This is easier than MCMC, and about as useful. We are doing the MCMC to get a sense of how MCMC works. The grid approach can provide a check to be sure your MCMC runs are behaving well.

Date: March 29, 2018.

The R code:

```
acl <- matrix(0, 1000, 3)
llhgit <- function(y, alph, cnst, x) {
    sum(y * (alph * x + cnst)) - sum(log(1 + exp(alph * x + cnst)))
}

x <- 20*runif(200)
y <- runif(200) > 1/(1 + exp(alph * x + cnst))
y <- as.numeric(y)
acl[1, 3] <- llhgit(y, 0, 0, x)
acl <- rbind(acl[1, ], matrix(0, 99999, 3))
for(it in 2:100000) {
    newac <- acl[it - 1, 1:2] +c(.02 * (runif(1) - .5), .2 * (runif(1) - .5))
    lpnew <- llhgit(y, newac[1], newac[2], x)
    if (exp(lpnew - acl[ it -1, 3]) > runif(1)) {
        acl[it, 1:2] <- newac
        acl[it, 3] <- lpnew
    } else {
        acl[it, ] <- acl[it-1, ]
    }
}
library("coda")
plot(mcmc(acl))
effectiveSize(acl)
```