ADVANCED BAYESIAN MODELING

EXAMPLE OF PRACTICAL MCMC:
RAT TUMOR DATA: MODEL 1

Rat Tumor Example

$$
\begin{aligned}
n_j &= \text{total number of rats in control group of experiment } j \\
y_j &= \text{number in control group of experiment } j \text{ that develop a tumor} \\
\theta_j &= \text{control group tumor probability in experiment } j \\
j &= 1, \ldots, 71
\end{aligned}
$$

Data values for n and y (`N` and `y`) in file `rattumor.txt`

Consider model specified in JAGS code file `rattumor1.bug`:

```
model {

  for (j in 1:length(y)) {
    y[j] ~ dbin(theta[j], N[j])
    theta[j] ~ dbeta(alpha, beta)
  }

  alpha ~ dexp(0.001)
  beta ~ dexp(0.001)

}
```

(Same as analyzed earlier)

This time, instead of using JAGS/`rjags` default options, we will

- use 4 chains
- initialize hyperparameters (α and β) to overdispersed combinations (different for each chain)
- run 1000 iterations for adaptation
- then run 1000 more iterations for burn-in
- finally run 2000 more iterations to check for convergence

Helpful: Determine crude preliminary values for parameters.

These will help determine what values could be considered "overdispersed."

Since the θs are probabilities, naive estimates are the empirical proportions

$$\hat{\theta}_j = \frac{y_j}{n_j}$$

The θs have a $\mathrm{Beta}(\alpha, \beta)$ distribution under the prior. In particular (BDA3, Table A.1):

$$\mathrm{E}(\theta \mid \alpha, \beta) = \frac{\alpha}{\alpha + \beta} \qquad \mathrm{var}(\theta \mid \alpha, \beta) = \frac{\alpha\beta}{(\alpha + \beta)^2(\alpha + \beta + 1)}$$

Solving these for α and β and then substituting the sample mean and variance of the $\hat{\theta}_j$s for $\mathrm{E}(\theta \mid \alpha, \beta)$ and $\mathrm{var}(\theta \mid \alpha, \beta)$ gives ...

```
> d <- read.table("rattumor.txt", header=TRUE)

> thetahat <- d$y/d$N

> Etheta <- mean(thetahat)

> Vtheta <- var(thetahat)

> ( alphahat <- Etheta * (Etheta * (1 - Etheta) / Vtheta - 1) )
[1] 1.373948

> ( betahat <- alphahat * (1/Etheta - 1) )
[1] 8.573899
```

These values will help to initialize the chains.

Recall: Initial values should be overdispersed – extremely large or small.

Hyperparameters α and β must be positive, so let's choose initial values about 100 times larger and smaller than we expect.

For the 4 chains:

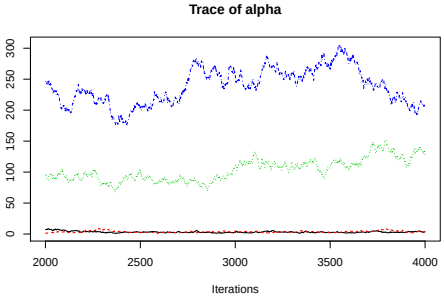
	Initial α	Initial β
Chain 1:	0.01	0.1
Chain 2:	100	0.1
Chain 3:	0.01	1000
Chain 4:	100	1000

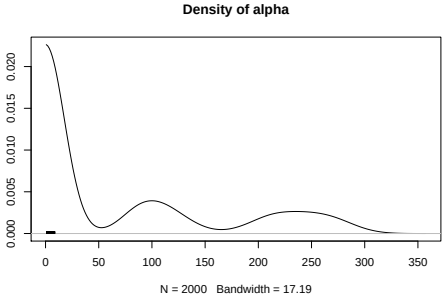
Set up JAGS model in R with 4 chains initialized, and perform 1000 iterations of adaptation:

```
> library(rjags)  # automatically loads coda package
...

> initial.vals <- list(list(alpha=0.01, beta=0.1),
+                      list(alpha=100,  beta=0.1),
+                      list(alpha=0.01, beta=1000),
+                      list(alpha=100,  beta=1000))

> m1 <- jags.model("rattumor1.bug", d, initial.vals, n.chains=4, n.adapt=1000)
...
```

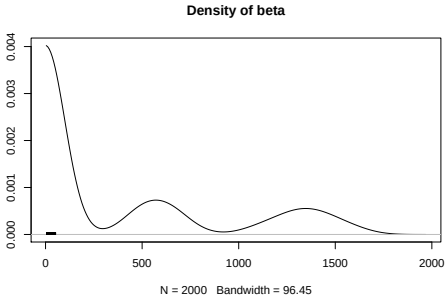
Perform 1000 iterations of burn-in, collect α and β for 2000 more iterations, and plot results:

```
> update(m1, 1000)  # burn-in
  |**************************************************| 100%

> x1 <- coda.samples(m1, c("alpha","beta"), n.iter=2000)
  |**************************************************| 100%

> plot(x1, smooth=FALSE)
```


Trace of alpha
0
50
100
150
200
250
300
2000
2500
3000
3500
4000
Iterations


Density of alpha
0.000
0.005
0.010
0.015
0.020
0
50
100
150
200
250
300
350
N = 2000 Bandwidth = 17.19


Trace of beta
0
500
1000
1500
2000
2500
3000
3500
4000
Iterations

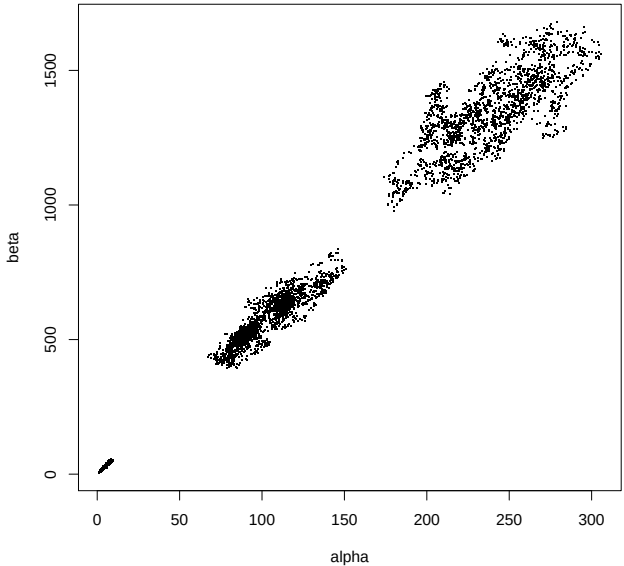

Density of beta
0.000
0.001
0.002
0.003
0.004
0
500
1000
1500
2000
N = 2000 Bandwidth = 96.45

Chains are far from convergence, even after 4000 iterations.

Could run more iterations (try yourself), but don't expect much improvement.

For a Gibbs sampler, slow convergence might be due to high posterior correlations:

```
> plot(as.matrix(x1)[,c("alpha","beta")], pch=".", cex=2)
```


beta
0
500
1000
1500
0
50
100
150
200
250
300
alpha

Results call into doubt earlier (default) analysis: There are regions of possibly high probability that were not sampled.

Fundamental problem: Priors were chosen close to flat for α and β, but using actual flat priors would give an improper posterior.

Better to try a more well-behaved prior ...