Running MCMC in JAGS via `runjags`

Stat 340, Fall 2021

For the remainder of the term we will use Just Another Gibbs Sampler (JAGS) to draw MCMC samples from our posterior distributions. In class, we won't be able to discuss every distribution or function you may need to specify you JAGS model, so be sure to keep the URL to the JAGS user's manual handy: https://github.com/aloy/math315-fall2019/blob/master/jags_user_manual.pdf.

Example: Survival analysis

Researchers are interested in modeling the survival times, measured in weeks, of patients who were diagnosed with leukemia. The patients were classified according to two characteristics of white bloods cells. In this example, we will only examine those whose blood is AG+. The sample consists of $n = 17$ times in weeks from diagnosis to death.

```r
times <- c(65, 156, 100, 134, 16, 108, 121, 4, 39, 143, 56, 26, 22, 1, 1, 5, 65)
```

After discussing the problem with the researchers, you decide to use the Weibull distribution to model the survival time of patient i, Y_i. The Weibull PDF and CDF are given below for your reference, but are available as functions in JAGS.

$$
\begin{aligned}
f(y|v, \lambda) &= \lambda \alpha y^{\alpha-1} e^{-\lambda y^{\alpha}}, \quad y, \alpha, \lambda > 0 \\
F(y|v, \lambda) &= 1 - e^{-\lambda y^{\alpha}}
\end{aligned}
$$

In addition, you decide to place independent gamma priors on λ and α.

Step 0. Load `runjags`

```r
library(runjags)
```

Step 1. Specify the model as a string

```r
model_string <- "
model{
  # Specify the likelihood
  for(i in 1:n) {
    y[i] ~ dweib(alpha, lambda)
  }

  # Specify the priors
  alpha ~ dgamma(1, 1)
  lambda ~ dgamma(1.53, 26.3)
}
"
```

Step 2. Define/load the data

First, let's get the data into R. Here, we're directly entering the data, but you'll typically load it from a .csv file.

```r
survival_times <- c(65, 156, 100, 134, 16, 108, 121, 4, 39, 143, 56, 26, 22, 1, 1, 5, 65)
```

Next, store your data in a list. Looking at our above `model_string`, JAGS will want a `y` vector and the number `n`.

```r
survival_data <- list(
  y = survival_times,
  n = length(survival_times)
)
```

Remember, JAGS likes things to be organized into lists!

Step 3. Define initial values

- If you don't specify initial values for your parameters, then JAGS will
- If not specified, then JAGS will use the mean or mode of the prior distribution
- More on this next time...

Step 4. Draw posterior samples using `run.jags()`

```r
posterior <- run.jags(
  model = model_string,
  n.chains = 1,
  data = survival_data,
  monitor = c("alpha", "lambda"),
  adapt = 1000,
  burnin = 5000,
  sample = 5000 #<<
)
```

- `model` string specifying the model
- `n.chains` the number of chains to run
- `data` a named list or data frame include the data and prior parameter values
- `monitor` character vector of the names of variables to monitor
- `adapt` number of samples drawn during initial sampling/adaptation phase
- `burnin` number of burn-in iterations, NOT including the adaptive iterations
- `sample` the total number of (additional) samples to draw

Step 5. Inspect the results

`runjags` provides some useful functions, such as `plot()` and `summarize()`.

```r
summary(posterior)
plot(posterior)
```

That said, for plotting posterior draws, I prefer to use functions in the `bayesplot` R package. Run the following code to explore what the functions do.

```r
library(coda)      # for as.mcmc
library(bayesplot) # for nice plotting functions


# Need to convert first
post_mcmc <- as.mcmc(posterior)

# Creating plots
mcmc_trace(post_mcmc)
mcmc_acf(post_mcmc)
mcmc_dens(post_mcmc)
mcmc_hist(post_mcmc)
mcmc_areas(post_mcmc)
mcmc_intervals(post_mcmc)
mcmc_scatter(post_mcmc)
```

Your turn: Revisiting election results

Now that we walked through one JAGS model, let's see if it makes sense by revisiting the differences in voting patterns between the 2016 and 2020 presidential elections.

Here is the model we wish to fit:

$$\begin{aligned}
Y_i | \mu, \sigma &\overset{\text{iid}}{\sim} \mathcal{N}(\mu, \sigma) \\
\mu &\sim \mathcal{N}(\mu_0, \sqrt{1/\phi_0}) \\
\phi = 1/\sigma^2 &\sim \text{Gamma}(a, b) \\
\mu &\perp \phi
\end{aligned}$$

and we'll use the following prior parameters which we decided produced rather uninformative priors:

$$\mu_0 = 0, \qquad \phi_0 = 1/1000, \qquad a = 0.1, \qquad b = 0.1.$$

To begin, load in your data

```
votes <- read.csv("https://aloy.github.io/stat340-notes-fa21/data/pres_pct_change_2016_2020.csv")
pct_change <- votes$pct_change_dem
```

and draw your random sample of 100 counties

```
set.seed(23537095)
sampled_change <- sample(na.omit(pct_change), 100)
```

Now, complete the following code chunk (i.e., fill in the blanks) to do the following:

i. Specify the model as a text string.
ii. Pass the model to JAGS, using 2000 burn-in samples and drawing 5000 MCMC samples after burn-in. Be sure to monitor μ and σ.

```
# Data management
y <- sampled_change
n <- length(y)

mcmc_data <- ___(y = y, n = n)

# Specify the model
vote_model_string <- "
model{
  # Specify the likelihood
  for(i in 1:n) {
    y[i] ~ dnorm(mu, phi) # JAGS uses precision, not SD
  }

  # Specify the priors
  mu ~ ___(0, 1/1000)
  phi ~ ___(.1, .1)

  # Calculate sigma to monitor
  sigma <- ___ / sqrt(___)
}
"

# Run MCMC via JAGS
vote_post <- ___(
  model = ___,
  n.chains = 1,
  data = ___,
  monitor = c(___, ___),
  adapt = 1000,
  burnin = ___,
  sample = ___,
  silent.jags = TRUE  # Eliminates progress bar
)

# Summarize the results
summary(___)

# Choose your favorite plotting method to look at the results
```

Follow-up questions:

1. Have the chains for μ and σ converged? How do you know?

2. Report the posterior mean and SD for each parameter.