Bayesian Analysis

Computer Lab 1 – Introduction to JAGS

1 What is JAGS?

JAGS is a free, open-source, cross-platform computer programme for doing Bayesian inference. Specifically, it samples from the posterior of a Bayesian model by using Gibbs Sampling (hence the acronym Just Another Gibbs Sampler). JAGS *only* performs the sampling, thus another program is required to process the output, etc. We will use the free software package `R` for this purpose as both JAGS and `R` run virtually identically on all common platforms (Windows, Mac, linux).

2 Downloading and installing JAGS

2.1 `R` and JAGS on a USB

To use the lab computers you will probably have to run JAGS and `R` from a USB as follows:

1. Copy the R-JAGS-USB.zip file from the Computer Labs blackboard folder.
2. Copy it to your USB and unzip it. There should be two folders, JAGS and R-3.3.2
3. You can now start `R` by double clicking on `R-3.3.2/bin/x64/Rgui.exe`
4. You need to tell `R` where the JAGS files are. Do this by running the following command in `R`:
   `Sys.setenv(JAGS_HOME="F:/R-JAGS-USB/JAGS/")`
   assuming that F is the name of the USB drive.
5. Load the rjags package in `R` by running the command
   `library(rjags)`
   (You can use the Packages dropdown menu, select "Load Package", then select rjags if you prefer.)

2.2 `R` and JAGS on your laptop

`R` can be installed from `https://ftp.heanet.ie/mirrors/cran.r-project.org/`
JAGS can be installed from `http://mcmc-jags.sourceforge.net/`

3 Getting help

The help menu in JAGS provides a number of useful links:

- The JAGS user manual. This is the main port of call for any problems or error messages. Perhaps the most useful section is that of the probability distributions implemented by JAGS in Table 5.3, page 27.

4 Format and Syntax of JAGS

Task:

- Download the file `dating.model` from the Computer Labs folder on blackboard.

The file dating.model (which implements the dating volcanic rock example we met in Lecture 2) specifies a Normal prior and likelihood. It can be opened with a text editor or in RStudio.
JAGS uses code similar to (but not exactly the same as) the `R` environment. One difference is that the order in which commands are issued is relatively unimportant. The specific requirements for your code to run are:

- The model section must start with the line `model` and contain any subsequent code in curly brackets.
- If a variable is to follow a distribution, the term $\sim$ must be used.
- If a variable is to equal a function of another variable, then `<-` must be used. This is the JAGS equivalent of the equals sign.
- If a distribution is to be used, it is usually prefixed with a `d`. For example, the normal distribution is `dnorm`, the gamma distribution is `dgamma`. More distributions can be found in the relevant section of the user manual. Remember: when specifying a normal distribution in JAGS the second argument is the precision rather than the variance. Be careful!
- Comments can be added by prefixing them with `#`.

5 Running a JAGS model in `R` (from package rjags)

We'll be using `R` to interface with JAGS (run, summarise output, plot, etc). If you haven't used `R` then a good place to start is `https://cran.r-project.org/manuals.html`
The most important `R` functions for us will be `jags.model()` and `jags.samples()`.

5.1 jags.model

In `jags.model` there are just two arguments required:

- *file* a file containing details of the likelihood and prior to be fitted. This is the above JAGS file.
- *data* a list containing the data and other fixed parameters. Vector observations can be included via, eg, `data1=list(x=c(2,4,6,8),y=c(1,3,5,7))`.

There are then several optional parts that can be passed to the model:

- *inits* a list of initial values, for specifying a first guess as to the values of the parameters.
- *n.chains = 1* the number of parallel chains for the model.
- *n.adapt=1000* the number of iterations for adaptation. JAGS may run an initial sampling phase during which the samplers adapt their behaviour to maximize efficiency.

5.2 jags.samples

In `jags.samples` there are just three arguments required:

- *model* the output of `jags.model` above.
- *variable.names* the names of the variables (surrounded by quotes) that are to be sampled.
- *n.iter* the number of samples required.

(Note: in later lectures we will cover the methods used to create these simulated values.)

Task:

- Create the data list by running the command
  `volcano=list(x=421, phi=64, theta0=370, phi0=400)`
- Create the JAGS model by running the command
  `jmodel=jags.model(file="dating.model", data=volcano)`
- Create 1000 samples from the posterior using the command
  `samps=jags.samples(jmodel,"theta",n.iter=1000)`
- Familiarise yourself with the process for running the model! This is going to be the same each time.

6 Analysing the JAGS output

After running the model for a number of iterations, we can explore the output.

Tasks:

- Plot the sample density for an estimate of the posterior probability density for theta
  `plot(density(samps$theta))`
- Plot the trace or history of the values sampled by JAGS from the posterior
  `plot(samps$theta,t='l')`
- Calculate the mean and standard deviation of the sampled values
  `mean(samps$theta)`
  `sd(samps$theta)`

7 Going Further

Tasks:

- Run the model for 100 iterations and look at the density and summary statistics. Run the model for a further 10,000 iterations and confirm that the model is converging to approximately the correct values we found in class ($N(414, 7.4^2)$)
- Try changing the initial values and look at the effect on the summary statistics. Does it affect your posterior mean and standard deviation?
- Try changing the prior values (ie `theta0` and `phi0`) and look at the effect on the posterior mean and standard deviation. What happens if the prior is very precise/diverse?