Bayesian estimation software

PUBH 8442: Bayes Decision Theory and Data Analysis

Eric F. Lock
UMN Division of Biostatistics, SPH
elock@umn.edu

04/8/2024

Bayesian estimation software

- Several stand-alone applications and add-ons to estimate Bayesian models
- Stand-alone applications:
  - Bayesian Inference using Gibbs Sampling (BUGS): winBUGS, openBUGS, and multiBUGS
  - Just Another Gibbs Sampler (JAGS), maintained by Martin Plummer and others
  - Stan, maintained by Andrew Gelman and others
- Packages in R
  - Package MCMC, maintained by Charles Greyer
  - Package Nimble, maintained by Chris Paciorek and others
  - Packages to run openBUGS, rjags, or STAN through R

winBUGS, openBUGS, and multiBUGS: history

- The BUGS project started in early 1990's from the MRC Biostatistics Unit, Cambridge
- WinBUGS first introduced in mid-90s
  - Became widely adopted software for Bayesian modeling in many application areas
- openBUGS was introduced in approximately 2009
  - Very similar to winBUGS, but open-source and slightly more flexible
- multiBUGS was introduced around 2020
  - Similar to openBUGS, but uses parallel computing to speed up computation.
- Development is now focused on multiBUGS

JAGS

- Just Another Gibbs Sampler (JAGS)
- First developed by Martyn Plummer in early 2000's
- Based on the BUGS language, with many similarities
- Usable on any operating system (windows, mac, linux)
- Well-integrated with R through the `rjags` package
- I cannot get OpenBUGS/MultiBugs to work on my Mac laptop (OS: Ventura)
  - rjags code for the examples in this slide deck: https://www.ericfrazerlock.com/rjags_examples.r

openBUGS: Installing

- Download openBUGS from this link.
- Straightforward in Windows or Linux
- For a Mac, this used to work for me:
  - Download Wine.app from the Wine Bottler website
  - Install Wine.app – this will facilitate running Windows programs.
  - Download the windows .exe file from openBUGS
  - Run this .exe file to install openBUGS (double-clicking should open it in Wine)
  - Run the OpenBUGS.exe from the application folder (again, this should open automatically in Wine)

openBUGS: input

- openBUGS is a mix of "point-and-click" menus and command code
- Input for a given application consists of the *model*, *data*, and *initialization*
- The *model* is given in BUGS language, which has syntax similar to R
  - Define sampling model, and the distribution for each parameter
  - BUGS recognizes several common distributions
  - Example syntax for normal-normal model with flat prior:

```
model{
  for(i in 1:n){
    y[i] ~ dnorm(mu, Prec) #2nd parameter is precision, NOT
variance
  }
  mu ~ dflat()
}
```

openBUGS: input

- Data and fixed parameters are specified in list form, with syntax similar to R:
  list(Prec=0.5, y=c(2.6,1.2,-0.4,3.7,3.1),n=5)

- Initial values are also given in list form
  list(mu = 0)

  - It is often not required to specify initial values, as these are provided automatically by openBUGS.

  - However, the default initializations may be poor

openBUGS: estimation approach

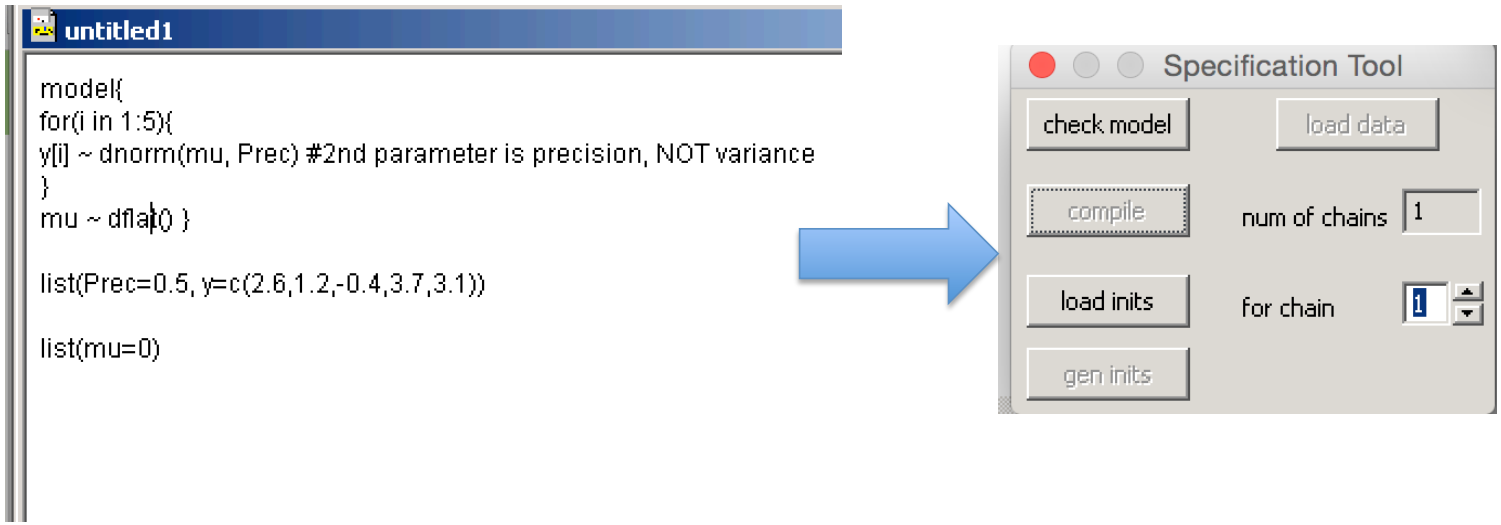
- openBUGS estimates the posterior through Gibbs sampling and MH-sampling
- Several known conjugate posteriors are recognized, allowing Gibbs sampling directly from full conditionals
- MH proposals used for unknown full conditionals
- User can specify
  - Number of MCMC chains (potentially with different initial values)
  - Total number of iterations and burn-in
  - Thinning number l - This will select every l'th draw for posterior inference

openBUGS: output

- Output for posterior inference:
  - Percentiles, summary statistics, and MCMC error for posterior draws
  - Kernel density estimates based on posterior draws
- Output for MCMC diagnostics:
  - Trace plots of mcmc draws ('history' option gives full chain)
  - Acceptance rate
  - Autocorrelation plots
- Output for model assessment:
  - Deviance information criterion (DIC) – will discuss this more later!

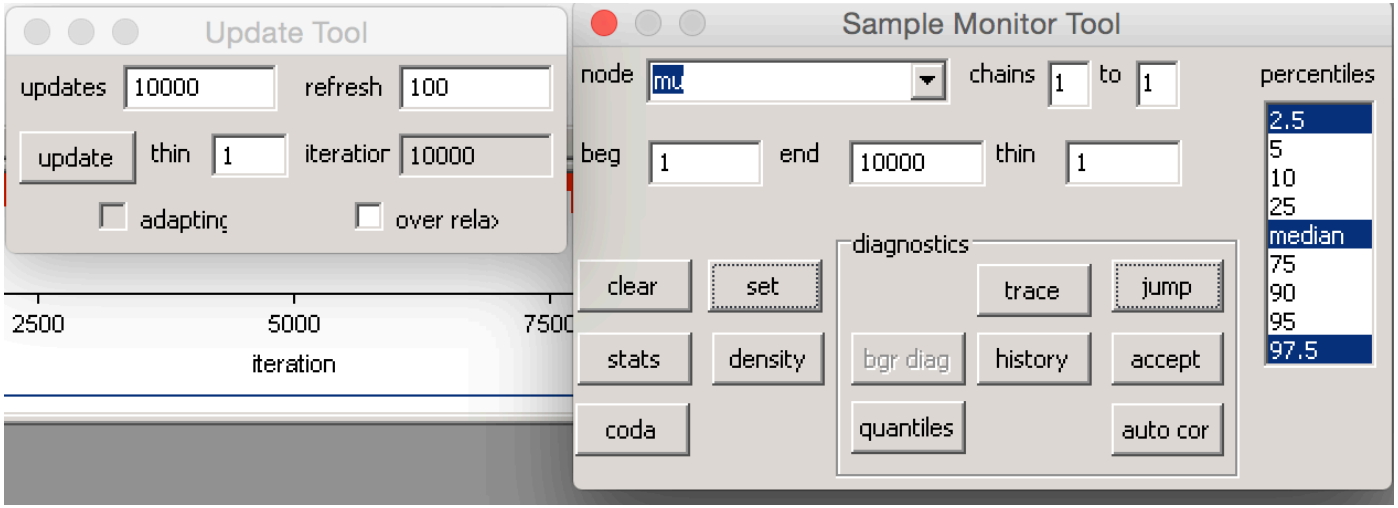
openBUGS screenshots

- Specify model, etc. in editor
- Select Model → Specification, then check, load, and compile model
  - Highlight relevant portion of text before each button



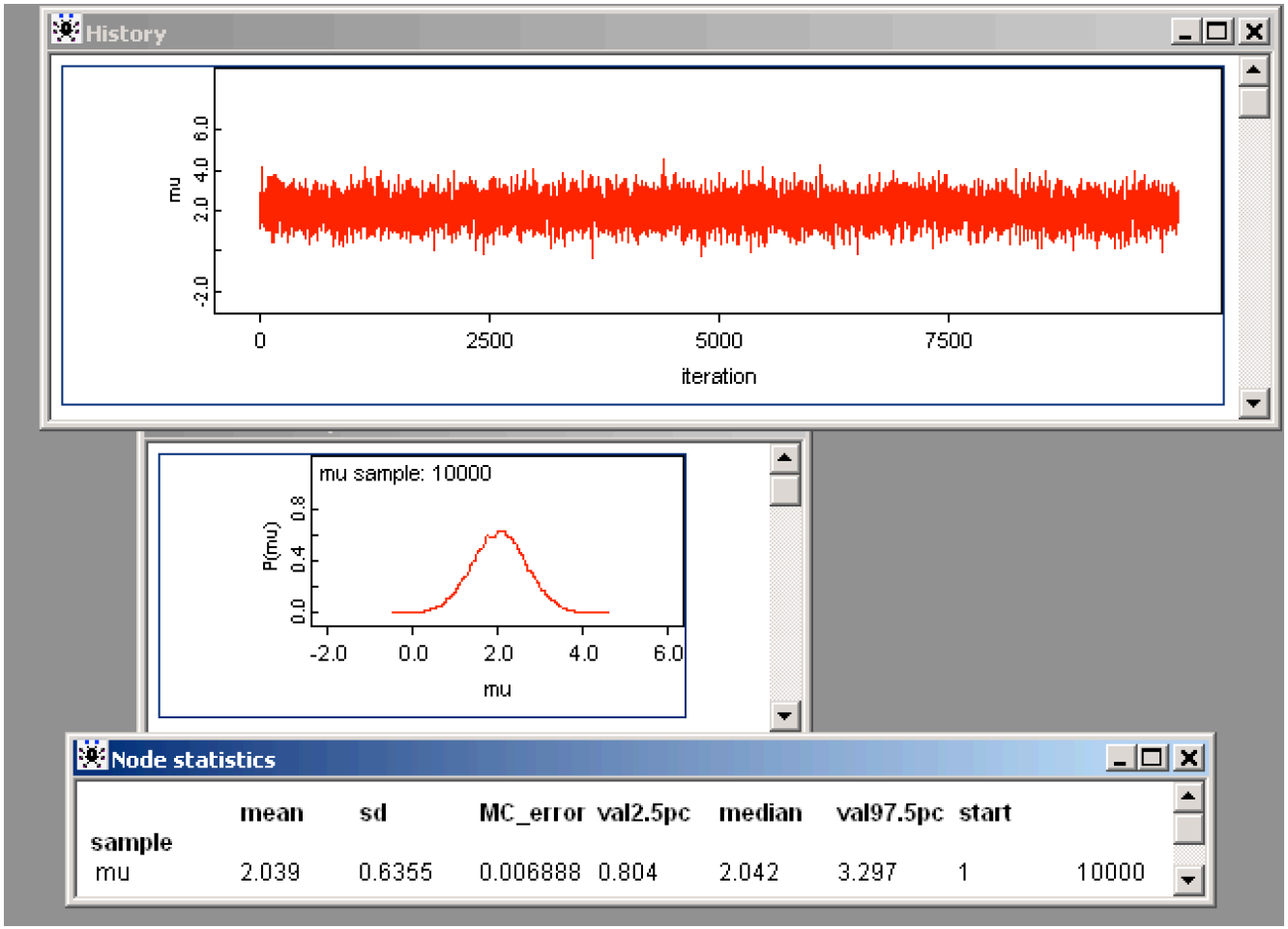
openBUGS screenshots

- For estimation select Model→Update Tool and Inference→Sample Monitor Tool
- Sample Monitor allows you to specify nodes (variables to track) and create output, Update Tool actually runs the MCMC



openBUGS screenshots

- Output:



Example: Beetle toxicity

- Flour beetles are exposed to a toxic substance at 4 different concentrations[1]
- Let x_i be the log-concentration for $i = 1, \ldots, 4$
- n_i is the number of beetles exposed at concentration i
- y_i is the number of beetles who die
- Model:

$$y_i \mid \theta_i \sim \text{Binomial}(\theta_i, n_i)$$

$$\log\left(\frac{\theta_i}{1-\theta_i}\right) = \alpha + \beta x_i$$

$$p(\alpha, \beta) = 1 \times \text{Normal}(\beta | 0, 1000)$$

[1] Example and data inspired by
http://math.tut.fi/~piche/bayes/winbugs_lecture.pdf

Example: Beetle toxicity

- Given data:

x_i	n_i	y_i
-0.863	5	0
-0.296	5	1
-0.053	5	3
0.727	5	5

- We are interested in LD50, the log-concentration at which 50% of beetles die

$$LD50 = -\frac{\alpha}{\beta}$$

- Estimate in openBUGS, running MCMC from two different initializations

Example: Beetle toxicity

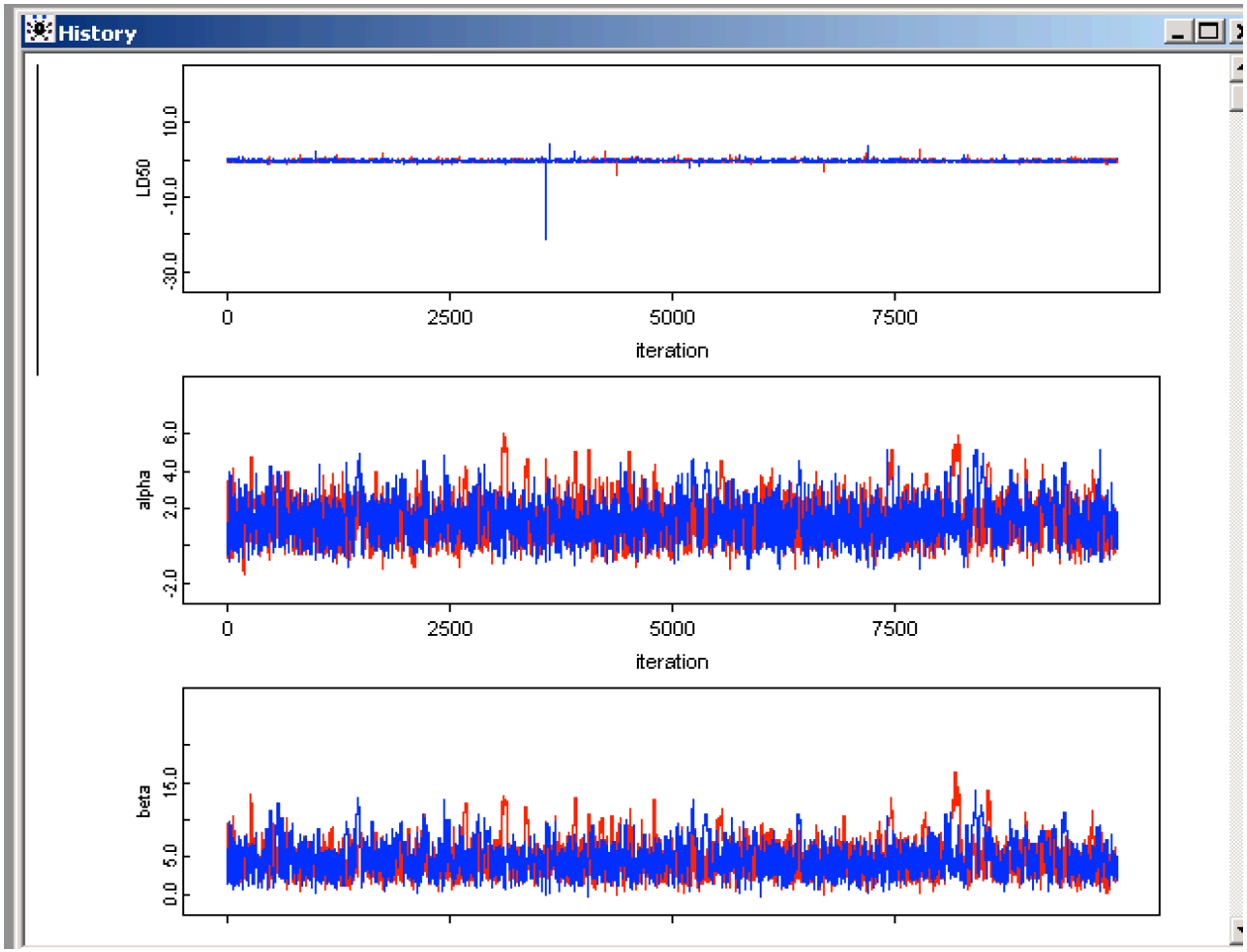
- BUGS code:

```
model{
   for(i in 1:nx){
     logit(theta[i]) <- alpha+beta*x[i]
     y[i] ~ dbin(theta[i],n[i])
  }
  alpha ~ dflat()
  beta ~ dnorm(0, 0.001)
  LD50 <- -alpha/beta }

#data
list(y=c(0,1,3,5), n=c(5,5,5,5), x=c(-0.863,-0.296,-0.53,0.727), nx=4)
#init 1:
list(alpha=0, beta=0)
#init 2:
list(alpha=0,beta=5)
```

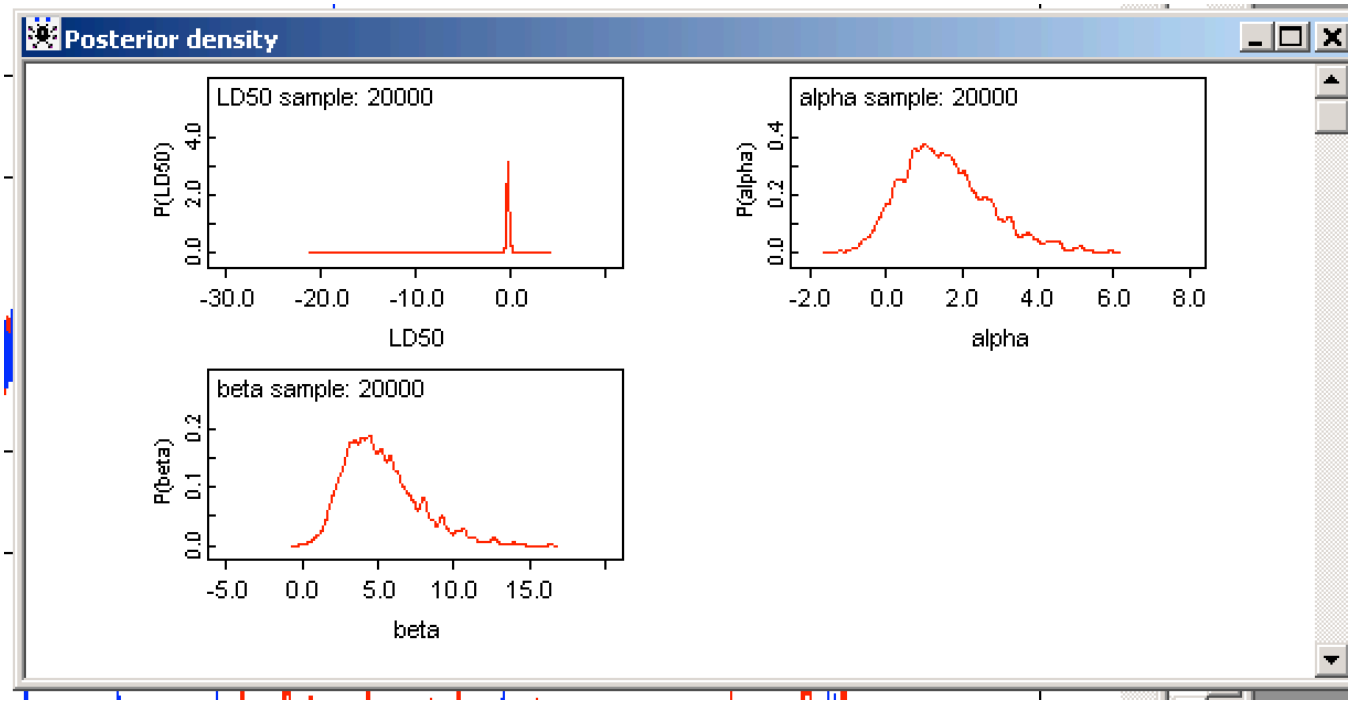
Example: Beetle toxicity

- Output:



Example: Beetle toxicity

- Output:



Example: Beetle toxicity

- Estimated LD50 is -0.2744, with 95% credible interval

$$LD50 \in (-0.5529, 0.1503)$$

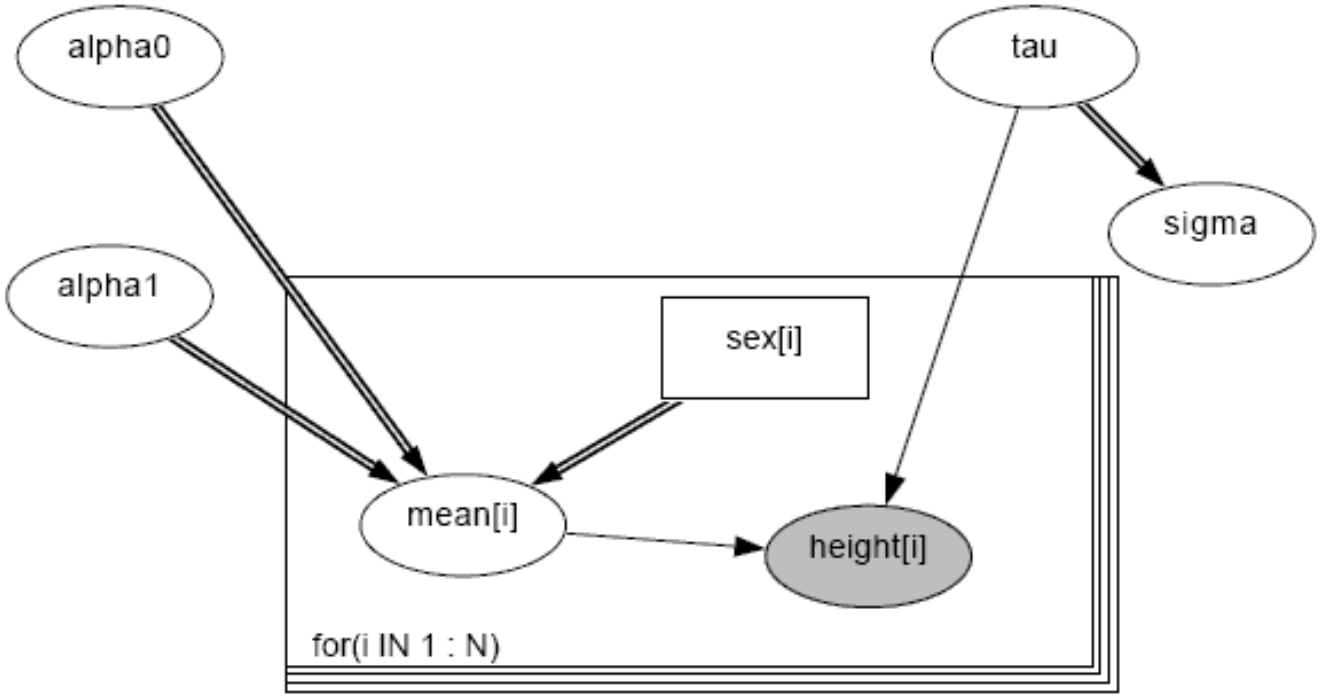
- Output:

Node statistics

	mean	sd	MC_error	val2.5pc	median	val97.5pc	start	sample
LD50	-0.2744	0.2554	0.003445	-0.5529	-0.2977	0.1503	2000	16002
alpha	1.586	1.198	0.04394	-0.3457	1.441	4.399	2000	16002
beta	5.245	2.437	0.09425	1.644	4.837	10.97	2000	16002

openBUGS doodle

- As an alternative to specifying model in BUGS syntax, the Doodle option can specify a hierarchical model graphically
- Example (http://web.engr.oregonstate.edu/~tgd/classes/519/assignment1.html):



R to BUGS

- For tips in saving data in R to BUGS format check out 'FROM R to WinBugs' at http://www.public.iastate.edu/~alicia/stat544/software.htm [link broken]
- Also see the R2OpenBUGS package: https://cran.r-project.org/web/packages/R2OpenBUGS/index.html
- Another good alternative option is RJAGS: https://cran.r-project.org/web/packages/rjags/index.html
- See course website for additional resources