

# Introduction to Bayesian Statistics with WinBUGS

## Part I—Introduction

Matthew S. Johnson  
Teachers College; Columbia University  
`johnson@tc.columbia.edu`

New York ASA Chapter  
CUNY Graduate Center  
New York, NY

December 17, 2009

# Outline of Today's Workshop

- ▶ Introduction to Bayesian Inference with WinBUGS
  - ▶ A review of classical/frequentist statistics.
  - ▶ The likelihood, prior and posterior.
  - ▶ Bayesian modeling in WinBUGS, OpenBugs, and JAGS.
- ▶ Monitoring convergence and making inferences from MCMC output.
  - ▶ Monte Carlo approximations for inference.
  - ▶ MCMC algorithms.
  - ▶ Monitoring convergence.
- ▶ Prior Selection, Bayesian hierarchical models and DoodleBugs.

# The goals of inference

- ▶ Statistical inference attempts to understand the random mechanism by which a given set of data (denoted  $\mathbf{y}$ ) was generated.
- ▶ We assume some probabilistic sampling model to describe the random phenomenon. The probability density (or mass) function is denoted by  $f_{\mathbf{y}|\theta}$ 
  1. The model may include information from *explanatory* variables such as predictors in a regression, etc.
  2. The model is typically defined by a small number of *parameters* (small relative to the sample size). Throughout this workshop, I will generically denote parameters by  $\theta$ .
- ▶ The sampling model defines the likelihood function

$$L(\theta; \mathbf{y}) \equiv f(\mathbf{y} | \theta) = \prod_{i=1}^n f(y_i | \theta)$$

# Classical Point Estimation

1. Assumes parameters are fixed quantities, but the values are unknown.
2. Point estimates are obtained in a variety of ways:
  - 2.1 Maximum likelihood estimates (MLEs) are the values of the parameters that maximize the likelihood function, i.e, they are the values of the parameters that maximize the joint density function of the observed data.
    - 2.1.1 MLEs are consistent.
    - 2.1.2 MLEs are asymptotically normal.
  - 2.2 Method of moments estimates estimate the moments of a parametric distribution with the moments of the empirical distribution. Under regularity conditions
    - 2.2.1 MOMs are consistent.
    - 2.2.2 MOMs are asymptotically normal.
  - 2.3 Uniform minimum variance unbiased estimates (UMVUE) and best linear unbiased estimates (BLUE). Usually difficult to find in most problems. Is unbiasedness really an important property to have?

# Frequentist Hypothesis Testing

3. The “precision” of an estimate is often measured with the mean squared error  $E[(\theta - \hat{\theta})^2]$ , which can be decomposed into the variance of the estimator and the squared bias of the estimator.
4. Hypothesis testing—Assume the null hypothesis is true and try to show that the data observed is unlikely to have come from the model under the null hypothesis.
  - 4.1 What is the null hypothesis and what is the alternative hypothesis?
  - 4.2 Should we ever believe a point null hypothesis, e.g.,  
 $H_0 : \mu = 0$ ?
  - 4.3 P-values are often misinterpreted as, “the probability that the null hypothesis is true.” The p-value is the probability that we would have observed data “more extreme” than the data at hand, if the null hypothesis were true.

# Frequentist Confidence Intervals

5. Confidence Intervals—A range of values for the unknown parameter that would not be rejected in a hypothesis test. In other words, the data would not be “extreme” if the true value of the parameter was one contained in the confidence interval.
  - 5.1 The random quantity is the interval not the parameter value.
  - 5.2 How do we correct for creating several intervals?  
Multidimensional confidence ellipses are rarely examined.
  - 5.3 Confidence intervals are often misinterpreted in the same way that hypothesis tests are misinterpreted, by discussing the parameter as the random quantity.

# Frequentist Prediction Intervals

6. Prediction Intervals—An interval estimate for a randomly selected future value drawn from the same distribution the data was drawn from.
- ▶  $(1 - \alpha)100\%$  of the prediction intervals constructed from the first  $n$  observations will contain the  $(n + 1)$ th observation.
  - ▶ There are two random components
    - ▶ The interval constructed from the  $n$  sampled observations.
    - ▶ The future observation.
  - ▶ Prediction intervals are difficult to construct unless the data was drawn from the normal distribution, or some other simple distribution.

# The Bayesian Approach

## The Random Components of a Bayesian Model

- ▶ The Bayesian approach assumes that the data was generated by some random mechanism ( $f_{y|\theta}$ ). That random mechanism defines the joint probability distribution of the data given the unknown parameters, i.e., the likelihood.
- ▶ The Bayesian also treats the unknown parameters as random quantities, and therefore must define a (prior) distribution ( $f_\theta$ ) that describes the uncertainty we have about the parameter values.

# The Posterior Distribution

Bayesian inferences are based on the *posterior distribution*, the conditional distribution of the parameter vector  $\theta$  given the data  $\mathbf{y}$ :

$$\begin{aligned}f_{\theta|\mathbf{y}}(\theta | \mathbf{y}) &= \frac{f_{\mathbf{y},\theta}(\mathbf{y}, \theta)}{f_{\mathbf{y}}(\mathbf{y})} \\ &= \frac{f_{\mathbf{y}|\theta}(\mathbf{y} | \theta)f_{\theta}(\theta)}{\int f_{\mathbf{y}|\theta}(\mathbf{y} | \theta)f_{\theta}(\theta)d\theta},\end{aligned}$$

which is exactly *Bayes Theorem*.

- ▶  $f_{\mathbf{y},\theta}(\mathbf{y}, \theta)$  is the joint density of the parameter and the data.
- ▶  $f_{\mathbf{y}}(\mathbf{y})$  is the marginal distribution of the data.

## More on the Posterior

$$f_{\theta|y}(\theta | \mathbf{y}) = \frac{f_{y,\theta}(\mathbf{y}, \theta)}{f_y(\mathbf{y})}$$

Given the data  $\mathbf{y}$ , the denominator in the definition of the posterior density is simply constant with respect to the quantity of interest ( $\theta$ ).

- ▶ The term ensures that the density  $f_{\theta|y}$  integrates to one.
- ▶ It is often convenient to write

$$\begin{aligned} f_{\theta|y}(\theta | \mathbf{y}) &\propto f_{y,\theta}(\mathbf{y}, \theta) \\ &\propto f_{y|\theta}(\mathbf{y} | \theta) f_{\theta}(\theta) \end{aligned}$$

and to worry about the proportionality constant later.

## Example 1

Consider the problem of estimating the probability that a respondent recalls an advertisement. A market researcher runs an experiment with three subjects. Two correctly recall the ad, and one does not.

A perfectly reasonable model for this data is the iid Bernoulli model. Suppose  $y_i$  denotes the observed binary response for subject  $i$ .

Then, the conditional density of the data vector  $\mathbf{y}$  given the parameter  $p$  is

$$f(\mathbf{y} \mid p) = p^2(1 - p)$$

## Example 1 Continued

For the Bayesian analysis we will assume a uniform prior on the success probability,  $p \sim U(0, 1)$ ,  $f(p) = 1$ .

The posterior density satisfies

$$\begin{aligned} f(p | \mathbf{y}) &\propto f(\mathbf{y} | p)f(p) \\ &\propto p^2(1 - p) \times 1 \end{aligned}$$

which is satisfied by the Beta(3,2) density, i.e.

$$\begin{aligned} p | \mathbf{y} &\sim \text{Beta}(3, 2) \\ f(p | \mathbf{y}) &= \frac{4!}{2!1!} p^2(1 - p) \end{aligned}$$

## Example 2

Consider a slightly more complicated example, where for we record recall and the age of each of the subjects (now 5).

Age ( $x$ )	10	30	50	70	90
Result	1	1	1	0	0

We want to perform inferences on the logistic regression model

$$f(y_i | p_i) = p_i^{y_i} (1 - p_i)^{1 - y_i}$$

$$\text{logit}(p_i) = \alpha + \beta x_i$$

## Example 2 Continued

The likelihood for this problem is:

$$f_{\mathbf{y}|\theta}(\mathbf{y} \mid \alpha, \beta, \mathbf{x}) = \frac{\exp\{3\alpha + 90\beta\}}{\prod_i [1 + \exp\{\alpha + \beta x_i\}]}$$

If we assume independent standard normal priors for  $\alpha$  and  $\beta$  the posterior distribution satisfies:

$$f_{\theta|\mathbf{y}}(\alpha, \beta \mid \mathbf{y}) \propto \frac{\exp\{3\alpha + 90\beta\}}{\prod_i [1 + \exp\{\alpha + \beta x_i\}]} \phi(\alpha)\phi(\beta)$$

There is no simple distribution that satisfies the proportionality constraint.

# Overview of MCMC

The goal of Markov chain Monte Carlo (MCMC) methods is to simulate a random walk process in the parameter space  $\boldsymbol{\theta}$  that converges to the joint posterior distribution  $f_{\boldsymbol{\theta}|\mathbf{y}}(\boldsymbol{\theta} | \mathbf{y})$ .

I will discuss three MCMC procedures later in the workshop

- ▶ The Gibbs sampler.
- ▶ The Metropolis-Hastings algorithm.
- ▶ The Slice sampler.

# Notes about MCMC

- ▶ Under relatively mild conditions the distribution of values sampled from MCMC will converge to the posterior distribution.
- ▶ The rate of convergence is not known. We don't know how long it will take until we are “close enough” to the correct distribution.  
We must monitor convergence.
- ▶ The values sampled from the Markov chain are likely to be correlated.  
We must account for correlated draws.

## Software for MCMC

- ▶ **WinBugs 1.4.3** available at <http://www.mrc-bsu.cam.ac.uk/bugs/> is a little out of date at this time, but is recommended for “standard use”
- ▶ OpenBugs, <http://mathstat.helsinki.fi/openbugs/>, is a little more experimental, but also more up-to-date (but still a couple of years old).
- ▶ **JAGS** (Just Another Gibbs Sampler): <http://www-fis.iarc.fr/~martyn/software/jags/> not the same as the BUGS programs, but works pretty much the modelling language is pretty much the same.

I have little experience with JAGS and would recommend OpenBugs over WinBugs, even though they are very similar.

## Using WinBugs and OpenBugs

1. From the `File` menu select `New` to open a new program editor.
2. Specify the Bayesian model in the editor window.
3. Once you specify the model you can `Check Model` by opening the `Specification...` window from under the `Model` pull-down menu.
4. Examine the lower left corner of the WinBugs parent window. This is where the results of your model check are reported. You hope to see the message “model is syntactically correct.” The `Check Model` feature is just checking the syntax. You may run into problems with your model, even though you passed the model check.

WinBugs Example

## Programming the model

Programming a model in WinBUGS is relatively easy. If you can write the model down on paper, then you can write it in BUGS.

*The model statement.*

The specification of a model in BUGS should all be contained within the the curly brackets `{}` of the `model` statement.

```
model {  
    MODEL STATEMENTS  
}
```

## Stochastic Nodes

Stochastic nodes are recognized by the presence of an equation with three components.

- ▶ The name of the random quantity (either data or parameter) appears on the left hand side of the equation.
- ▶ The  $\sim$  symbol appears in the middle, and
- ▶ A function name representing the distribution appears on the right hand side; the parameters of the distribution are passed as arguments to the function.

The distribution names are abbreviated and prefixed by the letter `d`, e.g., `dnorm`.

The following distributions are a subset of those available in BUGS

- ▶ *Normal*.  $x \sim \text{dnorm}(m, \text{prec})$ . **NOTE:** The second parameter of the normal distribution is the precision (1/variance) not the variance or standard deviation.
- ▶ *Bernoulli*.  $x \sim \text{dbern}(p)$ , where  $p$  is the success probability.
- ▶ *Binomial*.  $x \sim \text{dbin}(p, n)$ , where  $p$  is the success probability and  $n$  is the number of trials.
- ▶ *Categorical*.  $x \sim \text{dcat}(p[])$ , where  $p[]$  is a vector of probabilities that sum to one.
- ▶ *Uniform*.  $x \sim \text{dunif}(a, b)$ , where  $a$  &  $b$  are the limits of the uniform.
- ▶ *Beta*.  $x \sim \text{dbeta}(a, b)$
- ▶ *Gamma*.  $x \sim \text{dgamma}(a, b)$

## Logical nodes

Logical or deterministic nodes are those that are deterministic functions of other nodes in the model (i.e., there is no random component).

BUGS recognizes logical nodes by the presence of equation with the following components.

- ▶ The name of the variable (or a link function of the variable, e.g.,  $\text{logit}(p)$ ) on the left hand side.
- ▶ The arrow operator  $< -$  in the middle, and
- ▶ The expression of a mathematical operation on the right hand side. The expression can have the standard operators  $+$ ,  $-$ ,  $/$ ,  $*$ , as well as numerous mathematical functions, including:  $\text{exp}$ ,  $\text{log}$ ,  $\text{sin}$ ,  $\text{cos}$ ,  $\text{mean}$ ,  $\text{sum}$

## Constant nodes

Constant nodes are those that are specified in the data set, but do not appear on the left side of any equation in the model specification.

In general, each (non-constant) node should appear only once on the left hand side of an equation. The only exception is when you are modeling a transformation of the observed data, e.g.,

```
z <- log(x)
```

```
z ~ dnorm(0,1)
```

where  $x$  is an observed variable.

## Using arrays in WinBugs

- ▶ The indices of arrays (e.g, vectors, matrices) are contained within square matrices, e.g.,  $y_i = y[i]$
- ▶ The indices of multidimensional arrays are separated by commas, e.g.,  $X_{ij} = X[i, j]$ .
- ▶  $x[a:b] = (x_a, x_{a+1}, \dots, x_b)$
- ▶  $X[, j]$  represents the  $j$ th column of  $X$ .
- ▶  $X[i, ]$  represents the  $i$ th row of  $X$ .
- ▶  $x[]$  represents the entire vector  $x$ .

## Using loops

We often assume the elements of a vector are (conditionally) iid and so we will be defining the same structure for all elements of the vector (array). For loops come in handy for defining these types of repeated structures.

```
for (i in a:b){  
  x[i] ~ ddist(q,r)  
}
```

## WinBugs Syntax for our Examples

► Example 1

```
model{
  for(i in 1:3){
    y[i] ~ dbern(p)
  }
  p ~ dunif(0,1)
}
```

► Example 2

```
model{
  for(i in 1:5){
    logit(p[i]) <- alpha + beta*x[i]
    y[i] ~ dbern(p[i])
  }
  alpha ~ dnorm(0,1)
  beta ~ dnorm(0,1)
}
```

## Model Specification Errors

Some errors that I have encountered include:

- ▶ **“unknown type of probability density”**: Usually I've just made a typo or had the wrong syntax for a distribution.

```
y[i] ~dnrom(mu, sigma)
```

- ▶ **“unknown type of logical function”**: Typo for a function or I tried to use a function not implemented in WinBUGS.

```
zeta <- cso(4)
```

- ▶ **“expected left pointing arrow < – or twiddles ~.”** Usually has happened to me because I've used = instead of the < – or ~ notation that BUGS uses.

- ▶ **“invalid or unexpected token scanned.”** Usually just means that you've made a typo somewhere in the program.

```
y[i] ~ dnorm(mu, )
```

## More Specification Errors

- ▶ **“expected right parenthesis.”**: This happens when you forgot to close of a parenthesis somewhere or when you are including too many indices in an array.
- ▶ **“BugsVariable2”**: I haven't been able to determine what exactly this error message means. I've encountered it when I forgot to close of square brackets ([]) when indexing an array.  
$$y[i \sim \text{dnorm}(\mu, \sigma)$$

The easiest way to find your error is to look for where BUGS has placed the cursor within the program editor. It usually is very close to where the typo has occurred.

# Task 1

Specify a Bayesian model for a set of 8 exam scores, which are assumed to be normally distributed with unknown mean  $\mu$  and unknown variance  $\sigma^2$ .

- ▶ Assume a  $N(70, 10)$  prior for  $\mu$
- ▶ Assume a  $Gamma(1, 1)$  prior for the precision  $\tau^2 = 1/\sigma^2$ .

Check your model specification with WinBUGS.

## Loading Data/Constant Nodes

5. Once your model syntax has passed it is time to Load Data.
  - ▶ You can either list the data in the same file that contains the model specification or in a separate file/window.
  - ▶ In WinBugs data can be formatted in two ways:
    - (a) **S-format**: The S definition of the data starts with the `list` function, which tells BUGS that you will be listing the values of multiple variables.
    - (b) **Rectangular array format**: Standard format, but column names are a little different, and you must issue an `END` statement at the end of the data (and a return after the `END` statement).

## S/R formatted data

The `list` keyword is followed immediately by a `(`, and the data within that parenthesis and the closing `)`.

- ▶ **Scalar variables.** Within the list function scalars are defined simply with the `x=a` format where `a` is a number or `NA` (denoting missing data).
- ▶ **Vector variables.** The elements of a vector variable are defined within the `c()` function, e.g., `x=c(1,2,3,4)`.
- ▶ **Multidimensional arrays.** Multidimensional arrays are defined as a structure, a vector with `.Data` and `.Dim` attributes, e.g.,  
`x=structure(.Data=c(1,2,3,4,5,6),.Dim=c(2,3))`  
creates

$$x = \begin{pmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \end{pmatrix}$$

Note: Unlike S, BUGS fills the array in by row, not by column.

# Examples of Data Formatting

## S/R Data Formatting

- ▶ Example 1—Two successes, 1 failure.  
`list(y=c(1,1,0))`
- ▶ Example 2—Successes and failures by humidity.  
`list(y=c(1,1,1,0,0),  
x=c(10,30,50,70,90))`

# Examples of Data Formatting

## Rectangular Array

Example 1	Example 2
2 Success, 1 Failure	Success and failure by humidity
y[]	y[] x[]
1	1 10
1	1 30
0	1 50
END	0 70
	0 90
	END

Make sure you include a “return” after END.

## Task 1 Continued

The eight exam scores for our example are:

75, 95, 83, 87, 91, 77, 63, 82

Enter this data into WinBugs.

## Compiling the Model

6. In the Specification Window, type the `num of chains` you would like to run. Multiple chains are helpful for monitoring convergence of the MCMC algorithms. Then click `Compile`.

## Compilation Errors

- ▶ **“index out of range”**: You are trying to access an array element that doesn't exist. Usually means your model assumes there is more data than there actually is. Sometimes it is simply due to quirks of WinBUGS.
- ▶ **“undefined variable”**: You are trying to use a variable on the right hand side of an equation that has not been defined as data or given a prior distribution.

## Task 1 Continued

Enter 5 as the number of chains in the specification window and compile the model.

Was there an error?

## Initial Values in WinBUGS

7. After the model has been compiled the user must provide WinBUGS with initial values for the parameters (any unknown quantity). There are two ways to provide the initial values,
  - ▶ `load inits` allows users to specify their own initial values.
  - ▶ `gen inits` samples parameter values from their prior distributions (or approximations of the prior).

Extreme initial values can cause WinBUGS to produce an error (TRAP) when it attempts to update the model, because of numerical imprecision.

## Task 1 Continued

Generate initial values for the model.

## Monitoring Parameters and Drawing Samples from the Markov Chain

1. Select `Samples...` from the `Inference` menu.
2. Type the names of the variables you'd like to monitor in the space next to `node`. Click `set` after each variable name.
3. Select `Update...` from the `model` menu.
4. Type in the number of samples you'd like to draw. Start small, you can run more later.
5. Examine diagnostic plots—More on this later.
6. Once you've determined convergence has been met, calculate summary statistics—More on this later.

## Task 1 Continued

Choose to monitor the mean and variance of your model.

Update the Markov chain 1000 iterations.

## The age/recall example

```
model{
  for(i in 1:5){
    y[i] ~ dbern(p[i])
    logit(p[i]) <- alpha + beta*x[i]
  }
  alpha ~ dnorm(0,.01)
  beta ~ dnorm(0,.01)
}
# data
list(y=c(1,1,1,0,0), x=c(10,30,50,70,90))
# initial values #
list(alpha=-3, beta=3)
```

# Age and recall

Who says the relationship is linear?

```
model{
  for(i in 1:5){
    y[i] ~ dbern(p[i])
    logit(p[i]) <- sum(alpha[1:i])
  }
  alpha[1] ~ dnorm(0,1)
  for(i in 2:5){
    # increments are negative #
    alpha[i] ~ dnorm(0,1)I(,0)
  }
}
```

This type of model can be useful for growth curve analysis.

## Task 2: 1-Way ANOVA Example

We wish to compare student performances on a simple task. The students are randomly assigned to one of three treatment groups. The data is provided in the table below:

Treatment					
1	10	8	6	6	7
2	7	7	4	4	2
3	5	3	3	2	1

Develop a Bayesian analysis of variance model, where the within group variances are allowed to vary across groups.