Summary

1. Probability as a means of representing uncertainty
2. Bayesian direct probability statements about parameters
3. Probability distributions
4. Monte Carlo simulation
5. Implementation in WinBUGS (and DoodleBUGS) - Demo
6. Directed graphs for representing probability models
7. Examples

How did it all start?

In 1763, Reverend Thomas Bayes of Tunbridge Wells wrote

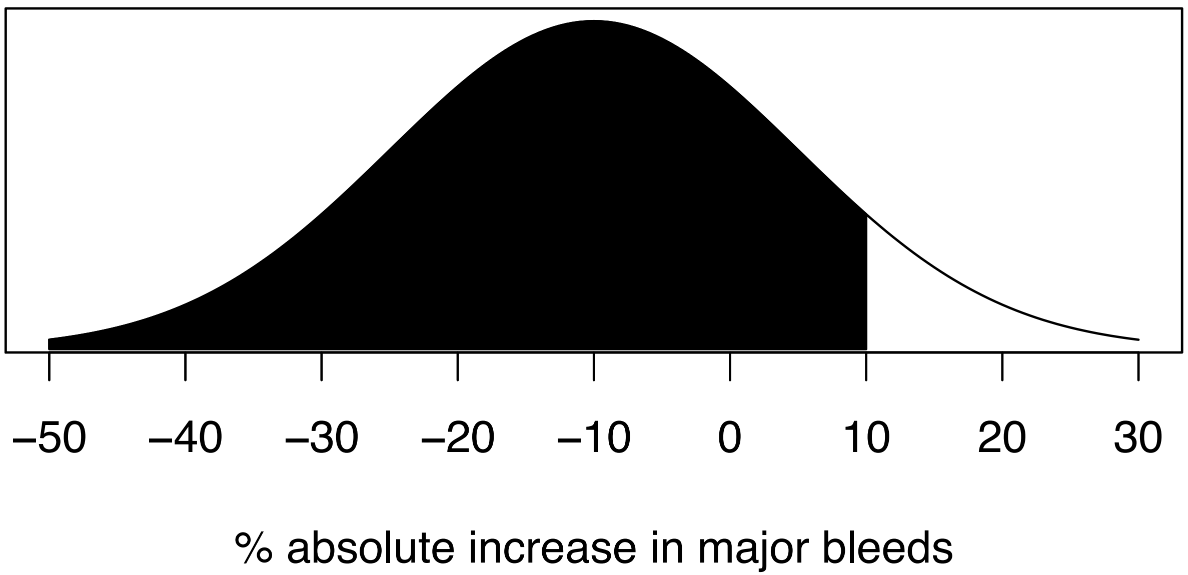
PROBLEM.

Given the number of times in which an unknown event has happened and failed: *Required* the chance that the probability of its happening in a ſingle trial lies ſomewhere between any two degrees of probability that can be named.

In modern language, given $r \sim \text{Binomial}(\theta, n)$, what is $\Pr(\theta_1 < \theta < \theta_2 | r, n)$?

Basic idea: Direct expression of uncertainty about unknown parameters

eg "There is an 89% probability that the absolute increase in major bleeds is less than 10 percent with low-dose PLT transfusions" (Tinmouth et al, Transfusion, 2004)



Why a direct probability distribution?

1. Tells us what we want: what are plausible values for the parameter of interest?

2. No *P-values*: just calculate relevant tail areas

3. No (difficult to interpret) *confidence intervals*: just report, say, central area that contains 95% of distribution

4. Easy to make predictions (see later)

5. Fits naturally into decision analysis / cost-effectiveness analysis / project prioritisation

6. There is a procedure for adapting the distribution in the light of additional evidence: i.e. *Bayes theorem* allows us to learn from experience

Inference on proportions

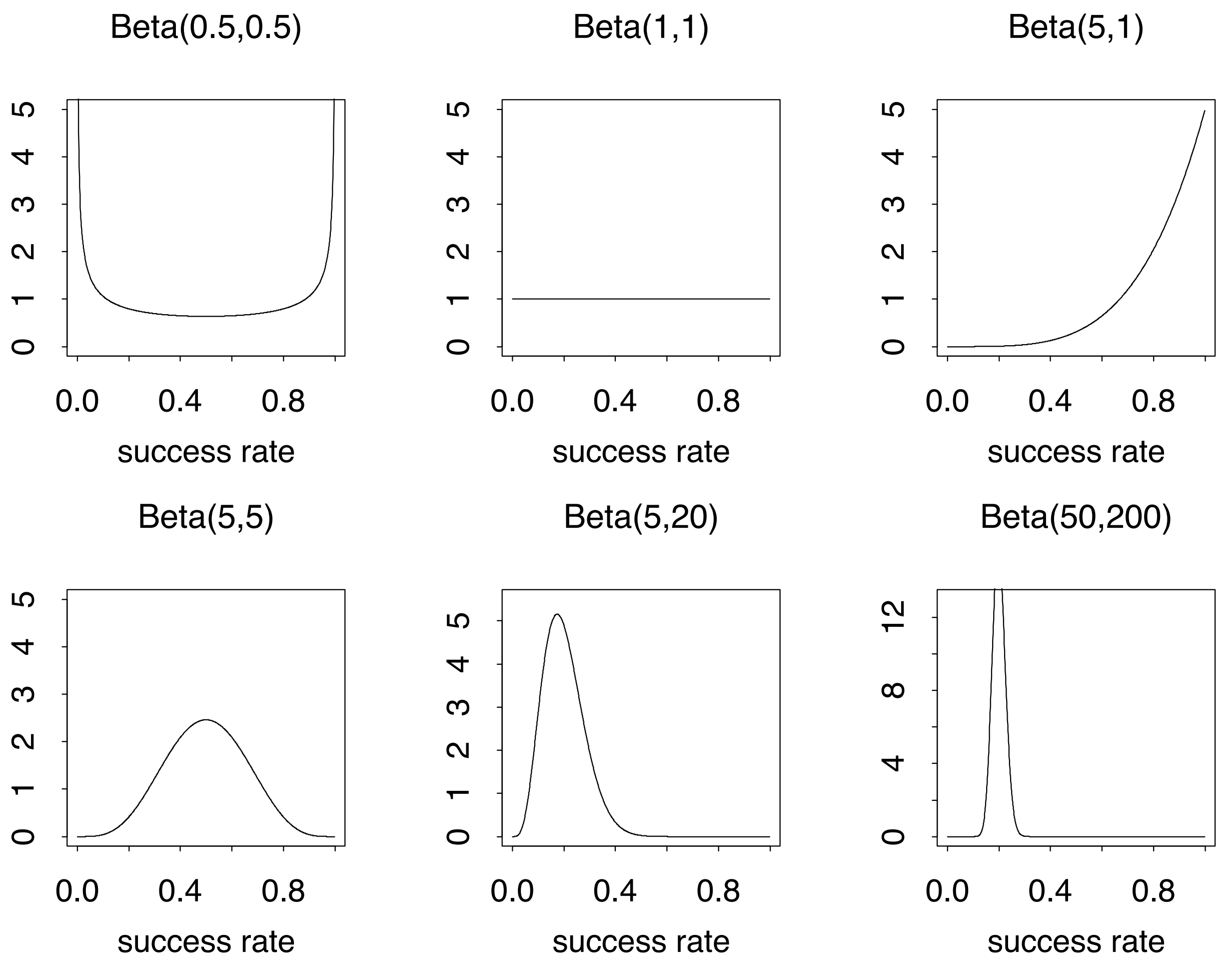
What is a reasonable form for a prior distribution for a proportion?

$\theta \sim$ Beta$[a, b]$ represents a beta distribution with properties:

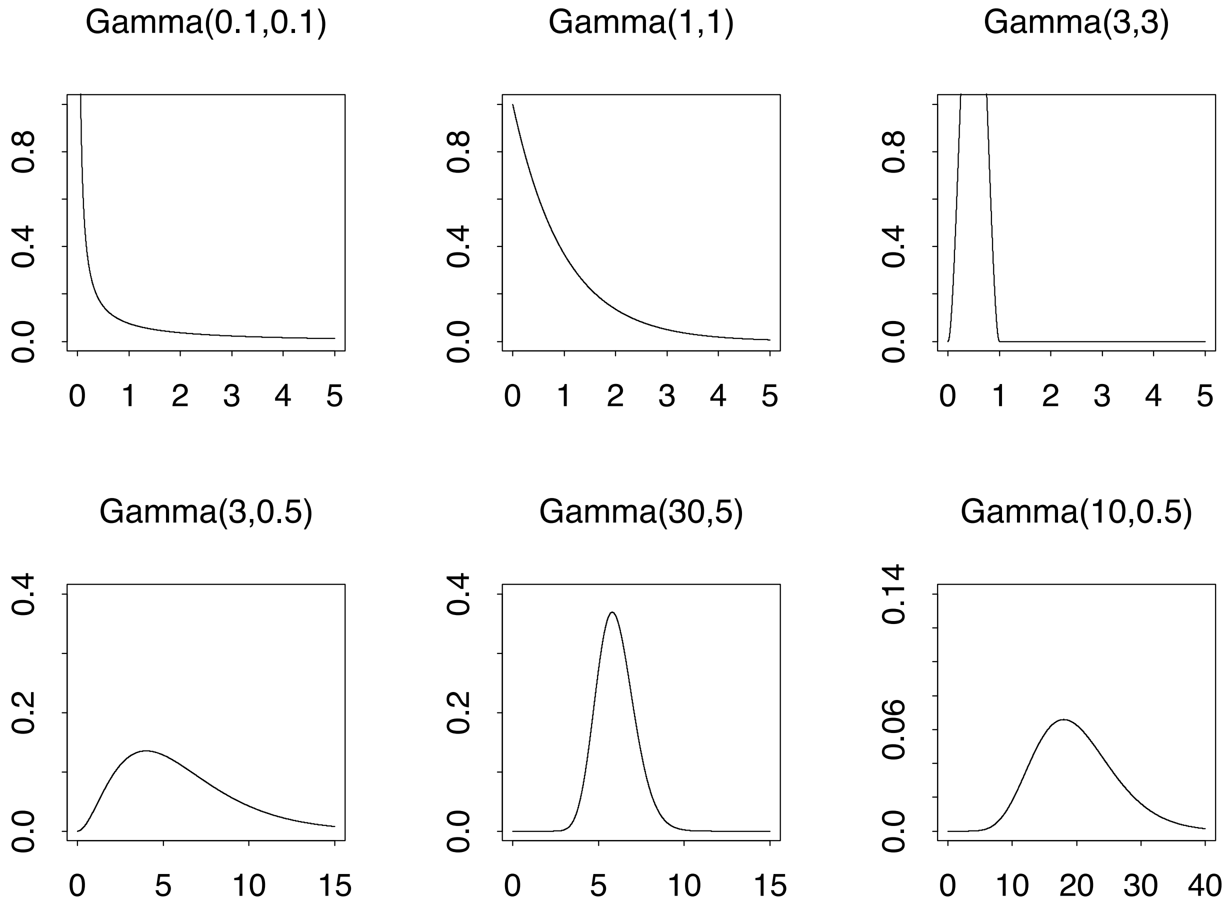
$$\begin{aligned} p(\theta|a,b) &= \frac{\Gamma(a+b)}{\Gamma(a)\Gamma(b)}\, \theta^{a-1}\, (1-\theta)^{b-1}; \qquad \theta \in (0,1) \\ \mathrm{E}(\theta|a,b) &= \frac{a}{a+b} \\ \mathrm{V}(\theta|a,b) &= \frac{ab}{(a+b)^2(a+b+1)}: \end{aligned}$$

WinBUGS notation: `theta ~ dbeta(a,b)`

Beta distribution



Gamma distribution



The Gamma distribution

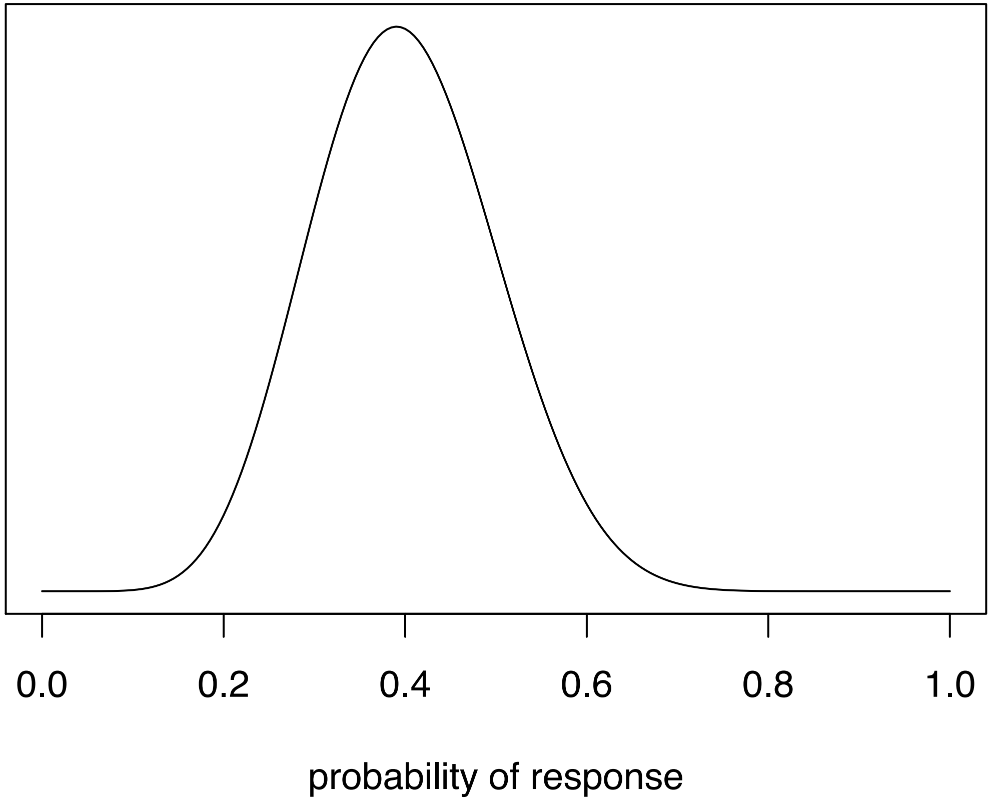
Flexible distribution for positive quantities. If $Y \sim \text{Gamma}[a, b]$

$$
\begin{aligned}
p(y|a,b) &= \frac{b^a}{\Gamma(a)} y^{a-1}\, e^{-by}; \qquad y \in (0, \infty) \\
\mathrm{E}(Y|a,b) &= \frac{a}{b} \\
\mathrm{V}(Y|a,b) &= \frac{a}{b^2}.
\end{aligned}
$$

- Gamma[1,b] distribution is exponential with mean $1/b$
- Gamma[$\frac{v}{2}, \frac{1}{2}$] is a Chi-squared χ^2_v distribution on v degrees of freedom
- $Y \sim$ Gamma[0.001,0.001] means that $p(y) \propto 1/y$, or that $\log Y \approx$ Uniform
- Used as conjugate prior distribution for inverse variances (precisions)
- Used as sampling distribution for skewed positive valued quantities (alternative to log normal likelihood) — MLE of mean is sample mean
- WinBUGS notation: `y ~ dgamma(a,b)`

Example: Drug

- Consider a drug to be given for relief of chronic pain
- Experience with similar compounds has suggested that annual response rates between 0.2 and 0.6 could be feasible
- Interpret this as a distribution with mean = 0.4, standard deviation 0.1
- A Beta[9.2,13.8] distribution has these properties



Beta[9.2, 13.8] prior distribution supporting response rates between 0.2 and 0.6,

Making predictions

Before observing a quantity Y, can provide its predictive distribution by integrating out unknown parameter

$$p(Y) = \int p(Y|\theta)p(\theta)d\theta.$$

Predictions are useful in e.g. cost-effectiveness models, design of studies, checking whether observed data is compatible with expectations, and so on.

If

$$\begin{aligned}\theta &\sim \text{Beta}[a,b]\\ Y_n &\sim \text{Binomial}(\theta,n),\end{aligned}$$

the exact predictive distribution for Y_n is known as the **Beta-Binomial**. It has the complex form

$$p(y_n) = \frac{\Gamma(a+b)}{\Gamma(a)\Gamma(b)} \left(\begin{array}{c} n \\ y_n \end{array} \right) \frac{\Gamma(a+y_n)\Gamma(b+n-y_n)}{\Gamma(a+b+n)}.$$

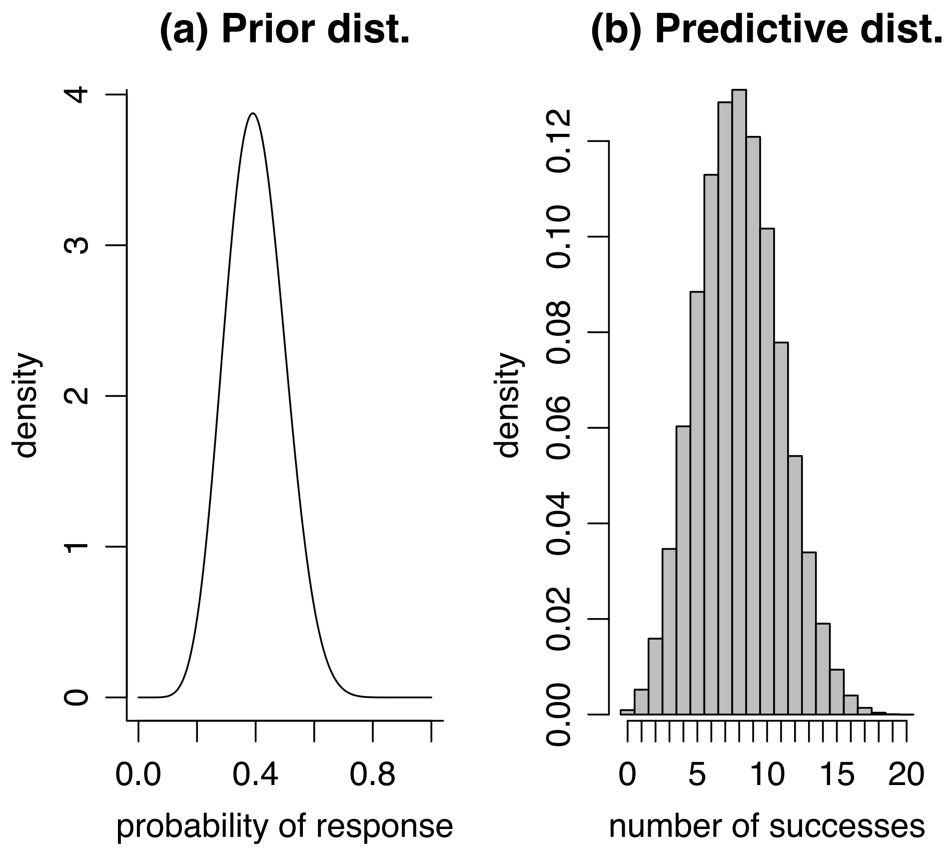
$$\mathsf{E}(Y_n) = n\frac{a}{a+b}$$

If $a = b = 1$ (Uniform distribution), $p(y_n)$is uniform over 0,1,...,n.

But in WinBUGS we can just write

```
theta ~ dbeta(a,b)
Y     ~ dbin(theta,n)
```

and the integration is automatically carried out and does not require algebraic cleverness.



(a) is the Beta prior distribution

(b) is the predictive Beta-Binomial distribution of the number of successes Y in the next 20 trials

From Beta-binomial distribution, can calculate $P(Y_n \geq 15) = 0.015$.

Example: a Monte Carlo approach to estimating tail-areas of distributions

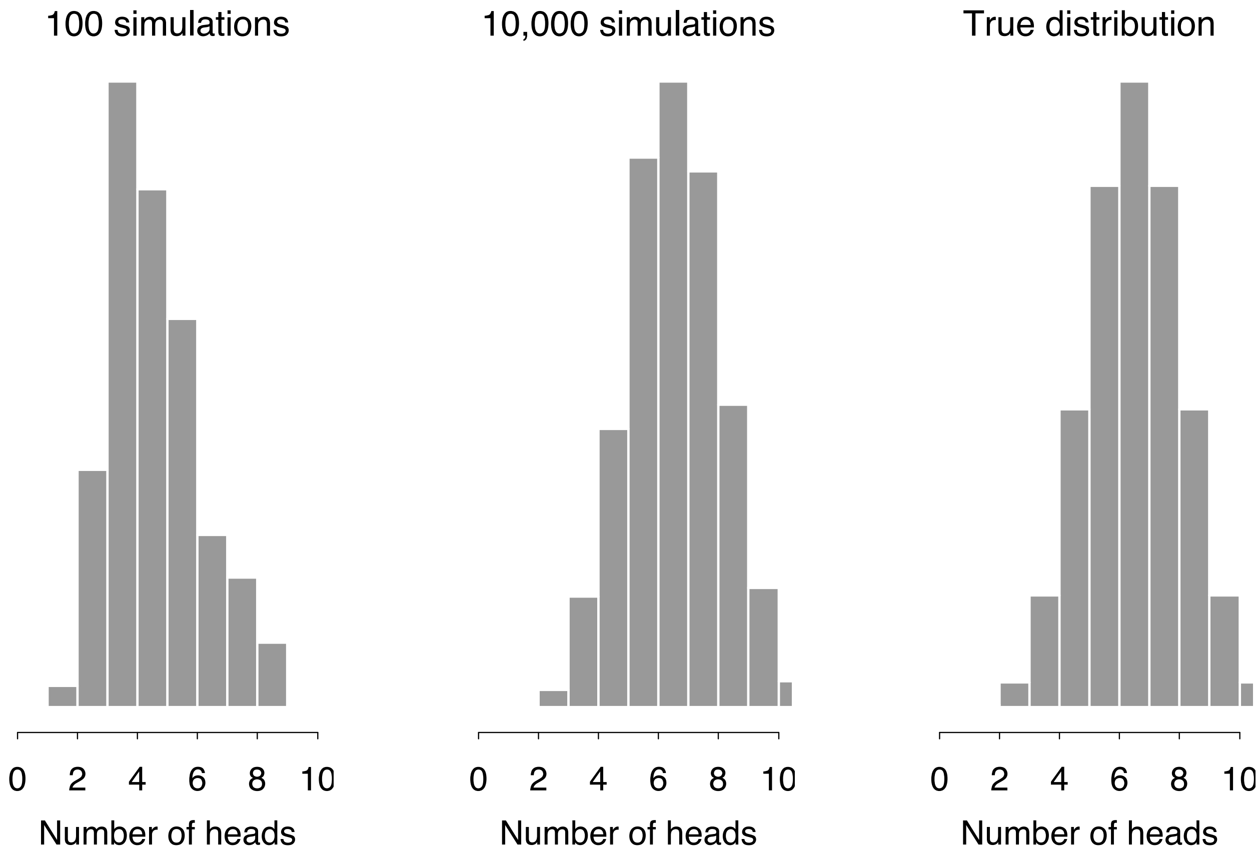
Suppose we want to know the probability of getting 8 or more heads when we toss a fair coin 10 times.

An *algebraic* approach:

$$
\begin{aligned}
\Pr(\geq 8 \text{ heads}) &= \sum_{z=8}^{10} p\left(z|\pi = \frac{1}{2}, n = 10\right) \\
&= \binom{10}{8}\left(\frac{1}{2}\right)^{8}\left(\frac{1}{2}\right)^{2} + \binom{10}{9}\left(\frac{1}{2}\right)^{9}\left(\frac{1}{2}\right)^{1} + \binom{10}{10}\left(\frac{1}{2}\right)^{10}\left(\frac{1}{2}\right)^{0} \\
&= 0.0547.
\end{aligned}
$$

A *physical* approach would be to repeatedly throw a set of 10 coins and count the proportion of throws that there were 8 or more heads.

A *simulation* approach uses a computer to toss the coins!



Proportion with 8 or more 'heads' in 10 tosses:

(a) After 100 'throws' (0.02); (b) after 10,000 throws (0.0577); (c) the true Binomial distribution (0.0547)

General Monte Carlo analysis - 'forward sampling'

Used extensively in risk modelling - can think of as 'adding uncertainty' to a spreadsheet

- Suppose have logical function f containing uncertain parameters
- Can express our uncertainty as a prior distribution
- Simulate many values from this prior distribution
- Calculate f at the simulated values ('iterations')
- Obtain an empirical predictive distribution for f
- Sometimes termed *probabilistic sensitivity analysis*
- Can do in Excel add-ons such as `@RISK` or `Crystal Ball`.

The BUGS program

Bayesian inference Using Gibbs Sampling

- Language for specifying complex Bayesian models
- Constructs object-oriented internal representation of the model
- Simulation from full conditionals using Gibbs sampling
- Current version (WinBUGS 1.4) runs in Windows
- 'Classic' BUGS available for UNIX but this is an old version

WinBUGS is freely available from `http://www.mrc-bsu.cam.ac.uk/bugs`

- Scripts enable WinBUGS 1.4 to run in batch mode or be called from other software
- Interfaces developed for R, Excel, Splus, SAS, Matlab
- OpenBUGS site `http://www.rni.helsinki.fi/openbugs` provides an open source version

Running WinBUGS for Monte Carlo analysis (no data)

1. Open *Specification tool* from *Model* menu.
2. Program responses are shown on bottom-left of screen.
3. Highlight `model` by double-click. Click on *Check model*.
4. Click on *Compile*.
5. Click on *Gen Inits*.
6. Open *Update* from *Model* menu, and *Samples* from *Inference* menu.
7. Type nodes to be monitored into *Sample Monitor*, and click *set* after each.
8. Type * into *Sample Monitor*, and click *trace* to see sampled values.
9. Click on *Update* to generate samples.
10. Type * into *Sample Monitor*, and click *stats* etc to see results on all monitored nodes.

Using WinBUGS for Monte Carlo

The model for the 'coin' example is

$$Y \sim \text{Binomial}(0.5, 10)$$

and we want to know $P(Y \geq 8)$.

This model is represented in the BUGS language as

```
model{
 Y     ~  dbin(0.5,10)
 P8   <-  step(Y-7.5)
 }
```

`P8` is a step function which will take on the value 1 if `Y -7.5` is ≥ 0, *i.e.* `Y` is 8 or more, 0 if 7 or less.

Running this simulation for 100, 10000 and 1000000 iterations, and then taking the empirical mean of `P8`, provided the previous estimated probabilities that Y will be 8 or more.

Some aspects of the BUGS language

- `<-` represents logical dependence, *e.g.* `m <- a + b*x`
- `~` represents stochastic dependence, *e.g.* `r ~ dunif(a,b)`
- Can use arrays and loops

```
for (i in 1:n){
  r[i] ~ dbin(p[i],n[i])
  p[i] ~ dunif(0,1)
  }
```

- Some functions can appear on left-hand-side of an expression, *e.g.*

```
logit(p[i])<- a + b*x[i]
log(m[i])  <- c + d*y[i]
```

- `mean(p[])` to take mean of whole array, `mean(p[m:n])` to take mean of elements m to n. Also for `sum(p[])`.
- `dnorm(0,1)I(0,)` means the prior will be restricted to the range $(0, \infty)$.

Functions in the BUGS language

- `p <- step(x-.7)` = 1 if `x` $\geq$ 0.7, 0 otherwise. Hence monitoring `p` and recording its mean will give the probability that `x`$\geq$ 0.7.
- `p <- equals(x,.7)` = 1 if `x` = 0.7, 0 otherwise.
- `tau <- 1/pow(s,2)` sets $\tau = 1/s^2$.
- `s <- 1/ sqrt(tau)` sets $s = 1/\sqrt{\tau}$.
- `p[i,k] <- inprod(pi[], Lambda[i,,k])` sets $p_{ik} = \sum_j \pi_j \Lambda_{ijk}$. `inprod2` may be faster.
- See 'Model Specification/Logical nodes' in manual for full syntax.

Some common Distributions

Expression	Distribution	Usage
`dbin`	binomial	`r ~ dbin(p,n)`
`dnorm`	normal	`x ~ dnorm(mu,tau)`
`dpois`	Poisson	`r ~ dpois(lambda)`
`dunif`	uniform	`x ~ dunif(a,b)`
`dgamma`	gamma	`x ~ dgamma(a,b)`

NB. The normal is parameterised in terms of its mean and *precision* $= 1/$ variance $= 1/\text{sd}^2$.

See 'Model Specification/The BUGS language: stochastic nodes/Distributions' in manual for full syntax.

Functions cannot be used as arguments in distributions (you need to create new nodes).

Drug example: Monte Carlo predictions

Our prior distribution for proportion of responders in one year θ was Beta$[9.2, 13.8]$.

Consider situation *before* giving 20 patients the treatment. What is the chance if getting 15 or more responders?

$$\begin{array}{rcll} \theta & \sim & \text{Beta}[9.2, 13.8] & \text{prior distribution} \\ y & \sim & \text{Binomial}[\theta, 20] & \text{sampling distribution} \\ P_{\text{crit}} & = & P(y \geq 15) & \text{Probability of exceeding critical threshold} \end{array}$$

```
# In BUGS syntax:
 model{
 theta     ~ dbeta(9.2,13.8)        # prior distribution
 y         ~ dbin(theta,20)         # sampling distribution
 P.crit   <- step(y-14.5)           # =1 if y >= 15, 0 otherwise

 }
```

WinBUGS output and exact answers

node	mean	sd	MC error	2.5%	median	97.5%	start	sample
theta	0.4008	0.09999	9.415E-4	0.2174	0.3981	0.6044	1	10000
y	8.058	2.917	0.03035	3.0	8.0	14.0	1	10000
P.crit	0.0151	0.122	0.001275	0.0	0.0	0.0	1	10000

Note that the mean of the 0-1 indicator `P.crit` provides the estimated tail-area probability.

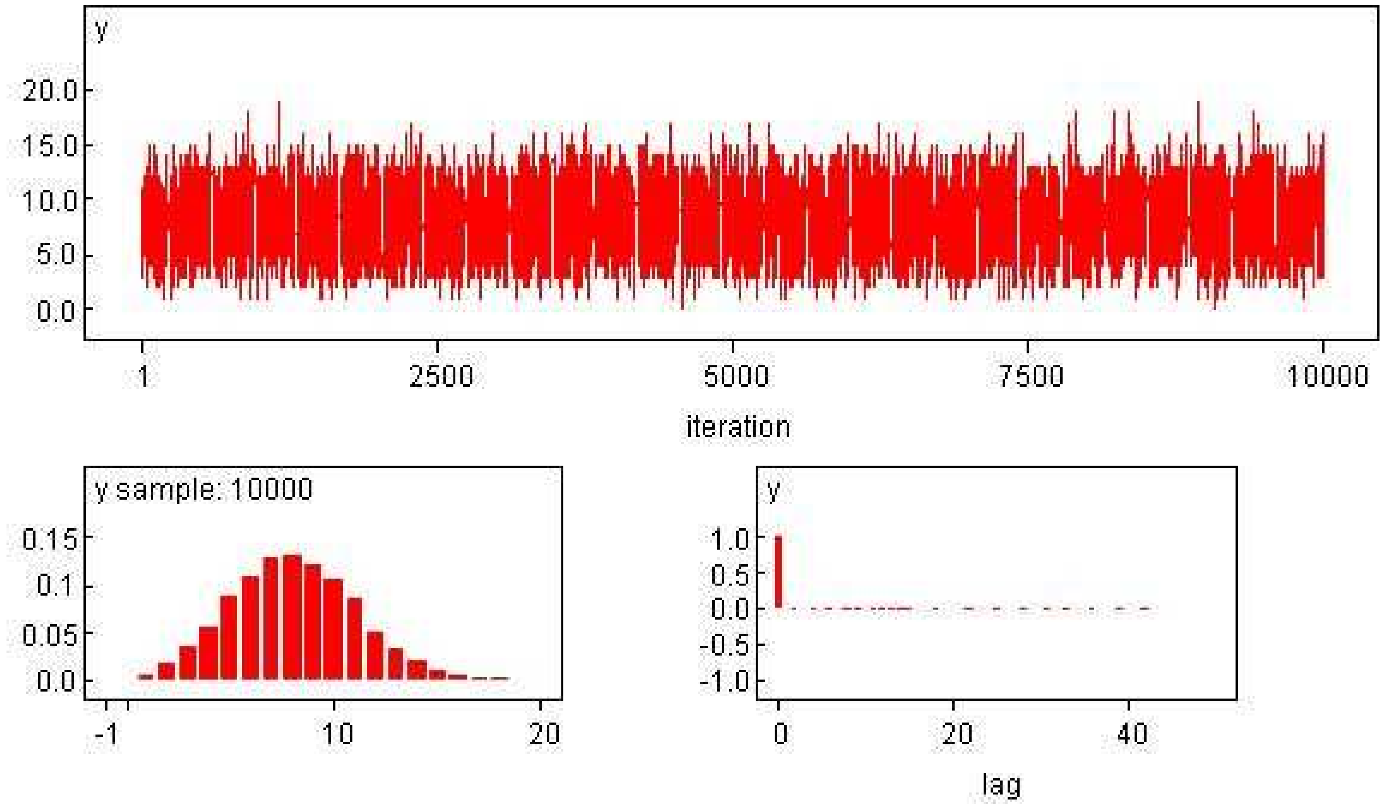
Exact answers from closed-form analysis:

- θ: mean 0.4 and standard deviation 0.1
- y: mean 8 and standard deviation 2.93.
- Probability of at least 15: 0.015

These are independent samples, and so MC error = $\text{SD}/\sqrt{\text{No.iterations}}$.

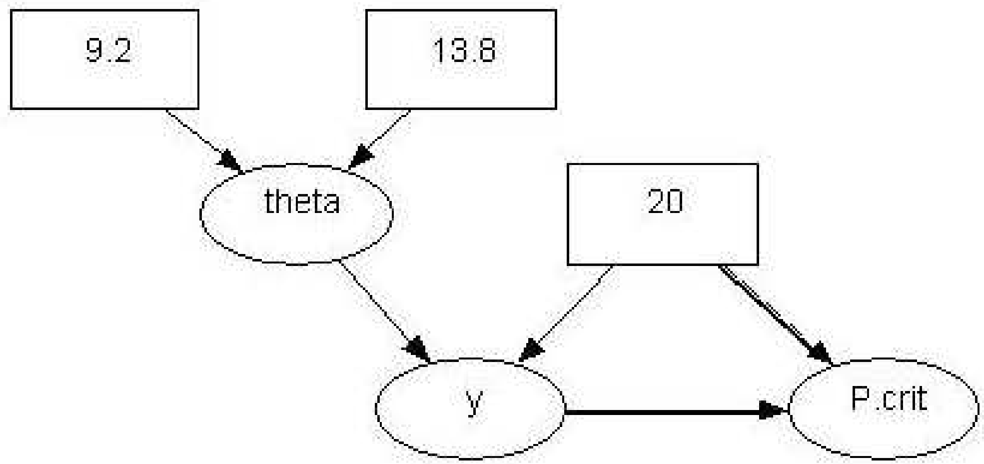
Can achieve arbitrary accuracy by running the simulation for longer.

WinBUGS output



Independent samples, and so no auto-correlation and no concern with convergence.

Graphical representation of models



- *Doodle* represents each quantity as a node in directed acyclic graph (DAG).
- Constants are placed in rectangles, random quantities in ovals
- Stochastic dependence is represented by a single arrow, and logical function as double arrow
- WinBUGS allows models to be specified graphically and run directly from the graphical interface
- Can write code from Doodles
- Good for explanation, but can be tricky to set up

Script for running Drug Monte Carlo example

Run from `Model/Script` menu

```
display('log')          # set up log file
check('c:/bugscourse/drug-MC')          # check syntax of model
#  data('c:/bugscourse/drug-data')      # load data file if there is one
compile(1)              # generate code for 1 simulations
# inits(1,'c:/bugscourse/drug-in1')     # load initial values if necessary
gen.inits()             # generate initial values for all unknown quantities
                        # not given initial values

set(theta)              # monitor the true response rate
set(y)                  # monitor the predicted number of successes
set(P.crit)             # monitor whether a critical number of successes occur
trace(*)                # watch some simulated values (although slows down simulation)
update(10000)           # perform 10000 simulations
history(theta)          # Trace plot of samples for theta
stats(*)                # Calculate summary statistics for all monitored quantities
density(theta)          # Plot distribution of theta
density(y)              # Plot distribution of y
```

Example: Power — uncertainty in a power calculation

- a randomised trial planned with n patients in each of two arms
- response with standard deviation $\sigma = 1$
- aimed to have Type 1 error 5% and 80% power
- to detect a true difference of $\theta = 0.5$ in mean response between the groups

Necessary sample size per group is

$$n = \frac{2\sigma^2}{\theta^2}(0.84 + 1.96)^2 = 63$$

Alternatively, for fixed n, the power is

$$\text{Power} = \Phi\left(\sqrt{\frac{n\theta^2}{2\sigma^2}} - 1.96\right).$$

Suppose we wish to express uncertainty concerning both θ and σ, e.g.

$$\theta \sim \mathrm{N}[0.5, 0.1^2], \quad \sigma \sim \mathrm{N}[1, 0.3^2].$$

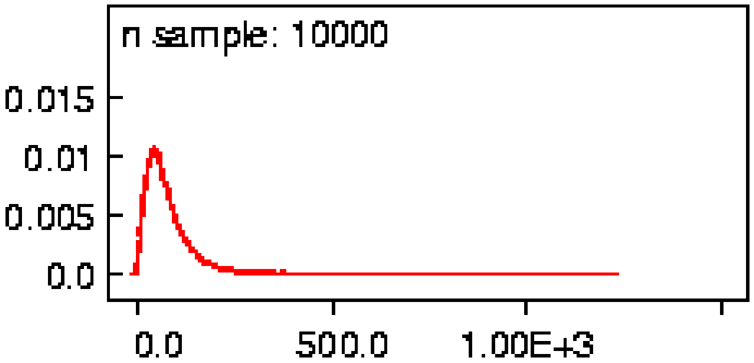
1. Simulate values of θ and σ from their prior distributions

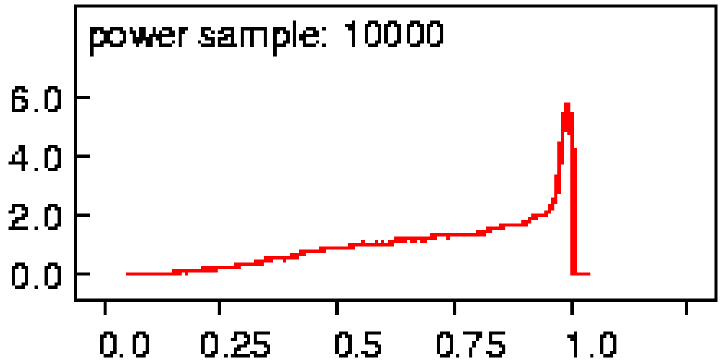
2. Substitute them in the formulae

3. Obtain a predictive distribution over n or Power

```
prec.sigma  <- 1/(0.3*0.3)      # transform sd to precision=1/sd2
prec.theta  <- 1/(0.1*0.1)
sigma       ~ dnorm(1, prec.sigma)I(0,)
theta       ~ dnorm(.5, prec.theta)I(0,)
n      <- 2 * pow(    (.84 +1.96) * sigma / theta ,  2 )
power  <- phi(  sqrt(63/2)* theta /sigma  -1.96  )
prob70 <-step(power-.7)
```

	Median	95% interval
n	62.5	9.3 to 247.2
Power (%)	80	29 to 100





For n= 63, the median power is 80%, and a trial of 63 patients per group could be seriously underpowered. There is a 37% chance that the power is less than 70%.