# Test Run to Check the Installation of OpenBUGS, JAGS, BRugs, R2OpenBUGS, Rjags & R2jags

The following annotated code is extracted from John Kruschke's R scripts, "E:\btut\r\BernBetaBugsFull.R" and "E:\btut\r\BernBetaJagsFull.R" that were downloaded from his website (http://indiana.edu/~kruschke/DoingBayesianDataAnalysis/). I have made some changes to this code.

The purpose of this annotated R code is simply to check that your installation of OpenBUGS, JAGS, BRugs, R2OpenBUGS, Rjags & R2jags are all running and talking to each other in the way that they should. *It is assumed that all of these programs have been installed* (see, e.g., "Installing BUGS and the R to BUGS Interface"; the installing.bugs.jags.pdf is available on the Psych 548 website). The R-code for this document is shown as text at the end of this file, and also as an ascii file, test.bugs.install.txt. If you want to run the code as you are reading this document, I recomment that *you load the text file, test.bugs.install.txt, into RStudio or your favorite programming editor*. It will be easier to transfer the code to R from a programming editor than from this pdf file.

| Section | Торіс   |  |
|---------|---|--|
| 1       | The binomial inference problem that will be used in this document |  |
| 2       | Checking the connection between R and OpenBUGS via BRugs          |  |
| 3       | Checking the connection between R and OpenBUGS via R2OpenBUGS     |  |
| 4       | Checking the connection between R and JAGS via rjags              |  |
| 5       | Checking the connection between R and JAGS via R2jags             |  |
| 6       | Text version of the R-code in Tables 1 - 4                        |  |

**Contents** (Cntrl-left click on a link to jump to the corresponding section)

# End of Contents Table

#### 1. The binomial inference problem that will be used in this document <u>toc</u>

All of the computations in this document make use of the same Bayesian inference problem.

| DATA                              | You have observed 14 independent flips of a coin which may or may not be biased. You observe 11 heads and 3 tails.   |
|-----------------------------------|--|
| PARAMETER                         | The probability of a heads is an unknown parameter called <b>theta</b> .   |
| PRIOR PROBABILITY<br>DISTRIBUTION | Prior to observing the coin flips, you believe that all values of <b>theta</b> are equally likely, so the prior probability distribution of <b>theta</b> is a uniform distribution over the [0, 1] interval. |
| BAYESIAN<br>INFERENCE PROBLEM     | What is the posterior probability distribution over <b>theta</b> after observing 11 heads and 3 tails?   |

One reason that this inference problem is used so often to teach Bayesian statistics is that it can be solved by many different methods. Studying how these different methods apply to the same problem helps students understand some of the more esoteric methods. Figure 1 shows the analytical solution to this problem. Given a uniform prior distribution over **theta**, the posterior distribution is a beta distribution with parameters 12 and 4 (I don't know of any widely accepted names for these parameters; often they are denoted as  $\alpha$  and  $\beta$ ). In Figure 1, the dotted line represents the prior probability distribution.



The remainder of this document solves this same problem using MCMC simulation software.

### 2. Checking the connection between R and OpenBUGS via BRugs. TOC

• Note: The **BRugs** documentation states that you may have problems running **BRugs** with the 32-bit version of R. Nevertheless, I had no problems running the following code with the 64-bit version of R.

**Table 1**. Test the **BRugs** interface between R and OpenBUGS. The code in this table is a subset for the code in Kruschke's script file, BernBetaBugsFull.R.

| # R Code                                | Explanation   |  |
|---|---|--|
| library(BRugs)                          | Attaches <b>BRugs</b> to the search path.                 |  |
| <pre>modelString = "</pre>              | THE MODEL.  |  |
| BUGS model specification begins         |   |  |
| model {                                 | Specify the model in BUGS language, but save it as a      |  |
| <pre># Likelihood:</pre>                | string in R:  |  |
| for ( i in 1:nFlips ) {                 |   |  |
| y[i] ~ dbern( theta )                   |   |  |
| }                                       |   |  |
| # Prior distribution:                   |   |  |
| theta ~ dbeta( priorA , priorB )        |   |  |
| priorA <- 1                             |   |  |
| priorB <- 1                             |   |  |
| }                                       |   |  |
| BUGS model specification ends.          |   |  |
| " # close quote to end modelString      |   |  |
| writeLines(modelString,con="model.txt") | Write the modelString to a file, using R commands:        |  |
| <pre>modelCheck( "model.txt" )</pre>    | Use BRugs to send the model.txt file to BUGS, which       |  |
|   | checks the model syntax:                                  |  |
| dataList = list(                        | # THE DATA.   |  |
| nFlips = 14 ,                           | Specify the data in R, using a list format compatible     |  |
| y = c(1,1,1,1,1,1,1,1,1,1,1,0,0,0)      | with BUGS: [This is not the only way to specify the       |  |
| )                                       | data for BUGS. JM will show you other ways.]              |  |
| modelData( bugsData( dataList ) )       | Use <b>BRugs</b> commands to put the data into a file and |  |
|   | ship the file to BUGS:                                    |  |
| modelCompile()                          | # INTIALIZE THE CHAIN.                                    |  |
| modelGenInits()                         | BRugs command tells BUGS to compile the model.            |  |
|   | BRugs command tells BUGS to randomly initialize a         |  |
|   | chain.  |  |

2

| <pre>samplesSet( "theta" )</pre>                             | <b># RUN THE CHAINS.</b><br>BRugs tells BUGS to keep a record of the sampled |
|--|--|
|  | "theta" values:  |
| chainLength = 50000  | R command defines a new variable that specifies an                           |
|  | arbitrary chain length:  |
| modelUpdate( chainLength )                                   | BRugs tells BUGS to generate a MCMC chain:                                   |
| <pre>thetaSample = samplesSample( "theta" )</pre>            | # EXAMINE THE RESULTS.   |
|  | BRugs asks BUGS for the sample values.                                       |
| mode(thetaSample)  | thetaSample should exist in your R data set. It                              |
| length(thetaSample)  | should be a vector of length 1,000 (the simulation                           |
| thetaSample[1:20]  | produced 1,000 estimates of theta. If this is in fact true,                  |
|  | then you have OpenBUGS successfully installed and the                        |
|  | Brugs interface is working properly with R.                                  |
| thetaSummary = samplesStats( "theta" )                       | BRugs asks BUGS for summary statistics. Again, if you                        |
| thetaSummary   | can see these statistics, then the <b>BRUGS</b> Interface is                 |
|  | The map mapping function is in the imfunction of $file^1$                    |
| res.param(thetaSample)                                       | You will have to attach this file to the search path if you                  |
|  | want to run this function res param makes a plot of                          |
| mtext (<br>"Estimated Posterior Distribution of ThetaSample" | the posterior distribution of a parameter.                                   |
| side = 3 cex = 2 line = 2)                                   | and posterior distribution of a parameter.                                   |
| mtext(   |  |
| "Estimates Computed by OpenBUGS Through the BRugs            |  |
| Interface",  |  |
| side = 3, $cex = 1.25$ , line = 0)                           |  |
| <pre>beta.exact = draw.beta(12, 4, plot.dist =</pre>         | I have added the true beta(12, 4) distribution to this plot,                 |
| FALSE)   | except that I displaced the curve by 0.005 to the left in                    |
| x = beta.exact[, 1] - 0.005                                  | order to be able to see both the true distribution and the                   |
| y = beta.exact[ , 2]   | distribution that was estimated by MCMC.                                     |
|  |  |
| lines(x, y, col = "red")                                     |  |

The text output shows a statistical summary:

 mean
 median
 mode
 bnd.low
 bnd.hi
 n.samples

 7.51e-01
 7.62e-01
 7.90e-01
 5.45e-01
 9.37e-01
 5.00e+04

It will be interesting to compare these results to other results computed below.

### 3. Checking the connection between R and OpenBUGS via R2OpenBUGS TOC

Table 2. Test the R2OpenBUGS interface between R and OpenBUGS.

| # R | Code                                   | Explanation  |
|-----|--|--|
| if  | ("package:R2WinBUGS" %in% search())    | This command detaches the <b>R2WinBUGS</b> package from the  |
|     | <pre>detach("package:R2WinBUGS")</pre> | search path in the case where it is present on the search<br>path. The reason for doing this is that the <b>bugs</b> function in |
|     |  | <b>R2WinBUGS</b> package can interfere with the use of the   |
|     |  | bugs function in the R2OpenBUGs package.   |

<sup>&</sup>lt;sup>1</sup> If you don't have the jmfuns.rda function, it can be downloaded from http://faculty.washington.edu/jmiyamot/downloads.htm or https://faculty.washington.edu/jmiyamot/p548/p548-set.htm

```
# The following cells (yellow background) are
# repeated from Table 1. They need to be
# repeated only if modelString and dataList
# have been removed from .GlobalEnv, e.g.,
# if you are starting a new R session.
modelString = "
                                                          THE MODEL.
BUGS model specification begins ...
                                                          Specify the model in BUGS language, but save it as a string
model {
                                                          in R:
     # Likelihood:
     for ( i in 1:nFlips ) {
         y[i] ~ dbern( theta )
     }
     # Prior distribution:
     theta ~ dbeta ( priorA , priorB )
    priorA <- 1
    priorB <- 1
}
... BUGS model specification ends.
" # close quote to end modelString
                                                          Write the modelString to a file, using R commands:
writeLines(modelString,con="model.txt")
dataList = list(
                                                          # THE DATA.
                                                          Specify the data in R, using a list format compatible with
    nFlips = 14 ,
                                                          BUGS: [This is not the only way to specify the data for
    y = c(1,1,1,1,1,1,1,1,1,1,1,0,0,0)
                                                          BUGS. JM will show you other ways.]
library (R2OpenBUGS)
                                                          Attach R2OpenBUGS to the search path.
                                                          WARNING: When you run the next command, your R
                                                          program will appear to freeze while it transfers the
                                                          computation to OpenBUGS. R will regain control after
                                                          OpenBUGS has finished the MCMC computation. On my
                                                          i5 laptop, the OpenBUGS computation takes about 5
                                                          seconds, but it could be longer on a slower computer. If
                                                          your computer freezes for more than 30 seconds, I would
                                                          assume that it is officially frozen, and you will have to force
                                                          the closure of R and OpenBUGS.
                                                          bugs sends the computation to OpenBUGS. When
bugs.out = bugs(
                                                          OpenBUGS has finished, it returns the results to R. The
    data = dataList,
                                                          results are stored in bugs.out which is a list structure.
    inits = NULL,
    parameters.to.save = "theta",
                                                          In future classes, we will thoroughly discuss the structure of
    model.file= "model.txt",
                                                          the bugs function and its output. For now, suffice it to say
   n.chains= 1,
                                                          that this function requests that OpenBUGS compute 1,000
   n.iter= 50000,
                                                          parameter estimates from the posterior distribution of the
    n.burnin= 0,
                                                          theta parameter, and stores these estimates in the
    n.thin= 1
                                                          bugs.out list.
    )
                                                          We see that bugs.out has many components.
names (bugs.out)
names(bugs.out$sims.list)
                                                          We see that the estimates of theta are stored in
                                                          bugs.out$sims.list.
                                                          We store the estimates of theta in est.theta.
est.theta = bugs.out$sims.list$theta
                                                          est.theta is a numeric vector of length 1,000,
mode(est.theta)
                                                          representing the 1,000 estimates of theta which have been
length(est.theta)
                                                          sampled via MCMC simulation from the posterior
est.theta[1:20]
                                                          distribution of theta.
```

4

| <pre>res.param(est.theta) mtext( "Estimated Posterior Distribution of est.theta",     side = 3, cex = 2, line = 2) </pre>            | The <b>res.param</b> function is in the <b>jmfuns.rda</b> file <sup>2</sup> .<br>You will have to attach this file to the search path if you want to run this function. <b>res.param</b> makes a plot of the posterior distribution of a parameter. |
|--|---|
| <pre>mtext( "Estimates Computed by OpenBUGS Through the R2OpenBUGS Interface", side = 3, cex = 1.25, line = 0)</pre>                 |   |
| <pre>beta.exact = draw.beta(12, 4, plot.dist = FALSE) x = beta.exact[, 1] - 0.005 y = beta.exact[, 2] lines(x, y, col = "rod")</pre> | I have added the true beta(12, 4) distribution to this plot,<br>except that I displaced the curve by 0.005 to the left in order<br>to be able to see both the true distribution and the<br>distribution that was estimated by MCMC.                 |
| lines(x, y, col = "red")   |   |

This analysis finds:

| mean     | median   | mode     | bnd.low  | bnd.hi   | n.samples |
|----------|----------|----------|----------|----------|-----------|
| 7.51e-01 | 7.62e-01 | 7.90e-01 | 5.45e-01 | 9.37e-01 | 5.00e+04  |

Not surprisingly, these results are exactly identical to the results produced by **BRugs** in Section 2. In both cases, the actual computation is performed by OpenBUGS. Next we will see check the performance of JAGS.

# 4. Checking the connection between R and JAGS via rjags TOC

 Table 3. Test the rjags interface between R and JAGS.

| # R Code   | Explanation  |
|--|--|
| <pre>rm(list = ls())</pre>   | Delete (remove) all objects<br>from .GlobalEnv, to clear the work space.   |
| <pre>graphics.off()</pre>  | Turn off all graphics windows, again to clear the work space.  |
| <pre>if ( .Platform\$OS.type != "windows" ) {   windows &lt;- function( ) X11( ) }</pre> | Sets the meaning of the "windows" function<br>so that it is compatible with either Microsoft<br>Windows or with a Mac or Linux<br>environment. |
| require(rjags)   |  |

5

<sup>&</sup>lt;sup>2</sup> If you don't have the jmfuns.rda function, it can be downloaded from http://faculty.washington.edu/jmiyamot/downloads.htm or https://faculty.washington.edu/jmiyamot/p548/p548-set.htm

| <pre>modelString = "</pre>                                      | # THE MODEL.                                 |
|---|--|
| <pre># JAGS model specification begins</pre>                    |  |
| model {   | Specify the model in JAGS language, but      |
| <pre># Likelihood:</pre>  | save it as a string in R:                    |
| <pre>for ( i in 1:nFlips ) {</pre>                              |  |
| y[i] ~ dbern( theta )   | It is interesting to note that the model     |
| }   | model language (for the same model) in       |
| # Prior distribution:   | JAGS (see Table 1).                          |
| theta ~ dbeta ( priorA , priorB )                               |  |
| priorP <- 1   |  |
|   |  |
| # JACS model specification ends                                 |  |
| # close quote to end modelString                                |  |
| <pre>writeLines(modelString con="model tyt")</pre>              | Write the modelString to a file using R      |
| wittenines (modelstling, con-model.txt)                         | commands:                                    |
| dataList = list(  | # THE DATA.                                  |
| nFlips = 14 ,   |  |
| y = c(1,1,1,1,1,1,1,1,1,1,1,0,0,0)                              | Specify the data in R, using a list format   |
| )   | compandle with JAGS:                         |
| <pre># initsList = list( theta = sum(dataList\$y) /</pre>       | # INTIALIZE THE CHAIN.                       |
| length(dataList\$y)   | Can be done automatically in jags.model() by |
|   | Commenting out inits argument.               |
|   | # DIPL THE CHAINS                            |
|   | # RON THE CHAINS.                            |
| parameters = c( "theta" )                                       | The parameter(s) to be monitored.            |
| adaptSteps = 500  | Number of steps to "tune" the samplers.      |
| burnInSteps = 1000  | Number of steps to "burn-in" the samplers.   |
| nChains = 3   | Number of chains to run.                     |
| numSavedSteps=50000   | Total number of steps in chains to save.     |
| thinSteps=1   | Number of steps to "thin" (1=keep every      |
|   | step).                                       |
| <pre>nIter = ceiling((numSavedSteps*thinSteps) / nChains)</pre> | Steps per chain.                             |
| <pre>jagsModel = jags.model(</pre>                              | Create, initialize, and adapt the model:     |
| "model.txt" , data=dataList , # inits=initsList ,               |  |
| n.chains=nChains , n.adapt=adaptSteps )                         |  |
| cat( "Burning in the MCMC chain\n" )                            | # Burn-in:                                   |
| update( jagsModel , n.iter=burnInSteps )                        |  |
| <pre>cat( "Sampling final MCMC chain\n" )</pre>                 | The saved MCMC chain:                        |
| <pre>codaSamples = coda.samples( jagsModel ,</pre>              | Tesulting codaSamples object has these       |
| <pre>variable.names=parameters, n.iter=nIter,</pre>             | indices:                                     |
| thin=thinSteps )  | and Samplas [[ abain Idy ]][ stan Idy        |
|   | paramIdy ]                                   |
|   | paramitax j                                  |
| <pre>mcmcChain = as.matrix( codaSamples )</pre>                 | # EXAMINE THE RESULTS.                       |
|   | Convert code object and game lasts           |
| thetaSample = mcmcChain   | matrix object for easier handling. But note  |
|   | that this concatenates the different chains  |
|   | into one long chain. Result is               |
|   | mcmcChain[ stepIdx , paramIdx ]              |

| mode(thetaSample)   | thetaSample should exist in your R data                |
|---|--|
| length(thetaSample)   | set. It should be a matrix with 50,001 rows            |
| <pre>thetaSample[1:20, ]</pre>                              | (the simulation produced 50,001 estimates of           |
|   | theta) and 1 column. If this is in fact true,          |
|   | then you have JAGS sucessfully installed               |
|   | and the <b>Rjags</b> interface is working properly     |
|   | with R.  |
| summary(thetaSample)  | BRugs asks BUGS for summary statistics.                |
|   | Again, if you can see these statistics, then the       |
|   | <b>BRugs</b> interface is working properly with R.     |
| res.param(thetaSample)                                      | The <b>res.param</b> function is in the                |
|   | jmfuns.rda file <sup>3</sup> . You will have to attach |
| mtext(  | this file to the search path if you want to run        |
| "Estimated Posterior Distribution of ThetaSample",          | this function. <b>res.param</b> makes a plot of        |
| side = 3, $cex = 2$ , line = 2)                             | the posterior distribution of a parameter.             |
| mtext(  |  |
| "Estimates Computed by JAGS Through the Rjags Interface",   |  |
| side = 3, cex = 1.25, line = 0)                             |  |
| <pre>beta.exact = draw.beta(12, 4, plot.dist = FALSE)</pre> | I have added the true beta(12, 4) distribution         |
| x = beta.exact[, 1] - 0.005                                 | to this plot, except that I displaced the curve        |
| y = beta.exact[, 2]   | by 0.005 to the left in order to be able to see        |
|   | both the true distribution and the distribution        |
| lines(x, y, col = "red")                                    | that was estimated by MCMC.                            |

Summary of Results from the Rjags Run:

| mean  | median | mode  | bnd.low | bnd.hi | n.samples |
|-------|--------|-------|---------|--------|-----------|
| 0.750 | 0.760  | 0.778 | 0.543   | 0.936  | 50001.000 |

Comparing these results to the results from **BRugs** and **R2OpenBUGS**, we can see that they are virtually identical.

## 5. Checking the connection between R and JAGS via R2jags TOC

 Table 4. Test the R2jags interface between R and JAGS.

| # R Code   | Explanation  |
|--|--|
| <pre>rm(list = ls())</pre>   | Delete (remove) all objects<br>from .GlobalEnv. to clear the work space.   |
| graphics.off()   | Turn off all graphics windows, again to clear<br>the work space.   |
| <pre>if ( .Platform\$OS.type != "windows" ) {   windows &lt;- function( ) X11( ) }</pre>   | Sets the meaning of the "windows" function<br>so that it is compatible with either Microsoft<br>Windows or with a Mac or Linux<br>environment. |
| library(R2jags)<br>search()  | Notice that the <b>R2jags</b> package requires<br>that <b>Rjags</b> and <b>R2WinBUGS</b> be on the<br>search path.                             |
| <pre># All of the following code (shaded yellow) is the<br/># same as the code for the Rjags example. It is<br/># reproduced here because it was deleted when we<br/># cleared the work space.</pre> |  |

<sup>&</sup>lt;sup>3</sup> If you don't have the jmfuns.rda function, it can be downloaded from http://faculty.washington.edu/jmiyamot/downloads.htm or https://faculty.washington.edu/jmiyamot/p548/p548-set.htm

| <pre>modelString = "</pre>                                      | # THE MODEL.   |
|---|--|
| <pre># JAGS model specification begins</pre>                    |  |
| model {   | Specify the model in JAGS language, but                          |
| <pre># Likelihood:</pre>  | save it as a string in R:  |
| for ( i in 1:nFlips ) {   | It is interesting to note that the model                         |
| y[i] ~ dbern( theta )   | language for OpenBUGS is identical to the                        |
| }<br># Duion dictuikution:                                      | model language (for the same model) in                           |
| # Prior distribution:<br>theta ~ dheta( priorA priorB )         | JAGS (see Table 1).  |
| priorA <- 1   |  |
| priorB <- 1   |  |
| }   |  |
| # JAGS model specification ends.                                |  |
| " # close quote to end modelString                              |  |
| <pre>writeLines(modelString,con="model.txt")</pre>              | Write the modelString to a file, using R                         |
|   | commands:  |
| dataList = list(  | # THE DATA.  |
| nFlips = 14 ,   |  |
| y = c(1,1,1,1,1,1,1,1,1,1,0,0,0)                                | Specify the data in R, using a list format compatible with JAGS: |
| <pre># initsList = list( theta = sum(dataList\$y) /</pre>       | # INTIALIZE THE CHAIN.   |
| length(dataList\$y)   | Can be done automatically in jags.model() by                     |
|   | commenting out inits argument.                                   |
|   | Otherwise could be established as:                               |
|   | # RUN THE CHAINS.  |
| <pre>parameters = c( "theta" )</pre>                            | The parameter(s) to be monitored.                                |
| adaptSteps = 500  | Number of steps to "tune" the samplers.                          |
| burnInSteps = 1000  | Number of steps to "burn-in" the samplers.                       |
| nChains = 3   | Number of chains to run.   |
| numSavedSteps=50000   | Total number of steps in chains to save.                         |
| thinSteps=1   | Number of steps to "thin" (1=keep every                          |
|   | step).   |
| <pre>nIter = ceiling((numSavedSteps*thinSteps) / nChains)</pre> | Steps per chain.   |
| # The following R2jags command has the same                     |  |
| # structure as the preceding Rjags computation.                 |  |
|   | The output of the jags analysis will be                          |
| jags.out = jags(  | saved in jags.out.   |
| data = dataList,  | Specify the data.  |
| <pre>inits = NULL,</pre>  | We could specify the initial values (start                       |
|   | values) for the 3 chains of sampled estimates.                   |
|   | By setting <b>inits</b> = <b>NULL</b> , we let <b>JAGS</b>       |
| narameters to save = narameters                                 | The parameter(s) to be monitored (returned                       |
| parameters.to.save - parameters,                                | to R for further analysis).                                      |
| n.iter= nIter + 1000,   | The number of iterations (samples of                             |
|   | estimates) to be drawn in each chain.                            |
| <pre>model.file= "model.txt",</pre>                             | The designation of a model file.                                 |
| n.chains= nChains,  | The number of chains of estimates.                               |
| n.burnin= burnInSteps,  | The number of "burn-in" samples that will be discarded.          |

9

| n.thin= thinSteps )       The rate at which samples will be saved from a chain, e.g., n.thin = 5 would mean that every 5th sample is saved.         # Next we need to find the component of the output # that contains the samples from the posterior # distribution of theta.       Names of the components of the list, jags.out, is agg.out, is a component, jags.out, is a component, jags.out, is a component, jags.out, is a component of BUGSoutput, is and it is guess on the first that the destimates of theta would be stored in the sime.list. (Also, Ihkow that the estimates of theta would be stored in the sime.list. (Also, Ihkow that R2WinBUGS.PQPenBUGS also stores the parameter estimates in a component called sime.list. (Also, Ihkow that R2WinBUGS.PQPenBUGS also stores the parameter estimates in a component called sime.list. (Also, Ihkow that R2WinBUGS.PQPenBUGS also stores the parameter estimates in a component called sime.list. (Also, Ihkow that R2WinBUGS.PQPenBUGS also stores the parameter estimates in a component called sime.list. (Also, Ihkow that R2WinBUGS.PQPenBUGS also stores the parameter estimates in a component called sime.list.cheta]         est.theta = jags.out\$BUGSoutput\$sims.list\$theta       Ihkow that the estimates of the the bugs function is in the sime lab, so we can expect parallels in how the functions in the same lab. so we can expect parameter.         "Estimated Posterior Distribution of est.theta", side = 3, cex = 2, line = 2)       The res.param function is in the jimfuns.rda file'. You will have to attach this file to the search path if you want to run this function. res.param makes all  |   |  |
|--|---|--|
| <pre>a chain eg. p. Kinh = 5 would mean time<br/>every 5th sample is saved.<br/># Next we need to find the component of the output<br/># that contains the samples from the posterior<br/># distribution of theta.<br/>names (jags.out)<br/>names (jags.out\$BUGSoutput)<br/>names (jags.out\$BUGSoutput)<br/>names (jags.out\$BUGSoutput\$sims.list)<br/>names (jags.out\$BUGSoutput\$sims.list]<br/>names (jags.out\$BUGSout</pre>   | n.thin= thinSteps )   | The rate at which samples will be saved from           |
| <pre># Next we need to find the component of the output # that contains the samples from the posterior # distribution of theta. names(jags.out) names(jags.out\$BUGSoutput) names(jags.out\$BUGSoutput\$sims.list) Names of the component, jags.out\$BUGSoutput\$sims.list) Names of the component, jags.out\$BUGSoutput\$sims.list If you are wondering how I knew that the estimates of theta would be stored in the simes.list.component of BUGSoutput\$sims.list. If you are wondering how I knew that the estimates of theta would be stored in the simes.list.component of BUGSoutput\$sims.list. If you are wondering how I knew that the estimates of theta would be stored in the simes.list.component of BUGSoutput\$sims.list. If you are wondering how I knew that the based this guess on the fact that the bugs function in R2OpenBUGS ald Solorput.l based this guess on the fact that the bugs function in R2OpenBUGS and R2jags were all written by researchers in the same lab, so we can expect parallels in how that rest.theta = jags.out\$BUGSoutput\$sims.list\$theta mode(est.theta) est.theta[1:20] res.param function is in the jmfuns.rdafile<sup>1</sup>. You will have to attach this file to the search paramitaks a plot of the posterior distribution of est.theta". side = 3, cex = 2, line = 2) mtext( "Stimates Computed by JAGS Through the R2jags Interface". side = 3, cex = 1.25, line = 0) beta.exact[, 1] - 0.005 y = beta.exact[, 2] Lines(x, y, col = "red")</pre>  |   | a chain, e.g., $\mathbf{n}$ . thin = 5 would mean that |
| <pre># Next we need to find the component of the output # that contains the samples from the posterior # distribution of theta. names(jags.out) names(jags.out\$BUGSoutput) names(jags.out\$BUGSoutput\$sims.list) Names of the component, jags.out\$BUGSoutput\$sims.list) Names of the component, jags.out\$BUGSoutput\$sims.list Names of the component, isse.list.Called Sume.list Names of the component, isse.list.Called Sume.list.Called S</pre>  |   | every 5th sample is saved.                             |
| <pre># that contains the samples from the posterior # distribution of theta. names(jags.out) Names of the component, jags.out\$BUGSoutput) Names of the component, jags.out\$BUGSoutput\$sims.list) Names of the component, jags.out\$BUGSoutput\$sims.list) Names of the component, jags.out\$BUGSoutput\$sims.list If you are wondering how I knew that the estimates of theta would be stored in the sims.list component of BUGSoutput\$sims.list. If you are wondering how I knew that the estimates of theta would be stored in the sims.list.(Also, I know that R2WinBUGS, R2OpenBUGS also stores the parameter estimates in a component called sims.list.(Also, I know that R2WinBUGS, R2OpenBUGS also stores the parameter estimates in a component called sims.list.(Also, I know that R2WinBUGS, R2OpenBUGS also stores the parameter estimates in a component called sims.list.(Also, I know that R2WinBUGS, R2OpenBUGS also stores the parameter estimates in a component called sims.list.(Also, I know that R2WinBUGS, R2OpenBUGS also stores the parameter estimates in a component called sims.list.(Also, I know that R2WinBUGS, R2OpenBUGS also stores the parameter estimates in a component called sims.list.(Also, I know that R2WinBUGS, R2OpenBUGS also stores the parameter estimates in a component called sims.list.(Also, I know that R2WinBUGS, R2OpenBUGS also stores the parameter estimates in a component called sims.list.(Also, I know that R2WinBUGS, R2OpenBUGS also stores the parameter estimates in a component called sims.list.(Also, I know that R2WinBUGS, R2OpenBUGS also stores the parameter estimates in a component called sims.list.component of BUGSoutput\$sims.list\$theta mode (est.theta) length (est.theta) length (est.theta) side = 3, cex = 2, line = 2) mtext( "Estimates Computed by JAGS Through the R2jags Interface", side = 3, cex = 1.25, line = 0) beta.exact[, 21 beta.exact[, 21 component intervent] lines(x, y, col = "red") lines(x, y, col = "red") </pre>   | # Next we need to find the component of the output          |  |
| <pre># distribution of theta. names(jags.out)</pre>  | # that contains the samples from the posterior              |  |
| <pre>names(jags.out) Names of the components of the list, jags.out. names(jags.out\$BUGSoutput) Names of the component, jags.out\$BUGSoutput\$sims.list) Names of the component, jags.out\$BUGSoutput\$sims.list. Names of the component, jags.out\$BUGSoutput\$sims.list. If you are wondering how I knew that the estimates of theta would be stored in the sims.list.component of BUGSoutput\$sims.list. Date that bugs function in R20penBUGS also stores the parameter estimates in a component called sims.list. (Also, I know that R2WinBUGS, R2OpenBUGS also stores the parameter estimates in a component called sims.list. (Also, I know that R2WinBUGS, R2OpenBUGS also stores the parameter estimates in a component called sims.list. (Also, I know that R2WinBUGS, R2OpenBUGS also stores the parameter estimates in a component called sims.list. (Also, I know that R2WinBUGS, R2OpenBUGS also stores the parameter estimates in a component called sims.list. (Also, I know that R2WinBUGS, R2OpenBUGS also stores the parameter estimates in a component called sims.list. (Also, I know that R2WinBUGS, R2OpenBUGS also stores the parameter estimates in a component called sims.list. (Also, I know that R2WinBUGS, R2OpenBUGS also stores the parameter estimates in a component called sims.list. (Also, I know that R2WinBUGS, R2OpenBUGS and R2jags were all written by researchers in the sing infuns.rda file<sup>1</sup>. You will have to attach this file to the search path if you want to run this function.res.param makes a plot of the posterior distribution of a parameter.  mtext( "Estimates Computed by JAGS Through the R2jags Interface". side = 3, cex = 1.25, line = 0) beta.exact [ , 2] beta.exact [ , 1] - 0.005 y = beta.exact [ , 2] lines(x, y, col = "red") </pre>  | <pre># distribution of theta.</pre>                         |  |
| jags.out.names (jags.out\$BUGSoutput)Names of the component,<br>jags.out\$BUGSoutput.names (jags.out\$BUGSoutput\$sims.list)Names of the component,<br>jags.out\$BUGSoutput\$sims.list.names (jags.out\$BUGSoutput\$sims.list)Names of the component,<br>jags.out\$BUGSoutput\$sims.list.If you are wondering how I knew that the<br>estimates of theta would be stored in the<br>sims.list.component of BUGSoutput\$based this guess on the fact that the bugs<br>function in R2OpenBUGS also stores the<br>parameter estimates in a component called<br>sims.list. (Also, I know that<br>R2WinBUGS.R2OpenBUGS and R2jags<br>were all writen by researchers in the same<br>lab, so we can expect parallels in how the<br>functions in these packages will work.)est.theta = jags.out\$BUGSoutput\$sims.list\$thetaThe res.param function is in the<br>jmfuns.rda file <sup>4</sup> . You will have to attach<br>this file to the search path if you want to run<br>this file to the search path if you want to run<br>this file to the search path if you want to run<br>this file to the search path if you want to run<br>this file to the search path if you want to run<br>this file to the search path if you want to run<br>this file to the search path if you want to run<br>this file to the search path if you want to run<br>this file to the search path if you want to run<br>this function. res.param makes a plot of<br>the posterior distribution of a parameter.mtext(<br>"Estimates Computed by JAGS through the R2jags Interface",<br>side = 3, cex = 1.25, line = 0)I have added the true beta(12, 4) distribution<br>to this plot, except that I displaced the curve<br>by 0.005 to the left in order to be able to see<br>both the true distribution and the distribution<br>that was estimated by MCMC.  | names(jags.out)   | Names of the components of the list,                   |
| <pre>names (jags.out\$BUGSoutput) Names of the component,     jags.out\$BUGSoutput\$sims.list) Names of the component,     jags.out\$BUGSoutput\$sims.list) Names of the component,     jags.out\$BUGSoutput\$sims.list. If you are wondering how I knew that the     estimates of theta would be stored in the     sims.list component of BUGSoutput\$, I     based this guess on the fact that the bugs     function in R2OpenBUGS also stores the     parameter estimates in a component called     sims.list.(Also,I know that     R2WinBUGS,R2OpenBUGS and R2jags     were all written by researchers in the same     lab, so we can expect parallels in how the     functions in these packages will work.) est.theta = jags.out\$BUGSoutput\$sims.list\$theta mode(est.theta) est.theta[1:20] res.param(est.theta) mtext( "Estimated Posterior Distribution of est.theta".     side = 3, cex = 2, line = 2) mtext( "Estimated Posterior Distribution of est.theta".     side = 3, cex = 1.25, line = 0) beta.exact[, 1] - 0.005 x = beta.exact[, 2]     lhave added the true beta(12, 4) distribution     to this plot, except that I displaced the curve     by OnO5 to the left in order to be able to see     both the true distribution     that Was estimated by MCMC. </pre>  |   | jags.out.  |
| jags.out\$BUGSoutput.<br>names(jags.out\$BUGSoutput\$sims.list)<br>Names of the component,<br>jags.out\$BUGSoutput\$sims.list.<br>If you are wondering how I knew that the<br>estimates of theta would be stored in the<br>sims.list component of BUGSoutput.I<br>based this guess on the fact that the bugs<br>function in R2OpenBUGS also stores the<br>parameter estimates in a component called<br>sims.list. (Also, I know that<br>R2WinBUGS, R2OpenBUGS also stores the<br>parameter estimates in a component called<br>sims.list. (Also, I know that<br>R2WinBUGS, R2OpenBUGS and R2jags<br>were all written by researchers in the same<br>lab, so we can expect parallels in how the<br>functions in these packages will work.)<br>est.theta[1:20]<br>res.param(est.theta)<br>est.theta[1:20]<br>res.param(est.theta)<br>mode(est.theta)<br>iside = 3, cex = 2, line = 2)<br>mtext(<br>"Estimated Posterior Distribution of est.theta",<br>side = 3, cex = 2, line = 2)<br>mtext(<br>"Estimates Computed by JAGS Through the R2jags Interface",<br>side = 3, cex = 1.25, line = 0)<br>beta.exact[, 1] - 0.005<br>x = beta.exact[, 2]<br>lines(x, y, col = "red")<br>Jags.out\$BUGSoutput\$sims.list\$theta<br>parameter definition and the distribution<br>that was estimated by MCMC.   | names (jags.out\$BUGSoutput)                                | Names of the component,                                |
| <pre>names(jags.out\$BUGSoutput\$sims.list) Names of the component, jags.out\$BUGSoutput\$sims.list. If you are wondering how I knew that the estimates of theta would be stored in the sims.list component of BUGSoutput,I based this guess on the fact that the bugs function in R2OpenBUGS also stores the parameter estimates in a component called sims.list. (Also, I know that R2WinBUGS,R2OpenBUGS and R2jags were all written by researchers in the same lab, so we can expect parallels in how the functions in these packages will work.) est.theta = jags.out\$BUGSoutput\$sims.list\$theta mode(est.theta) length(est.theta) est.theta[1:20] res.param(est.theta) mtext( "Estimated Posterior Distribution of est.theta", side = 3, cex = 2, line = 2) mtext( "Estimates Computed by JAGS Through the R2jags Interface", side = 3, cex = 1.25, line = 0) beta.exact[, 1] - 0.005 x = beta.exact[, 2] lhave added the true beta(12, 4) distribution to this plot, except that I displaced the curve y = beta.exact[, 2] lhave added the true beta(12, 4) distribution to this plot, except that I displaced the curve by 0.005 to the left in order to be able to see both the true distribution that was estimated by MCMC.</pre>   |   | jags.out\$BUGSoutput.                                  |
| <pre>jags.out\$BUGSoutput\$sims.list.<br/>If you are wondering how I knew that the<br/>estimates of theta would be stored in the<br/>sims.list.component of BUGSoutput, I<br/>based this guess on the fact that the bugs<br/>function in R2OpenBUGS allo stores the<br/>parameter estimates in a component called<br/>sins.list.(Also,I know that<br/>R2WinBUGS,R2OpenBUGS and R2jags<br/>were all written by researchers in the same<br/>lab, so we can expect parallels in how the<br/>functions in these packages will work.)<br/>est.theta = jags.out\$BUGSoutput\$sims.list\$theta<br/>mode(est.theta)<br/>est.theta[1:20]<br/>res.param(est.theta)<br/>est.theta[1:20]<br/>res.param(est.theta)<br/>mtext(<br/>"Estimated Posterior Distribution of est.theta",<br/>side = 3, cex = 2, line = 2)<br/>mtext(<br/>"Estimates Computed by JAGS Through the R2jags Interface",<br/>side = 3, cex = 1.25, line = 0)<br/>beta.exact[, 1] - 0.005<br/>y = beta.exact[, 2]<br/>lines(x, y, col = "red")</pre>  | names(jags.out\$BUGSoutput\$sims.list)                      | Names of the component,                                |
| If you are wondering how I knew that the<br>estimates of theta would be stored in the<br>sime.list component of BUGSoutput, I<br>based this guess on the fact that the bugs<br>function in R2OpenBUGS also stores the<br>parameter estimates in a component called<br>sime.list.(Also,I know that<br>R2WinBUGS, R2OpenBUGS and R2jags<br>were all written by researchers in the same<br>lab, so we can expect parallels in how the<br>functions in these packages will work.)est.theta = jags.out\$BUGSoutput\$sims.list\$thetamode(est.theta)<br>est.theta[1:20]res.param(est.theta)<br>side = 3, cex = 2, line = 2)<br>mtext(<br>"Estimated Posterior Distribution of est.theta",<br>side = 3, cex = 1.25, line = 0)The res.param function is in the<br>jmfruns.rda file <sup>4</sup> . You will have to attach<br>this file to the search path if you want to run<br>this function. res.param makes a plot of<br>the posterior distribution of a parameter.x = beta.exact[, 1] - 0.005<br>y = beta.exact[, 2]I have added the true beta(12, 4) distribution<br>to this plot, except that I displaced the curve<br>by 0.005 to the left in order to be able to see<br>both the true distribution and the distribution<br>that was estimated by MCMC.   |   | jags.out\$BUGSoutput\$sims.list.                       |
| <pre>If you are wondering how I knew that the estimates of theta would be stored in the sims.list component of BUGSoutput,I,I based this guess on the fact that the bugs function in R2OpenBUGS also stores the parameter estimates in a component called sims.list.(Also,I know that R2WinBUGS,R2OpenBUGS and R2jags were all written by researchers in the same lab, so we can expect parallels in how the functions in these packages will work.) est.theta = jags.out\$BUGSoutput\$sims.list\$theta mode(est.theta) length(est.theta) est.theta[1:20] res.param(est.theta) mtext( "Estimated Posterior Distribution of est.theta", side = 3, cex = 2, line = 2) mtext( "Estimates Computed by JAGS Through the R2jags Interface", side = 3, cex = 1.25, line = 0) beta.exact = draw.beta(12, 4, plot.dist = FALSE) x = beta.exact[, 1] - 0.005 y = beta.exact[, 2] lhave added the true beta(12, 4) distribution lines(x, y, col = "red")</pre>  |   |  |
| <pre>estimates of theta would be stored in the<br/>sims.list component of BUGSoutput, I<br/>based this guess on the fact that the bugs<br/>function in R2OpenBUGS also stores the<br/>parameter estimates in a component called<br/>sims.list. (Also, I know that<br/>R2WinBUGS, R2OpenBUGS and R2jags<br/>were all written by researchers in the same<br/>lab, so we can expect parallels in how the<br/>functions in these packages will work.)</pre><br>est.theta = jags.out\$BUGSoutput\$sims.list\$theta<br>mode(est.theta)<br>length(est.theta)<br>est.theta[1:20]<br>res.param(est.theta)<br>mtext(<br>"Estimated Posterior Distribution of est.theta",<br>side = 3, cex = 2, line = 2)<br>mtext(<br>"Estimates Computed by JAGS Through the R2jags Interface",<br>side = 3, cex = 1.25, line = 0)<br>beta.exact = draw.beta(12, 4, plot.dist = FALSE)<br>x = beta.exact[, 2]<br>heta.exact[, 3]<br>heta.exact[, 3]<br>heta.exact[, 4]<br>heta.exact[, 4]<br>heta.exact[, 4]<br>heta.exact[, 4]<br>heta.exact[, 4]<br>heta.exact[, 4]<br>heta.exact[, 4]<br>heta.ex |   | If you are wondering how I knew that the               |
| <pre>sims.list component of BUGSoutput, I<br/>based this guess on the fact that the bugs<br/>function in R2OpenBUGS also stores the<br/>parameter estimates in a component called<br/>sims.list. (Also, I know that<br/>R2WinBUGS, R2OpenBUGS and R2jags<br/>were all written by researchers in the same<br/>lab, so we can expect parallels in how the<br/>functions in these packages will work.)</pre><br>est.theta = jags.out\$BUGSoutput\$sims.list\$theta<br>mode(est.theta)<br>length(est.theta)<br>est.theta[1:20]<br>res.param(est.theta)<br>est.theta[1:20]<br>res.param(est.theta)<br>mtext(<br>"Estimated Posterior Distribution of est.theta",<br>side = 3, cex = 2, line = 2)<br>mtext(<br>"Estimates Computed by JAGS Through the R2jags Interface",<br>side = 3, cex = 1.25, line = 0)<br>beta.exact = draw.beta(12, 4, plot.dist = FALSE)<br>x = beta.exact[, 1] - 0.005<br>y = beta.exact[, 2]<br>lines(x, y, col = "red")   |   | estimates of theta would be stored in the              |
| <pre>based this guess on the fact that the bugs<br/>function in R2OpenBUGS also stores the<br/>parameter estimates in a component called<br/>sims.list. (Also, I know that<br/>R2WinBUGS, R2OpenBUGS and R2jags<br/>were all written by researchers in the same<br/>lab, so we can expect parallels in how the<br/>functions in these packages will work.)</pre> est.theta = jags.out\$BUGSoutput\$sims.list\$theta mode(est.theta) length(est.theta) est.theta[1:20] res.param(est.theta) mtext( "Estimated Posterior Distribution of est.theta",<br>side = 3, cex = 2, line = 2) mtext( "Estimates Computed by JAGS Through the R2jags Interface",<br>side = 3, cex = 1.25, line = 0) beta.exact = draw.beta(12, 4, plot.dist = FALSE) x = beta.exact[, 1] - 0.005 y = beta.exact[, 2] lhave added the true beta(12, 4) distribution<br>to this plot, except that I displaced the curve<br>by 0.005 to the left in order to be able to see<br>both the true distribution and the distribution<br>that was estimated by MCMC.   |   | sims.list component of BUGSoutput, I                   |
| <pre>function in R2OpenBUGS also stores the parameter estimates in a component called sims.list. (Also, I know that R2WinBUGS, R2OpenBUGS and R2jags were all written by researchers in the same lab, so we can expect parallels in how the functions in these packages will work.) est.theta = jags.out\$BUGSoutput\$sims.list\$theta mode(est.theta) length(est.theta) est.theta[1:20] res.param(est.theta) mtext( "Estimated Posterior Distribution of est.theta", side = 3, cex = 2, line = 2) mtext( "Estimates Computed by JAGS Through the R2jags Interface", side = 3, cex = 1.25, line = 0) beta.exact = draw.beta(12, 4, plot.dist = FALSE) x = beta.exact[, 1] - 0.005 y = beta.exact[, 2] lines(x, y, col = "red") </pre>  |   | based this guess on the fact that the <b>bugs</b>      |
| <pre>parameter estimates in a component called<br/>sims.list.(Also, I know that<br/>R2WinBUGS, R2OpenBUGS and R2jags<br/>were all written by researchers in the same<br/>lab, so we can expect parallels in how the<br/>functions in these packages will work.)</pre> est.theta = jags.out\$BUGSoutput\$sims.list\$theta mode(est.theta) length(est.theta) est.theta[1:20] res.param(est.theta) mtext( "Estimated Posterior Distribution of est.theta",<br>side = 3, cex = 2, line = 2) mtext( "Estimates Computed by JAGS Through the R2jags Interface",<br>side = 3, cex = 1.25, line = 0) beta.exact [ , 1] - 0.005 y = beta.exact[ , 2] lines(x, y, col = "red")  The res.parameter estimates in a component called sims.list.(Also,I know that R2WinBUGS,R2OpenBUGS and R2jags were all written by researchers in the same lab, so we can expect parallels in how the functions in these packages will work.)   |   | function in <b>R2OpenBUGS</b> also stores the          |
| <pre>sims.list.(Also, I know that R2WinBUGS, R2OpenBUGS and R2jags were all written by researchers in the same lab, so we can expect parallels in how the functions in these packages will work.) est.theta = jags.out\$BUGSoutput\$sims.list\$theta mode(est.theta) length(est.theta) est.theta[1:20] res.param(est.theta) mtext( "Estimated Posterior Distribution of est.theta", side = 3, cex = 2, line = 2) mtext( "Estimates Computed by JAGS Through the R2jags Interface", side = 3, cex = 1.25, line = 0) beta.exact[, 1] - 0.005 y = beta.exact[, 2] lines(x, y, col = "red")</pre>  |   | parameter estimates in a component called              |
| R2WinBUGS, R2OpenBUGS and R2jags<br>were all written by researchers in the same<br>lab, so we can expect parallels in how the<br>functions in these packages will work.)est.theta = jags.out\$BUGSoutput\$sims.list\$thetamode (est.theta)<br>length (est.theta)<br>est.theta[1:20]res.param (est.theta)<br>mtext(<br>"Estimated Posterior Distribution of est.theta",<br>side = 3, cex = 2, line = 2)<br>mtext(<br>"Estimates Computed by JAGS Through the R2jags Interface",<br>side = 3, cex = 1.25, line = 0)beta.exact = draw.beta(12, 4, plot.dist = FALSE)<br>y = beta.exact[, 1] - 0.005<br>y = beta.exact[, 2]I have added the true beta(12, 4) distribution<br>to this plot, except that I displaced the curve<br>by 0.005 to the left in order to be able to see<br>both the true distribution and the distribution<br>that was estimated by MCMC.  |   | sims.list. (Also, I know that                          |
| <pre>were all written by researchers in the same<br/>lab, so we can expect parallels in how the<br/>functions in these packages will work.)<br/>est.theta = jags.out\$BUGSoutput\$sims.list\$theta<br/>mode(est.theta)<br/>length(est.theta)<br/>est.theta[1:20]<br/>res.param(est.theta)<br/>mtext(<br/>"Estimated Posterior Distribution of est.theta",<br/>side = 3, cex = 2, line = 2)<br/>mtext(<br/>"Estimates Computed by JAGS Through the R2jags Interface",<br/>side = 3, cex = 1.25, line = 0)<br/>beta.exact = draw.beta(12, 4, plot.dist = FALSE)<br/>x = beta.exact[, 1] - 0.005<br/>y = beta.exact[, 2]</pre>  |   | R2WinBUGS, R2OpenBUGS and R2jags                       |
| lab, so we can expect parallels in how the<br>functions in these packages will work.)est.theta = jags.out\$BUGSoutput\$sims.list\$thetamode (est.theta)length (est.theta)est.theta[1:20]res.param(est.theta)mtext (<br>"Estimated Posterior Distribution of est.theta",<br>side = 3, cex = 2, line = 2)<br>mtext (<br>"Estimates Computed by JAGS Through the R2jags Interface",<br>side = 3, cex = 1.25, line = 0)beta.exact = draw.beta(12, 4, plot.dist = FALSE)<br>y = beta.exact[, 2]I have added the true beta(12, 4) distribution<br>to this plot, except that I displaced the curve<br>by 0.005 to the left in order to be able to see<br>both the true distribution and the distribution<br>that was estimated by MCMC.   |   | were all written by researchers in the same            |
| <pre>functions in these packages will work.) est.theta = jags.out\$BUGSoutput\$sims.list\$theta mode(est.theta) length(est.theta) est.theta[1:20] res.param(est.theta) mtext( "Estimated Posterior Distribution of est.theta", side = 3, cex = 2, line = 2) mtext( "Estimates Computed by JAGS Through the R2jags Interface", side = 3, cex = 1.25, line = 0) beta.exact = draw.beta(12, 4, plot.dist = FALSE) x = beta.exact[, 1] - 0.005 y = beta.exact[, 2] lines(x, y, col = "red")</pre> The res.param function is in the immediate function is interface", side = 3, cex = 1.25, line = 0) lines(x, y, col = "red")  |   | lab, so we can expect parallels in how the             |
| <pre>est.theta = jags.out\$BUGSoutput\$sims.list\$theta mode(est.theta) length(est.theta) est.theta[1:20] res.param(est.theta) mtext( "Estimated Posterior Distribution of est.theta",     side = 3, cex = 2, line = 2) mtext( "Estimates Computed by JAGS Through the R2jags Interface",     side = 3, cex = 1.25, line = 0) beta.exact = draw.beta(12, 4, plot.dist = FALSE) x = beta.exact[, 1] - 0.005 y = beta.exact[, 2] lines(x, y, col = "red")</pre>  |   | functions in these packages will work.)                |
| <pre>mode (est.theta) length (est.theta) est.theta[1:20]  res.param(est.theta) mtext( "Estimated Posterior Distribution of est.theta",     side = 3, cex = 2, line = 2) mtext( "Estimates Computed by JAGS Through the R2jags Interface",     side = 3, cex = 1.25, line = 0) beta.exact = draw.beta(12, 4, plot.dist = FALSE) x = beta.exact[, 1] - 0.005 y = beta.exact[, 2] lines(x, y, col = "red")</pre> The res.param function is in the jmfuns.rda file <sup>4</sup> . You will have to attach this file to the search path if you want to run this function. res.param makes a plot of the posterior distribution of a parameter. I have added the true beta(12, 4) distribution to this plot, except that I displaced the curve by 0.005 to the left in order to be able to see both the true distribution and the distribution that was estimated by MCMC.   | est.theta = jags.out\$BUGSoutput\$sims.list\$theta          |  |
| <pre>length (est.theta) est.theta[1:20]  res.param(est.theta)  mtext( "Estimated Posterior Distribution of est.theta",     side = 3, cex = 2, line = 2) mtext( "Estimates Computed by JAGS Through the R2jags Interface",     side = 3, cex = 1.25, line = 0) beta.exact = draw.beta(12, 4, plot.dist = FALSE) x = beta.exact[, 1] - 0.005 y = beta.exact[, 2]  lines(x, y, col = "red") </pre> The res.param function is in the jmfuns.rda file <sup>4</sup> . You will have to attach this file to the search path if you want to run this function.res.param makes a plot of the posterior distribution of a parameter.   | mode(est.theta)   |  |
| <pre>est.theta[1:20] res.param(est.theta) mtext( "Estimated Posterior Distribution of est.theta", side = 3, cex = 2, line = 2) mtext( "Estimates Computed by JAGS Through the R2jags Interface", side = 3, cex = 1.25, line = 0) beta.exact = draw.beta(12, 4, plot.dist = FALSE) x = beta.exact[, 1] - 0.005 y = beta.exact[, 2] lines(x, y, col = "red")</pre> The res.param function is in the jmfuns.rda file <sup>4</sup> . You will have to attach this file to the search path if you want to run this function. res.param makes a plot of the posterior distribution of a parameter. I have added the true beta(12, 4) distribution to this plot, except that I displaced the curve by 0.005 to the left in order to be able to see both the true distribution and the distribution that was estimated by MCMC.  | length(est.theta)   |  |
| <pre>res.param(est.theta) The res.param function is in the jmfuns.rda file<sup>4</sup>. You will have to attach this file to the search path if you want to run "Estimated Posterior Distribution of est.theta",     side = 3, cex = 2, line = 2) mtext( "Estimates Computed by JAGS Through the R2jags Interface",     side = 3, cex = 1.25, line = 0) beta.exact = draw.beta(12, 4, plot.dist = FALSE) x = beta.exact[, 1] - 0.005 y = beta.exact[, 2] lines(x, y, col = "red")</pre>  | est.theta[1:20]   |  |
| <pre>immext( immext() immext()</pre>   | res.param(est.theta)  | The <b>res.param</b> function is in the                |
| <pre>mtext( "Estimated Posterior Distribution of est.theta", side = 3, cex = 2, line = 2) mtext( "Estimates Computed by JAGS Through the R2jags Interface", side = 3, cex = 1.25, line = 0) beta.exact = draw.beta(12, 4, plot.dist = FALSE) x = beta.exact[, 1] - 0.005 y = beta.exact[, 2] lines(x, y, col = "red")</pre> this file to the search path if you want to run this function. res.param makes a plot of the posterior distribution of a parameter. It have added the true beta(12, 4) distribution to this plot, except that I displaced the curve by 0.005 to the left in order to be able to see both the true distribution and the distribution that was estimated by MCMC.  |   | jmfuns.rda file <sup>4</sup> . You will have to attach |
| <pre>"Estimated Posterior Distribution of est.theta",<br/>side = 3, cex = 2, line = 2)<br/>mtext(<br/>"Estimates Computed by JAGS Through the R2jags Interface",<br/>side = 3, cex = 1.25, line = 0)<br/>beta.exact = draw.beta(12, 4, plot.dist = FALSE)<br/>x = beta.exact[, 1] - 0.005<br/>y = beta.exact[, 2]<br/>lines(x, y, col = "red")</pre> this function. res.param makes a plot of<br>the posterior distribution of a parameter.<br>It have added the true beta(12, 4) distribution<br>to this plot, except that I displaced the curve<br>by 0.005 to the left in order to be able to see<br>both the true distribution and the distribution<br>that was estimated by MCMC.   | mtext(  | this file to the search path if you want to run        |
| <pre>side = 3, cex = 2, line = 2) mtext( "Estimates Computed by JAGS Through the R2jags Interface", side = 3, cex = 1.25, line = 0) beta.exact = draw.beta(12, 4, plot.dist = FALSE) x = beta.exact[, 1] - 0.005 y = beta.exact[, 2] lines(x, y, col = "red")</pre> the posterior distribution of a parameter. the posterior distribution of a parameter.  | "Estimated Posterior Distribution of est.theta",            | this function. <b>res.param</b> makes a plot of        |
| <pre>mtext( "Estimates Computed by JAGS Through the R2jags Interface",     side = 3, cex = 1.25, line = 0) beta.exact = draw.beta(12, 4, plot.dist = FALSE) x = beta.exact[, 1] - 0.005 y = beta.exact[, 2] lines(x, y, col = "red") I have added the true beta(12, 4) distribution to this plot, except that I displaced the curve by 0.005 to the left in order to be able to see both the true distribution and the distribution that was estimated by MCMC.</pre>  | side = 3, $cex = 2$ , line = 2)                             | the posterior distribution of a parameter.             |
| <pre>"Estimates Computed by JAGS Through the R2jags Interface",<br/>side = 3, cex = 1.25, line = 0) beta.exact = draw.beta(12, 4, plot.dist = FALSE) x = beta.exact[, 1] - 0.005 y = beta.exact[, 2] lines(x, y, col = "red")</pre> I have added the true beta(12, 4) distribution to this plot, except that I displaced the curve by 0.005 to the left in order to be able to see both the true distribution and the distribution that was estimated by MCMC.   | mtext (   |  |
| side = 3, cex = 1.25, line = 0)beta.exact = draw.beta(12, 4, plot.dist = FALSE)x = beta.exact[, 1] - 0.005y = beta.exact[, 2]lines(x, y, col = "red")  | "Estimates Computed by JAGS Through the R2jags Interface",  |  |
| <pre>beta.exact = draw.beta(12, 4, plot.dist = FALSE) x = beta.exact[, 1] - 0.005 y = beta.exact[, 2] lines(x, y, col = "red")</pre> I have added the true beta(12, 4) distribution to this plot, except that I displaced the curve by 0.005 to the left in order to be able to see both the true distribution and the distribution that was estimated by MCMC.  | side = 3, $cex = 1.25$ , line = 0)                          |  |
| x = beta.exact[, 1] - 0.005to this plot, except that I displaced the curve<br>by 0.005 to the left in order to be able to see<br>both the true distribution and the distribution<br>that was estimated by MCMC.  | <pre>beta.exact = draw.beta(12, 4, plot.dist = FALSE)</pre> | I have added the true beta(12, 4) distribution         |
| y = beta.exact[ , 2]by 0.005 to the left in order to be able to see<br>both the true distribution and the distribution<br>that was estimated by MCMC.  | x = beta.exact[, 1] - 0.005                                 | to this plot, except that I displaced the curve        |
| <b>lines(x, y, col = "red")</b> both the true distribution and the distribution that was estimated by MCMC.  | y = beta.exact[, 2]   | by 0.005 to the left in order to be able to see        |
| lines (x, y, col = "red") that was estimated by MCMC.  |   | both the true distribution and the distribution        |
|  | <pre>lines(x, y, col = "red")</pre>                         | that was estimated by MCMC.                            |

Summary of Results from the R2jags Run:

| mean  | median | mode  | bnd.low | bnd.hi | n.samples |
|-------|--------|-------|---------|--------|-----------|
| 0.750 | 0.760  | 0.781 | 0.542   | 0.934  | 50001.000 |

Obviously, the results are similar from all four methods. The plots of the estimated posterior distributions are shown on the next page.

<sup>&</sup>lt;sup>4</sup> If you don't have the jmfuns.rda function, it can be downloaded from http://faculty.washington.edu/jmiyamot/downloads.htm or https://faculty.washington.edu/jmiyamot/p548/p548-set.htm



**Figure 2**. Posterior distributions of **theta** estimated by four different methods. The thin red line shows the beta(12, 4) distribution (exact posterior distribution); it has been displaced to the left by 0.005 in order to make it more visible.

#### 6. Text version of the R-code in Tables 1 - 4.

# The code shown below is also available in a separate text file called 'test.bugs.install.txt'.# You may want to copy this code from the current document and paste it into a script file# within RStudio. Then you can run it in RStudio.

# Table 1. Test the BRugs interface between R and OpenBUGS.

```
# The code in this table is a subset for the code in
# Kruschke's script file, BernBetaBugsFull.R.
library(BRugs)
modelString = "
BUGS model specification begins ...
model {
    # Likelihood:
    for ( i in 1:nFlips ) {
       y[i] ~ dbern( theta )
    }
    # Prior distribution:
   theta ~ dbeta ( priorA , priorB )
   priorA <- 1
   priorB <- 1
}
... BUGS model specification ends.
" # close quote to end modelString
writeLines(modelString,con="model.txt")
modelCheck( "model.txt" )
dataList = list(
   nFlips = 14 ,
   y = c(1,1,1,1,1,1,1,1,1,1,1,1,0,0,0)
)
modelData( bugsData( dataList ) )
modelCompile()
modelGenInits()
samplesSet( "theta" )
chainLength = 50000
modelUpdate( chainLength )
thetaSample = samplesSample( "theta" )
mode(thetaSample)
length(thetaSample)
thetaSample[1:20]
thetaSummary = samplesStats( "theta" )
thetaSummary
res.param(thetaSample)
mtext(
"Estimated Posterior Distribution of ThetaSample",
   side = 3, cex = 2, line = 2)
mtext(
"Estimates Computed by OpenBUGS Through the BRugs Interface",
   side = 3, cex = 1.25, line = 0)
beta.exact = draw.beta(12, 4, plot.dist = FALSE)
x = beta.exact[, 1] - 0.005
y = beta.exact[ , 2]
lines(x, y, col = "red")
# Table 2. Test the R2OpenBUGS interface between R and
# OpenBUGS.
if ("package:R2WinBUGS" %in% search())
   detach("package:R2WinBUGS")
```

```
# The following cells (yellow background) are
# repeated from Table 1. They need to be
# repeated only if modelString and dataList
# have been removed from .GlobalEnv, e.g.,
# if you are starting a new R session.
modelString = "
BUGS model specification begins ...
model {
    # Likelihood:
    for ( i in 1:nFlips ) {
        y[i] ~ dbern( theta )
    }
    # Prior distribution:
    theta ~ dbeta( priorA , priorB )
    priorA <- 1
    priorB <- 1
}
... BUGS model specification ends.
" # close quote to end modelString
writeLines(modelString,con="model.txt")
dataList = list(
    nFlips = 14 ,
    y = c(1,1,1,1,1,1,1,1,1,1,1,0,0,0)
)
library (R2OpenBUGS)
bugs.out = bugs(
   data = dataList,
   inits = NULL,
   parameters.to.save = "theta",
   model.file= "model.txt",
   n.chains= 1,
   n.iter= 50000,
   n.burnin= 0,
   n.thin= 1
   )
names (bugs.out)
names(bugs.out$sims.list)
est.theta = bugs.out$sims.list$theta
mode(est.theta)
length(est.theta)
est.theta[1:20]
res.param(est.theta)
mtext(
"Estimated Posterior Distribution of est.theta",
   side = 3, cex = 2, line = 2)
mtext(
"Estimates Computed by OpenBUGS Through the R2OpenBUGS Interface",
   side = 3, cex = 1.25, line = 0)
beta.exact = draw.beta(12, 4, plot.dist = FALSE)
x = beta.exact[, 1] - 0.005
y = beta.exact[ , 2]
```

```
lines(x, y, col = "red")
# Table 3. Test the rjags interface between R and JAGS.
rm(list = ls())
graphics.off()
if ( .Platform$OS.type != "windows" ) {
  windows <- function( ... ) X11( ... )
ł
require(rjags)
modelString = "
# JAGS model specification begins ...
model {
    # Likelihood:
   for ( i in 1:nFlips ) {
       y[i] ~ dbern( theta )
    }
    # Prior distribution:
   theta ~ dbeta( priorA , priorB )
   priorA <- 1
   priorB <- 1
ł
# ... JAGS model specification ends.
" # close quote to end modelString
writeLines(modelString,con="model.txt")
dataList = list(
   nFlips = 14,
   y = c(1,1,1,1,1,1,1,1,1,1,1,0,0,0)
)
# initsList = list( theta = sum(dataList$y) /
                                                   length(dataList$y)
parameters = c( "theta" )
adaptSteps = 500
burnInSteps = 1000
nChains = 3
numSavedSteps=50000
thinSteps=1
nIter = ceiling((numSavedSteps*thinSteps) / nChains)
jagsModel = jags.model(
   "model.txt" , data=dataList , # inits=initsList ,
   n.chains=nChains , n.adapt=adaptSteps )
cat( "Burning in the MCMC chain... \n" )
update( jagsModel , n.iter=burnInSteps )
cat( "Sampling final MCMC chain... n" )
codaSamples = coda.samples( jagsModel ,
   variable.names=parameters, n.iter=nIter,
   thin=thinSteps )
mcmcChain = as.matrix( codaSamples )
thetaSample = mcmcChain
mode(thetaSample)
length(thetaSample)
thetaSample[1:20, ]
```

```
summary(thetaSample)
res.param(thetaSample)
mtext(
"Estimated Posterior Distribution of ThetaSample",
   side = 3, cex = 2, line = 2)
mtext(
"Estimates Computed by JAGS Through the Rjags Interface",
   side = 3, cex = 1.25, line = 0)
beta.exact = draw.beta(12, 4, plot.dist = FALSE)
x = beta.exact[, 1] - 0.005
y = beta.exact[, 2]
lines(x, y, col = "red")
#_____
# Table 4. Test the R2jags interface between R and JAGS.
rm(list = ls())
graphics.off()
if ( .Platform$OS.type != "windows" ) {
  windows <- function( ... ) X11( ... )
ł
library(R2jags)
search()
# All of the following code (shaded yellow) is the
# same as the code for the Rjags example. It is
# reproduced here because it was deleted when we
# cleared the work space.
modelString = "
# JAGS model specification begins ...
model {
    # Likelihood:
   for ( i in 1:nFlips ) {
       y[i] ~ dbern( theta )
    }
    # Prior distribution:
    theta ~ dbeta( priorA , priorB )
   priorA <- 1
   priorB <- 1
}
# ... JAGS model specification ends.
" # close quote to end modelString
writeLines (modelString, con="model.txt")
dataList = list(
   nFlips = 14 ,
   y = c(1,1,1,1,1,1,1,1,1,1,1,0,0,0)
)
# initsList = list( theta = sum(dataList$y) /
                                                    length(dataList$y)
parameters = c( "theta" )
adaptSteps = 500
burnInSteps = 1000
nChains = 3
numSavedSteps=50000
```

```
thinSteps=1
nIter = ceiling((numSavedSteps*thinSteps) / nChains)
# The following R2jags command has the same
# structure as the preceding Rjags computation.
jags.out = jags(
   data = dataList,
   inits = NULL,
   parameters.to.save = parameters,
   n.iter= nIter + 1000,
   model.file= "model.txt",
   n.chains= nChains,
   n.burnin= burnInSteps,
   n.thin= thinSteps )
# Next we need to find the component of the output
# that contains the samples from the posterior
# distribution of theta.
names(jags.out)
names(jags.out$BUGSoutput)
names(jags.out$BUGSoutput$sims.list)
est.theta = jags.out$BUGSoutput$sims.list$theta
mode(est.theta)
length(est.theta)
est.theta[1:20]
res.param(est.theta)
mtext(
"Estimated Posterior Distribution of est.theta",
   side = 3, cex = 2, line = 2)
mtext(
"Estimates Computed by JAGS Through the R2jags Interface",
   side = 3, cex = 1.25, line = 0)
beta.exact = draw.beta(12, 4, plot.dist = FALSE)
x = beta.exact[, 1] - 0.005
y = beta.exact[ , 2]
lines(x, y, col = "red")
```