An Introduction to **iBUGS**

Yihui Xie*and Jiebiao Wang†

March 28, 2013

iBUGS is an R package which aims to make it easier to call WinBUGS[1], OpenBUGS[2] and JAGS[3] in R, under the operating systems of Windows and Linux. The computation is done respectively by **R2WinBUGS** (Sturtz *et~al.*, 2005), **BRugs** (Thomas *et~al.*, 2006) and **R2jags** (Su and Yajima, 2012) , and the GUI is created by **gWidgetsRGtk2** (Lawrence and Verzani, 2012). To start the GUI, simply type *library(iBUGS)* in R. In case you close the window carelessly, you may start it again by *iBUGS()*.

```r
## the GUI will show up once the package is loaded
library(iBUGS)
## or call iBUGS() to generate another GUI
iBUGS()
```

1 Motivation

We were thinking about writing this package when we saw the default value for the argument `bugs.directory` was "`c:/Program Files/WinBUGS14/`" in the main function *bugs()* of **R2WinBUGS**, as we believed it is not a too difficult task to use R to find the installation directories of WinBUGS and OpenBUGS for most Windows users. We tried the Windows registry approach[4] but found it was not general enough, so we switched to another "brute-force" way: searching for WinBUGS and OpenBUGS in the directory defined by the environment variable "`ProgramFiles`". Most Windows users will install programs in this directory, so we no longer need to manually specify "`bugs.directory = c:/Program Files/WinBUGS14/`".

Based on this trivial motivation, we think we can also make other arguments easier to specify, and generalize this to JAGS and Linux. For instance, we can analyze the BUGS model code and guess the parameter names using regular expressions; then put them in a list and select the ones we are interested in. In this case, we do not need to type the names of parameters one by one, which will otherwise be tedious in R or WinBUGS/OpenBUGS. When we were new to WinBUGS, we often forgot to specify the parameters (node) to monitor in the "Sampler Monitor Tool" panel, and the consequence was we got nothing after a long long waiting and began to regret clicking the "Update" menu too fast.

2 The GUI

The main interface of **iBUGS** looks quite simple as shown in Figure 1. The text box is the place to write the model; it comes with a default sketch of the model. The button "Open" and "Save" can be used to open and save a BUGS model respectively. There will be a tooltip hanging around the text box and showing the path

*Department of Statistics, Iowa State University. Email: xie@yihui.name (or xie@iastate.edu). Homepage: `http://yihui.name`

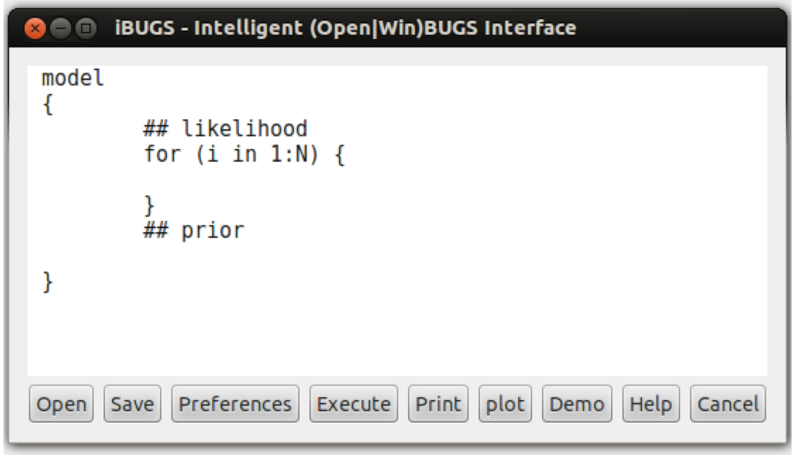
†Department of Health Studies, University of Chicago. Email: randel.wang@gmail.com.

[1] **B**ayesian inference **U**sing **G**ibbs **S**ampling: `http://www.mrc-bsu.cam.ac.uk/bugs/`

[2] Open-source version of BUGS: `http://www.openbugs.info`

[3] **J**ust **A**nother **G**ibbs **S**ampler: `http://www-fis.iarc.fr/~martyn/software/jags/`

[4] `http://yihui.name/en/2010/03/looking-for-software-paths-in-windows-registry/`



```
model
{
        ## likelihood
        for (i in 1:N) {

        }
        ## prior

}
```

Figure 1: The main interface of **iBUGS**

of the current file when you move the mouse over it (the tooltip will be empty if you did not open a model and have not saved the current one).

When you finished writing the model, you need to make sure the data objects you mentioned in the model are already in the current R session. Then open the "Preferences" panel (Figure 2): all the arguments for the function *bugs()* are listed there. If you are familiar with the function *bugs()*, we will not need to explain anything here. The `data` list is read from R's workspace; note it even includes the names from the objects that are *attach()*ed to the R session. For example, the code below can make the two objects `x` and `y` in `dat` visible to **iBUGS**:

```r
dat = list(x = 1:3, y = rnorm(5))
## personally we don't recommend attaching R objects that is a
## bad habit
attach(dat)
```

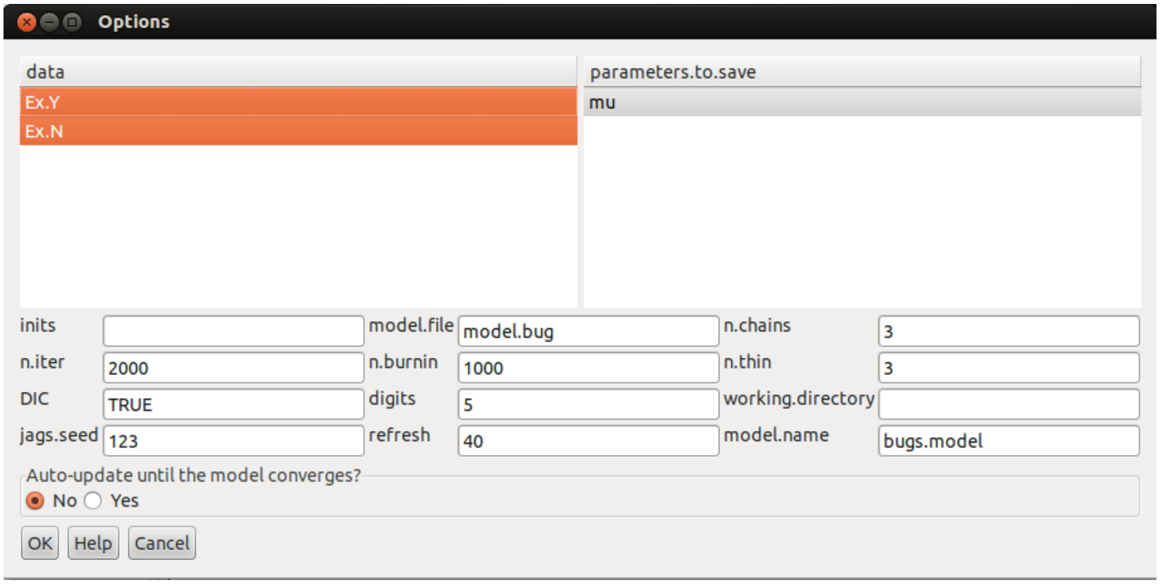
The parameter names, as introduced before, can be found out automatically from the BUGS model. Other options are from the parameters of the *bugs()* function. Note that in the preference panel, there is an additional option "`model.name`" (default to be "`bugs.model`"); the results will be saved to a file with this name in the working directory, so that you can do further analysis with this object by *dget(bugs.options()$model.name)*. The choice of "auto-update" makes **iBUGS** intelligent enough to automatically judge convergence with *autojags()* from package **R2jags**. The "Help" button in the preference panel can read the help pages of *bugs()* or *jags()* correspondingly, so you do not need to turn to *?bugs* or *?jags* each time they use it.

After clicking "OK" button, you can click "Execute" on the main interface to run the model. There is also a simple demo and you can test if **iBUGS** works for you. The buttons of "Print" and "Plot" are used to demonstrate the results, thus you almost do not need to write any code.

3 Technical Details

3.1 WinBUGS / OpenBUGS directory

Most Windows system comes with an environment variable "`ProgramFiles`", which records the default directory to install new software packages. We can search in this directory for WinBUGS or OpenBUGS. Using file manipulation functions such as *list.files()* as well as regular expressions, we can test if any BUGS package has been installed; e.g.



Options

data	parameters.to.save
Ex.Y	mu
Ex.N	

inits		model.file	model.bug	n.chains	3
n.iter	2000	n.burnin	1000	n.thin	3
DIC	TRUE	digits	5	working.directory	
jags.seed	123	refresh	40	model.name	bugs.model

Auto-update until the model converges?
● No ○ Yes

Figure 2: The preference panel

```
if (nzchar(prog <- Sys.getenv("ProgramFiles")) && length(bugs.dir <- list.files(prog,
    "^(Open|Win)BUGS.*")) && length(bugs.exe <- dirname(list.files(file.path(prog,
    bugs.dir), pattern = "(Open|Win)BUGS.*\\.exe$", full.names = TRUE,
    recursive = TRUE)))) {
    ## if we can find OpenBUGS, use it prior to WinBUGS
    program = ifelse(length(grep("OpenBUGS", bugs.exe)), "OpenBUGS",
        "WinBUGS")
    ## ignore multiple directories if (several versions of) BUGS
    ## installed in multiple places
    bugs.directory = bugs.exe[grep(program, bugs.exe)][1]
}
```

Note if both WinBUGS and OpenBUGS are detected, **iBUGS** will prefer OpenBUGS, as we know the development of WinBUGS has stopped. The JAGS directory can also be detected in the same way.

3.2 GUI construction

Building a GUI in R with the **gWidgets** (Verzani, 2012) package is quite easy and even dynamic. That means you can generate GUI components dynamically whenever you need them. The **gWidgets** package comes with 4 types of interface, namely GTK+ (**gWidgetsRGtk2**), tcltk (**gWidgetstcltk**), Java (**gWidgetsrJava**) and WWW (**gWidgetsWWW**). You do not need to deal with these four specific packages – just play with **gWidgets** and specify your GUI type. Personally we like the GTK+ interface most, so we made it the default one for **iBUGS**. Here is a short demo:

```
library(gWidgets)
options(guiToolkit = "RGtk2")
## create a window and add a button to it
gw = gwindow("GUI Demo")
gb = gbutton("Click me!", container = gw, handler = function(h, ...) {
    svalue(h$obj) = paste(svalue(h$obj), "haha!")
})
```

In **gWidgets**, there are several widgets available (text boxes, buttons, drop-down list, ...), and you just need to think about how to arrange them. You can attach an event (handler) to a widget so that this event will be called when users take an action to the widget.

4 Future Work

There is a list of things in our mind:

1. We know little about Mac, but we guess with proper Wine emulation specifications, **iBUGS** can also work (but not so intelligent); we will appreciate help from Mac users;

2. We have not made up my mind yet: is it worth providing a menu for diagnostics? (e.g. Plot ▷ Gelman-Rubin-Brooks.) Sounds like we are re-inventing the wheel.

Acknowledgment

Yihui would like to thank Prof Di Cook for her support and ideas, Ted Peterson for the IT support, and Dr Alyson Wilson for introducing him to the Bayesian world.

References

Lawrence M, Verzani J (2012). *gWidgetsRGtk2: Toolkit implementation of gWidgets for RGtk2*. R package version 0.0-81, URL `http://CRAN.R-project.org/package=gWidgetsRGtk2`.

Sturtz S, Ligges U, Gelman A (2005). "R2WinBUGS: A Package for Running WinBUGS from R." *Journal of Statistical Software*, **12**(3), 1–16. URL `http://www.jstatsoft.org`.

Su YS, Yajima M (2012). *R2jags: A Package for Running jags from R*. R package version 0.03-08, URL `http://CRAN.R-project.org/package=R2jags`.

Thomas A, O'Hara B, Ligges U, Sturtz S (2006). "Making BUGS Open." *R News*, **6**(1), 12–17. URL `http://cran.r-project.org/doc/Rnews/`.

Verzani J (2012). *gWidgets: gWidgets API for building toolkit-independent, interactive GUIs*. R package version 0.0-52, URL `http://CRAN.R-project.org/package=gWidgets`.