P.O. Box 614
Augusta, ME 04332-0614
info@metruninstitute.org

2 Tunxis Road, Suite 112
Tariffville, CT 06081
Phone: 860.735.7043
Fax: 860.760.6014

bugsParallel: Open Source Software Tools for Parallel Computation of Multiple MCMC Chains with WinBUGS

Installation and Use Instructions

1. Background

When used for a Bayesian analysis, Markov chain Monte Carlo (MCMC) simulations generate samples that approximate the joint posterior distribution of the sampled parameters. The sequential nature of MCMC simulation limits the benefits of parallel computation when applied to a single chain. However parallel computation of multiple chains is an "embarassingly parallel" problem that can substantially reduce computation time and is relatively easy to implement using freely available software.

bugsParallel is a set of R [5] functions for distributed computing with WinBUGS 1.4.* [3]. It is a modified subset of the R2WinBUGS package [7]. It uses the R package Rmpi [9] to implement a network of workstations and the rlecuyer package [6] for parallel random number generation. This assures appropriately independent initial estimates when they are specified via a function that uses random number generation to generate the initial estimates. The seeds used by each WinBUGS chain are generated in bugsParallel by the master node. The master generates a vector of random seeds—one per chain. The seed vector and chain identifier are passed to each slave that uses the appropriate element of that vector. The seed is passed to WinBUGS via the undocumented set.seed WinBUGS script command.

bugsParallel has been installed successfully on Windows XP and Mac OS X 10.5. Implementation involves installation of the following freely distributed software components:

- Microsoft Windows implementation:
  - MPICH2 1.0.8 for Windows [2], an implementation of the Message-Passing Interface (MPI) for parallel computation,
  - R 2.6.2 for Windows,
  - R packages Rmpi 0.5-5 and rlecuyer 0.1.
- Mac OS X implementation:
  - OpenMPI [1] (pre-installed on MAC OS X 10.5).
  - R 2.8.1 for Mac OS X
  - Wine 1.0.1 [8]
  - R packages Rmpi 0.5-6 and rlecuyer 0.1.

- WinBUGS 1.4.3

bugsParallel is currently provided as an R script that the user accesses via a source command in a user written R script.

The version numbers cited above are known to result in successful installations. Other versions may be acceptable but they have not been tested by the authors at the time of this writing. Rmpi 0.5-5 did not run successfully with R versions 2.7.* or 2.8.* for Windows, thus the reason for using an earlier version of R for Windows. A Windows binary version of Rmpi 0.5-6 is expected soon and that may resolve the problem.

The software components required for implementing bugsParallel are contained in the zip file called bugsParallel.zip or may be obtained via internet download. For each computer in your cluster place a copy of bugsParallel.zip in any convenient folder and unzip the file to extract its contents. This should produce a folder called bugsParallel.

2. Windows XP Installation and Use

2.1. **Installing Rmpi.**

(1) Create users with the same username and password on all computers in your cluster.
   - I think they may have to have administrator privileges but I havent checked that. I am certain that all remote nodes must have the same username and password (at least using the setup approach described here), but I think the master may have a different username. I accidentally ran a job from a master with a different username than the remotes and it worked OK.

(2) Install Microsoft .Net Framework 2.0
   (a) Double-click on dotnetfx.exe and follow on-screen instructions.
   (b) Run Windows Update and install .Net service pack 1.

(3) Install MPICH2 1.0.8
   (a) Double-click on mpich2-1.0.8-win-ia32.msi and follow on-screen instructions. Accept defaults except when you get to the dialog window headed "Select Installation Folder", select the radio button labeled "Everyone".
   (b) Run wmpiconfig and specify the "hosts", e.g., "localhost:3 wrgdesktop:2" for my setup where I both my laptop and desktop have 2 processors and I am using the laptop as the "master". For the "master" specify one more than the number of processors. This means 1 master + 2 slaves for a dual processor machine.
   (c) Run wmpiregister to put the username and password in the registry.

(4) Install R 2.6.2, Rmpi 0.5-5 and rlecuyer 0.1
   (a) Double-click on R-2.6.2-win32.exe. Follow the on-screen instructions. Accept all defaults.
   (b) Launch R-2.6.2 by double-clicking the icon on your desktop.
   (c) To install Rmpi 0.5-5 select "Install package(s) from local zip file" from the packages menu. Navigate the file browser to the bugsParallel directory and select Rmpi_0.5-5.zip.
   (d) To install rlecuyer 0.1 select "Install package(s) from local zip file" from the packages menu. Navigate the file browser to the bugsParallel directory and select rlecuyer_0.1.zip.
   (e) Put the shortcut "mpiexec Rgui 2.6.2" in a convenient location like your desktop. This is only required for computers you are likely to use as a "master" node.

(5) Adjust firewall settings
   (a) This is not necessary if all processors are in the same computer.

(b) From the Start menu open the control panel. Then select the Security Center. Click on the Windows Firewall. Go to the Exceptions tab and add the following programs to the exceptions list.
   (i) C:\Program Files\R\R-2.6.2\bin\Rgui.exe
   (ii) C:\Program Files\R\R-2.6.2\bin\Rterm.exe
   (iii) C:\Program Files\MPICH2\bin\smpd.exe
   (iv) C:\Program Files\MPICH2\bin\mpiexec.exe

Consistency in paths across the machines seems to be critical because Rmpi often specifies files using full paths. For example it did not work when 2 different versions of R were used because it looked for the R files in the same path on the remote machine as the master machine.

2.2. **Rmpi example.** The following example illustrates replacing lapply with mpi.applyLB to do a loop in R. mpi.applyLB distributes the computations to multiple nodes. For this example the elapsed time for the mpi.applyLB loop was 93 s on a 4 processor setup and the lapply loop took 297 s. To use this script copy the text to a text file and change the number of slave nodes ("nslaves") in the call to mpi.spawn.Rslaves to a number less than or equal to the number of processors on your network. Open the script file in R and execute it.

```
library(Rmpi)
mpi.spawn.Rslaves(nslaves=4)

cpt1 = function(x,dose,cl,v,sd.cl,sd.v,nsim){
    cpt1.1 = function(x,cl,v){
        exp(-cl*x/v)/v
    }
    cl = exp(rnorm(nsim,log(cl),sd.cl))
    v = exp(rnorm(nsim,log(v),sd.v))
    dose*apply(sapply(1:length(cl),function(i,x,cl,v) cpt1.1(x,cl[i],v[i]),
        x=x,cl=cl,v=v),1,mean)
}

nrep = 2000
cl = exp(rnorm(nrep,log(10),0.2))
v = exp(rnorm(nrep,log(100),0.2))
sd.cl = exp(rnorm(nrep,log(0.2),0.2))
sd.v = exp(rnorm(nrep,log(0.2),0.2))
xx = 0:12

system.time(
junk <- mpi.applyLB(1:length(cl),function(i,cpt1,xx,dose,cl,v,sd.cl,sd.v,nsim){
    cpt1(xx,dose,cl[i],v[i],sd.cl[i],sd.v[i],nsim)},
    cpt1=cpt1,xx=xx,dose=100,cl=cl,v=v,sd.cl=sd.cl,sd.v=sd.v,nsim=10000)
)

system.time(
junk <- lapply(1:length(cl),function(i,cpt1,xx,dose,cl,v,sd.cl,sd.v,nsim){
    cpt1(xx,dose,cl[i],v[i],sd.cl[i],sd.v[i],nsim)},
    cpt1=cpt1,xx=xx,dose=100,cl=cl,v=v,sd.cl=sd.cl,sd.v=sd.v,nsim=10000)
)
```

```
mpi.close.Rslaves()
```

2.3. **Using bugsParallel.** The folder named "FXaExample1" contains an example that illustrates the use of bugsParallel. This serves as a template, an instructive example, and a test of the installation. WinBUGS 1.4.* (`http://www.mrc-bsu.cam.ac.uk/bugs/welcome.shtml`) must be installed on each computer in the cluster you will be using. To run the example do the following:

(1) Place the FXaExample1 folder in any convenient location.
(2) Do the same with the bugsTools folder
(3) Launch R 2.6.2 by double-clicking on the shortcut named "mpiexec Rgui 2.6.2".
(4) Install the R package called coda by selecting "Install package(s) from local zip file" from the packages menu. Navigate the file browser to the bugsParallel directory and select coda_0.13-1.
(5) Open the R script file named FXaExample1.R using "Open script..." in the File menu.
(6) Change second line of the script to indicate the full path name of the FXaExample1 folder on your computer.
(7) Change third line of the script to indicate the full path name of the bugsTools folder on your computer.
(8) Change the line "nSlaves = 2" to specify a number less than or equal to the number of nodes in your cluster.
(9) Run the script by making the script the active window and then selecting "Run all" from the Edit menu.
    - The statement containing the call to bugsParallel required an lapsed time of ~ 160 s on cluster of 2 dual processor machines running MS Windows XP—a laptop with an Intel Core 2 Duo T7600 (2.33 GHz/2 GB RAM and a desktop with an Intel Xeon 5150 (2.66 GHz/3 GB RAM).
(10) When the script has finished running end the R session using the command "mpi.quit()" rather than the usual "q()". This is necessary to properly end the MPI processes.
(11) The script should have created a subdirectory of the R working directory called "FXaExample1" containing the results of the WinBUGS analysis. The contents should resemble those in the supplied "FXaExample1Results" directory.

3. Mac OS X 10.5 Installation

3.1. **Installing Rmpi and Wine.** To date the authors only have experience using bugsParallel on a single Mac machine with multiple processors. In principle it should work across a network of machines but no instructions are provided here for configuring such a network.

(1) Install Wine
    - The authors have experience with Wine installed via MacPorts [4] and with Darwine. The instructions here are for the former.
    (a) Install MacPorts following instructions on the MacPorts website (`http://www.macports.org/index.php`).
    (b) Open an xterm window.
    (c) Install Wine using the command "sudo port install wine". This takes a while, particularly the "building wine" step.
(2) Install R, Rmpi and rlecuyer
    (a) Download and install the current version of R for Mac OS X from `http://cran.r-project.org/bin/macosx/` or any of the many CRAN mirror sites.

(b) Launch R by double-clicking on the R.app file in the Applications folder or click on the R icon in the dock.
(c) To install Rmpi open the Package Installer from the "Packages & Data" menu. Change the "Packages Repository" to "CRAN (sources)" (text selection box in upper left hand corner of R Package Installer window). If a list of packages does not already appear then click on the "Get List" button. Scroll down until you find Rmpi. Select Rmpi by clicking on it. Then click on the "Install Selected" button. It will take a few minutes to install. You will see the progress of the installation in the R Console.
(d) Install rlecuyer by again opening the Package Installer. This time specify "CRAN (binaries)" for the "Packages Repository". If a list of packages does not already appear then click on the "Get List" button. Scroll down until you find rlecuyer. Select rlecuyer by clicking on it. Then click on the "Install Selected" button.

3.2. **Rmpi example.** The Rmpi example script in section 2.2 also works on the Mac and should be executed to test the installation.

3.3. **Using bugsParallel.** The folder named "FXaExample1" contains an example that illustrates the use of bugsParallel. This serves as a template, an instructive example, and a test of the installation. WinBUGS 1.4.* (`http://www.mrc-bsu.cam.ac.uk/bugs/welcome.shtml`) must be installed on each computer in the cluster you will be using. To run the example do the following:

(1) Place the FXaExample1 folder in any convenient location.
(2) Do the same with the bugsTools folder
(3) Launch R.
(4) Install the R package called coda by opening the Package Installer. Specify "CRAN (binaries)" for the "Packages Repository". If a list of packages does not already appear then click on the "Get List" button. Scroll down until you find coda. Select coda by clicking on it. Then click on the "Install Selected" button. Alternatively you may prefer to install R2WinBUGS and its dependencies (including coda) in the same manner except to remember to check the "install dependencies" box.
(5) Open the R script file named FXaExample1.R using "Open Document..." in the File menu.
(6) Change second line of the script to indicate the full path name of the FXaExample1 folder on your computer.
(7) Change third line of the script to indicate the full path name of the bugsTools folder on your computer.
(8) Change the line "nSlaves = 2" to specify a number less than or equal to the number of nodes in your cluster.
(9) Run the script by making the script the active window and then selecting "Run all" from the Edit menu.
(10) When the script has finished running end the R session using the command "mpi.quit()" rather than the usual "q()". This is necessary to properly end the MPI processes.
(11) The script should have created a subdirectory of the R working directory called "FXaExample1" containing the results of the WinBUGS analysis. The contents should resemble those in the supplied "FXaExample1Results" directory.

If you have any difficulties with the installation or just have some questions about bugsParallel, please contact Bill Gillespie at billg@metrumrg.com.

References

[1] Edgar Gabriel, Graham E. Fagg, George Bosilca, Thara Angskun, Jack J. Dongarra, Jeffrey M. Squyres, Vishal Sahay, Prabhanjan Kambadur, Brian Barrett, Andrew Lumsdaine, Ralph H. Castain, David J. Daniel, Richard L. Graham, and Timothy S. Woodall. Open MPI: Goals, concept, and design of a next generation MPI implementation. In *Proceedings, 11th European PVM/MPI Users' Group Meeting*, Budapest, Hungary, September 2004.

[2] William Gropp, Ewing Lusk, David Ashton, Darius Buntinas, Ralph Butler, Anthony Chan, Rob Ross, Rajeev Thakur, and Brian Toonen. *MPICH2 Users Guide, Version 0.4*. Mathematics and Computer Science Division, Argonne National Laboratory, 2004. http://phase.hpcc.jp/mirrors/mpi/mpich2/downloads/mpich2-doc-user.pdf.

[3] D.J. Lunn, A. Thomas, N. Best, and D. Spiegelhalter. WinBUGS – a bayesian modelling framework: concepts, structure, and extensibility. *Statistics and Computing*, 10:325–337, 2000.

[4] The MacPorts website. http://www.macports.org/.

[5] R Development Core Team. *R: A Language and Environment for Statistical Computing*. R Foundation for Statistical Computing, Vienna, Austria, 2008. ISBN 3-900051-07-0.

[6] Hana Sevcikova. *The relecuyer Package*. http://cran.r-project.org/web/packages/rlecuyer/index.html.

[7] Andrew Gelman Sibylle Sturtz, Uwe Ligges. R2WinBUGS: A package for running WinBUGS from R. *Journal of Statistical Software*, 12(3), 2005. http://www.jstatsoft.org/v12/i03.

[8] WineHQ website. http://www.winehq.org/.

[9] Hao Yu. *The Rmpi Package*. http://www.stats.uwo.ca/faculty/yu/Rmpi/.