

Package ‘R2jags’

November 12, 2009

Version 0.01-27

Date 2009-11-9

Title A Package for Running jags from R

Author Yu-Sung Su <ys463@columbia.edu>, Masanao Yajima
<yajima@stat.columbia.edu>,

Maintainer Yu-Sung Su <ys463@columbia.edu>

Depends methods, R(>= 2.7.0), coda, R2WinBUGS, rjags(>= 1.0.3-8)

SystemRequirements jags (>= 1.0.3)

Description Using this package to call jags from R.

License GPL (>= 2)

Repository CRAN

Date/Publication 2009-11-12 11:36:57

R topics documented:

| | |
|-----------------------|---|
| attach.jags | 2 |
| autojags | 3 |
| jags | 4 |
| jags2bugs | 7 |
| recompile | 8 |
| traceplot | 8 |

| | |
|--------------|-----------|
| Index | 10 |
|--------------|-----------|

`attach.jags`*Attach/detach elements of jags objects to search path*

Description

These are wrapper functions for `attach.bugs` and `detach.bugs`, which attach or detach three-way-simulation array of bugs object to the search path. See `attach.all` for details.

Usage

```
attach.jags(x, overwrite = NA)
detach.jags()
```

Arguments

| | |
|------------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <code>x</code> | An <code>rjags</code> object. |
| <code>overwrite</code> | If <code>TRUE</code> , objects with identical names in the Workspace (<code>.GlobalEnv</code>) that are masking objects in the database to be attached will be deleted. If <code>NA</code> (the default) and an interactive session is running, a dialog box asks the user whether masking objects should be deleted. In non-interactive mode, behaviour is identical to <code>overwrite=FALSE</code> , i.e. nothing will be deleted. |

Details

See `attach.bugs` for details

Author(s)

Yu-Sung Su <ys463@columbia.edu>,

References

Sibylle Sturtz and Uwe Ligges and Andrew Gelman. (2005). “R2WinBUGS: A Package for Running WinBUGS from R.” *Journal of Statistical Software* 3 (12): 1–6.

Examples

```
# See the example in ?jags for the usage.
```

`autojags`*Function for auto-updating jags until the model converges*

Description

The `autojags` takes a `rjags` object as input. `autojags` will update the model until it converges.

Usage

```
## S3 method for class 'rjags':
update(object, n.iter=1000, n.thin=1,
        refresh=n.iter/50, progress.bar = "text", ...)
autojags(object, n.iter=1000, n.thin=1, Rhat=1.1, n.update=2,
         refresh=n.iter/50, progress.bar = "text", ...)
```

Arguments

| | |
|---------------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <code>object</code> | an object of <code>rjags</code> class. |
| <code>n.iter</code> | number of total iterations per chain, default=1000 |
| <code>n.thin</code> | thinning rate. Must be a positive integer, default=1 |
| <code>...</code> | further arguments pass to or from other methods. |
| <code>Rhat</code> | convergence criterion, default=1.1. |
| <code>n.update</code> | the max number of updates, default=2. |
| <code>refresh</code> | refresh frequency for progress bar, default is <code>n.iter/50</code> |
| <code>progress.bar</code> | type of progress bar. Possible values are “text”, “gui”, and “none”. Type “text” is displayed on the R console. Type “gui” is a graphical progress bar in a new window. The progress bar is suppressed if <code>progress.bar</code> is “none” |

Author(s)

Yu-Sung Su <ys463@columbia.edu>

References

Gelman, A., Carlin, J.B., Stern, H.S., Rubin, D.B. (2003): *Bayesian Data Analysis*, 2nd edition, CRC Press.

Examples

```
# see ?jags for an example.
```

jags

Run jags from R

Description

The `jags` function takes data and starting values as input. It automatically writes a `jags` script, calls the model, and saves the simulations for easy access in R.

Usage

```
jags(data, inits, parameters.to.save, model.file="model.bug",
      n.chains=3, n.iter=2000, n.burnin=floor(n.iter/2),
      n.thin=max(1, floor((n.iter - n.burnin) / 1000)),
      DIC=TRUE, working.directory=NULL,
      refresh = n.iter/50, progress.bar = "text")
```

```
jags2(data, inits, parameters.to.save, model.file="model.bug",
       n.chains=3, n.iter=2000, n.burnin=floor(n.iter/2),
       n.thin=max(1, floor((n.iter - n.burnin) / 1000)),
       DIC=TRUE, jags.path="",
       working.directory=NULL, clearWD=TRUE,
       refresh = n.iter/50)
```

Arguments

| | |
|---------------------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <code>data</code> | a vector or list of the names of the data objects used by the model, or a list of the data objects themselves. |
| <code>inits</code> | a list with <code>n.chains</code> elements; each element of the list is itself a list of starting values for the BUGS model, <i>or</i> a function creating (possibly random) initial values. If <code>inits</code> is <code>NULL</code> , JAGS will generate initial values for parameters. |
| <code>parameters.to.save</code> | character vector of the names of the parameters to save which should be monitored. |
| <code>model.file</code> | file containing the model written in BUGS code. |
| <code>n.chains</code> | number of Markov chains (default: 3) |
| <code>n.iter</code> | number of total iterations per chain (including burn in; default: 2000) |
| <code>n.burnin</code> | length of burn in, i.e. number of iterations to discard at the beginning. Default is <code>n.iter/2</code> , that is, discarding the first half of the simulations. If <code>n.burnin</code> is 0, <code>jags()</code> will run 100 iterations for adaption. |
| <code>n.thin</code> | thinning rate. Must be a positive integer. Set <code>n.thin > 1</code> to save memory and computation time if <code>n.iter</code> is large. Default is <code>max(1, floor(n.chains * (n.iter - n.burnin) / 1000))</code> which will only thin if there are at least 2000 simulations. |

| | |
|-------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| DIC | logical; if TRUE (default), compute deviance, pD, and DIC. The rule $pD = \text{var}(\text{deviance}) / 2$ is used. |
| working.directory | sets working directory during execution of this function; This should be the directory where model file is. |
| jags.path | directory that contains the jags executable. The default is "". |
| clearWD | indicating whether the files 'data.txt', 'inits[1:n.chains].txt', 'codalIndex.txt', 'jagsscript.txt', and 'CODAchain[1:nchains].txt' should be removed after jags has finished, default=TRUE. |
| refresh | refresh frequency for progress bar, default is <code>n.iter/50</code> |
| progress.bar | type of progress bar. Possible values are "text", "gui", and "none". Type "text" is displayed on the R console. Type "gui" is a graphical progress bar in a new window. The progress bar is suppressed if <code>progress.bar</code> is "none" |

Details

To run:

1. Write a **BUGS** model in an ASCII file.
2. Go into R.
3. Prepare the inputs for the `jags` function and run it (see Example section).
4. The model will now run in **JAGS**. It might take awhile. You will see things happening in the R console.

BUGS version support:

- **jags** 1.0.3default

Author(s)

Yu-Sung Su <ys463@columbia.edu>, Masanao Yajima <yajima@stat.columbia.edu>

References

Plummer, Martyn (2003) "JAGS: A program for analysis of Bayesian graphical models using Gibbs sampling." <http://citeseer.ist.psu.edu/plummer03jags.html>.

Gelman, A., Carlin, J. B., Stern, H.S., Rubin, D.B. (2003) *Bayesian Data Analysis*, 2nd edition, CRC Press.

Sibylle Sturtz and Uwe Ligges and Andrew Gelman. (2005). "R2WinBUGS: A Package for Running WinBUGS from R." *Journal of Statistical Software* 3 (12): 1–6.

Examples

```
# An example model file is given in:
model.file <- system.file(package="R2jags", "model", "schools.txt")
# Let's take a look:
file.show(model.file)
```

```

# data
J <- 8.0
y <- c(28.4,7.9,-2.8,6.8,-0.6,0.6,18.0,12.2)
sd <- c(14.9,10.2,16.3,11.0,9.4,11.4,10.4,17.6)

jags.data <- list("y","sd","J")
jags.params <- c("mu","sigma","theta")
jags.inits <- function(){
  list("mu"=rnorm(1),"sigma"=runif(1),"theta"=rnorm(J))
}

#####
# using jags #
#####
jagsfit <- jags(data=jags.data, inits=jags.inits, jags.params,
  n.iter=5000, model.file=model.file)

# display the output
print(jagsfit)
plot(jagsfit)

# traceplot
traceplot(jagsfit)

# if the model does not converge, update it!
jagsfit.upd <- update(jagsfit, n.iter=1000)
print(jagsfit.upd)
plot(jagsfit.upd)

# or auto update it until it converges! see ?autojags for details
jagsfit.upd <- autojags(jagsfit)

# to get DIC or specify DIC=TRUE in jags() or do the following
dic.samples(jagsfit.upd$model, n.iter=1000, type="pD")

# attach jags object into search path see "attach.bugs" for details
attach.jags(jagsfit.upd)

# this will show a 3-way array of the bugs.sim object, for example:
mu

# detach jags object into search path see "attach.bugs" for details
detach.jags()

# to pick up the last save session
# for example, load("RWorkspace.Rdata")
recompile(jagsfit)
jagsfit.upd <- update(jagsfit)

#####
# using jags2 #
#####

```

```
## jags cannot be updated, but produces coda files
## You may need to edit "jags.path",
## also you need write access in the working directory:
## currently works only under Windows OS
## e.g. setwd("c:/")

## NOT RUN HERE
#jagsfit <- jags2(data=jags.data, inits=jags.inits, jags.params,
# n.iter=5000, model.file=model.file)
#print(jagsfit)
#plot(jagsfit)
```

jags2bugs

Read jags output files in CODA format

Description

This function reads Markov Chain Monte Carlo output in the CODA format produced by **jags** and returns an object of class `mcmc.list` for further output analysis using the **coda** package.

Usage

```
jags2bugs(path=getwd(), parameters.to.save,
          n.chains=3, n.iter=2000, n.burnin=1000, n.thin=2,
          DIC=TRUE)
```

Arguments

| | |
|---------------------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <code>path</code> | sets working directory during execution of this function; This should be the directory where CODA files are. |
| <code>parameters.to.save</code> | character vector of the names of the parameters to save which should be monitored. |
| <code>n.chains</code> | number of Markov chains (default: 3) |
| <code>n.iter</code> | number of total iterations per chain (including burn in; default: 2000) |
| <code>n.burnin</code> | length of burn in, i.e. number of iterations to discard at the beginning. Default is <code>n.iter/2</code> , that is, discarding the first half of the simulations. |
| <code>n.thin</code> | thinning rate, default is 2 |
| <code>DIC</code> | logical; if TRUE (default), compute deviance, pD, and DIC. The rule <code>pD=var(deviance)/2</code> is used. |

Author(s)

Yu-Sung Su <ys463@columbia.edu>, Masanao Yajima <yajima@stat.columbia.edu>

 recompile

Function for recompiling rjags object

Description

The `recompile` takes a `rjags` object as input. `recompile` will re-compile the previous saved `rjags` object.

Usage

```
recompile(object, n.iter, refresh, progress.bar)
## S3 method for class 'rjags':
recompile(object, n.iter=100, refresh=n.iter/50,
          progress.bar = "text")
```

Arguments

| | |
|---------------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <code>object</code> | an object of <code>rjags</code> class. |
| <code>n.iter</code> | number of iteration for adapting, default is 100 |
| <code>refresh</code> | refresh frequency for progress bar, default is <code>n.iter/50</code> |
| <code>progress.bar</code> | type of progress bar. Possible values are “text”, “gui”, and “none”. Type “text” is displayed on the R console. Type “gui” is a graphical progress bar in a new window. The progress bar is suppressed if <code>progress.bar</code> is “none” |

Author(s)

Yu-Sung Su <ys463@columbia.edu>

Examples

```
# see ?jags for an example.
```

 traceplot

Trace plot of bugs object

Description

Displays a plot of iterations vs. sampled values for each variable in the chain, with a separate plot per variable.

Usage

```
traceplot(x, ...)  
## S4 method for signature 'rjags':  
traceplot(x, mfrow = c(1, 1), varname = NULL,  
  match.head = TRUE, ask = TRUE,  
  col = rainbow( x$n.chains ),  
  lty = 1, lwd = 1, ...)
```

Arguments

| | |
|------------|--------------------------------------------------------------------------------------------|
| x | A bugs object |
| mfrow | graphical parameter (see <code>par</code>) |
| varname | vector of variable names to plot |
| match.head | matches the variable names by the beginning of the variable names in bugs object |
| ask | logical; if TRUE, the user is <i>asked</i> before each plot, see <code>par(ask=.)</code> . |
| col | graphical parameter (see <code>par</code>) |
| lty | graphical parameter (see <code>par</code>) |
| lwd | graphical parameter (see <code>par</code>) |
| ... | further graphical parameters |

Author(s)

Masanao Yajima <yajima@stat.columbia.edu>.

See Also

[densplot](#), [plot.mcmc](#), [traceplot](#)

Index

*Topic **IO**

jags2bugs, 6

*Topic **file**

jags2bugs, 6

*Topic **hplot**

traceplot, 8

*Topic **interface**

attach.jags, 1

jags, 3

*Topic **models**

autojags, 2

jags, 3

recompile, 7

attach.all, 1

attach.bugs, 1, 2

attach.jags, 1

autojags, 2

densplot, 9

detach.bugs, 1

detach.jags (*attach.jags*), 1

jags, 3

jags2 (*jags*), 3

jags2bugs, 6

mcmc.list, 6

plot.mcmc, 9

recompile, 7

rjags-class (*jags*), 3

traceplot, 8, 9

traceplot, mcmc.list-method

(*traceplot*), 8

traceplot, rjags-method

(*traceplot*), 8

traceplot.default (*traceplot*), 8

update.rjags (*autojags*), 2