Appendix B:
A Guide to Installation and Use of *bayesm*

An R package is a pre-packaged collection of functions, datasets and documentation which can be easily installed and updated from within R. The package resides on the CRAN network of world-wide mirror sites so that its availability is assured. ***bayesm*** is an R package that implement the methods in this book.

1. Installing *bayesm*

bayesm can be installed from a CRAN mirror site by selecting the "Packages" tab on the toolbar in the R software package. Users must first set the CRAN mirror, then install ***bayesm*** directly from CRAN by simply clicking on "bayesm" in the pop-up window displayed. You can also install ***bayesm*** by using the R command, `install.packages("bayesm")`.

The install command downloads the content of the package and builds a directory tree in the `library` subdirectory of the R install. If you upgrade your version of R, you must re-install ***bayesm.***

2. Using *bayesm*

Once ***bayesm*** has been installed, you must "load" the package for each R session. This is done with the command `library(bayesm)`. In order to avoid giving this command each time you invoke R, it is convenient to include the command in the `.Rprofile` file. `.Rprofile` is executed each time R is invoked. `.Rprofile` should be located in directory pointed to by the environmental variable, HOME. Environmental variables are set by right-clicking on the "My Computer" icon on your desktop, selecting "Properties", and then selecting the "Advanced" tab.

Sample contenxt of .Rprofile are given below:

```
# Example of .Rprofile
     .First <- function() {
        library(bayesm)
      }
     .Last <- function()  cat("\n   Goodbye!\n\n")
```

Once loaded, you can use any of the more than 40 functions in ***bayesm*** just by inserted the function call in your source code. For all except the most trivial problems, it is best to edit a source file with your code and then "source" this file into R. See appendix A for more details.

The following tips are useful in using ***bayesm***:

1. If you don't know much about R, read appendix A. ***bayesm*** has many functions defined in it. The "turn-key" or "end-user" functions start with the letter `r`, e.g. `rmnpGibbs` is the Gibbs Sampler for the multinomial probit model.
2. Check the examples. Each function has an example file. To view the example for a function, use the R command `?`, e.g. `?rmnpGibbs`. The example will be listed at the bottom of the displayed help text. You can also find the examples in the R program directory tree, e.g. C:\Program Files\R\rwXXXX\library\bayesm\R-ex. You may have to unzip these files.
3. The best way is work with ***bayesm*** functions is to copy the examples into a .R file and then edit the file to read in your own data and run the function. At first, use as many defaults as possible (e.g. Priors) to make sure that the function is working properly on your example.

3. Obtaining Help on *bayesm*

Once ***bayesm*** has been loaded, you can obtain help in various ways.

To see a list of all the functions in ***bayesm***, use the command, `library(help=bayesm)`. This command will produce only brief summaries of each function in ***bayesm***.

To learn more, use the `help()` or `?` commands as for any function in R. This will provide the standard information on usage, arguments and output from the function. The most useful aspect of these files is the examples section. For example, `?rmnlIndepMetrop` produces:

```
rmnlIndepMetrop            package:bayesm            R Documentation

MCMC Algorithm for Multinomial Logit Model
Description:
     'rmnIndepMetrop' implements Independence Metropolis for the MNL.

Usage:
     rmnlIndepMetrop(Data, Prior, Mcmc)

Arguments:
    Data: list(p,X,y)
   Prior: list(A,betabar)  optional
    Mcmc: list(R,keep,nu)

Details:
     Model:   Pr(y=j) = exp(x_j'beta)/sum_k{e^{x_k'beta}}.

     Prior:   beta ~ N(betabar,A^{-1})

     list arguments contain:
   'p' number of alternatives
   'X' nobs*m x nvar matrix
   'y' nobs vector of multinomial outcomes (1,..., m)
   'A' nvar x nvar pds prior prec matrix (def: .01I)
   'betabar' nvar x 1 prior mean (def: 0)
   'R' number of MCMC draws
   'keep' MCMC thinning parm: keep every keepth draw (def: 1)
   'nu' degrees of freedom parameter for independence t density (def:
        6)

Value:
     a list containing:
betadraw: R/keep x nvar array of beta draws
 acceptr: acceptance rate of Metropolis draws
```

```
See Also:
     'rhierMnlRwMixture'

Examples:
     ##
     if(nchar(Sys.getenv("LONG_TEST")) != 0) # set env var LONG_TEST to run
     {

     set.seed(66)
     n=200; p=3; beta=c(1,-1,1.5,.5)
     simout=simmnl(m,n,beta)
     A=diag(c(rep(.01,length(beta)))); betabar=rep(0,length(beta))

     R=2000
     Data=list(y=simout$y,X=simout$X,p=p);
     Mcmc=list(R=R,keep=1) ; Prior=list(A=A,betabar=betabar)
     out=rmnlIndepMetrop(Data=Data,Prior=Prior,Mcmc=Mcmc)
     cat(" Betadraws ",fill=TRUE)
     mat=apply(out$betadraw,2,quantile,probs=c(.01,.05,.5,.95,.99))
     mat=rbind(beta,mat); rownames(mat)[1]="beta"; print(mat)
     }
```

The examples section could be clipped out and used as a template for running your own data.

To list all functions related to a specific topic, use the command `help.search("topic")`. For example, `help.search("bayes")` will produce:

```
Help files with alias or concept or title matching 'bayes' using fuzzy
matching:



breg(bayesm)            Posterior Draws from a Univariate Regression
                        with Unit Error Variance
eMixMargDen(bayesm)     Compute Marginal Densities of A Normal Mixture
                        Averaged over MCMC Draws
logMargDenNR(bayesm)    Compute Log Marginal Density Using
                        Newton-Raftery Approx
rbprobitGibbs(bayesm)   Gibbs Sampler (Albert and Chib) for Binary
                        Probit
rhierLinearModel(bayesm)
                        Gibbs Sampler for Hierarchical Linear Model
rhierMnlRwMixture(bayesm)
                        MCMC Algorithm for Hierarchical Multinomial
                        Logit with Mixture of Normals Heterogeneity
rivGibbs(bayesm)        Gibbs Sampler for Linear "IV" Model
rmnlIndepMetrop(bayesm)
                        MCMC Algorithm for Multinomial Logit Model
rmnpGibbs(bayesm)       Gibbs Sampler for Multinomial Probit
rmultireg(bayesm)       Draw from the Posterior of a Multivariate
                        Regression
rmvpGibbs(bayesm)       Gibbs Sampler for Multivariate Probit
rnmixGibbs(bayesm)      Gibbs Sampler for Normal Mixtures
rscaleUsage(bayesm)     MCMC Algorithm for Multivariate Ordinal Data
                        with Scale Usage Heterogeneity.
runireg(bayesm)         Draw from Posterior for Univariate Regression
runiregGibbs(bayesm)    Gibbs Sampler for Univariate Regression
```

Note that this list may contain functions from other packages. To restrict search to only *bayesm*, use the command, `help.search("topic",package="bayesm")`

The "manual" for *bayesm* is available in the help menu in R. Use the "help" item from the console window and click on "html help," then "packages," and then "bayesm." This will give you a linked html document with all of the defined functions.

A very nice looking manual for *bayesm* can be found from this html menu under the top link "browse directory" in the html page for *bayesm*. This is a pdf file created from a LaTeX version of the documentation. This same file can be found by going to the CRAN web site and clicking on the "packages" menu and then clicking on *bayesm*. The file `bayesm.pdf` is the manual for the most recent version of the package. Google "CRAN" to find a link to the CRAN web site.

4. Tips on Using MCMC Methods

All of the important functions in *bayesm* are implementations of various MCMC methods. The following tips are useful:

1. If you are unfamiliar with MCMC methods, read chapter 3. Try some of our test examples first, before trying your own data.
2. The "output" of an MCMC method is a set of draws of the parameters. You must decide how many draws to make and also how to analyze the draws produced. Unlike most classical methods, the MCMC methods provide an estimate of the entire posterior distribution, not just a few moments. Summarize the distribution by using histograms or quantiles. Resist the temptation to simply report the posterior mean and posterior standard deviation! For non-normal distributions, these moments have little meaning!
3. Most of the MCMC methods implemented in *bayesm* run very fast so it is possible to make 10,000s of draws even for relatively large datasets in less than ½ hour. Use this power where possible. Only the hierarchical models, `rhierLinearModel`, `rhierBinLogit`, `rmnlRwMixture`, and `rscaleUsage` will take appreciably longer (> one half hour, in some cases several hours will be required).
4. If you are having problems with using too much memory, set `keep` in the `Mcmc` parameter list to more than 1.

5. Extending and Adapting Our Code

We hope that many researchers will find our functions useful. Some will want to adapt and extend our code. To find the R and C source code, go to the CRAN web site, click on the "packages" menu link. Zipped files with source for the package are available. When this file is "unzipped" it will create a directory structure. Look at the directories "R" for source for the functions and "src" for the C/C++ code.

6. Updating *bayesm*

From time to time, we will release new versions of ***bayesm***. To obtain the latest release, use the `update.packages()` function or use the menu item, "update packages," from the R console window "packages" toolbar item. If you would like to be informed about releases of R packages, consider subscribing to the R-help or R-packages email lists. See the mailing lists link on the R homepage.