How to use RSGHB

Jeffrey Dumont

6 April 2013

Abstract

This vignette describes the process for specifying, estimating and analyzing the output of a choice model with RSGHB. The document is structured into two case studies using different model structures. The synthetic choice data used in this document (as well as number of other examples using different model structures) can be downloaded from RSGHB github page - `https://github.com/jeffdumont/RSGHB`

RSGHB modeling file structure

The typical RSGHB model file has the following 4 main sections. We will walk through each of these 4 sections in the examples that follow.

1. Setup and data preparation
2. Setting the controls for model estimation
3. Defining the likelihood function
4. Calling the doHB function to start the estimation process

EXAMPLE 1: MNL Model with Fixed Parameters

In this section, the code for estimating a multinomial logit model with fixed (non-random) parameters is explained. In this example, synthetic respondents were presented with a choice between two travel alternatives - one that is toll free but slower and one that was priced but faster. Each respondent was presented with a panel of 8 choice tasks.

Setup and Data Preparation

```
library(RSGHB)

# Sets the working directory. RSGHB generates a number of output files
# that will be written to this location.
setwd("C:/Work/Code/HB/RSGHB.git/RSGHB/vignettes/Example1")

# ------------------ DATA PREPARATION ------------------

# RSGHB assumes that respondents are identified with a ID column. The
# choicedata data.frame needs to be sorted by respondent and then
# experiment.
choicedata <- read.table("data_simulated.dat", sep = "\t", header = T)

# We typically work with one one row per choice observations. This isn't
# necessary however but it does lend it self to faster computation of the
# choice likelihoods
head(choicedata)
```

```
##     ID thecount tt1 tt2 toll2 asc1 Choice
## 1 8738        1  60  51  1.25    1      1
## 2 8738        2  60  51  0.75    1      1
## 3 8738        3  63  59  0.50    1      2
## 4 8738        4  60  54  0.75    1      1
## 5 8738        5  60  54  0.50    1      2
## 6 8738        6  63  54  0.75    1      1


# We can then specify any variables from the choicedata data.frame that
# you'd like to use in the utility equations in the likelihood function
# below. These can be any variables within the data or transformations of
# those variables. This example comes from transport so each alternative
# is defined by travel times and toll costs.
TT1 <- choicedata$tt1
TT2 <- choicedata$tt2
TOLL2 <- choicedata$toll2

# Here we specify the choice vectors. Note in this example there are only
# two alternatives. Also, dummying coding the choice vector is not
# necessary but allows for easier coding of the likelihood.
choice1 <- (choicedata$Choice == 1)
choice2 <- (choicedata$Choice == 2)

# Frequency of choice for the first alternative
table(choice1)

## choice1
## FALSE  TRUE
##  3560  6682


# Frequency of choice for the second alternative.
table(choice2)

## choice2
## FALSE  TRUE
##  6682  3560
```

Controling the Estimation Process

There are number of options for controling the estimation process. Please see the help file for doHB or the final section of this document for more details. Note that a number of controls have default values and do not need to be directly specified if the default is acceptable.

```
# ------------------ ESTIMATION CONTROL ------------------

# Setting control list for estimation (see ?doHB for more estimation
# options)

# modelname is used for naming the output files
modelname <- "MNL"

# gVarNamesFixed contains the names for the fixed (non-random) variables
# in your model. This will be used in output and also when displaying
# iteration detail to the screen.
gVarNamesFixed <- c("ASC1", "BTime", "BCost")

# FC contains the starting values for the fixed coefficients.
FC <- c(0, 0, 0)
```

```
# ITERATION SETTINGS

# gNCREP contains the number of iterations to use prior to convergence
gNCREP <- 30000
# gNEREP contains the number of iterations to keep for averaging after
# convergence has been reached
gNEREP <- 20000
# gNSKIP contains the number of iterations to do in between retaining
# draws for averaging
gNSKIP <- 1
# gINFOSKIP controls how frequently to print info about the iteration
# process
gINFOSKIP <- 250

# To simplify the doHB functional call, we put all of the control
# parameters into a single list that can be passed directly to doHB.
control <- list(modelname = modelname, gVarNamesFixed = gVarNamesFixed, FC = FC,
    gNCREP = gNCREP, gNEREP = gNEREP, gNSKIP = gNSKIP, gINFOSKIP = gINFOSKIP)
```

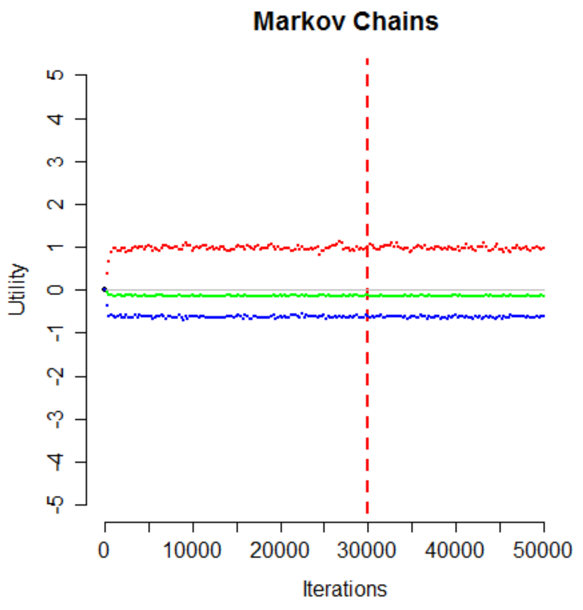
Writing the likelihood function

RSGHB is expecting the user-specificed likelihood function to take the parameters *fc* and *b* (even if they are not used within the function calculate the likelihood). The *fc* parameter is a vector of fixed coefficients (they do not vary across individuals in your data). The *b* parameter is a matrix of individual coefficients which are generated from the random coefficients in the model. In this example, we only focus on the *fc* vector.

It is important to note that the computation of the likelihood is the most computational taxing part of the estimation process. So coding the likelihood efficiently is essential to reduce run time of the model.

```
# ------------------ likelihood ------------------

likelihood <- function(fc, b) {

    # defining the parameters

    # using cc var to index the fc vector simplifies the addition/subtraction
    # of new parameters
    cc <- 1
    ASC1 <- fc[cc]
    cc <- cc + 1
    Btime <- fc[cc]
    cc <- cc + 1
    Btoll <- fc[cc]
    cc <- cc + 1

    # utility functions
    v1 <- ASC1 + Btime * TT1
    v2 <- Btime * TT2 + Btoll * TOLL2

    # mnl probability statement
    p <- (exp(v1) * choice1 + exp(v2) * choice2)/(exp(v1) + exp(v2))

    return(p)
}
```

Figure 1: Plotting of the Markov Chains during estimation



Estimating the model

To start the model estimation process, the analyst needs to call the *doHB* function passing in the *likelihood* function, the *choicedata* data.frame and the *control* list.

```
doHB(likelihood, choicedata, control)
```

RSGHB will first perform a series of diagnostics on your model to catch common errors in model setup. In addition, it will provide you with some basice summary statistics on your choice data and model. Before estimation begins, RSGHB will present you with a confirmation prompt allowing you to cancel the model estimation.

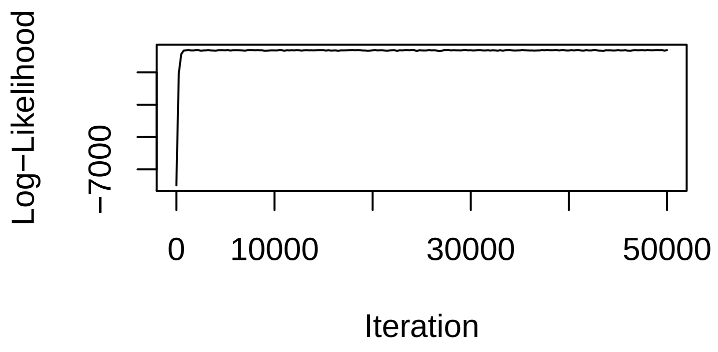
During the estimation, current estimates of the markov chains will be plotted to the screen. This plot is updated based on the control parameter *gINFOSKIP* (see figure 1). In addition, it will provide numerical iteration details in the R Console.

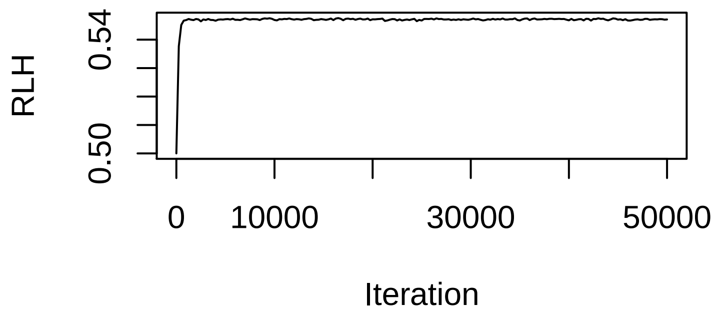
Evaluating the output

There are two main output files for this particular model - *log* file and the *_F* file. RSGHB comes with some basic tools for plotting the contents of these files.

The *.log* file contains some statistics that can be used to understand if model convergence has been reached. Because this model contains only fixed coefficents, the log file contains just the root likelihood (RLH) and log-likelihood at each iteration defined by *gINFOSKIP*.

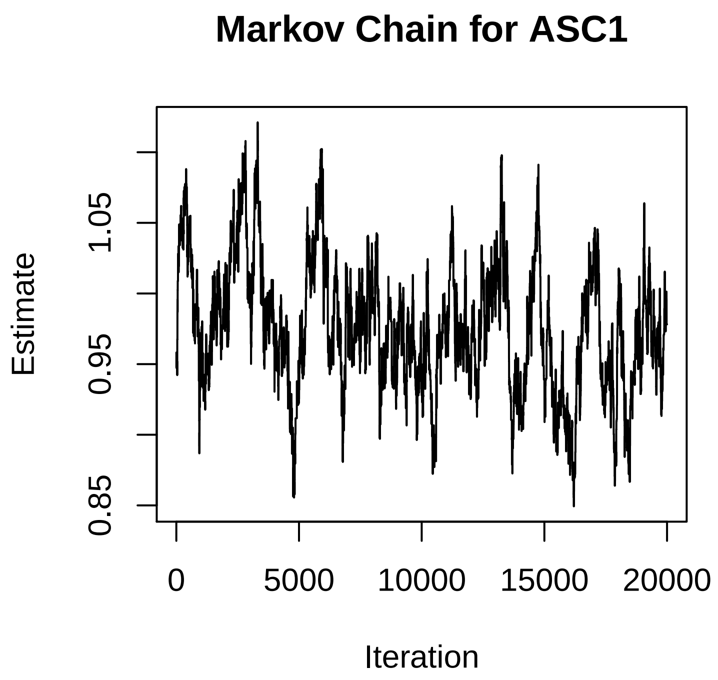
```
setwd("C:/Work/Code/HB/RSGHB.git/RSGHB/vignettes/Example1")

# plots the contents of the log file
plotLog(modelname)
```





The _F file contains the set of fixed (non-random) coefficients for each iteration after convergence of the markov chain.

```
setwd("C:/Work/Code/HB/RSGHB.git/RSGHB/vignettes/Example1")

Ffile <- read.table(paste(modelname, "_F.csv", sep = ""), header = T, sep = ",")

head(Ffile)

##   iteration   ASC1   BTime   BCost
## 1         1 0.9476 -0.1295 -0.6205
## 2         2 0.9582 -0.1289 -0.6147
## 3         3 0.9582 -0.1289 -0.6147
## 4         4 0.9582 -0.1289 -0.6147
## 5         5 0.9582 -0.1289 -0.6147
## 6         6 0.9573 -0.1348 -0.6224


plot(Ffile[, 2], main = "Markov Chain for ASC1", ylab = "Estimate", xlab = "Iteration",
    type = "l")
```



EXAMPLE 2: MNL Model with Random Coefficients

In this section, we expand on the model estimated in EXAMPLE 1 by allowing the coefficients to vary across the individuals in our dataset. This type of model is referred to by many names - Random Coefficients Logit, Random Parameters Logit or Mixed Logit.

Setup and Data Preparation

The setup and data preparation are very similar to the first model.

```
library(RSGHB)

# Sets the working directory. RSGHB generates a number of output files
# that will be written to this location.
setwd("C:/Work/Code/HB/RSGHB.git/RSGHB/vignettes/Example2")

# ------------------ DATA PREPARATION ------------------

# RSGHB assumes that respondents are identified with a ID column. The
# choicedata data.frame needs to be sorted by respondent and then
# experiment.
choicedata <- read.table("data_simulated.dat", sep = "\t", header = T)

# We typically work with one one row per choice observations. This isn't
# necessary however but it does lend it self to faster computation of the
# choice likelihoods
head(choicedata)
```

```
##     ID thecount tt1 tt2 toll2 Choice
## 1 8738        1  60  51  1.25      1
## 2 8738        2  60  51  0.75      2
## 3 8738        3  63  59  0.50      1
## 4 8738        4  60  54  0.75      1
## 5 8738        5  60  54  0.50      1
## 6 8738        6  63  54  0.75      1


# We can then specify any variables from the choicedata data.frame that
# you'd like to use in the utility equations in the likelihood function
# below. These can be any variables within the data or transformations of
# those variables. This example comes from transport so each alternative
# is defined by travel times and toll costs.
TT1 <- choicedata$tt1
TT2 <- choicedata$tt2
TOLL2 <- choicedata$toll2

# Here we specify the choice vectors. Note in this example there are only
# two examples. Also, dummying coding the choice vector is not necessary
# but allows for easier coding of the likelihood.
choice1 <- (choicedata$Choice == 1)
choice2 <- (choicedata$Choice == 2)

# Frequency of choice for the first alternative
table(choice1)
```

```
## choice1
## FALSE  TRUE
##  4223  6019
```

```
# Frequency of choice for the second alternative.
table(choice2)
```

```
## choice2
## FALSE  TRUE
##  6019  4223
```

Controling the Estimation Process

To allow for mixing of the parameters, we need to specify a few more controls to pass into the *doHB* function.

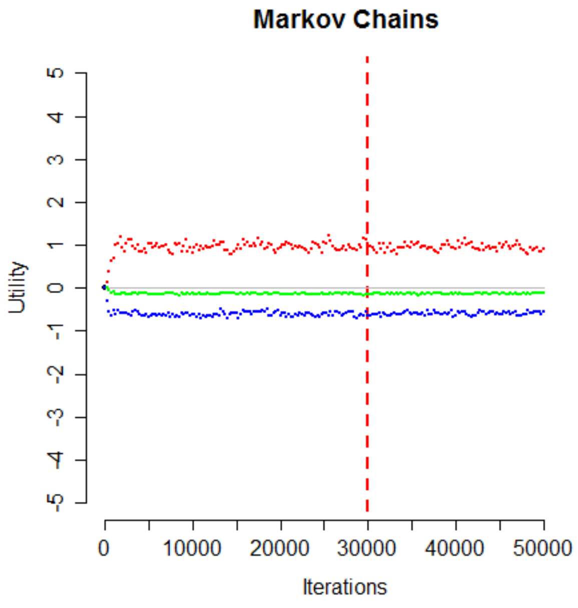
```
# ------------------ ESTIMATION CONTROL ------------------

# Setting control list for estimation (see ?doHB for more estimation
# options)

# modelname is used for naming the output files
modelname <- "MNL"

# gVarNamesNormal provides names for the random parameters
gVarNamesNormal <- c("ASC1", "BTime", "BCost")

# gDIST specifies the type of continuous distribution to use for the
# random parameters. gDIST must have an entry for each value in
# gVarNamesNormal The options are: 1. normal 2. log-nomal 3. negative
# log-normal 4. normal with all values below zero massed at zero 5.
# Johnson SB with a specified min and max
# In this example, we use normal distributions for all 3 of the
# parameters.
gDIST <- c(1, 1, 1)

# svN contains the starting values for the means of the normal
# distributions for each of the random parameters
svN <- c(0, 0, 0)

# ITERATION SETTINGS

# gNCREP contains the number of iterations to use prior to convergence
gNCREP <- 30000
# gNEREP contains the number of iterations to keep for averaging after
# convergence has been reached
gNEREP <- 20000
# gNSKIP contains the number of iterations to do in between retaining
# draws for averaging
gNSKIP <- 1
# gINFOSKIP controls how frequently to print info about the iteration
# process
gINFOSKIP <- 250

# To simplify the doHB functional call, we put all of the control
# parameters into a single list that can be passed directly to doHB.
control <- list(modelname = modelname, gVarNamesNormal = gVarNamesNormal, gDIST = gDIST,
    svN = svN, gNCREP = gNCREP, gNEREP = gNEREP, gNSKIP = gNSKIP, gINFOSKIP = gINFOSKIP)
```

Writing the likelihood function

To introduce mixing into the model, we switch from using *fc* vector to using the *b* matrix in the coding of the likelihood. The *b* matrix contains the individual conditionals for the sample-level random coefficients. The matrix *b* has one row per observation (an individual's coefficients are repeated across their choice observations automatically by RSGHB) and one column for each of the random parameters.

```
likelihood <- function(fc, b) {

    # the change from using fc to b is the only change in the likelihood
    # function required to allow for mixing.
    cc <- 1
    ASC1 <- b[, cc]
    cc <- cc + 1
    Btime <- b[, cc]
    cc <- cc + 1
    Btoll <- b[, cc]
    cc <- cc + 1

    v1 <- ASC1 + Btime * TT1
    v2 <- Btime * TT2 + Btoll * TOLL2

    p <- (exp(v1) * choice1 + exp(v2) * choice2)/(exp(v1) + exp(v2))

    return(p)
}
```

Figure 2: Plotting of the Markov Chains during estimation



Section 4: Estimating the model

Again, to start the model estimation process, the analyst needs to call the *doHB* function passing in the *likelihood* function, the *choicedata* data.frame and the *control* list.

```
doHB(likelihood, choicedata, control)
```

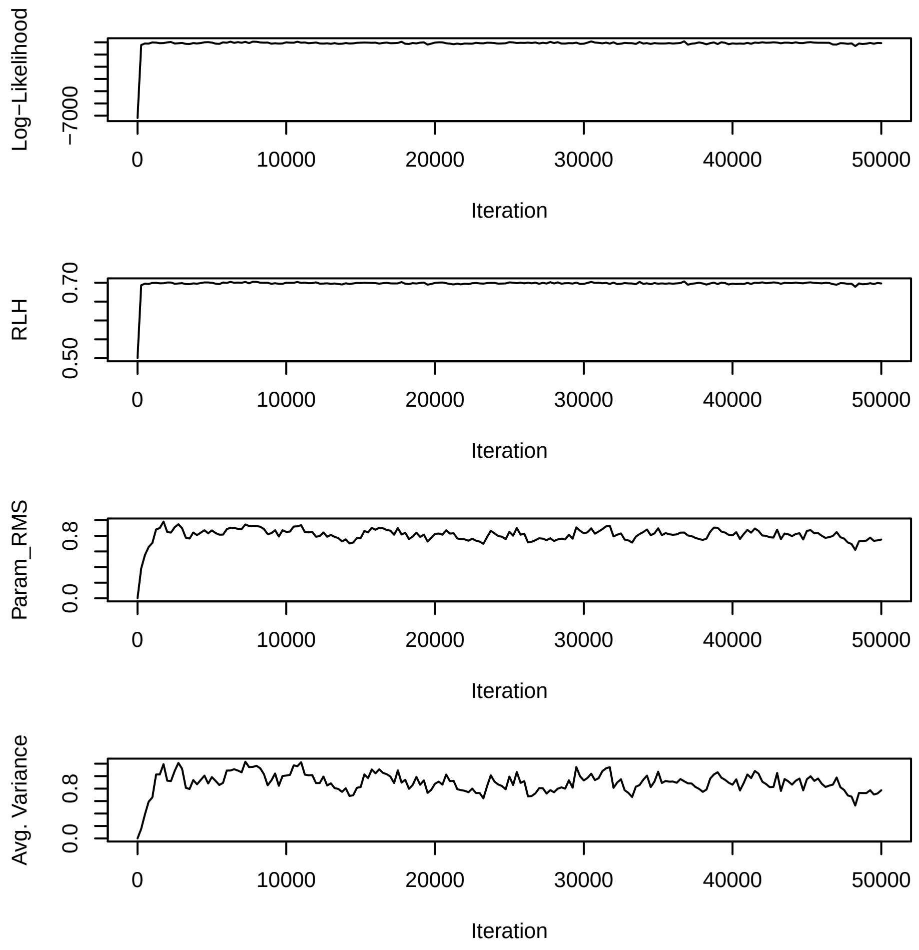
Evaluating the output

As in the first example, current estimates of the markov chains will be plotted to the screen - s ee figure 2. In this model, these represent the means of the underlying normals for the random parameters.

There are more output files for this model. RSGHB comes with some basic tools for plotting the contents of these files.

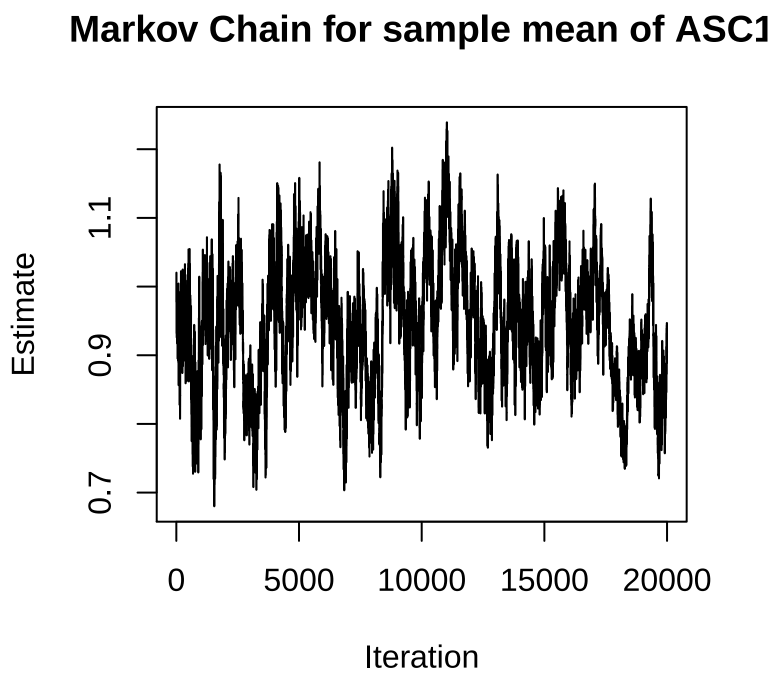
The *.log* file contains some statistics that can be used to understand if model convergence has been reached. Because this model includes random coefficients, the log file now contains the average variance and parameter root mean square (RMS) at each iteration.

```
setwd("C:/Work/Code/HB/RSGHB.git/RSGHB/vignettes/Example2")

# plots the contents of the log file
plotLog(modelname)
```



The _*A* file contain the sample-level means of the underlying normal at each iteration.

```
setwd("C:/Work/Code/HB/RSGHB.git/RSGHB/vignettes/Example2")

Afile <- read.table(paste(modelname, "_A.csv", sep = ""), header = T, sep = ",")

head(Afile)

##   iteration   ASC1   BTime   BCost
## 1         1 0.9775 -0.1483 -0.6599
## 2         2 0.9866 -0.1401 -0.6580
## 3         3 1.0055 -0.1513 -0.6767
## 4         4 1.0203 -0.1340 -0.6463
## 5         5 0.9973 -0.1358 -0.6371
## 6         6 0.9784 -0.1320 -0.6372


plot(Afile[, 2], main = "Markov Chain for sample mean of ASC1", ylab = "Estimate",
    xlab = "Iteration", type = "l")
```



The *_B* file contains the average across iterations of the individual level draws for the underlying normals for the random parameters. The *_Bsd* file provides the standard deviations of those individual draws.

```
setwd("C:/Work/Code/HB/RSGHB.git/RSGHB/vignettes/Example2")

Bfile <- read.table(paste(modelname, "_B.csv", sep = ""), header = T, sep = ",")

head(Bfile)

##   Respondent   ASC1    BTime   BCost
## 1       8738 1.1536 -0.01117 -0.6535
## 2       8740 1.1617 -0.01307 -0.5311
## 3       8741 0.7044 -0.56812 -0.5114
## 4       8742 1.1573 -0.27454 -0.6487
## 5       8744 0.4327 -0.01753 -0.5461
## 6       8745 0.9851 -0.29169 -0.5596


BSDfile <- read.table(paste(modelname, "_Bsd.csv", sep = ""), header = T, sep = ",")

head(BSDfile)

##   Respondent   ASC1  BTime  BCost
## 1       8738 0.6587 0.1601 0.3968
## 2       8740 0.6595 0.1644 0.3825
## 3       8741 0.6689 0.2004 0.3839
## 4       8742 0.6696 0.1563 0.3985
## 5       8744 0.6565 0.1587 0.4008
## 6       8745 0.6284 0.1607 0.3728
```

The *_C* file contains the average across iterations of the individual level draws for the random parameters including the appropriate transformations. The *_Csd* file provides the standard deviations of those individual draws. These two files are equivalent to the conditional distributions from models estimated using Maximum Simulated Likelihood methods.

```
setwd("C:/Work/Code/HB/RSGHB.git/RSGHB/vignettes/Example2")

Cfile <- read.table(paste(modelname, "_C.csv", sep = ""), header = T, sep = ",")

head(Cfile)

##   Respondent    RLH   ASC1    BTime   BCost
## 1       8738 0.6726 1.1536 -0.01117 -0.6535
## 2       8740 0.6303 1.1617 -0.01307 -0.5311
## 3       8741 0.8423 0.7044 -0.56812 -0.5114
## 4       8742 0.5145 1.1573 -0.27454 -0.6487
## 5       8744 0.4823 0.4327 -0.01753 -0.5461
## 6       8745 0.4940 0.9851 -0.29169 -0.5596


CSDfile <- read.table(paste(modelname, "_Csd.csv", sep = ""), header = T, sep = ",")

head(CSDfile)

##   Respondent   ASC1  BTime  BCost
## 1       8738 0.6587 0.1601 0.3968
## 2       8740 0.6595 0.1644 0.3825
## 3       8741 0.6689 0.2004 0.3839
## 4       8742 0.6696 0.1563 0.3985
## 5       8744 0.6565 0.1587 0.4008
## 6       8745 0.6284 0.1607 0.3728
```

The _D file contains a row-based representation of the sample covariance for each iteration. Note the use below of the xpnd function to convert to a matrix representation of the sample covariance.

```
setwd("C:/Work/Code/HB/RSGHB.git/RSGHB/vignettes/Example2")

Dfile <- read.table(paste(modelname, "_D.csv", sep = ""), header = T, sep = ",")

head(Dfile)

##   iteration      X      X.1     X.2       X.3     X.4    X.5
## 1         1 0.4060 -0.01448 0.06579 0.0043159 0.01233 0.1091
## 2         2 0.3520 -0.01918 0.06392 0.0025858 0.01657 0.1119
## 3         3 0.3981 -0.02825 0.06619 0.0052280 0.01506 0.1159
## 4         4 0.4019 -0.02315 0.06263 0.0119998 0.01127 0.1031
## 5         5 0.4149 -0.02466 0.06762 0.0005846 0.01413 0.1070
## 6         6 0.4390 -0.02598 0.06316 0.0033442 0.01099 0.1015


# building the covariance matrix

covMat <- xpnd(colMeans(Dfile[-1]))

rownames(covMat) <- c("ASC1", "BTime", "BCost")
colnames(covMat) <- c("ASC1", "BTime", "BCost")

covMat

##           ASC1    BTime    BCost
## ASC1   0.46179 -0.06016 -0.03624
## BTime -0.06016  0.07694  0.02735
## BCost -0.03624  0.02735  0.15864
```

RSGHB Control Parameters

Here is a list of the user-specified control parameters.

gVarNamesNormal - A vector of charater-based names for the random parameters.
Default: NULL

gVarNamesFixed - A vector of character-based names for the fixed parameters.
Default: NULL

gDIST - A vector of integers (1-5) which indicate which type of distribution should be applied to the random coefficients - 1 = Normal, 2 = Postive Log-Normal, 3 = Negative Log-Normal, 4 = Censored Normal, 5 = Johnson SB. There should be an element for each name in gVarNamesNormal.
Default: NULL

FC - A vector of starting values for the fixed coefficients. There should be an element for each name in gVarNamesFixed.
Default: NULL

svN - A vector of starting values for the means of the underlying normals for the random parameters. There should be an element for each name in gVarNamesNormal.
Default: NULL

gNCREP - Number of burn-in iterations to use prior to convergence.
Default: 100000

gNEREP - Number of iterations to keep for averaging after convergence has been reached.
Default: 100000

gNSKIP - Number of iterations in between retaining draws for averaging.
Default: 1

gINFOSKIP - Number of iterations in between printing/saving information about the iteration process.
Default: 250

modelname - The model name which is used for creating output files.
Default: paste(”HBModel”,round(runif(1)*10000000,0),sep=””))

gSIGDIG - The number of significant digits for reporting purposes.
Default: 10

priorVariance - The amount of prior variance assumed.
Default: 2.0

degreesOfFreedom - Additional degrees of freedom for the prior covariance matrix (not including the number of parameters.
Default: 5

rho - The initial proportionality fraction for the jumping distribution for the Metropolis-Hastings algorithm for the random parameters. This fraction is adjusted by the program after each iteration to attain an acceptance rate of about 0.3.
Default: 0.1

rhoF - The proportionality fraction for the jumping distribution for the Metropolis-Hastings algorithm for the fixed parameters. Unlike rho, this value is not adjusted as the markov chain proceeds.
Default: 0.0001

gFULLCV - A number that indicates if a full or independent covariance structure should be used for the random parameters. A value of 1 indicated full and 0 for an independent structure.
Default: 1

gMINCOEF - A vector of minimums for the Johnson SB distributions. If Johnson SB is used, each random parameter needs an element but only the elements that correspond to a 5 in gDIST are used.
Default: 0

gMAXCOEF - Like gMINCOEF but for the maximum of the Johnson SB distribution.
Default: 0

gStoreDraws - A boolean value to store the draws for the individual level coefficients.
Default: F

gSeed - The random seed.
Default: 0

constraintsNorm - This is a list of monotonic constraints to be applied during estimation. The structure of the constraints is c(param1number - inequality - param2number). For constraints relative to 0, use 0 instead of the param2number. For the inequality, use 1 for ¡ and 2 for ¿. Example

```
constraintsNorm <- list(c(5,1,0),c(6,1,5),c(7,1,6),c(8,1,7))
```

would constrain the 5th parameter ¡ 0, the 6th parameter ¡ 5th parameter, the 7th parameter ¡ the 6th parameter, etc.
Default: NULL

nodiagnostics - If set to TRUE, the diagnostic report will not be reported to the screen with a prompt to continue. This makes batch processing easier to implement.
Default: FALSE