HSROC : An R package for Bayesian meta-analysis of diagnostic test accuracy

Ian Schiller[3], Nandini Dendukuri[1,2,3],

1: Department of Epidemiology and Biostatistics, McGill University, Montreal

2: Technology Assessment Unit, McGill University Health Center, Montreal

3: Division of Clinical Epidemiology, McGill University Health Center, Montreal

July 23, 2012

1   Introduction

This document is intended to serve as a guide for the usage of the HSROC package to be used within the free statistical software environment, R [6]. Therefore, the main goal is to describe the functions in the package via different examples without emphasing the statistical theory behind them. The likelihood of the HSROC model and prior distributions appear in the appendix. The interested reader can learn more about the statistical theory in [1].

The HSROC package consists of 4 functions and 2 data sets. The two main functions are *HSROC* and *HSROCSummary*. *HSROC*, which must be run first, is used to implement a Gibbs sampler while *HSROCSummary* produces summaries for the HSROC model parameters. The remaining 2 functions are secondary functions : *simdata* simulates a dataset based on the HSROC diagnostic meta-analysis model, while *beta.parameter* returns the shape parameters of the $Beta(\alpha, \beta)$ probability corresponding to a given range.

In the following sections, we will present 3 examples to explain how to make use of the functions within the HSROC package. In section 2 we present a simple example where a test under evaluation is compared to a perfect reference test when both tests are independent given the true disease status. In section 3, we present an example where the test under evaluation is now compared to an imperfect reference test, while both tests remain conditionally independent from each other. In section 4 we show how to simulate data from a HSROC diagnostic meta-analysis model

2 Example 1 : Meta-Analysis in the presence of a gold standard reference test assuming conditional independence

We start with the simplest mode, where the reference test is assumed to be a gold standard (i.e. sensitivity and specificity of the reference test both equal to 100%).

For this example, we use data on magnetic resonance (MR) imaging from 10 studies reviewed in a study by Scheidler et al [5]. This is a subset of the illustration dataset used in the paper describing the HSROC model of Rutter and Gatsonis [4].

2.1 Data preparation

After having installed the package, the library can be loaded with the following command :

```
> library(HSROC)
```

The data on MR imaging is included in the library and can be loaded as follows :

```
> data(MRI)
> MRI
   ++ +- -+ --
1   9  2  2  44
2   3  6  5  32
3   3  2  1  16
4   3  1 12  44
5   1  1  6  16
6   7  2 22 167
```

```
 7  12  4  4  29
 8  23  5 14 230
 9   8  5  5  53
10  16  2  2  22
```

The columns $++, +-, -+, --$ represent the results of the cross tabulation between MRI (the test under evaluation) and histologic/cytologic specimens obtained by surgery or lymph node biopsy (reference test). The colummn headings $++, +-, -+, --$ correspond to (MRI $+$, reference $+$), (MRI $+$, reference -), (MRI -, reference $+$) and (MRI -, reference -), respectively.

In order to estimate the parameters of the conditional independence model, we use the function *HSROC*. The arguments of the function are as follows :

```
> args(HSROC)
function (data, iter.num, init = NULL, sub_rs = NULL, first.run = TRUE,
    path = getwd(), refresh = 100, prior.SEref = NULL, prior.SPref = NULL,
    prior_PI = c(0, 1), prior_LAMBDA = c(-3, 3), prior_THETA = c(-1.5,
        1.5), prior_sd_alpha = list(0, 2, "sd"), prior_sd_theta = list(0,
        2, "sd"), prior_beta = c(-0.75, 0.75))
```

This function results in drawing a sample from the posterior distribution of the model via a single chain gibbs sampler.

A number of arguments, such as those determining the prior distributions, have default values. In particular, the default values *prior.SEref = NULL* and *prior.SPref = NULL* define a gold standard reference test. The Gibbs sampler requires initial values which we will provide in this example. Therefore, the only argument for which we will alter the default in this example, is the *init* argument. When this argument is left equal to its default value, the initial values required by the Gibbs sampler are randomly selected by the *HSROC* function itself based on the prior distributions.

The *init* argument is a list object composed of the initial values of the within-study parameters as the first element of the list and the initial values of the between-study parameters as the second element of the list. The within-study element must be a matrix-like object with each column corresponding to a different parameter and each row corresponding to a different study, while the between-study element must be a vector with each element corresponding to a different parameter. Suppose that the following inital values are desired for each of the within-study parameters α_i (diagnostic accuracy of study i), θ_i (the cut-off value for defining a positive test in study i), S_{1i} (sensitivity of study i for the test under evaluation), C_{1i} (specificity of study i for the test under evaluation), and π_i (the prevalence of study i), for the MR meta-analysis

```
> init.alpha = c(2.51, 2.54, 3.81, 2.41, 2.64, 2.70, 3.31, 3.39, 3.11, 2.99)
> init.theta = c(-0.51, -0.39, 0.33, -2.06, -0.14, -0.08, 1.11, 0.38, -0.86,
+-0.38)
> init.s1 = rep(0.9,10)
> init.c1 = rep(0.9,10)
> init.pi = c(0.38, 0.17, 0.78, 0.07, 0.74, 0.84, 0.52, 0.95, 0.07, 0.56)
```

We first create a matrix of within-study initial values

```
> init_within = cbind(init.alpha, init.theta, init.s1, init.c1, init.pi)
```

The ordering of the initial values in the *cbind* function above is important and must not be altered. Placing *init.theta* before *init.alpha*, that is

```
> init_within = cbind(init.theta, init.alpha, init.s1, init.c1, init.pi)
```

will result in initialising the α_i parameters with the initial values of θ_i and vice-versa. This could possibly lead to slower convergence of the Gibbs sampler in case the starting values are not suitable.

Now, for the between-study parameters Θ (the overall mean cut-off value for defining a positive test), σ_θ (the between-study standard deviation in the cut-off), Λ (the overall diagnostic accuracy), σ_α (the between-study standard deviation of the difference in means), and β (the logarithm of the ratio of the standard deviation of test results among patients with the disease and among patients without the disease), let's suppose the following starting values are to be used :

```
> init.THETA = -0.16
> init.sig.theta = 0.75
> init.LAMBDA = 2.58
> init.sig.alpha = 0.5
> init.beta = 0.25
```

We then create the vector of between-study initial values as follows :

```
> init_between = c(init.THETA, init.sig.theta, init.LAMBDA, init.sig.alpha, init.beta)
```

For the same reason discussed above, the ordering in the vector above must be kept as is.

Finally, we simply need to merge the within-study and between-study initial values created above into a list, as follow

```
> init = list(init_within, init_between)
```

2.2 Running the Gibbs sampler

To complete the function call we need to provide two additional arguments *data* and *iter.num* for the data set and number of gibbs sampler iterations. In this example, we will run the Gibbs sampler for 50,000 iterations. We thus make the following call

```
> HSROC(data=MRI, iter.num=50000, init=init  )
```

Rather than manage very large matrices of posterior samples for each parameter within R, that grow in size as the number of iterations increases, the function creates text files in the default working directory, or in any specified working directory through the *path* argument. This should help computational speed as the number of iterations increase. Once the function has reached the selected number of iterations, the following message will appear

```
[1] The files created during the Gibbs sampler process are in "C:\... "
```

where C : \... is the working directory where the text files were created and saved.

2.3 Interpreting the output files

The *HSROCSummary* function can be used to obtain descriptive statistics and graphs using the posterior sample. The arguments of the function are as follows

```
> args(HSROCSummary)
function (data, burn_in = 0, iter.keep = NULL, Thin = 1, sub_rs = NULL,
    point_estimate = c("median", "mean"), summary.path = getwd(),
    chain = getwd(), tv = NULL, digit = 6, print_plot = FALSE,
    plot.ind.studies = TRUE, conf_region = TRUE, predict_region = TRUE,
```

```
    col.pooled.estimate = "red", col.predict.region = "blue",

    lty.conf.region = "dotdash", lty.predict.region = "dotted",

    region_level = 0.95, trunc_low = 0.025, trunc_up = 0.025)
```

For our example, we call the function as follows :

```
> HSROCSummary(data = MRI, burn_in=10000, Thin=2, print_plot=TRUE )
```

The descriptive statistics created by the function are discussed in section 2.3.1. The argument *data* simply needs to be set equal to the dataset. In the case of *burn_in*, it indicates how many burn-in iterations are to be dropped before calculation of the estimates. The argument *Thin* defines the thinning interval. Finally, $print\_plot = TRUE$ allows the creation of graphical tools to help the user assess if the convergence of the Gibbs sampler has been achieved. The different plots produced by the function will be discussed in section 2.3.2.

2.3.1 Descriptive statistics

The *HSROCSummary* function returns a summary of the results in the R GUI and more detailed results in a text file within the working directory. In the R GUI it lists the point estimates and 95% highest posterior density (HPD) credible intervals for the between-study and within-study parameters. The text file that is saved in the working directory is divided into three sections.

The first section (displayed in Figure 1) lists some of the general settings. The number of gibbs sampler chains used in this example was 1. A total of 50,000 iterations were used with a burn in of

10,000. Every second value was kept for estimation (the thinning interval). This left us with a sample of 20,000 values drawn from the posterior distribution of each parameter we are interested in. This section also lists the location of the file on the hard drive, the date of creation of the summary file and the nature of the reference test (perfect or imperfect). The data set is also listed.

The next section of the output gives a summary of the prior distributions used (Figure 2), which happen to be the default prior distributions in this example. The prior distribution used for the prevalence was a Beta(1,1), distribution, that is a beta distribution with scale and shape parameters equal to 1. This is equivalent to a uniform prior for the prevalence, allowing all possible

Figure 2

values to be equally likely. The log of the ratio between the two standard deviations, β was assumed to follow a $U(-0.75, 0.75)$ distribution. The overall mean cut-off Θ was assigned a uniform distribution ranging from -1.5 to 1.5. The pooled diagnostic accuracy Λ followed a uniform distribution over -3 to 3. Finally, parameters σ_α and σ_θ were both assumed to follow $U(0, 2)$ distribution.

The final section lists descriptive statistics for the parameters of the model. In addition to the point estimate and highest posterior density (HPD) intervals, it also includes Monte Carlo (MC) error and standard deviation of the posterior sample. The MC error is calculated via the batch mean method described in Ntzoufras [3]. For this reason, the user may note that no estimation will be provided if less than 100 iterations are left, once the burn in and thinning are taken into account. In this situation, the following message would be displayed

```
> Error in HSROCSummary(data = MRI, burn_in = 1, Thin = 1, i = 50, print_plot = T) :
   You don't have enough iterations to estimate the MC error.
  After taking into account the "burn in" and "thinning interval",
  you need at least 100 iterations to proceed.
```

The addition of the MC error and the standard error provide means for examining the precision of the estimation. One might want to run the Gibbs sampler until the MC error is sufficiently small. For example, a desirable criterion might be to have the MC error of each parameter smaller than 10% of its posterior standard deviation. The estimates of the between-study parameters and within-study parameters are shown in Figures 3 and 4 respectively.

For example, the cut-off parameter Θ was estimated to be 0.900203 with a 95% highest posterior density interval of (0.435669, 1.368114). The standard deviation and MC error were estimated to be 0.239167 and 0.011831, respectively. We see that the MC error is smaller than 10% of the posterior standard deviation for Θ. Other estimates included here are those of Λ, β, σ_θ and σ_α. The estimates of the pooled sensitivity and specificity, denoted S overall and C overall, are obtained as functions of Θ, Λ and β. A similar analysis of the MR imaging data was performed by Rutter and Gatsonis [4] using an HSROC model with a logit link function. They reported a pooled sensitivity estimate of 0.541 with a 95% equal-tail credible interval of (0.286, 0.771) and a pooled specificity estimate of 0.953 with a 95% equal-tail credible interval of (0.907, 0.981). Even though both models used a different link function, it is reassuring to notice that our estimates of S overall and C overall are satisfactorily close to the estimates of Rutter and Gatsonis. The difference noticed between their credible intervals and


```
Summary for N = 20000  - Notepad
File  Edit  Format  View  Help

          BETWEEN_STUDY parameters (Point estimate = median )


          Estimate Standard_Dev MC_error C.I._lower C.I._upper
THETA         0.900203  0.239167  0.011831  0.435669  1.368114
LAMBDA        2.053754  0.356305  0.019747  1.315465  2.697137
beta         -0.34803   0.295007  0.021779 -0.749267  0.259118
sigma.alpha   0.51137   0.295974  0.007239  0.000375  1.082589
sigma.theta   0.511308  0.191878  0.002683  0.228293  0.926484
S overall               0.556529  0.102008  0.001173  0.350376  0.75549
C overall               0.948203  0.024638  0.000397  0.897168  0.984585
```

Figure 3

ours comes from the fact that the *HSROCSummary* function returns 95% highest posterior density intervals.

Figure 4 displays a portion of the within-study parameter estimates. We see that estimates for individual studies are grouped by parameters.

For example, the estimates of θ_1 (the cut-off in study 1), θ_2, θ_3, ..., θ_{10} are shown in the first sub-section of the within-study parameters section. Just like the between-study estimates of Figure 3, the standard deviation, MC error and 95% HPD interval are given.

2.3.2 The graphical summary

The *HSROCSummary* function creates various plots to help the user judge if the descriptive statistics of section 2.3.1 are reliable. Among them, a trace plot for each parameter can be used to help evaluate whether the Gibbs sampler has converged. Each trace plot is

Figure 4

a scatter plot of the posterior sample of a single parameter vs the iteration number of the Gibbs sampler. The trace plots for the sensitivity of MRI in studies 1, 2, 3, 6, 7 and 8 are shown in Figure 5.
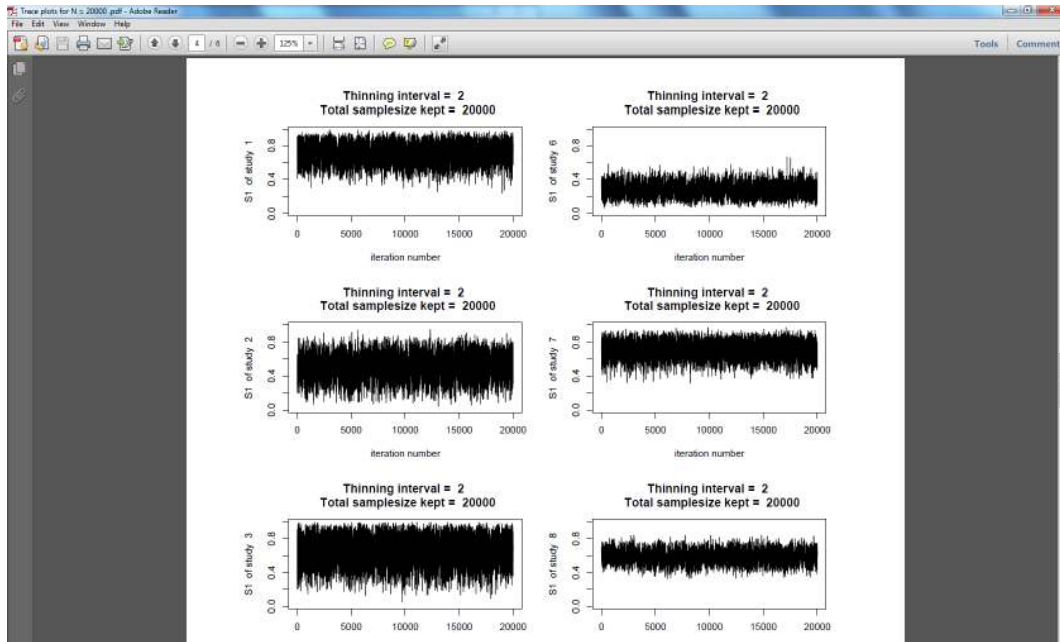
Figure 5

For our example, we can see that convergence seems to be achieved fairly quickly.

Another graphical summary produced by the *HSROCSummary* function is the density plot. It plots a smoothed posterior kernel density function for each parameter. Figure 6 shows density plots for some of the between-study parameters.

Finally, the function also produce a summary receiver operating characteristic (SROC) curve. Finally, the function also produce a summary receiver operating characteristic (SROC) curve. The SROC curve summarizes the relationship between sensitivity and (1 - specificity) across studies, taking into account the between-study heterogeneity. The SROC curve for the MRI data is shown in Figure 7. Individual studies are depicted by a clear circle. The radius of the circle is proportional to the sample size of the study. The red circle marks the pooled sensitivity and specificity across the 10 studies in this meta-analysis. The 95% prediction region is defined by the blue dotted-curve. The red dot-dashed-curve marks the boundary of the 95% credible region for the pooled estimates of sensitivity and specificity across the 10 studies.
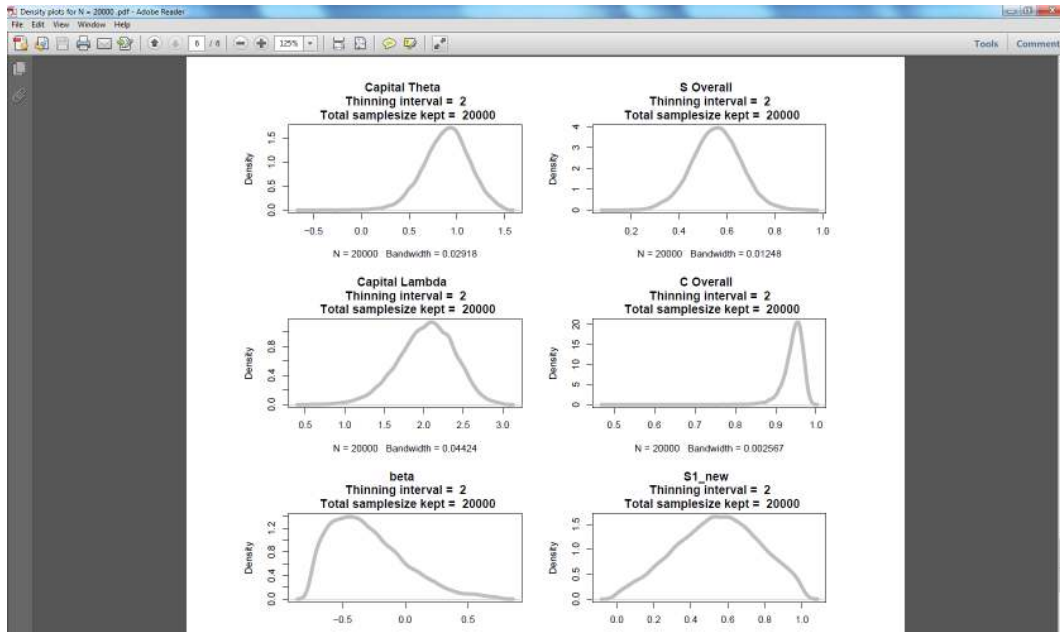
Figure 6

2.3.3 Multiple chains to assess convergence

Gelman and Rubin [2] recommend running the Gibbs sampler multiple times starting from different initial values in order to assess convergence.

In our example, we have run a single chain so far with initial values given in section 2.1. Let's say we would like to run two more chains. The idea is to repeat sections 2.1 and 2.2 with a different set of initial values and a different working directory as many times as desired. Here to run 2 more chains, we would repeat steps 2.1 and 2.2 two more times.

Instead of using the default working directory, let's suppose we had previously assigned a directory to the path argument in the *HSROC* function.

```
> dir.create("C:/MRI/Chain1")
> HSROC(data=MRI, iter.num=50000,  init=init,  path="C:/MRI/Chain1"  )
```

Now to run the second chain, we would make the following call
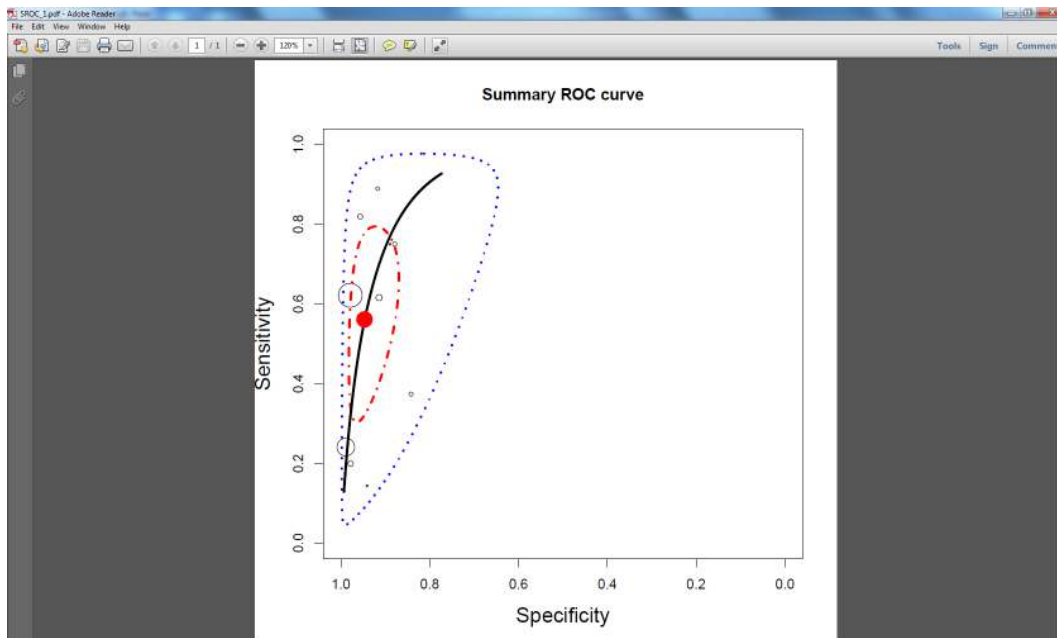
Figure 7

```
> dir.create("C:/MRI/Chain2")
> HSROC(data=MRI, iter.num=50000,  init=init2, path="C:/MRI/Chain2"   )
```

where *init2* is our second set of initial values. Finally, to run our third chain, we simply run

```
> dir.create("C:/MRI/Chain3")
> HSROC(data=MRI, iter.num=50000, init=init3, path="C:/MRI/Chain3"   )
```

For the sake of efficiency, one might want to run all 3 chains at the same time by running each chain in separate R windows. In other words, we could open 3 different R sessions and run the *HSROC* function with respective initial values and working directory, one in each R session, simultaneously.

Once all 3 chains have reached the desired number of iterations, a single call to the function *HSROCSummary* will summarize all 3 chains.

```
> HSROCSummary(data = MRI, burn_in=10000, Thin=2, print_plot=TRUE,
```

```
+     path="C:/MRI/All_Chains", chain=list("C:/MRI/Chain1","C:/MRI/Chain2",

+     "C:/MRI/Chain3") )
```

Through the *path* argument we define a new working directory to save all results and plots produced by the function. The *chain* argument points to the working directories where the posterior samples from each chain were previously saved. The structure of the output remains the same as the one described in section 2.3 except for the trace plot which overlays the posterior samples of all 3 chains simultaneously (see Figure 8).
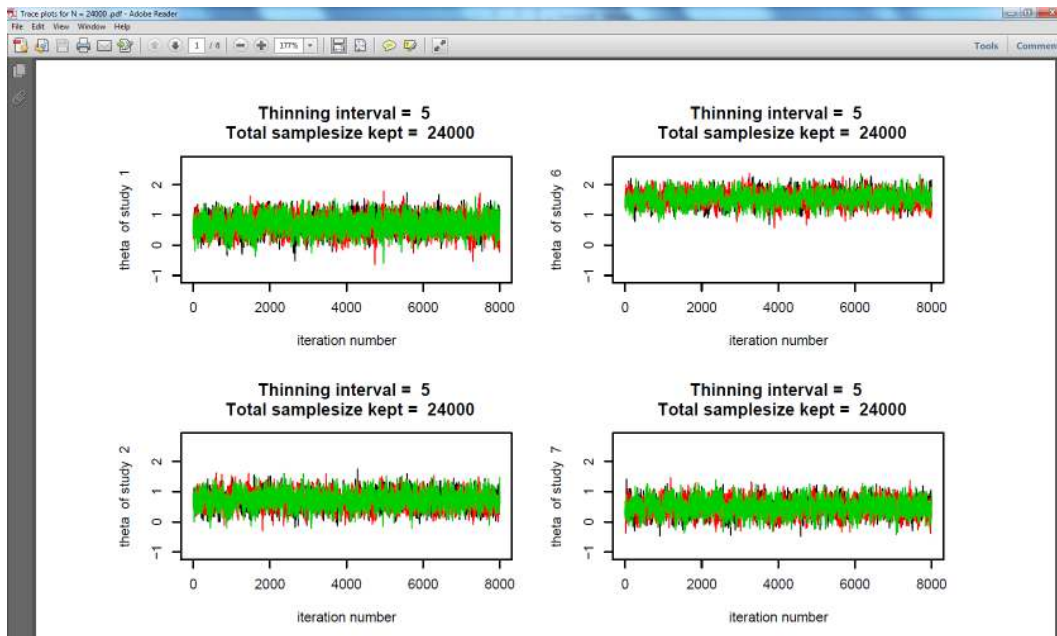

Figure 8

In our example, we can see that all 3 chains (each chain represented by a different color) converged quickly to a common region of the parameter space as is desirable.

3 Example 2 : Meta-Analysis in the presence of multiple imperfect reference tests assuming conditional independence

In this example we consider a situation where we no longer have a perfect reference standard. We will use data from a meta-analysis of TB pleuritis where 3 different imperfect reference standards were used. We assume that the test under evaluation is independent of the reference test in each study given the true disease status. The data set consists of 11 primary studies of in-house nucleic acid amplification tests of the IS6110 target, a commonly used rapid test for tuberculous pleuritis.

3.1 Data preparation

The data set is avalaible as part of the HSROC package.

```
> data(In.house)
> In.house
   ++ +- -+ --
1  11  1 14 75
2   1  1  3 25
3   8  0  1 16
4  16  6  0 43
5  16  0  1 56
6   9  0  6 10
7  13  0  4 25
8  17  2  4 84
9   7  0  3 13
10 14  1 19 97
```

```
11 31  7 11 63
```

In order to take into account the multiple imperfect reference standards, we must make changes to the default values of three arguments of the *HSROC* function seen in section 2.1. First, the argument *sub_rs* must be set to reflect the desired number of reference standards, second, the *init* argument must now include initial values of sensitivity and specificity of the reference standards and third, prior distributions must be provided for the sensitivity and specificity of the reference standards via the arguments *prior.SEref* and *prior.SPref*.

In our example, there were 3 reference standards. The first reference standard was used in studies 1 and 2, the second reference standard in studies 3 and 4 and finally, studies 5 to 11 used the third reference standard. The *sub_rs* argument, is a list variable where the first element of the list corresponds to the number of different reference standards used. The remaining elements specify the study numbers that used each reference test. For the TB pleuritis example :

```
> REFSTD = list(3, 1:2, 3:4, 5:11)
```

In general, if the dataset includes k reference tests, *sub_rs* must comprise $k + 1$ elements. The next step is to define the initial values for the sensitivity and specificity of the reference standards. We define

```
> init.s2 = c(0.4, 0.45, 0.8)
> init.c2 = rep(0.95,3)
```

the initial values of the 3 reference standards. That is, reference standard 1 has sensitivity = 40% and specificity = 95%, reference standard 2 has sensitivity = 45% and specificity = 95% and reference standard 3 has sensitivity = 80% and specificity = 95%. We now combine *init.s2* and *init.c2* into a single matrix

```
> init_ref = rbind(init.s2, init.c2)
```

the first row being the sensitivities and second row being the specificities. We then put these initial values together with those defined in section 2.1 as follows,

```
> init = list(init_within, init_between, init_ref)
```

The prior information on the sensitivity and specificity of the reference tests can be provided in terms of the plausible ranges of these parameters. Based on a literature review, it was assumed that reference test 1 has a sensitivity ranging from 20% to 60%, reference test 2 has a sensitivity ranging from 20% to 70% and reference test 3's sensitivity is ranging from 70% to 90%. It is also assumed that the specificity of all 3 tests is between 90% and 100%. This information is expressed as :

```
> S2.low = c(0.2, 0.2, 0.7) ;  S2.up = c(0.6, 0.7, 0.9) ;
> C2.low = rep(0.9,3) ; C2.up = rep(1,3) ;
```

3.2   Running the Gibbs sampler

To run the Gibbs sampler, we make the following call to the *HSROC* function

```
> HSROC(data=In.house,  iter.num=50000, init=init, sub_rs=REFSTD,
+    prior.SEref=c(S2.low, S2.up), prior.SPref=c(C2.low, C2.up))
```

3.3   Interpreting the output files

This section is very similar to section 2.3. We will only discuss the new features involved when using a model with multiple imperfect reference tests. Previously, the *HSROCSummary*

function was called using *data, burn_in, Thin* and *print_plot* arguments, leaving all other arguments at their respective default value. Now in this section, in addition to these 4 arguments, we also need to define the *sub_rs* argument in the same way as in section 3.1.

```
> HSROCSummary(data = In.house, burn_in=10000, Thin=2,  sub_rs=REFSTD,
+    print_plot=TRUE )
```

3.3.1 Descriptive statistics

The summary output of this section is similar to the one seen in section 2.3.1. Of course, the main difference being that we now have extra information on the reference tests.

It also returns the prior ranges translated in terms of beta distribution parameters. For example, we provided a prior range of 20% up to 60% for the sensitivity of the reference standard for studies 1 and 2. This is transformed into a $Beta(\alpha = 9.2, \beta = 13.8)$ prior distribution by equating the centre of the range to the mean of the Beta distribution $(\alpha/(\alpha + \beta))$ and 25% of the range to the standard deviation $(\alpha\beta/((\alpha + \beta)^2(\alpha + \beta + 1)))$
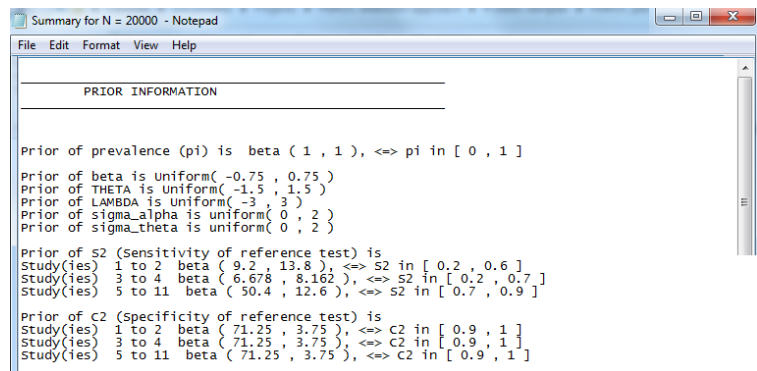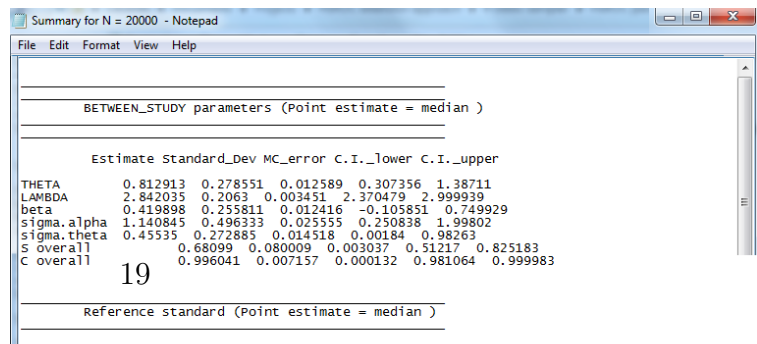

Figure 9

The between-study parameters section of the summary output text file now includes the estimates of sensitivity and specificity for each reference standard (see figure 10).

19

For example, the sensitivity of the reference standard in studies 1 and 2 was estimated to be 0.579393 (0.421502, 0.735355). The MC error (0.000782) is lower than 10% of the standard deviation (0.080355). Similarly, estimation of the specificity of the reference standard in studies 1 and 2 is 0.937374 with 95% HPD interval given by (0.881911, 0.984617). The MC error (0.000673) is also well below 10% of the standard deviation (0.027641).

3.3.2   The graphical summary

Besides the plots presented in section 2.3.2, 2 more files with plots are created when the model allows for imperfect reference standards. The first contains trace plots for the sensitivity and specificity of the different reference standards while the other contains their density plots. Figure 11 shows both trace and density plots for some of these parameters in the TB pleuritis example.
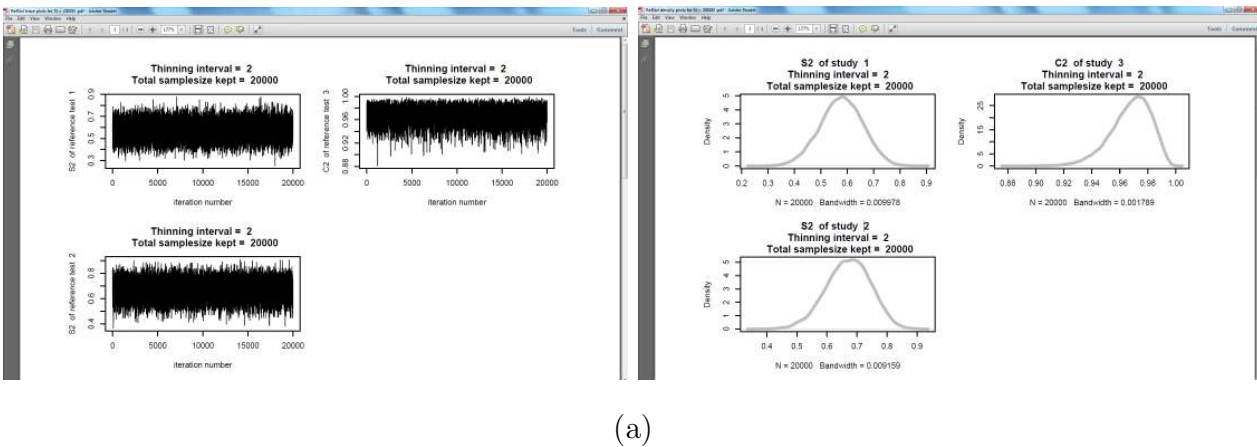

(a)   (b)

Figure 11

4 Simulating data sets

4.1 Simulation of data arising from a HSROC model assuming conditional independence between index and reference tests

The *HSROC* package contains a function that allows simulation of a data set coming from a HSROC diagnostic meta-analysis model. The arguments of the function are

```
> args(simdata)
function (N, n, n.random = "FALSE", sub_rs, prev, se_ref = NULL,
    sp_ref = NULL, T, range.T=c(-Inf, Inf), L, range.L=c(-Inf, Inf),
    sd_t, sd_a, b, path=getwd())
```

The user must specify the values of the between-study parameters (Θ, Λ, β, σ_α and σ_θ), the number of studies desired and the number of individuals within each study. In addition, the sensitivity and specificity of each reference test must be specified in case of non gold standard tests.

Let's suppose we want to generate a dataset of 6 studies with 20 individuals within each study and with the following parameters

```
> beta = 0.25
> LAMBDA = 2.5
> sd_alpha = 0.75 ;
> THETA = -0.2
> sd_theta = 0.50 ;
> pi = c(0.15,0.25,0.10,0.12,0.22,0.18)
```

The following call would generate the desired data under the assumption that a perfect reference test is used in each study.

```
> sim.data = simdata(N=6, n=20, prev=pi,  T=THETA, L=LAMBDA,
+    sd_t=sd_theta, sd_a=sd_alpha, b=beta)
```

The resulting data looks like

```
$Data
     ++ +- -+ --
[1,]  3  6  0 11
[2,]  6  1  0 13
[3,]  2  0  1 17
[4,]  2  0  2 16
[5,]  4  5  0 11
[6,]  2  0  1 17




$'Within study parameters'
         alpha      theta       ++         -- prev
[1,] 1.475586 -0.2052966 0.7973728 0.7268779 0.15
[2,] 2.657826 -0.5538340 0.9516953 0.8101039 0.25
[3,] 2.906462  0.6132198 0.7707464 0.9903992 0.10
[4,] 3.604026  0.1193441 0.9312215 0.9852665 0.12
[5,] 3.119060 -0.9392145 0.9862777 0.7589440 0.22
[6,] 3.269660  0.5130067 0.8389143 0.9925297 0.18


$'Between study parameters'
     THETA sigma theta    LAMBDA sigma alpha      beta  Overal ++  Overall --
 -0.2000000    0.5000000 2.5000000    0.7500000 0.2500000 0.8996607  0.8829387


$'Reference standard'
```

```
[1] "PERFECT"
```

The n argument can be modified e.g. n = c(20,50,25,35,105,15) so each study has a unique sample size provided by the user, resulting in the data below :

```
$Data
      ++ +- -+ --
[1,]  1  1  0 18
[2,]  7  0  3 40
[3,]  3  5  0 17
[4,]  4  2  1 28
[5,] 26 23  0 56
[6,]  3  2  0 10
```

Alternatively, a unique sample size can be selected randomly within a range as follow

```
> n = seq(50,250,1)
```

resulting in the data below :

```
$Data
      ++ +- -+   --
[1,] 11  1  0   42
[2,] 41 95  6   74
[3,] 22  8  1 206
[4,] 25 56  0 146
[5,] 12  7  0   46
[6,] 11  0 10   89
```

In each case described above, the function will also create 3 files in the directory specified by the *path* argument summarizing the dataset. Those files can be used within the *HSROC-Summary* function to compare the estimates of parameters to their true values. Further help on this function can be obtained by typing

```
> help(simdata)
```

To simulate a dataset for an HSROC model with imperfect reference standards we must provide the values of each reference test's sensitivity and specificity. Let's suppose we want to generate data for 6 studies with 2 different reference standards where the first reference test is to be applied over the first 4 studies while the other test will be applied over the remaining 2 studies.

```
> REFSTD = list(2, 1:4, 5:6)
> s2 = c(0.6, 0.75)
> c2 = c(0.95, 0.7)
```

In the example above, sensitivity and specificity of the first reference test are 60% and 95% respectively. Sensitivity and specificity of the second reference test are 75% and 70% respectively. To get a dataset of 6 studies with 200 individuals within each study assuming the same between-study parameters as in section 4.1, we run

```
> simdata(N=6, n=200, sub_rs=REFSTD, se_ref=s2, sp_ref=c2, prev=pi,
+     T=THETA, L=LAMBDA, sd_t=sd_theta, sd_a=sd_alpha, b=beta)
```

resulting in

```
$Data
```

```
      ++  +- -+  --
[1,] 14  14  7 165
[2,] 27  62 13  98
[3,] 15 101 10  74
[4,] 12  75  8 105
[5,] 44  31 40  85
[6,] 25  17 47 111
```

$'Within study parameters'

```
        alpha        theta         ++          -- prev
[1,] 2.586503  0.2755630 0.8154357 0.9622734 0.15
[2,] 1.288232 -0.1718890 0.7642751 0.7037109 0.25
[3,] 1.688717 -0.9207889 0.9403522 0.4654919 0.10
[4,] 2.990638 -1.1974804 0.9912584 0.6321283 0.12
[5,] 2.973003 -0.7219818 0.9743511 0.8068411 0.22
[6,] 2.176751  0.2464142 0.7712676 0.9347985 0.18
```

$'Between study parameters'

```
     THETA sigma theta     LAMBDA sigma alpha      beta   Overal ++  Overall --
 -0.2000000   0.5000000  2.5000000   0.7500000  0.2500000   0.8996607   0.8829387
```

$'Reference standard'

```
      1    2
s2 0.60 0.75
c2 0.95 0.70
```

5   Appendix : Likelihood function and prior distributions for the case when the same imperfect reference standard is used in all studies

The likelihood function of the observed data across the J studies can be expressed in terms of the sensitivity and specificity of each test, and the prevalence in the jth study, ($P(D = 1|\text{Study} = j) = \pi_j$), as follows:

$$L(\Theta, \Lambda, S_2, C_2, \sigma_\alpha^2, \sigma_\theta^2, \beta, \pi_j, \alpha_j, \theta_j, j = 1, \ldots, J | t_{1j}, t_{2j}, j = 1, \ldots, J)$$

$$= \prod_{j=1}^{J} (\pi_j \Phi(-\frac{\theta_j - \frac{\alpha_j}{2}}{exp(\frac{\beta}{2})}) S_2 + (1 - \pi_j) \Phi(-\frac{\theta_j + \frac{\alpha_j}{2}}{exp(\frac{-\beta}{2})})(1 - C_2))^{\sum t_{1j} t_{2j}}$$

$$\times (\pi_j \Phi(-\frac{\theta_j - \frac{\alpha_j}{2}}{exp(\frac{\beta}{2})})(1 - S_2) + (1 - \pi_j) \Phi(-\frac{\theta_j + \frac{\alpha_j}{2}}{exp(\frac{-\beta}{2})}) C_2)^{\sum t_{1j}(1 - t_{2j})}$$

$$\times (\pi_j \Phi(\frac{\theta_j - \frac{\alpha_j}{2}}{exp(\frac{\beta}{2})}) S_2 + (1 - \pi_j) \Phi(\frac{\theta_j + \frac{\alpha_j}{2}}{exp(\frac{-\beta}{2})})(1 - C_2))^{\sum(1 - t_{1j}) t_{2j}}$$

$$\times (\pi_j \Phi(\frac{\theta_j - \frac{\alpha_j}{2}}{exp(\frac{\beta}{2})})(1 - S_2) + (1 - \pi_j) \Phi(\frac{\theta_j + \frac{\alpha_j}{2}}{exp(\frac{-\beta}{2})}) C_2)^{\sum(1 - t_{1j})(1 - t_{2j})},$$

where all sums are from 1 to J.

The pooled 'difference in means' parameter was assumed to have prior density $\Lambda \sim U(-3, 3)$. The log of the ratio between the two standard deviations, β was assumed to follow a U(-0.75, 0.75) distribution. The pooled 'cut-off' parameter, Θ was assumed to follow a U(-1.5, 1.5) distribution. Parameters σ_α and σ_θ were assumed to follow U(0,2) distributions. For the π_j, S_2 and C_2 parameters, we assumed a Beta distribution.

References

[1] N. Dendukuri, I. Schiller, L. Joseph, and M. Pai. Bayesian meta-analysis of the accuracy of a test for tuberculous pleuritis in the absence of a gold standard reference. *Biometrics*, Published online, May 8 2012.

[2] A. Gelman and D. B. Rubin. Inferences from iterative simulation and multiple sequences (with discussion). *Statistical Science*, 7:457–511, 1992.

[3] Ioannis Ntzoufras. *Bayesian Modeling Using WinBUGS*. John Wiley and Sons, Hoboken, NJ, 2009.

[4] C. M. Rutter and C. A. Gatsonis. A hierarchical regression approach to meta-analysis of diagnostic accuracy evaluations. *Statistics in Medicine*, 20(19):2865–2884, 2001.

[5] J. Scheidler, H. Hricak, KK. Yuk, L. Subak, and MR. Segal. Radiological evaluation of lymph node metastases in patients with cervical cancer : a meta-analysis. *Journal of American Medical Association*, 278(13):1096–1101, 1997.

[6] R Development Core Team. *R: A Language and Environment for Statistical Computing*. R Foundation for Statistical Computing, Vienna, Austria, 2010. ISBN 3-900051-07-0.