MISA Package Example

Melanie Wilson

February 18, 2011

The functions in the MISA package focus on intermediate throughput case-control association studies, where the outcome of interest is often a binary disease state and where the genetic markers have been chosen to capture variation in a set of related genes, such as those involved in a specific biochemical pathway. Given this data, we are interested in addressing two questions: "To what extent does the data support an overall association between the pathway and outcome of interest?" and "Which markers or genes are most likely to be driving this association?" To address both of these questions,this package performs a Bayesian model search technique that utilizes Evolutionary Monte Carlo and searches over models including main effects of all genetic markers and marker-specific genetic effects in a computationally efficient manner. The package incorporates functions that:

- Calculate marginal Bayes factors under the log-additive, dominant and recessive parameterizations of genotype that can be used as a screen to reduce the number of SNPs included in the model search.
- Samples a set of models using the EMC algorithm from the posterior.
- Assess convergence of the model search algorithm.
- Calculates multi-level posterior quantities of interest (SNP, gene and global Bayes factors).
- Summarizes results with image plots of the SNPs and genes included in the top models (based on model posterior probabilities).

This vignette is an example of using the `MISA` package in R. We will first load the library (where the lib.loc argument may be needed if MISA is not in the R library tree):

```
> library(MISA)
```

1 Load Simulated Data

Simulated data for the problem is in the datasets `dna.snp.full`. There are 62 variables in the data set: the response `case` is the disease status of each

individual, `age` is a covariate that will be included in every model, and there are 60 SNPs, parameterized as the number of rare alleles (0, 1 or 2) that an individual carries. The data set `dna.snp` has a subset of the full set of SNPs (19). This subset is determined by the marginal Bayes factor screen `bf4assoc` on the full data set `dna.snp.full`. More information on the simulation (including assumed odds ratio and genetic parametrizations of the SNPs) is found in the data set `sim.info`. In particular, we assume that there are two associated genes and 4 unassociated genes. In each of the two associated genes we assume that one of the SNPs is the source of the association. SNP "rs10889710" in gene "GADD45A" was assumed to have an odds ratio of 1.25 and a dominant genetic parametrization, and SNP "rs1470383" in gene "MDM2" was assumed to have an odds ratio of 1.75 and a log-additive genetic parametrization. We begin the analysis by loading the full data set that will be used for the remainder of the analyses:

```
> data(dna.snp.full)
> p.full <- dim(dna.snp.full)[2] - 2
> data(sim.info)
```

2 Marginal Bayes Factor Screen

A marginal Bayes factor screen is then first used to reduce the number of SNPs that will go into the MISA analysis. This is done by the following R statements:

```
> marg.bf <- bf4assoc(D = as.numeric(dna.snp.full$case) -
+     1, X = as.matrix(dna.snp.full$age), XS = as.matrix(dna.snp.full[,
+     -c(1, 2)]), Ns = p.full, Nx = 1, snpsd = 0.25,
+     Prior = 0, MinCount = 1.9, MaxIt = 1000, RelTol = 1e-07)

[1] -0.76126  0.13417

> max.bf <- apply(marg.bf[, 3:5], 1, max)
> dna.snp <- dna.snp.full[, c(TRUE, TRUE, max.bf >
+     1)]
> p <- dim(dna.snp)[2] - 2
```

We note that we are using Prior=0, which is the normal prior on the genetic effects and snpsd=0.25 which is is the assumed standard deviation of the mean zero normal prior. We then are interested in computing the maximum marginal Bayes factor for each SNP across the 3 genetic parametrizations (log-additive, dominate, recessive) and allowing the SNPs with a maximum marginal Bayes factor greater than one to pass the screen and go on for further analysis. The SNPs that have passed the screen are in the data set `dna.snp`.

3 Model Search Algorithm

Once we run the marginal screen to subset the initial full set of SNPs we run `Gene.EMC` to sample a set of models from the model space based on the Evolutionary Monte Carlo model search algorithm. For the purpose of this example we just run 10 iterations. However, for the two output data sets `emc.out.1` and `emc.out.2` that were used to analyze the simulation we ran each for 100,000 iterations. The EMC algorithm was initialized with two random SNPs in the model.

```
> start.snp <- rep(FALSE, p)
> start.snp[c(3, 7)] <- TRUE
> emc.out <- Gene.EMC(data = dna.snp, force = c("age"),
+     fitness = "AIC.BB", b = p, a = 1, start.snps = start.snp,
+     n.iter = 10, N = 5, tmax = 5, tlone = 1, qm = 0.25,
+     display.acc = TRUE, display.acc.ex = FALSE,
+     log = FALSE)

MISA v2.11.1-1.0.1
Date: Fri Feb 18 14:20:50 2011
User: gl37
Node: tack.isds.duke.edu


Iter = 10
Gene.EMC(data = dna.snp, force = c("age"), fitness = "AIC.BB",
    b = p, a = 1, start.snps = start.snp, n.iter = 10, N = 5,
    tmax = 5, tlone = 1, qm = 0.25, display.acc = TRUE, display.acc.ex = FALSE,
    log = FALSE)

Checking parameters ... OK
Checking data ...
Warnings:
    Converting levels into numeric case/control status: case=1 control=0.

Iter     1: Cost = 1545.427
  Accept rates of mutation and crossover are: 0.000 1.000
Iter     2: Cost = 1545.427
  Accept rates of mutation and crossover are: 0.000 1.000
Iter     3: Cost = 1545.427
  Accept rates of mutation and crossover are: 0.800 1.000
Iter     4: Cost = 1535.450
  Accept rates of mutation and crossover are: 0.800 1.000
Iter     5: Cost = 1545.992
  Accept rates of mutation and crossover are: 0.800 1.000
Iter     6: Cost = 1543.761
  Accept rates of mutation and crossover are: 0.800 1.000
```

```
Iter     7: Cost = 1545.427
  Accept rates of mutation and crossover are: 0.800 0.833
Iter     8: Cost = 1545.992
  Accept rates of mutation and crossover are: 0.800 0.857
Iter     9: Cost = 1545.992
  Accept rates of mutation and crossover are: 0.800 0.875
Iter    10: Cost = 1545.427
  Accept rates of mutation and crossover are: 0.600 0.875
```

We note that the covariate "age" is included as a design variable in every model, including the null model. Also, fitness="AIC.BB" denotes that we use AIC to approximate the marginal likelihood of each model and that we place a Beta-Binomial prior with hyper-parameters $a = 1$ and $b = p$ on the model space. The output of the function `Gene.EMC` can be printed to the screen or written to a user-specified output file. This output allows the user to keep an eye on acceptance rates of the mutation and crossover steps (display.acc=TRUE). Although not shown here, we can also output the acceptance rates of the exchange step between each chain in the population to make sure that the temperature scheme is appropriately tuned (display.acc.ex=TRUE). If you see low acceptance rates between chains this may mean that the difference in temperatures between adjacent chains is too far apart and may need to be decreased.

4 Assessing Convergence of Model Search Algorithm

To assess convergence, we run two independent runs of the EMC algorithm for 100,000 iterations each and save the output from the function `Gene.EMC` (`emc.out.1` and `emc.out.2`). Both output data sets are loaded below and convergence diagnostics are plotted in Figure 1 and are made by the R statements:

```
> data(emc.out.1)
> data(emc.out.2)
> emc.out <- list(emc.out.1, emc.out.2)
> converge.EMC(emc.out, plot.type = "all", bandwidth = 1000,
+     b = p, a = 1)
```

We assess convergence of the sampling algorithm by using graphical diagnostics that summarize two independent, long runs of the algorithm. These diagnostics are plotted in Figure 1 for two independent runs of the simulation, each 100,000 iterations long. The upper left panel depicts end–to–end trace plots of the cost values associated with the models sampled in the two runs. The first is plotted in blue and the second in red. This plot is used to verify that the algorithm's movement around the model space is adequate. If this is the case, the sampled model will change frequently and the associated cost values will vary around an average cost so that points in the plot appear to fall within a common horizontal band. If there appears to be drift at the start of either of the trace plots,

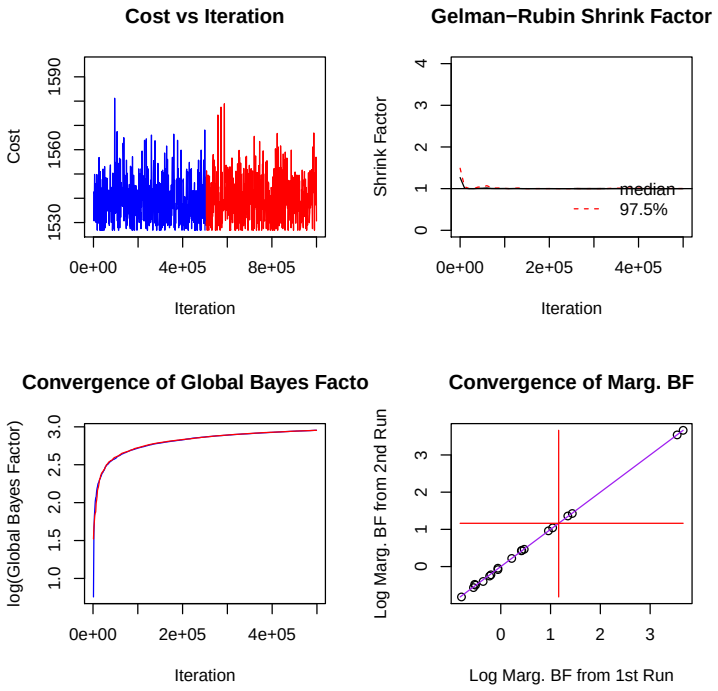

Figure 1: Convergence diagnostics for EMC.Upper left panel:Trace plot of the cost values of the models visited over each iteration for each of the two independent runs. Upper right panel: Plot of the Gelman-Rubin convergence diagnostic of the cost values of the models visited in the two independent chains. Lower left panel: Plot of the global log Bayes factor computed across iterations for each independent chain. Lower right panel: Plot of the SNP Bayes factors for one independent run vs. another independent run

the associated iterations, representing a period of burn–in, should be removed from the analysis. In addition, if the sampled cost value changes infrequently, indicating that the algorithm is not moving adequately, the algorithm should be restarted with a larger maximum temperature or a larger number of parallel chains so that adjacent chains can communicate better.

The upper right panel in the figure depicts a plot of the Gelman–Rubin shrink factor computed for the sampled cost values. The shrink factor should approach 1 as the cost values of the models converge. The lower left panel of the figure shows the logarithm global Bayes factor for each of the two independent runs (the first in blue and the second in red) as a function of iteration. The estimated Bayes factor, which is a lower bound on the value we would compute were we able to enumerate all models, will increase with every new model sampled. As

unique models are sampled less frequently as the algorithm runs, the Bayes factor will begin to plateau suggesting that the value of additional samples is diminishing. The plots of the two runs should begin to converge, with neither making large jumps in the later iterations. Finally, the lower right panel plots the marginal Bayes factors for each of the SNPs for the two independent runs of the algorithm. This plot enables us to determine if the values of the marginal SNP Bayes factors are consistent across two independent runs of the algorithm. Points in the scatter plot will be near the diagonal when MCMC sampling variability for estimating the associated marginal Bayes factor is low. If the plot shows significant deviation from the diagonal, the algorithm has not yet converged and should be allowed to run longer.

5 Calculation of Posterior Quantities of Interest

Given a sample from the model space produced by the output of the function `Gene.EMC` that has passed the above convergence diagnostics, we are able to compute several multi-level posterior quantities of interest: (1) Global Bayes factor giving the evidence of at least one association in the set of tested SNPs, (2) Gene Bayes factors giving the evidence of at least one association within the set of tested SNPs within a particular gene, and (3) SNP Bayes factors giving the marginal evidence of an association with the function `post.prob`. We first take the two independent runs of the algorithm used to determine convergence and combine the results.

```
> data(emc.out.1)
> data(emc.out.2)
> combo.out <- combine.EMC(list(emc.out.1, emc.out.2))
```

Next we calculate global and marginal posterior probabilities based on the unique sampled models in the two runs of the model search algorithm. In particular, we can look at the global posterior probability and Bayes factor giving the evidence of at least one associated marker in the data set. Here, the global Bayes factor is calculated as 20.38 giving strong evidence of at least one associated marker in the analysis.

```
> data(sim.info)
> post.prob.out <- post.prob(combo.out, sim.info,
+     b = p)
> post.prob.out$BF

               Prior= 1
Post.Prob     0.9539645
Bayes.Factor 20.7223458
Prior.Odds    1.0000000
```

6 Summary of Results

Summaries of the marginal SNP Bayes factor and gene Bayes factor results are plotted in Figure 2 and 3 and are made by the following R statements using the output from the function `post.prob`:

```
> model.inc(post.prob.out, num.models = 100, num.snps = p,
+     inc.typ = "s")

> model.inc(post.prob.out, num.models = 100, num.genes = 6,
+     inc.typ = "g")
```

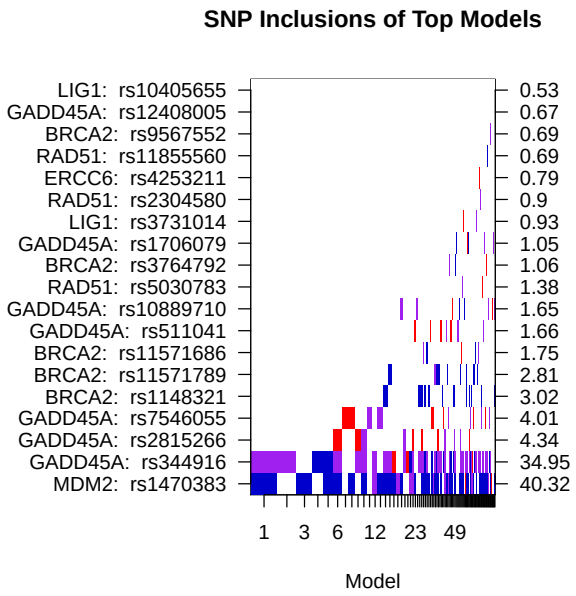

Figure 2: Image plot of SNP inclusions in the top 100 models.

Figures 2 and 3 are image plots of the SNP and gene inclusion indicators for the top 100 Models. SNPs/Genes are ordered based on their marginal Bayes Factors which are plotted on the right axis. The columns correspond to models and have width proportional to the estimated model probability, and the models are plotted in descending order of posterior support. The color of the inclusion block in the SNP plot corresponds to the genetic parameterization of the SNP in that model. Purple corresponds to a log–additive parameterization, red to a

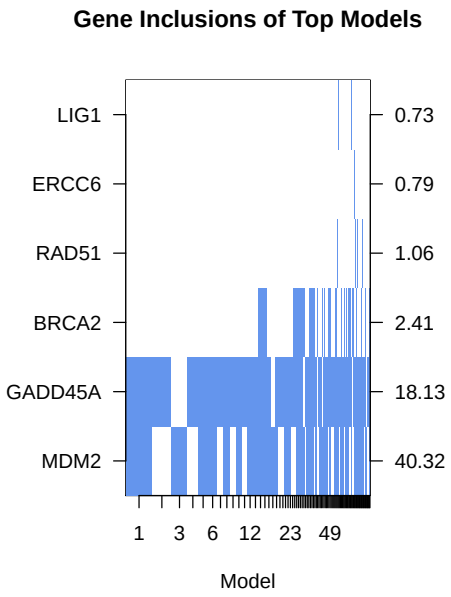

Figure 3: Image plot of gene inclusions in the top 100 models.

dominant parameterization and blue to a recessive parameterization.The color in the gene plot is chosen to be neutral since the genetic parameterizations are not defined at the gene level. We notice that SNPs within both associated genes have marginal SNP Bayes factors suggesting strong evidence of an association and that the marginal gene Bayes factors for the associated genes also suggest strong evidence of an association.