Manual for MCMC_IBDfinder version 1.0
Software for estimating probabilities of IBD sharing among multiple individuals.

Ida Moltke
ida@binf.ku.dk

March 2011

Contents

1 Introduction

This is the manual for MCMC_IBDfinder version 1. Check for the latest version of the software package and this manual at `http://people.binf.ku.dk/ida/Software/MCMC_IBDfinder/`.

MCMC_IBDfinder is a C++ program that estimates posterior probabilities of *identity by descent* (IBD) sharing among $N > 1$ individuals using unphased single nucleotide polymorphism (SNP) data.

The underlying method is based on a Hidden Markov Model (HMM), where the possible hidden states in each locus are all the possible partitionings of the $2N$ chromosomes into IBD sets. Even for a small numbers of individuals the amount of possible partitionings into IBD sets in any one locus is gigantic, e.g. for 25 individuals and thus 50 chromosomes the number of partitionings is $1.86 \cdot 10^{27}$. For this reason it is intractable to obtain posterior probabilities using standard HMM inference methods. The probabilities for specific set partionings are therefore obtained using Markov Chain Monte Carlo (MCMC) simulations.

This means that what the program does in practice is to produce a lot of samples from the posterior distribution of IBD set partitionings from which the user can then approximate the probabilities he/she is interested in.

The sampling process can be somewhat time consuming and it might seem tedious that the probabilities has to be approximated afterwards by the user. However, first of all the MCMC approach makes it possible to solve this seemingly intractable problem. Second, the MCMC approach gives the user the chance to get probabilities for any IBD configuration of interest. The MCMC approach in other words makes it possible to answer a wide range of questions which other IBD programs cannot currently answer directly in a probabilistically sound manner.

Note that this software package is licensed under the gpl licence `http://www.gnu.org/licenses/gpl.html` and comes under no warranty.

2 Getting started

Before the program can be used it has to be downloaded, unpacked and compiled. This can all be done using only a few commands. The following section will describe how.

2.1 Technical requirements to your computer

MCMC_IBDfinder is a C++ program commandline program. In order to be able to compile the program `g++` should be installed. The program has been successfully compiled on ubuntu 10.10 and MAC OS 10.5.8 with `g++` compilers 4.4.5 and 4.2 respectively. And the program should work on both linux and MAC computers.

2.2 Where to download the program

Get the latest version of the package on `http://people.binf.ku.dk/ida/Software/MCMC_IBDfinder/`.

2.3 How to install the program

Start by unpacking the downloaded file with your favorite unpack program. For instance run the command `tar xfvz MCMC_IBDfinder_v1.0.tar.gz` in linux.

The unpacked folder will be called MCMC_IBDfinder_v1.0 and will be structured as follows: The `src` subfolder contains the C++ source files, the `data` subfolder contains the example data used in this manual and the `man` subfolder contains this document.

To install the program simply go to the subfolder called `src` and type `make`. The program should then be compiled using `g++` and an executable '`MCMC_IBDfinder`' should be located in the `src` subfolder.

Installation commands

```
tar xfvz MCMC_IBDfinder_v1.0.tar.gz
cd MCMC_IBDfinder_v1.0/src
make
```

3 File formats

3.1 Input files

MCMC_IBDfinder requires three input files: a genotype file, a posmaf file and an initz file. These files should all be tab delimited.

3.1.1 The genotype file

The genotype file contains the unphased genotypes for all individuals. It consists of one line with genotypes for each individual in the data set of interest. The genotypes should be given in the order they appear in the genome.

A genotype is given in terms of the number of minor alleles (the least common of the two possible alleles in the population). Hence in a locus where the major allele is A and the minor allele is C, AA is denoted 0, AC is denoted 1 and CC is denoted 2.

For an example of a genotype file see table 1.

1	2	2	0	0	2
2	1	2	0	1	1
1	1	1	1	0	2

Table 1: Example of a genotype file for 3 individuals each with genotype data from 6 SNP loci.

Note, that the program cannot handle missing data, hence loci containing missing data should not be included in the genotype file.

3.1.2 The posmaf file

The posmaf file contains two lines both of length x, where x is the number of loci for which there is data in the genotype file. The first line consists of the genetic positions of the x loci (measured in cM). The second line consists of the minor allele frequency of each of the x loci. The minor allele frequencies should be estimated from population data such as HapMap.

For an example of a posmaf file see table 2.

0.1	0.3	0.5	0.8	1.0	1.2
0.2	0.1	0.3	0.1	0.4	0.2

Table 2: Example of a posmaf file for a data set with 6 SNP loci.

3.1.3 The initz file

The initz file contains the overall IBD configuration which is used to inititalize the program. It contains one row of numbers for each individual in the data set and two consecutive columns for each locus. Hence line i consists of a number for locus 0 of individual i's first chromosome, followed by a number for locus

0 of individual i's the second chromosome, followed by a number for locus 1 of individual i's first chromosome, followed by a number for locus 1 of individual i's second chromosome etc.

The numbers must range from 0 to k, where k is a user defined maximum on the number of IBD sets present in any given locus. The numbers for a given locus indicate the IBD set partitioning in this locus; chromosomes with the same number $n > 0$ are in the same IBD set (e.g. they are IBD) and chromosomes with different numbers are not in the same IBD set (e.g. they are not IBD). Chromosomes with the number 0 are not IBD with any other chromosome in the data set.

For an example of a initz file see table 3.

1	2	1	2	1	2	1	2	1	2	1	1
1	2	1	2	1	2	1	2	1	2	1	1
1	0	1	0	1	0	1	0	1	0	1	1

Table 3: Example of a initz file for a data set with 3 individuals and 6 SNP loci. In the first 5 loci all 3 individuals share their first chromosome IBD and the two first individuals also share their second chromosome IBD. In the last locus all 6 chromosomes are IBD.

Note that this is only the initial IBD configuration and does in theory not matter at all for the final results. It only matters for the time it takes to get the results. For most purposes a file with only zeroes or a file with only ones will be fine.

3.2 Output file

The output is per default printed to screen, but if you specify an output filename the output is instead printed to the file. This is recommended since the amount of output is often rather large. The output starts with a list of all the parameter values that were used when running the program. This list is followed by information about the MCMC samples produced by the program.

The information that is printed for each MCMC sample consists of $N + 1$ lines, where N is the number of individuals being analysed. The first line starts with ">" and is followed by 6 numbers: the sample/iteration number, the sampled value of λ, the sampled value of ρ, the logarithm of the transition probability of the sample, the logarithm of the emission probability of the sample and the time since the sampling started measured in seconds[1]. Hence if the log-likelihood of the sample is wanted this can be achieved by adding the logarithm of the transition probability and the logarithm of the emission probability.

The remaining N lines describe the sampled IBD configuration. This is represented exactly as it is in the initz file with the only exceptions that all lines starts with a "z".

An example of the output can be seen on the next page.

[1]For an explanation of the parameters λ and ρ and of the transition and emission probabilities, see [Moltke et al. (2011)]

example of output

```
You are running MCMC_IBDfinder version 1.0 with the following parameter values:

Data parameters
-g      Name of file with genotypes             verysmallexamplegenotypefile.txt
-p      Name of file with pos maf info          verysmallexampleposmaffile.txt

Initialisation parameters
-z      Initial value of z (filename)           verysmallexampleinizfile.txt
-l      Initial value of lambda                 0.200000
-r      Initial value of rho                    0.200000
-s      Seed for random numbers                 100

Run parameters
-e      Genotyping error rate on allele level   0.010000
-f      Index of first SNP                      0
-k      Max number of IBD sets                  2
-ni     Number of iterations                    1
-ns     Number of SNPs to include in MCMC       All

Move parameters
-nl     Number of moves of type l               0
-nr     Number of moves of type r               0
-nlw    Number of moves of type lw              1
-nrw    Number of moves of type rw              1
-nzreg  Number of moves of type zreg            350
-nz     Number of moves of type z               150
-nzbor  Number of moves of type zbor            200
-nzmreg Number of moves of type zmreg           0
-nzxorr Number of moves of type zxorr           150
-nznes  Number of moves of type znes            0
-nzsnes Number of moves of type zsnes           0
-nzcp   Number of moves of type zcp             50
-nzbc   Number of moves of type zbc             100

Output parameters
-o      Output destination (filename/screen)    Screen
-t      Thinning t (print every t iteration)    1

MCMC has been started...

> 0     0.200000        0.200000        -8.927122       -31.064723      0
z 1     2       1       2       1       2       1       2       1       2       1       2
z 1     2       1       2       1       2       1       2       1       2       1       2
z 1     0       1       0       1       0       1       0       1       0       1       0
> 1     0.181560        0.174060        -11.398768      -16.858966      0
z 2     0       2       2       2       2       2       2       2       2       2       2
z 0     2       0       2       0       2       0       2       0       2       0       2
z 2     0       2       0       2       0       2       0       2       0       2       0
```

4 How to run the software

The program is a command line program, hence it is run by typing a command in a terminal window. The command should be written on one line (but has been split up in several lines in some of the examples below for improved readability). The order of the arguments does not matter.

4.1 Getting help

To get help simply type:

This should be run from the src folder

```
./MCMC_IBDfinder -h
```

This will both give you a list of all options and a list of all default parameter values.

4.2 Basic usage

The basic run command looks like this:

This should be run from the src folder

```
./MCMC_IBDfinder -g yourgenotypefile.txt  -p yourposmaffile.txt
 -z yourinitzfile.txt
```

The "`-g -p -z`" parameters are required and they are explained below:

-g Genotype file with individuals as rows and columns as SNPs. The genotypes should be tab delimited. See Section 3.1.1 for more elaborate info.

-p Posmaf file with one line with genetic positions and one line with minor allele frequencies. Tab should be used as delimiter. See Section 3.1.2 for more elaborate info.

-z Initz file with the initial IBD configuration. Tab should be used as delimiter. See Section 3.1.3 for more elaborate info.

Note that in the program the symbol z is used to denote the IBD configuration as opposed to x in the paper.

4.3 Optional parameters

Besides the three required parameters, there are a number of additional optional parameters. Below is a list with an explanation of each of the parameters and what the default value it is set to if the user does not chose to set it.

4.3.1 Initialisation parameters

-l initial value of lambda. Default value is 0.2.

-r initial value of rho. Default value is 0.2.

-s intitial seed for random numbers. Default value is 0.

4.3.2 Run parameters

-e the genotyping error rate. Should be set according to the data you use. The default value is set to 0.01.

-f the first SNP to include in the analysis. Can be used to indicate that only a region of the genome should be analysed. Default value is 0.

-k the maximum number of IBD sets allowed in each locus. The higher k is, the slower the program is. Default value is set to 2.

-ni number of iterations to run the MCMC algorithm for. Default is 10. This is very low for almost any data set.

-ns number of loci to include in the analysis. Can be used to indicate that only a region of the genome should be analysed. Default value is all loci starting from the value indicated by the -f option.

4.3.3 Move parameters

In the MCMC algorithm a lot of different moves are possible. The default is a mixture of them. This mixture can be changed. The different moves are described in detail in the supplementary material for the paper describing the method that this software is an implementation of. Here is a list of the options that allow the user to change the number of each of the moves that in total makes it up for one iteration of the MCMC algorithm:

-nlw number of 'parameter λ' moves. Default value is 1.

-nrw number of 'parameter ρ' moves. Default value is 1.

-nz number of 'single site Z' moves. Default value is 150.

-nzreg number of 'region Z' moves. Default value is 350.

-nzbor number of 'border Z' moves. Default value is 200.

-nzbc number of 'block state change Z' moves. Default value is 100.

-nzcp number of 'copy Z' moves. Default value is 50.

-nzxorr number of 'IBD expansion/reduction Z' moves. Default value is 150.

4.3.4 Output parameters

-o name of output file. If not specified output will be printed to screen.

-t thinning. The state of the markov chain is printed every tth iteration. Default value is 1.

4.4 How to run a full analysis

To be able to run a full analysis of a given data set using this software some knowledge about MCMC is needed. This section will therefore start with a very brief introduction to MCMC. For a much more elaborate description, see [Gilks et al. (1996)].

4.4.1 Very brief introduction to MCMC

The goal in MCMC is to sample from a given distribution, D. The distribution this software uses MCMC to sample from is the posterior distribution of IBD configurations, i.e. the distribution of IBD configurations given some SNP chip genotype data. This is done by continuously sampling IBD configurations based on the previously sampled IBD configuration in a manner so the samples are eventually correlated samples from D. When this point is reached the chain of samples is said to have converged.

What is important to note is:

- the samples will not from the very beginning be samples from D
- it is hard based only on one run of the software to decide if and when convergence has been reached
- the samples are correlated

When analyzing data, these issues have to be taken into account.

First, in order to be able to assess if convergence has been reached **one should always run more than one chain** (with different seeds and starting parameter values). Because once they have both converged they should both supply samples from D and thus from the same distribution. Hence by comparing result from more chains it is easier to determine when convergence has been reached.

Second, to ensure that the samples that used in later analyses are all samples from the D **all samples before convergence was reached should be discarded**. This is called burnin.

Third, since the samples are correlated **it is often a good idea only to keep every tth sample**. If nothing else then too avoid having to store too many samples. This is called thinning.

Once the above has been done, one can use the samples that are left to approximate posterior probablities very simply: the probability of any given IBD (sub-)configuration can be approximated by the fraction of samples in which the IBD configuration is present. So for instance if one wishes to approximate the probability that the first three individuals share at least one chromosome IBD in locus 10 and this is the case in 2000 out of 10000 samples then the probability of this is simply 0.2 (approximately).

If one is interested in the full posterior distribution of IBD configurations this can of course be approximated using the exact same procedure. The same is true for the modes of the posterior. Hence the MCMC approach gives very high flexibility in the questions that can be answered.

4.4.2 Assumptions that should be taken into account

The model underlying this software is based on a number of assumptions. The most important ones are:

- all individuals in a given data set are from the same population.
- the genotype data does not contain linkage disequilibrium (LD).
- the genotype data does not contain any missing data.

In practice this means that input data sets should be pruned for LD and missing data before running this software.

4.4.3 Outline of the steps of an analysis

Based on the above introduction to MCMC and the underlying model an analysis should consist of the following step:

1. Prune away LD and missing data from the data set.
2. Run two or more chains with different seeds and start values.
3. Check if and when the chains have converged.
4. Discard all samples before convergence (burn-in samples)
5. If high correlation betweeen samples discard all samples but every tth sample.
6. Approximate posterior probability of a (sub-)IBD configuration by the fraction of times it is present in the samples that remains after step 4.

4.4.4 Example analysis

In this section we will present a toy example of running an analysis using MCMC_IBDfinder. We will use the files called examplegenotypefile.txt, exampleposmaffile.txt, exampleinitzfile0.txt and exampleinitzfile1.txt. These files can all be found in the data subfolder of the MCMC_IBDfinder package.

The files contain artificial data from 3 individuals at 20 SNP loci with no LD and no missing data. Let us say we are interested in answering how probable it is that all three individuals share at least one chromosome IBD in SNP locus 20. To answer we follow the above analysis outline:

Step 1: Remove LD and missing data This first step is not necessary for this data set, since the data is simulated in a manner so it does not contain neither LD nor missing data.

If we had not known that the data was LD free, we should have tested for presence of LD and removed it if it was present. This could for instance have been be done by calculating pairwise r^2 for all pairs of SNP loci and discarding those loci with with an associated r^2 value higher than a given threshold. Note that to do this properly we would have to use data from more than the three individuals that this very small data set consists of. If the three individuals are Europeans LD we could for instance have based the LD calculations on a bigger data set consisting of both the three individuals and the 60 unrelated HapMap individuals with European ancestry.

Step 2: Run two or more chains We run two MCMC chains with different seeds and different starting values.

This should be run from the data folder

```
time ../src/MCMC_IBDfinder -g examplegenotypefile.txt
-p exampleposmaffile.txt -z exampleinitzfile0.txt
-s 3 -k 1 -ni 100000 -t 5 -o example_t5_0.out

time ../src/MCMC_IBDfinder -g examplegenotypefile.txt
-p exampleposmaffile.txt -z exampleinitzfile1.txt
-s 1117 -k 1 -ni 100000 -t 5 -o example_t5_1.out
```

Note that to avoid saving too many samples, we only store every 5th sample. Since consecutive MCMC samples are correlated this should only lead to a very small loss of information.

Step 3: Check for convergence One simple (but not always sufficient) way to check if convergence has been reached is to look at a statistic called 'potential scale reduction factor' (PSRF) introduced by [Gelman and Rubin (1992)]. This statistic is based on the output from two or more chains and it is in broad terms a measure of the ratio between the variance within each of the chains and the variance between them. It will be close to 1 if convergence has been reached.

We can calculate PSFR in R using the package *coda*. In case this packages is not already installed you can do this as follows in R

This should be run in R from the data folder

```
# Install coda
install.packages('coda')
```

When *coda* has been installed you can calculate and plot PSFR along the chains as follows.

This should be run in R from the data folder

```
# Load convergence assessment package
library("coda")

# Get sample info
chain0info = read.table("example_0_t5_locus20.allIBD")
chain1info = read.table("example_1_t5_locus20.allIBD")

# Plot
bothchaininfos = mcmc.list(mcmc(chain0info),mcmc(chain1info))
varnames(bothchaininfos) = ""
gelman.plot(bothchaininfos,main="Convergence check")
```

Note that here the file example_0_t5_locus20.allIBD is a file that has been derived from the general MCMC output files. It contains a 0 or a 1 for each MCMC sample in the first chain; a 0 if the three individuals do not share at least one chromosome IBD in locus 20 in the sample and a 1 if they do. The file example_0_t5_locus20.allIBD contains the same for the second chain.

The resulting plot can be seen in figure 1. As can be seen PSRF seems reach a stable level close to 1 after the first 9000 samples. This is likely to indicate that convergence has been reached after the first 9000 samples.

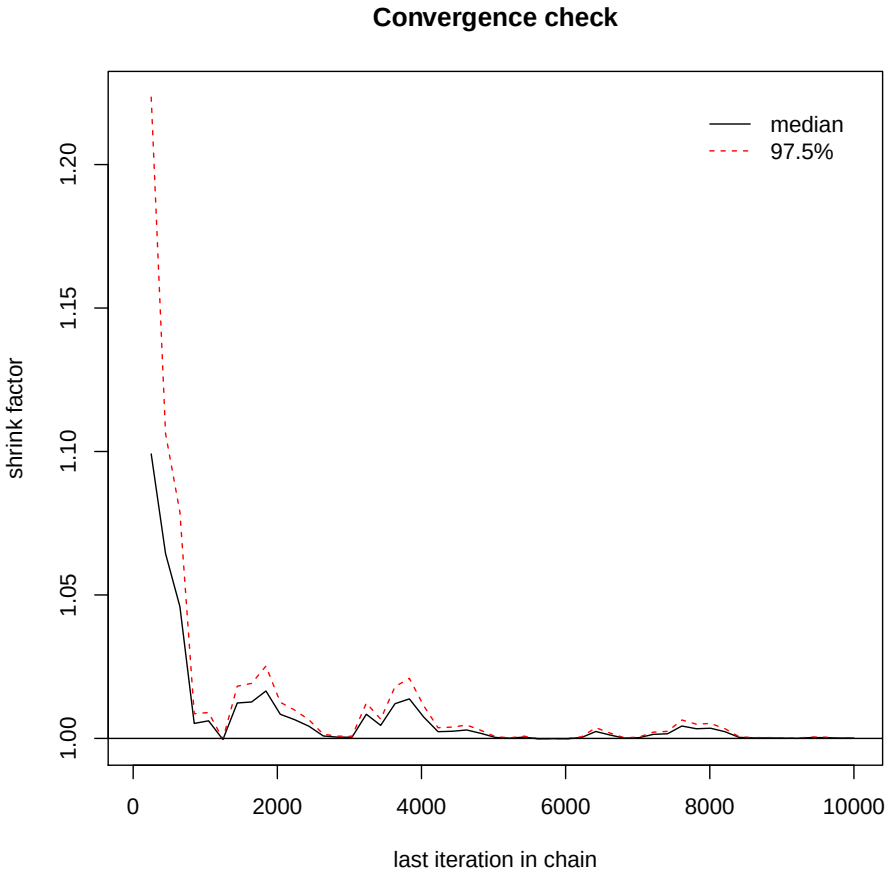

Figure 1: Plot of convergence statistic.

Step 4: Discard burn-in samples Based on the result in previous step we discard the first 9000 samples as burnin. Hence we now have 1000 samples left in each chain.

Step 5: Thinning To save harddisk space we already used a thinning factor of 5 when we ran MCMC_IBDfinder. To test if this leaves us with samples that

are not too correlated we check the chains for autocorrelation using the autocorr function in R:

This should be run in R from the data folder

```
autocorr(mcmc(chain0info[9000:10000,]),lag=c(1:5,10))
autocorr(mcmc(chain1info[9000:10000,]),lag=c(1:5,10))
```

For chain 0 for instance this gives the following output:

Result of running autocorr

```
              [,1]
Lag 1  -0.01390051
Lag 2  -0.01394101
Lag 3  -0.01398151
Lag 4  -0.04070254
Lag 5  -0.01406251
Lag 10  0.03909604
```

which indicates that there is there is very little autocorrelation, and hence that there is no reason to thin the chains any further.

Step 6: Approximate posterior probability We finally count the number of samples in which all three individuals share at least one chromosome IBD in locus 20. In chain this amounts to 961 out of the 1000 samples we were left with after burnin. And thus the posterior probability that all three individuals share at least one chromosome IBD in locus ten is approximately $961/1000 = 0.961$.

5 Further comments

5.1 Run time

The program is very CPU intensive and so far we have not used the software for more than 30 individuals. The lower the parameter k is set the faster it runs. For example run times see [Moltke et al. (2011)].

5.2 Citations

If you use this software for an analysis please cite [Moltke et al. (2011)].

6 Contact and bugreporting

The author can be contacted by emailing to **ida@binf.ku.dk**. Please provide your operating system, compiler version, the MCMC_IBDfinder version, and of course a small description of the problem you have encountered.

References

[Gelman and Rubin (1992)] Gelman A and Rubin DB (1992) *Inference from iterative simulation using multiple sequences.* Statistical Science, 7, 457-511.

[Gilks et al. (1996)] Gilks WR, Richardson S and Spiegelhalter DJ (1996). *Markov Chain Monte Carlo In Practice.* Chapman and Hall/CRC.

[Moltke et al. (2011)] Moltke I, Albrechtsen A, Hansen TvO, Nielsen FC and Nielsen R (2011). *A Method for Detecting IBD Regions Simulataneously in Multiple Individuals – with Applications to Disease Genetics.* Genome Research, Epub ahead of print.