AutoMix

Release 1.3

Jan 29, 2020

Contents

The AutoMix package is a C program for Unix-like systems, implementing the automatic reversible jump MCMC sampler of the same name described in Chapters 4, 5, and 6 of David Hastie's Ph.D. thesis included in *docs/thesis*.

Warning: Potential users should carefully understand the limitations of using the AutoMix sampler. The reliability of results from this sampler depends on many factors, including the scaling of the parameters and the degree of multimodality of the within-model conditionals of the target distribution.

CHAPTER 1

Usage

To run the sampler for a particular problem the program must be compiled with a user-provided file containing four C functions that define the problem in question. Examples of such files for problems considered within the aforementioned thesis are bundled with the archived AutoMix software. More details of the functions are provided below.

The output of the sampler is in the form of several text files summarizing the MCMC sampler. We discuss these files in section *Output Files*. Subsequent analysis may be performed on the output with the use of statistical packages. We recommend R as a good free choice of such a package.

1.1 Contents:

1.1.1 Compilation

To compile the AutoMix sampler for the examples included within the AutoMix package, the user should edit the *Makefile* (supplied) if required (for example the compile flags, or libraries)

Optimized Compilation

Typing:

```
make
```

in the shell, in the AutoMix folder where the AutoMix distribution was unzipped to, will compile all the programs that were distributed with this package.

Compile with Debug Information

To compile all programs with debug information (*-g* flag enabled), type:

```
make DEBUG=1
```

The supplied programs have all been debugged but we acknowledge that use of a debugger can often help to understand how the program works.

Any of the programs can also be made individually, with the appropriate make command.

Example

For example:

```
make amtoy1
```

will make amtoy1. See the Makefile for further examples.

The executables have the form *amNAME*, where *NAME* is the name of the example.

Clean

To remove the executables and object (.o) files, type:

```
make clean
```

To compile the sampler for a new example, the Makefile can be edited to contain the appropriate commands (remembering to include command for compiling any dependencies) for the new program.

1.1.2 Running AutoMix

The sampler is run by typing the name of the executable, followed by run-time flags separated by spaces. The run-time flags control a number of options within the sampler. If flags are not supplied default values are used. The flags can be used in any order.

Command-line Arguments

The flags can be summarised as follows (I is assumed to be a positive integer):

-m <mode>

Controls the mode of the sampler.

- Mode 0 is mixture fitting.
- Mode 1 skips stage 1 and 2 if a file containing the mixture parameters is supplied.
- Mode 2 fits AutoMix version of AutoRJ sampler (see Green, 2003 - full reference in thesis).

Default uses *D=0*.

-n <iters>

Run the sampler for max(iters, nkk * 10000, 100000) iterations in the stage 1 RWM for each model k.

Default value is 100,000.

-N <iters>

Run the sampler for *iters* Reversible jump iterations in stage 3.

Default uses I=100000.

-s <seed>
Initialises the random number generator with seed I.

Default uses clock as seed.

-a <adapt>
Controls whether or not adaptation is done in stage 3 RJ.

If *adapt=0* no adaptation is done, if *adapt=1* adaptation is done. Default is 1.

-p <perm>
Controls whether or not random permutation is done in stage 3 RJ.

If *perm=0* no permutation is done, if *perm=1* permutation is done. Default is 0.

-t <dof>
Controls whether standard Normal or T distributed variables are used in RWM and in RJ moves.

If *dof=0* Normal variables are used, otherwise t-distributed variables with dof degrees of freedom are used. Default is 0.

-f <basestring>
Uses the string *basestring* as the basis for filenames (e.g. if *basestring=output*, filenames will be named *output_log.data*, *output_mix.data* etc).

Default is "output".

[-h, --help]
Prints help information on command line arguments and exits.

[-v, --version]
Prints AutoMix version number and exits.

Example

As an example, typing:

```
amtoy1 -m 0 -N 1000000 -p 1 -f toy1
```

runs the optimized mixture fitting version of the toy1 problem (see thesis, section 5.5.1) with 1 million RJ sweeps, enabling permutation and storing the output in files of the type *toy1_*.data*. Running the sampler produces a summary of how the run is progressing.

For each of the models:

- In **stage 1** a countdown of the number of iterations remaining is printed to screen;
- In **stage 2** a summary of the mixture fitting is printed to screen. This summary consists of a countdown of the number of components in the current mixture, with the iteration number that the last component was removed and an indicator *n* if the component was annihilated naturally, and *f* if the annihilation was forced.
- In the RJ **stage 3** a countdown of the number of iterations remaining is printed to screen. No summary statistics are printed to screen. Instead all output from the sampler is written to files.

1.1.3 Writing User Functions

For each example to which the user wishes to apply the AutoMix sampler the user must supply a file (which is linked at compile time, see Makefile for examples) containing four user functions written in C.

Warning: A familiarity with the C programming language is assumed.

These functions must have names and arguments detailed below and return the following information:

1. A function that returns the number of models:

```
int get_nmodels(void)
```

2. A function to load the dimension of each model:

```
void load_model_dims(int nmodels, int *model_dims)
```

   Given the number of models *nmodels*, on exit from the function the vector *model_dims* should contain the dimensions of the models.

3. A function to get initial conditions (possibly random) for the RWM for a given model:

```
void get_rwm_init(int model_k, int mdim, double *rwm)
```

   Given the model index k, and the dimension *model_k* of that model, on exit from the function the vector *rwm* should contain the initial conditions. In particular, rwm[j-1] should contain the (possibly random) intial state for component j of the parameter vector associated with model k, (j=1,. . . ,mdim).

   If random initial states are used, the user must also declare the random number functions in the file (see the example files).

4. A function to return the log of the target function pi evaluated at a given point in the state space, up to an additive constant:

```
void logpost(int model_k, int mdim, double *theta, double* lp, double *llh)
```

   Given the model index *model_k*, and parameter vector *theta* (of dimension *mdim*), the function must load in *lp* the log of the target function (up to an additive constant) evaluated at this point. If pi is a posterior distribution, the double *llh* should contain the log-likelihood evaluated at this point (although this is only necessary for returning the log-likelihood to output file, and can contain any other value if preferred).

The examples provided, with comments, show typical examples of these user files for the problems under consideration.

1.1.4 Output Files

The following files are returned by the AutoMix sampler (assuming the filestem is "output")

output_ac.data

A file containing the estimated autocorrelation coefficient for the chain. The coefficients go up to the order that was used to compute the autocorrelation time using Sokal's method (see Green and Han, 1992)

output_adapt.data

A file summarising the AAP adaptation process for each model - adapting the scale parameter of the single component RWM. For a model with nk components, there are *2*nk* columns, with the odd columns showing the evolution of the scale parameters and the even columns showing the evolution of the average RWM acceptance rate for the corresponding scale parameter.

output_cf.data

A file summarising the evolution of the mixture fitting for each model. In particular, for each model the first column records the number of mixture components at the current iteration, the third column summarises the cost function (to be minimised to find the best mixture) and the final column is an indicator function which takes the value 1 if a component is annihilated naturally at that iteration, 2 if a component is removed by forced annihiliation. (The second column is a term that contibutes to the cost function, see Figueiredo and Jain, 2002).

output_k.data

A file recording the model index at each sweep of the RJ sampler (after burn-in).

output_log.data

A log file, containing important run statistics, including run-time options, run seed, mixture and RWM parameters for each model, and acceptance rates and autocorrelation statistics for the RJ stage 3. Model probabilities are also returned in this file.

output_lp.data

A file recording the log posterior (1st column) and log likelihood (2nd column) at each sweep of the RJ sampler (after burn-in).

output_mix.data

A file containing the fitted mixture parameters for the each model. The file is for use by the AutoMix program and is not easily readable by the user. The file contains the number of models and the dimension of each model. Then for each model in turn, the file records the adapted random walk scaling parameters, the number of components of the mixture fitted to that model, and for each component in turn the weight, the mean and the lower triangle of the B matrix. It is this file that is required by the AutoMix sampler if it is run in mode 1.

output_pk.data

A file containing the evolution of the model jumping proposal parameters for the RJ stage 3. Column k is the probability of jumping to model k.

and for each model k. . .

output_thetak.data

A file recording the parameters conditional on model k at each sweep.

1.1.5 License

The AutoMix sampler is free for personal and academic use, but users must reference the sampler as instructed below. For commercial use permission must be sought from the author. To seek permission for such use please send an e-mail to d_hastie@hotmail.com outlining the desired usage.

Use of the AutoMix sampler is entirely at the user's own risk. It is the responsibility of the user to ensure that any conclusions made through the use of the AutoMix sampler are valid. The author accepts no responsibility whatsoever for any loss, financial or otherwise, that may arise in connection with the use of the sampler.

The AutoMix sampler is available from http://www.davidhastie.me.uk/AutoMix. The sampler may be modified and redistributed as desired but the author encourages users to register at the above site so that notice can be received of updates of the software.

Before use, please read this README file bundled with this software.

Index

Symbols

C