Running Coalescent Analyses With coalescentMCMC

Emmanuel Paradis

April 22, 2022

Coalescent analysis is a powerful approach to investigate the demography of populations using genetic data. The coalescent is a random process describing the coalescent times of a genealogy with respect to population size and mutation rate. In the majority of cases, the genealogy of individuals within a population is unknown. So a coalescent analysis typically integrates over the likely genealogies to make inference on the dynamics of the population. This uses computer-intensive methods such as Monte Carlo simulations of Markov chains. Besides, if priors are defined on the distributions of the parameters, Bayesian inference can be done. Several methods have been proposed for such integrations, although currently there is no consensus on which method is the best, or which ones are the most appropriate in some circumstances [1].

coalescentMCMC aims to provide a general framework to run coalescent analyses. In its current version, the package provides a simple MCMC algorithm based on Hastings's ratio.

coalescentMCMC has three main groups of functions that have different roles:

- the function `coalescentMCMC` itself which runs the chain;
- some functions doing operations on tree which are called by the previous one to move from one tree to another;
- some functions to infer demography from genealogies under various coalescent models which are typically used to analyse the output of a chain run.

The motivating idea behind coalescentMCMC is that the user can have full control over the analysis. The options of the main function are:

```
> library(coalescentMCMC)
> args(coalescentMCMC)

function (x, ntrees = 3000, model = "constant", tree0 = NULL,
    printevery = 100, degree = 1, nknots = 0, knot.times = NULL,
    moves = 1:6)
NULL
```

where `ntrees` are the number of trees to output, and `tree0` is the initial tree (if not provided, a UPGMA tree from a JC69-based distance matrix is used). `model` is either `NULL` in which case Θ is assumed to be constant, or `"time"` in which case a model where Θ follows an exponential growth is used.[1] Finally, `printevery` is an integer controlling the print frequency of the progress of the chain.

The current implementation uses neighborhood rearrangement as proposed in [2] calling the function `NeighborhoodRearrangement` at each step of the chain. Five other tree moves are availables taken from [**?**]. This can modified by using other functions described in `?treeOperators`.

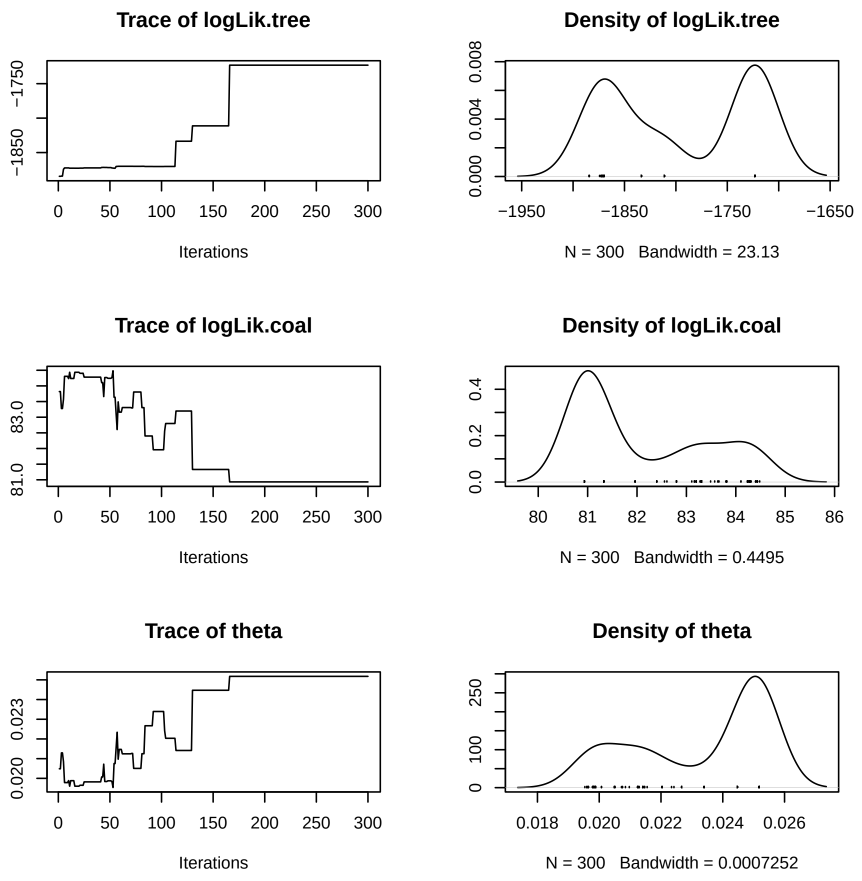
Let us now consider a very simple analysis with the woodmouse data available in `ape`. For the purpose of this vignette, we run a very light analysis in order to produce small outputs in a reasonable time.

```
> data(woodmouse)
> out <- coalescentMCMC(woodmouse, ntrees = 300)

Running the Markov chain:
  Number of generations to run: 200
Generation    Nb of accepted trees
   300                 162
Done.
```

The output object is of class `"coda"`, so we can visualise it with the package of the same name (which has already been loaded):

```
> plot(out)
```

[1]Other models will be implemented later. See the vignette "CoalescentModels".



The log-likelihood was relatively stable between −1870 and −1880. The trees are stored in a special place of the memory (an 'environment' in R's jargon) from where they can be retrieved with a specific function:

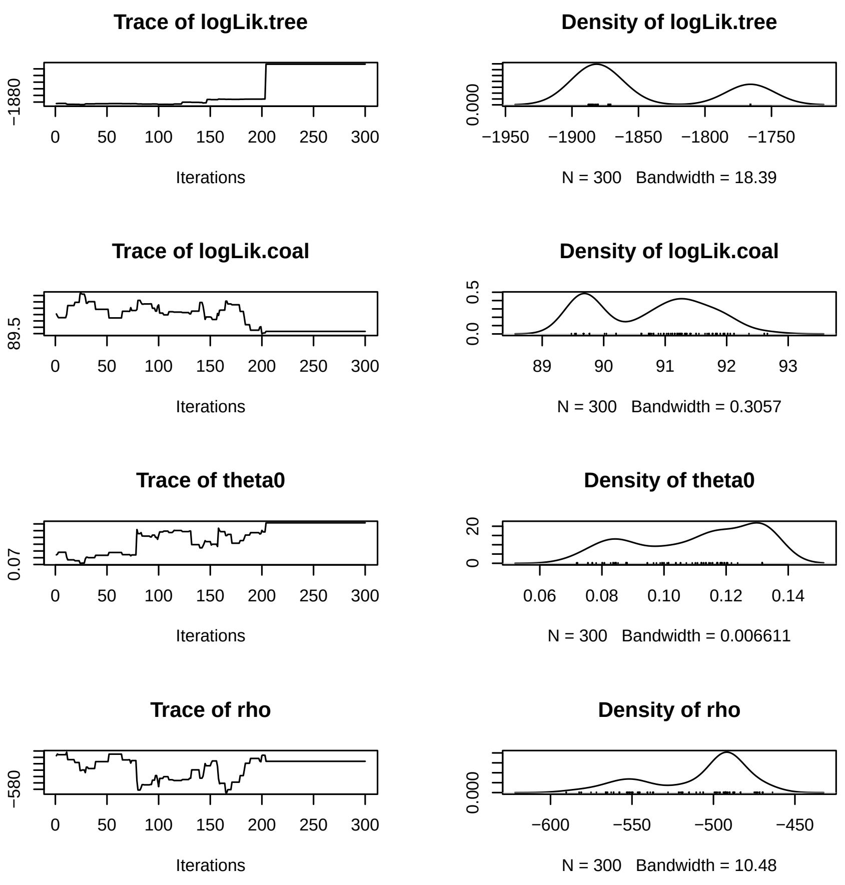
```
> TR <- getMCMCtrees()
> TR

300 phylogenetic trees
```

Note that the trees generated during the burn-in period are not output, but the corresponding values of log-likelihood and Θ are. Hence `out` has 400 rows.

```
> dim(out)

[1] 300   3

> colnames(out)

[1] "logLik.tree" "logLik.coal" "theta"
```

We now run a model of time-dependent coalescent where Θ follows an exponential change through time:

```
> out2 <- coalescentMCMC(woodmouse, ntrees = 300, model = "time")

Running the Markov chain:
  Number of generations to run: 300
Generation    Nb of accepted trees
   300                 154
Done.

> plot(out2)
```



The change in log-likelihood along the chain is similar to what was observed above. The object `out2` has now three columns:

```
> dim(out2)

[1] 300   4

> colnames(out2)

[1] "logLik.tree" "logLik.coal" "theta0"      "rho"
```

If we try to extract the trees as previously done and R is running in interactive mode, we will be asked which list of trees to extract:

```
> getMCMCtrees()
Several lists of MCMC trees are stored:

1 : TREES_001
2 : TREES_002

Return which number?
```

It is also possible to extract the trees of a specific chain with its number, which is useful when the R code is not run interactively (such as this vignette):

```
> TR2 <- getMCMCtrees(2)
```

The parameters of the MCMC runs are stored separately and can be extracted with:

```
> getMCMCstats()

MCMC chain summaries (chains as columns):

                     001 002
Nb of generations    300 300
Nb of accepted moves  49 111
```

We can now compare both coalescent models: the two hypotheses under consideration are:

- H_0: Θ is constant;
- H_1: Θ changes through time following an exponential model.

Other things that could be done with simple R commands include:

- Compute confidence intervals around $\hat{\Theta}_0$ and $\hat{\rho}$ (alternatively, posterior distributions of these parameters if a Bayesian sampling is done);
- Re-run the chain(s) with different initial trees, for instance to run branching chains taking a tree from `TR` or `TR2`.
- Use other models of time-dependent coalescent (see the other vignette in this package).

References

[1] J. Felsenstein. Trees of genes in populations. In O. Gascuel and M. Steel, editors, *Reconstructing Evolution: New Mathematical and Computational Advances*, pages 3–29. Oxford University Press, Oxford, 2007.

[2] M. K. Kuhner, J. Yamato, and J. Felsenstein. Estimating effective population size and mutation rate from sequence data using Metropolis-Hastings sampling. *Genetics*, 140:1421–1430, 1995.