SSA: Inference of Maximum Likelihood Phylogenetic Trees Using a Stochastic Search Algorithm

Version 1.0

Laura Salter
Department of Mathematics and Statistics
University of New Mexico
Albuquerque, NM 87131
salter@stat.unm.edu

If you use this program in a publication, please cite the following reference:

Salter, L.A. and D.K. Pearl (2001) Stochastic Search Strategy for Estimation of Maximum Likelihood Phylogenetic Trees. *Systematic Biology* 50(1): 7-17.

About the Program

SSA is a program for inferring maximum likelihood phylogenies from DNA sequences. Two versions of the program are available: one which assumes a molecular clock and one which does not make this assumption. The method for searching the space of trees for the maximum likelihood (ML) tree is based on a simulated-annealing type algorithm and is described in the reference above. The program implements Felsenstein's F84 model of nucleotide substitution [1] and associated sub-models (i.e., JC69 [2] and K2P [3]). The program estimates the ML tree and branch lengths, and can optionally estimate the transversion/transversion ratio. Upon termination, the program returns the k trees of highest likelihood found during the search, where k can be set by the user.

Program Availability

SSA is written in ANSI C. Executables for SSA are available for free at www.stat.unm.edu/~salter/software/ssa/ssa.html for several operating systems. The programs have been successfully compiled using the Gnu C Compiler in Unix (Sun/Sparc) and Linux, and executables for these operating systems, as well as for the PC, are available. Anyone who has purchased the book Numerical Recipes in C by Press et al. can request the source code for the program from the author at salter@stat.unm.edu.

Downloading the Program

A zip file containing executables, an example, and documentation can be downloaded from www.stat.unm.edu/~salter/software/ssa/ssa.html. The executables can then be used directly after unzipping the file.

Using the Program

Input File

SSA requires DNA sequence data to be placed in an input file called "infile" in the directory from which the program will be run. The format of the input file is essentially PHYLIP format, with the exception that five additional variables are required to be specified on the first line of the file. These variables are used to aid in reading in the data when the data are in interleaved format. Thus the first line of the file consists of the following variables, separated by spaces:

```
nseq nsites interleaved? no_leaved length_leaved cont_length last_length
```

where

`nseq` = the number of sequences
`nsites` = the number of sites
`interleaved?` = 1 if the data are in sequential format, 2 if the data are in interleaved format
`no_leaved` = number of interleaved groups of data
`length_leaved` = total length of each interleaved section of data
`cont_length` = length of each contiguous group of data that is not blank separated
`last_length` = length of the last section of data

The data then appear following the first line. Each taxa must be given a name that is 10 characters or less, and may include any combination of characters, numbers, or symbols.

As an example, consider the data set below consisting of mtDNA with 231 sites for 5 primate species. The data is shown in sequential format.

Example data in sequential format:

```
5 231 1 0 0 0 0
Gibbon    ACTATACCCACCCAACTCGACCTACACCAATCCCCACATAGCACACAGACCAACAACCTCCCAC
CTTCCATACCAAGCCCCGACTTTACCGCCAACGCACCTCATCAAAACATACCTACAACACAAACAAATGCCCCC
CCACCCTCCTTCTTCAGCCCACTAGACCATCCTACCTTCCTAGCACGCCAAGCTCTCTACCATCAAACGCACAA
CTTACACATACAGAACCAC
Orangutan ACCCCACCCGTCTACACCAGCCAACACCAACCCCCACCTACTATACCAACCAATAACCTCTCAA
CCCCTAAACCAAACACTATCCCCAAAACCAACACACTCTACCAAAATACACCCCCAATTCACATCCGCACACCC
CCACCCCCCCCTGCCCAGTCCATCCCATCACCCTCTCCTCCCAACACCCTAAGCCACCTTCCTCAAAATCCAAAA
CCCACACAACCGAAACAAC
Gorilla   ACCCCATTTATCCATAAAAACCAACACCAACCCCCATCTAACACACAAACTAATGACCCCCCAC
CCTCAAAGCCAAACACCAACCCTATAATCAATACGCCTTATCAAAACACACCCCCAACATAAACCCACGCACCC
CCACCCCTTCCGCCCAGCTCACCACATCATCTCTCCCCTTCAACACCTCAATCCACCTCCCCCCAAATACACAA
TTCACACAAACAATACCAC
Chimp     ACCCCATCCACCCATACAAACCAACATTACCCTCCATCCAATATACAAACTAACAACCTCCCAC
TCTTCAGACCGAACACCAATCTCACAACCAACACGCCCCGTCAAAACACCCCTTCAGCACAAATTCATACACCC
CTACCTTTCCTACCCAGTTCACCACATCATCCCCCCCCTCTCAACATCTTGACTCGCCTCTCTCCAAACACACAA
TTCACGCAAACAACGCCAC
Human     ACCCCACTCACCCATACAAACCAACACCACTCTCCACCTAATATACAAATTAATAACCTCCCAC
CTTCAGAACTGAACGCCAATCTCATAACCAACACACCCCATCAAAGCACCCCTCCAACACAAACCCGCACACCT
CCACCCCCCCTCGTCTAGCTTACCACGTCATCCCTCCCTCTCAACACCTTAACTCACCTTCTCCCAAACGCACAA
TTCGCACACACAACGCCAC
```

Settings File

Settings for the program are specified in the "settings" file. All available user options are set in this file, which must be present in the directory from which the program is run. The parameters must be set in a specified order, which is given in the file included in the package and is also described on the next page.

Example settings file for trees with the molecular clock assumption:

```
model:              2
include_gaps:       0
burnin:             100
pburnin:            0.05
pbound_un:          0.05
cbound:             10
print_data_ind:     0
print_data_phylip:  0
cbeta:              0.25
abeta:              0.5
percent_mu:         0.25
num_saved_trees:    10
bound_total_iter:   500000
nj_lik:             -891.85
est_Kprime:         1
lin_rates:          0
P_accept:           0.90
delta_L_accept:     10.0
length_est:         0
num_boot:           100
seedj:              12345
seedk:              56789
```

We now discuss the various parameters.

`model`: Selects the model to be used in the estimation. 1 = HKY85, 2 = F84, 3 = JC69, 4 = K2P (Note: The F84 model should be used over the HKY85 model, as the HKY85 model is not fully supported).

`include_gaps`: Specifies whether or not sites with gaps in the alignment should be included in the analysis. 0=omit sites with gaps, 1=use sites with gaps

`burnin`: Number of iterations in the burnin period, which is used to estimate some parameters used in the search procedure. For most problems, the default value of 100 iterations should be adequate.

`pburnin`: To ensure adequate time during the burnin period, this parameter is used to determine the number of iterations needed to guarantee that the probability of not selecting a node for rearrangement is at most `pburnin`. The total number of iterations used in the burnin period is this number plus `burnin`.

`pbound_un`: This probability is used in setting the stopping rule for termination of the algorithm. It is used to determine the number of iterations needed to guarantee that the probability of not selecting a node for rearrangement is at most `pbound_un`. This is used to determine the value of x in [5].

`cbound`: This parameter is also used in setting the stopping rule for termination of the algorithm. It specified the number of iterations that must be exceeded since the proposal of an uninvestigated topology in order for the algorithm to terminate.

`print_data_ind`: Print the data to standard output in the format used internally in the program.

`print_data_phylip`: Print the data to standard output in PHYLIP format.

`cbeta`: Used to specify the rate of cooling. In particular, `cbeta` is the numerator of the parameter β, which controls the rate at which the probability of accepting a tree with a lower value of the log likelihood is accepted. It is the parameter c in [5].

`abeta`: Used to specify the rate of cooling. In particular, `abeta` specifies the precise linear combination of the number of taxa and the average log likelihood per site that are used in the denominator of the parameter β. It is the parameter a in [5].

`percent_mu`: Percentage change in log likelihood after which iterations in the search algorithm will be temporarily suspended so that parameters may be estimated for the current tree. For most problems, the default value of 0.25 should be adequate.

`num_saved_trees`: Number of trees retained by the algorithm. For example, if `num_saved_trees` is set to 10 (the default) then the 10 trees of highest likelihood encountered during the search procedure will be written to the output files.

`bound_total_iter`: Bound on the total number of iterations that the algorithm will perform. This option simply prevents the algorithm from running for an indefinite period of time, or may be used to terminate a search after a specified period of time. This bound should be set to some large number, generally several hundred thousand. In practice, the algorithm will generally terminate well before this bound is reached.

`nj_lik`: Value of the log likelihood from the neighbor-joining tree, or some reasonably "good" tree. This is used to control the rate of cooling in the stochastic search.

`est_Kprime`: Specifies whether the transition/transversion ratio should be estimated or held fixed at 2.0 during the tree estimation procedure. 0 = set the transition/transversion ratio to 2.0, 1 = estimate the transition/transversion ratio

`lin_rates`: (molecular clock version only) This option is not presently available, and should be set to 0.

`P_accept`: If the transition/transversion ratio will be estimated, this option specifies the probability of accepting a tree with a log likelihood that is lower by the amount `delta_L_accept` (see below) during a second, smaller stochastic search. This option has no effect if the transition/transversion ratio is fixed.

`delta_L_accept`: If the transition/transversion ratio will be estimated, this is the difference in log likelihood that will be accepted with probability `P_accept` (see above) during a second, smaller stochastic search. This option has no effect if the transition/transversion ratio is fixed.

`gamma_par`: (Only in non-molecular clock version) Parameter of the gamma distribution used in modifying the branch lengths of the proposal tree following local rearrangement. Each affected branch length is multiplied by a random observation from a gamma distribution with mean 1.0 and variance 1/`gamma_par`. The default value for `gamma_par` is 350.0.

`length_est`: Specifies whether optimal branch lengths should be estimated by SSA for the `num_saved_trees` best trees found by the algorithm. The algorithms in SSA that optimize the branch lengths are very slow, so an alternative to using SSA for the optimization is to run the file `paupfile.nex` in PAUP* to compute optimal branch lengths.

`num_boot`: Estimates of the variance of the transition/transversion ratio estimate may be obtained using a bootstrap procedure as described in [6]. If `num_boot` is > 0, the output file `paupfile.nex` will contain the required PAUP* commands to perform the bootstrap analysis. Output from PAUP* must then be summarized to obtain the variance estimate (i.e., the variance of the transition/transverion parameter estimates from the bootstrap replicates found in the file `lscores.boot` must be computed).

`seedj`: One of two random number seeds that must be set by the user.

`seedk`: One of two random number seeds that must be set by the user.

Output Files

The program writes several output files to the current directory upon completion of the algorithm. These are described below.

`results`: The results file contains summary information concerning the data and the progress of the algorithm. The first section of the results file summarizes the data. It gives the num-

ber of sites in the data set, the number of unique sites, information about whether or not gaps were included, base frequencies, the initial value of the instantaneous rate parameter (for rooted trees), and the initial value of the transition/transversion ratio.

The next section gives details of the algorithm's progress, including quantities such as the user parameters set to control the algorithm, the number of iterations used by the algorithm, and the number of trees accepted by the algorithm. Final estimates of parameters are also given. For rooted trees, the total length of the tree is initially set to 1.0, and the instantaneous rate parameter is estimated during the algorithm. Upon completion of the algorithm, the total length of the tree is recomputed based on the estimate of the instantaneous rate parameter, and all tree results are printed with this adjustment. This makes the tree parameterization the same as that used in PAUP* and PHYLIP. For unrooted trees, such an estimate is not needed. If the corresponding option is selected, the estimate of the transition/transversion rate and the transition/transversion parameter will also be given.

`besttree`: This file gives information concerning the k best trees found by the algorithm. The trees themselves are not printed here; rather, information such as the value of the maximized log likelihood, the value of the maximum likelihood estimate of the transition/transversion ratio (if estimated), the iteration at which the tree was first encountered, and the number of times the tree was encountered by the algorithm is given.

`treefile`: This file contains a list of the k best trees found by the algorithm in Newick format. Each tree is preceded by the estimate of the transition/transversion ratio in brackets. Additionally, the best tree found by the algorithm (which will be one of the trees in the list) is printed again at the bottom of the file. The `treefile` is useful for input into other programs, such as PAUP*, PHYLIP, and tree-drawing programs (the brackets and the transition/transversion ratio estimate preceding each tree will first need to be removed).

`paupfile.nex`: This is a Nexus-formatted file that is ready to be used in PAUP*. Specifically, it gives the data, the `num_saved_trees` best trees found by SSA, and the commands necessary to compute optimal branch lengths and parameter values with PAUP* using whatever substitution model was selected for use in SSA. Additionally, if `num_boot` > 0, the PAUP* commands required to perform a bootstrap analysis with `num_boot` samples are included so that estimates of the variance of the transition/transversion parameter estimate may be obtained. See [6] for details. The file can also be easily modified to run other analyses in PAUP*.

Examples of all four files for the example data set are given in the Appendix.

Running the Program

There are several steps involved in running the program, which are outlined below. Details of the implementation are given in the next section.

1. Use a program such as PAUP* or PHYLIP to obtain the log likelihood of the neighbor-

joining tree, or some other "reasonably good" tree. Modify the `nj_lik` option in the settings file to include this log likelihood. Change any other options of interest in the settings file, and set the two random number seeds.

2. Format the data as specified above and place them in a file called "infile" in the directory from which the program will be run.

3. Run the program (`ssa_mc` for the molecular clock version, or `ssa_nmc` for the non-molecular clock version).

4. Upon termination of the program, examine the contents of the three output files (results, besttree, treefile) to check that the program has completed successfully.

5. The k trees listed in file treefile should be optimized in PAUP* or PHYLIP to obtain optimal branch lengths. The branch length optimization routines implemented in SSA perform reasonably well, but slight increases in log likelihood values (generally less than 1.0) were sometimes obtained using the routines in PAUP*, and it is therefore recommended that users check results there.

Details of the Implementation

The details of the SSA algorithm are fully described in [5] and will only be briefly reviewed here so that implementation issues may be discussed. The algorithm is based on a simulated annealing algorithm which has been modified for use with phylogenies. The algorithm works by considering at each iteration a current phylogeny from the set of all possibilities phylogenies for *nseq* taxa. From each current phylogeny, a new phylogeny is proposed, by modifying the topology (branching pattern) of the tree and the lengths of the branches within that tree. Following generation of the proposal tree, the log likelihoods of the two trees (the current tree and the proposal tree) are compared. If the proposal tree has a higher log likelihood, then it will become the current tree for the next iteration. If the proposal tree has a lower log likelihood, then it becomes the current tree with probability proportional to the difference in log likelihood between the two trees. The idea behind the algorithm is that since a tree with a lower log likelihood always has some probability of being accepted by the algorithm, the search should be less likely to become trapped in locally optimal portions of the tree space than are existing uphill search strategies, such as those implemented in PAUP* and PHYLIP.

The probability of accepting a tree of lower log likelihood than the current tree is decreased as the algorithm proceeds, with the idea that eventually the search should settle on the optimal tree as moves to trees of lower log likelihood become less likely to be accepted. The manner in which this probability is lower is called the *cooling schedule* in the simulated annealing literature. The parameters controlling the rate of cooling are `abeta` and `cbeta`, which the user may specify in the settings file. `cbeta` must be a number between 0 and 1, where values closer to 0 represent slower cooling (increased search time but less chance of returning a locally optimal solution) and values closer to 1 result in more rapid cooling (shorter search time but trees found are more likely to be only locally optimal). `abeta` is also between 0 and 1, and specifies the relative weighting given to the number of sequences

and the average log likelihood per site in specifying the difficulty of the problem. Values closer to 0 put more weight on the number of sequences. The default values given in the settings files have been thoroughly tested and should be adequate for most problems, and thus the user will not generally need to change these setting.

The algorithm terminates when one of two conditions are satisfied: (1) a sufficient number of trees have been proposed from the current tree without any of them resulting in acceptance, or (2) the search is alternating between a collection of high-likelihood trees that are separated from one another by a single rearrangement, and a sufficient number of iterations have passed since any alternative trees have been accepted. The parameters which specify the exact conditions for termination of the algorithm are `pbound_un` and `cbound`. See [5] for details. These parameters should not need to be modified by most users.

The code used to implement the algorithm is elementary, and is not optimized for either speed or memory efficiency. However, in comparisons with PAUP* and PHYLIP, the performance of the algorithm has been favorable (see [5]). Random number generation is accomplished by inclusion of the "randlib" package. Branch length optimization is performed using routines from "Numerical Recipes in C" by Press et al. [4]. The PC executables were compiled using Visual C++ 6.0. Thanks to graduate student Wenbin Luo for providing these and some related necessary debugging. Unix/Linux versions have been tested with the Gnu C compiler.

Additionally, it is worth mentioning that the most time-consuming portion of the algorithm is the optimization of branch lengths for the k best trees (due to inefficient programming). Hence setting k to be relatively low will decrease the required run time. However, it is the author's philosophy that it is important to gain information not only about the maximum likelihood tree, but about other trees of high likelihood, since it is often the case that there will be many trees with likelihoods nearly as high as the ML tree. For this reason, it is not recommended to use k less than 5.

While the stochastic nature of SSA should help to avoid many of the difficulties associated with locally optimal trees, there is no guarantee that the ML tree found on any particular run of the algorithm is the true globally optimal tree. Thus it is advised that the program be run several times with different random number seeds provided by the user in the settings file, and that the estimate of the ML tree be taken as the tree of maximum likelihood found in any of these runs.

I would greatly appreciate hearing about any successes and/or bugs associated with use of the program. I can't promise that I will be able to respond quickly to reported bugs. However, please e-mail me the files `infile`, `results`, `besttree`, and `treefile`, as well as any error messages you get from the program, for the run in which the problem occurred, and I will eventually respond to your query. Please e-mail all comments to salter@stat.unm.edu.

Acknowledgments

The majority of this work was completed under the direction of Professor Dennis Pearl at The Ohio State University as part of the completion of my Ph.D. research. Two students at the University of New Mexico, James Degnan and Wenbin Luo, subsequently contributed to the development of this software. James Degnan contributed to the development of the program for non-molecular clock trees, and was supported by an RAC grant from the University of New Mexico. Wenbin Luo developed the PC executables, and assisted with testing and debugging of both the molecular clock and non-molecular clock versions.

Appendix

Example: Estimation of the ML tree under the assumption of a molecular clock for 5 primate mtDNA sequences.

`infile` used in the example:

```
5 231 1 0 0 0 0
Gibbon    ACTATACCCACCCAACTCGACCTACACCAATCCCCACATAGCACACAGACCAACAACCTCCCAC
CTTCCATACCAAGCCCCGACTTTACCGCCAACGCACCTCATCAAAACATACCTACAACACAAACAAATGCCCCC
CCACCCTCCTTCTTCAGCCCACTAGACCATCCTACCTTCCTAGCACGCCAAGCTCTCTACCATCAAACGCACAA
CTTACACATACAGAACCAC
Orangutan ACCCCACCCGTCTACACCAGCCAACACCAACCCCCACCTACTATACCAACCAATAACCTCTCAA
CCCCTAAACCAAACACTATCCCCAAAACCAACACACTCTACCAAAATACACCCCCAATTCACATCCGCACACCC
CCACCCCCCCCTGCCCAGTCCATCCCATCACCCTCTCCTCCCAACACCCTAAGCCACCTTCCTCAAAATCCAAAA
CCCACACAACCGAAACAAC
Gorilla   ACCCCATTTATCCATAAAAACCAACACCAACCCCCATCTAACACACAAACTAATGACCCCCCAC
CCTCAAAGCCAAACACCAACCCTATAATCAATACGCCTTATCAAAACACACCCCCAACATAAACCCACGCACCC
CCACCCCTTCCGCCCAGCTCACCACATCATCTCTCCCCTTCAACACCTCAATCCACCTCCCCCCAAATACACAA
TTCACACAAACAATACCAC
Chimp     ACCCCATCCACCCATACAAACCAACATTACCCTCCATCCAATATACAAACTAACAACCTCCCAC
TCTTCAGACCGAACACCAATCTCACAACCAACACGCCCCGTCAAAACACCCCTTCAGCACAAATTCATACACCC
CTACCTTTCCTACCCAGTTCACCACATCATCCCCCCCTCTCAACATCTTGACTCGCCTCTCTCCAAACACACAA
TTCACGCAAACAACGCCAC
Human     ACCCCACTCACCCATACAAACCAACACCACTCTCCACCTAATATACAAATTAATAACCTCCCAC
CTTCAGAACTGAACGCCAATCTCATAACCAACACACCCCATCAAAGCACCCCTCCAACACAAACCCGCACACCT
CCACCCCCCTCGTCTAGCTTACCACGTCATCCCTCCCTCTCAACACCTTAACTCACCTTCTCCCAAACGCACAA
TTCGCACACACAACGCCAC
```

`settings` file used in the example:

```
model:              2
include_gaps:       0
burnin:             100
pburnin:            0.05
pbound_un:          0.05
cbound:             10
print_data_ind:     0
print_data_phylip:  0
cbeta:              0.25
abeta:              0.5
percent_mu:         0.25
num_saved_trees:    10
bound_total_iter:   500000
nj_lik:             -891.85
est_Kprime:         1
lin_rates:          0
P_accept:           0.90
delta_L_accept:     10.0
length_est:         0
num_boot:           100
seedj:              12345
seedk:              56789
```

`results` file output by the program:

```
MAXIMUM LIKELIHOOD ESTIMATION USING SIMULATED ANNEALING
      UNDER THE ASSUMPTION OF A MOLECULAR CLOCK

-----------------------------------------------------------------

The number of unique site patterns is 62

Sites with a gap in at least one taxa have been excluded - the
 total number of sites used in the computations is 231

Empirical Base Frequencies:

   A   0.337662
   G   0.042424
   C   0.465801
   T   0.154113

The F84 Model will be used

Transition/transversion ratio = 2.000000

Transition/transversion parameter = 2.509181

The scaling factor for the branch lengths is 0.707351

The likelihood of the initial tree is -1024.100910

-----------------------------------------------------------------


Information about the simulated annealing procedure:

 The likelihood at end of burnin is -1067.412451

 Total iterations in first anneal is 1050,
  total iterations in second anneal is 572

 Time to perform annealing is 39.810000

 Time to perform tree optimization is 88.700000
```

(continued on next page)

(continued from previous page)

```
 The number of iterations during the burnin period is 115

 The number of iterations after last unique tree proposed is
  at least 50

 The number of trials within each tree must be at least 45

 The value of beta is 0.056428

 The estimated value of co is 100.267913

 The number of iterations was 1622

 The final value of ci was 3.013137

 The overall number of topology changes was 564

 The total number of trees accepted was 1009

 The final estimate of mu is 0.104225

 The estimate of mu associated with the best tree is 0.151537

 The estimate of the transition/transversion parameter associated
  with the best tree is 54.465444

 The estimate of the transition/transversion ratio associated
  with the best trees is 35.845467

 The final estimate of the transition/transversion parameter is
  10.925629

 The corresponding estimate of the transition/transversion ratio is
  7.482662

 The scaled estimate of mu depending on the transition/transversion
  ratio is 0.250164

 The maximum unconstrained likelihood is -773.651103

===============================================================
```

`besttree` file output by the program (only the first 5 trees are shown - for this example 10 total trees would be printed in the file):

```
MAXIMUM LIKELIHOOD ESTIMATION USING SIMULATED ANNEALING
      UNDER THE ASSUMPTION OF A MOLECULAR CLOCK

The 10 trees with the best likelihoods were:


Tree Number 1

First hit:  681
Max ln L for tree:  -862.066831
Transition/transversion ratio:  22.454550
Number of times hit:  26
Hit in first annealing pass:  1


Tree Number 2

First hit:  399
Max ln L for tree:  -866.167269
Transition/transversion ratio:  21.639693
Number of times hit:  30
Hit in first annealing pass:  1


Tree Number 3

First hit:  683
Max ln L for tree:  -862.504565
Transition/transversion ratio:  10.601938
Number of times hit:  125
Hit in first annealing pass:  1


Tree Number 4

First hit:  343
Max ln L for tree:  -866.393094
Transition/transversion ratio:  9.892350
Number of times hit:  69
Hit in first annealing pass:  1


Tree Number 5

First hit:  346
Max ln L for tree:  -871.101365
Transition/transversion ratio:  7.365176
Number of times hit:  39
Hit in first annealing pass:  1
```

`treefile` output by the program:

```
[22.454550] ((Orangutan,Gibbon),(Gorilla,(Chimp,Human)));

[21.639693] ((Chimp,(Human,Gorilla)),(Gibbon,Orangutan));

[10.601938] (Orangutan,(Gibbon,(Gorilla,(Human,Chimp))));

[9.892350] (Orangutan,(Gibbon,(Chimp,(Gorilla,Human))));

[7.365176] (Orangutan,(Gibbon,(Human,(Chimp,Gorilla))));

[12.145771] (Gibbon,(Orangutan,(Gorilla,(Human,Chimp))));

[11.022469] (Gibbon,(Orangutan,(Chimp,(Gorilla,Human))));

[7.347838] (Gibbon,(Orangutan,(Human,(Gorilla,Chimp))));

[11.958220] (Gibbon,(Orangutan,(Chimp,(Gorilla,Human))));

[11.069885] (Gibbon,((Chimp,Human),(Orangutan,Gorilla)));

The best tree and branch lengths found by the algorithm:

(Gibbon:1.000000,(Orangutan:0.556750,(Gorilla:0.276440,
(Human:0.184168,Chimp:0.184168):0.092272):0.280310):0.443250);
```

`paupfile.nex` output by the program:

```
#NEXUS

  Begin data;
  Dimensions ntax=5 nchar=231;
  Format datatype=nucleotide gap=- missing=? matchchar=.;
  Matrix
Gibbon    ACTATACCCACCCAACTCGACCTACACCAATCCCCACATAGCACACAGACCAACAACCTCCCAC
CTTCCATACCAAGCCCCGACTTTACCGCCAACGCACCTCATCAAAACATACCTACAACACAAACAAATGCCCCC
CCACCCTCCTTCTTCAGCCCACTAGACCATCCTACCTTCCTAGCACGCCAAGCTCTCTACCATCAAACGCACAA
CTTACACATACAGAACCAC
Orangutan ACCCCACCCGTCTACACCAGCCAACACCAACCCCCACCTACTATACCAACCAATAACCTCTCAA
CCCCTAAACCAAACACTATCCCCAAAACCAACACACTCTACCAAAATACACCCCCAATTCACATCCGCACACCC
CCACCCCCCCCTGCCCAGTCCATCCCATCACCCTCTCCTCCCAACACCCTAAGCCACCTTCCTCAAAATCCAAAA
CCCACACAACCGAAACAAC
Gorilla   ACCCCATTTATCCATAAAAACCAACACCAACCCCCATCTAACACACAAACTAATGACCCCCCAC
CCTCAAAGCCAAACACCAACCCTATAATCAATACGCCTTATCAAAACACACCCCCAACATAAACCCACGCACCC
CCACCCCTTCCGCCCAGCTCACCACATCATCTCTCCCCTTCAACACCTCAATCCACCTCCCCCCAAATACACAA
TTCACACAAACAATACCAC
Chimp     ACCCCATCCACCCATACAAACCAACATTACCCTCCATCCAATATACAAACTAACAACCTCCCAC
TCTTCAGACCGAACACCAATCTCACAACCAACACGCCCCGTCAAAACACCCCTTCAGCACAAATTCATACACCC
CTACCTTTCCTACCCAGTTCACCACATCATCCCCCCCTCTCAACATCTTGACTCGCCTCTCTCCAAACACACAA
TTCACGCAAACAACGCCAC
Human     ACCCCACTCACCCATACAAACCAACACCACTCTCCACCTAATATACAAATTAATAACCTCCCAC
CTTCAGAACTGAACGCCAATCTCATAACCAACACACCCCATCAAAGCACCCCTCCAACACAAACCCGCACACCT
CCACCCCCCTCGTCTAGCTTACCACGTCATCCCTCCCTCTCAACACCTTAACTCACCTTCTCCCAAACGCACAA
TTCGCACACACAACGCCAC
;
 End;

Begin trees;

tree mytree=[&R]((Orangutan,Gibbon),(Gorilla,(Chimp,Human)));

tree mytree=[&R]((Chimp,(Human,Gorilla)),(Gibbon,Orangutan));

tree mytree=[&R](Orangutan,(Gibbon,(Gorilla,(Human,Chimp))));

tree mytree=[&R](Orangutan,(Gibbon,(Chimp,(Gorilla,Human))));

tree mytree=[&R](Orangutan,(Gibbon,(Human,(Chimp,Gorilla))));
```

(continued on next page)

(continued from previous page)

```
tree mytree=[&R](Gibbon,(Orangutan,(Gorilla,(Human,Chimp))));

tree mytree=[&R](Gibbon,(Orangutan,(Chimp,(Gorilla,Human))));

tree mytree=[&R](Gibbon,(Orangutan,(Human,(Gorilla,Chimp))));

tree mytree=[&R](Gibbon,(Orangutan,(Chimp,(Gorilla,Human))));

tree mytree=[&R](Gibbon,((Chimp,Human),(Orangutan,Gorilla)));

End;

Begin paup;
set criterion=likelihood;
lset nst=2 basefreq=empirical tratio=estimate variant=F84 clock=yes;
lscores;
End;
```

References

[1] J. Felsenstein. Distance methods for inferring phylogenies: A justification. *Evolution*, 38:16–24, 1984.

[2] T. H. Jukes and C. R. Cantor. Evolution of protein molecules. In H. N. Munro, editor, *Mammalian Protein Metabolism*, pages 21–132. Academic Press, New York, 1969.

[3] M. Kimura. A simple method for estimating evolutionary rate of base substitutions through comparative studies of nucleotide sequences. *J Mol Evol*, 16:111–120, 1980.

[4] W. H. Press, S. A. Teukolsky, W. T. Vetterling, and B. P. Flannery. *Numerical Recipes in C*. Cambride University Press, Second edition, 1992.

[5] L. Salter and D. Pearl. Stochastic search strategy for estimation of maximum likelihood phylogenetic trees. *Syst Biol*, 50(1):7–17, 2001.

[6] Q. Wang, L. Salter, and D. Pearl. Estimation of evolutionary parameters with phylogenetic trees. *J Mol Evol, in revision*, 2002.