Phylogenetic Analysis by Maximum Likelihood (PAML)

Version 1.2, November 1996

Ziheng Yang

 The package can be obtained by anonymous ftp (binary) from ftp.bio.indiana.edu (129.79.225.25) under directory Incoming/ or molbio/evolve. DOS and MAC executables may also be available. Comments, suggestions, and complaints are appreciated. If you wish to receive update notice, please send me an email message.

Suggested citation:

Yang, Z. 1996. Phylogenetic Analysis by Maximum Likelihood (PAML), Version 1.2. Department of Integrative Biology, University of California at Berkeley.

The author can be reached at

Ziheng Yang

Department of Integrative Biology

University of California

Berkeley, CA 94720-3140

Fax: (510) 643 6264

Email: `ziheng@mws4.biol.berkeley.edu`

`http://mw511.biol.berkeley.edu/ziheng/ziheng.html`

Table of Contents

0 What Is New?

1. I have done some changes with the user interface. Name of files (sequence data file, control file, tree structure file) are now specified in the control files (The default are `baseml.ctl`, `codeml.ctl`, `pamp.ctl`, `mcmctree.ctl`). The command line for executing a program is now *ProgramName ControlFileName* with *ControlFileName* optional. I have added some text in the output file of `baseml` so that the results are eassier to understand. Choose `verbose=0` in the control file if you do not want the detailed output.
2. A new program `mcmctree` is included, which performs Bayesian estimation of phylogenies using Markov chain Monte Carlo methods.
3. A number of minor changes are made, such as using species names in the parenthesis tree representation.

1 General Description

This package consists of several programs for phylogenetic analyses of homologous molecular sequence data using the method of maximum likelihood (ML). The programs are written in ANSI C and should work with any compatible compiler. The programs can be used for (1) estimation of branch lengths in a phylogenetic tree and as well as other parameters in the evolutionary model such as the transition/transversion rate ratio, the shape parameter of the gamma distribution for variable substitution rates at sites, and rate parameters for different genes; (2) test of hypotheses concerning sequence evolution, such as rate constancy and independence among sites and rate constancy among lineages (the molecular clock); (3) calculation of substitution rates at sites and reconstruction of ancestral nucleotide or amino acid sequences, and (4) phylogenetic tree reconstruction. Tree making is not a strong point of the programs.

Two programs, `baseml` and `basemlg`, are for analyzing nucleotide sequences. A third program, `codeml`, is formed by merging two old programs: `codonml` which implements the codon-based models of Goldman and Yang (1994a) for protein-coding DNA sequences and `aaml` which implements models for amino acid sequences. These two are now distinguished by a variable named `seqtype` in `codeml`, which is 1 for codon sequences and 2 for amino acid sequences. In this document I use `codonml` and `aaml` to mean `codeml` with `seqtype` = 1 and 2, respectively. The programs `baseml`, `codonml`, and `aaml` use similar algorithms to fit models, the difference being that the unit of evolution in the substitution model, referred to as a "site" in the sequence, is a nucleotide, a codon, or an amino acid for the three programs, respectively. Markov process models are used to describe substitutions between nucleotides, codons or amino acids, with substitution rates assumed to be either constant or variable among sites. A *discrete-gamma* model (Yang, 1994c) is used in `baseml`, `codonml` and `aaml` to accommodate rate variation among sites, by which rates for sites come from four or eight categories used to approximate the continuous gamma distribution. When rates are variable at sites, a correlation of rates between adjacent sites can be accounted for using an *auto-discrete-gamma* model (Yang, 1995). The program `basemlg` fits the model of (continuous) gamma rates for sites (Yang, 1993). This algorithm is extremely slow and is useful for data of four to six sequences only. The details of the models are described in §6, and an example data set is analyzed in §9.

General assumptions of the models (programs):

(1) Substitutions occur independently in different lineages;

(2) Substitutions occur independently among sites (except for the auto-discrete-gamma model which account for the correlation of substitution rates at neighboring sites);

(3) The process of substitution is described by a time-homogeneous Markov process. Further restrictions may be placed on the structure of the *rate matrix* of the process and lead to different substitution models;

(4) The process of substitution is stationary. In other words, the frequencies of nucleotides (`baseml` and `basemlg`), codons (`codonml`), or amino acids (`aaml`) have remained constant over the time period covered by the data.

The existence of a molecular clock (rate constancy among lineages) is not necessary but can be imposed. Variation (and dependence) of rates at sites is allowed by the discrete-gamma (or auto-discrete-gamma) models implemented in `baseml`, `codonml` and `aaml`. The (independent) continuous gamma model is implemented in `basemlg` only.

The sequences must be aligned, and sites involving gaps will be removed from all sequences before analysis, with appropriate adjustment to the sequence length. Note that all sites in the sequence should be included in the analysis, including the "invariant" or "non-informative" sites.

A summary of the types of analyses performed by different programs in the package is given below.

Program	Functions
`baseml`	ML analysis of nucleotide sequences: estimation of tree topology, branch lengths, and substitution parameters under a variety of nucleotide substitution models (JC69, K80, F81, F84, HKY85, TN96, REV); constant or gamma rates for sites; molecular clock (rate constancy among lineages) or no clock, among-gene and within-gene variation of substitution rates; models for combined analyses of multiple gene data; calculation of substitution rates at sites; reconstruction of ancestral nucleotides.
`basemlg`	ML analysis with a continuous gamma distribution of rates among sites, under a variety of substitution models (JC69, K80, F81, F84 and HKY85).
`codonml` (codeml with `seqtype=1`)	ML analysis of protein-coding DNA sequences using the codon-based model of Goldman and Yang (1994); calculation of the codon-usage table; estimation of synonymous and nonsynonymous substitution rates.
`aaml` (`codeml` with `seqtype =2`)	ML analysis of amino acid sequences under several amino acid substitution models (Poisson, Proportional, Dayhoff, Jones *et al*., Goldman and Yang, REV); constant or gamma rates for sites; molecular clock (rate constancy among lineages) or no clock, among-gene and within-gene variation of substitution rates; models for combined analyses of multiple gene data; calculation of substitution rates at sites;

	reconstruction of ancestral amino acid sequences.
`pamp`	Parsimony-based analyses for a given tree topology: estimation of the substitution pattern by the method of Yang and Kumar (1996); estimation of the gamma parameter for variable rates across sites by the method of moments, the method of Sullivan *et al.* (1995), and the method of Yang and Kumar (1996); reconstruction of ancestral character states using the algorithm of Hartigan (1973) and an improved parsimony method.
`mcmctree`	Bayesian estimation of phylogenies using DNA sequence data (Rannala and Yang, 1996; Yang and Rannala, submitted). Markov chain Monte Carlo method for calculating posterior probabilities of trees (labeled histories).
`listtree`	This does miscellaneous things, such as listing all rooted and unrooted trees for a given number of species, generating random trees with branch lengths, and calculating tree bipartition distances. Later I will include more things in this program such as simulating data, and also change the program name.

2 List of Files

The following files are included in the package:

File	Descriptions
`baseml.c`	various models for nucleotide sequences
`codeml.c`	models for codon (seqtype = 1) and amino acid (seqtype = 2) sequences
`basemlg.c`	models of (continuous) gamma rates for sites for nucleotide sequences
`pamp.c`	parsimony analyses of nucleotide or amino acid sequences
`mcmctree.c`	Markov chain Monte Carlo algorithm for Bayes estimation of phylogenies
`listtree.c`	small gadgets
`treesub.c`	a few functions
`treespace.c`	a few functions
`tools.c`	my toolkit
`tools.h`	header file
`baseml.ctl`	control file for running baseml and basemlg
`codeml.ctl`	control file for codeml (i.e., codonml and aaml)
`pamp.ctl`	control file for pamp
`mcmctree.ctl`	control file for mcmctree
`grantham.dat`	amino acid distance matrix (Grantham 1974), used by codonml
`dayhoff.dat`	Dayhoff et al. (1978)
`jones.dat`	Jones et al. (1978)
`trees.4s`	tree structure file for 4-sequence data
`trees.5s`	tree structure file for 5-sequence data
`brown.nuc`	the 895-bp mtDNA data of Brown et al. (1982)
`mtpriom9.nuc`	the extended data set of Brown et al. (1982) with 9 primates, 888 sites
`abglobin.nuc`	the concatenated α- and β-globin genes, example data for condonml

`abglobin.aa`	the concatenated α- and β-globins, translated from abglobin.nuc
`paml.gcc`	batch file for compilation using the GNU gcc compiler
`paml.acc`	batch file for compilation using the SUN acc compiler
`paml1_2.doc`	this document (Microsoft Word 6 document)

The programs, when executed, produce result files, which are not listed. Some other files with names `rub`, `lfh`, `rst1`, or `InitVal` may also be created, which can be ignored.

3 Using the Programs

3.1 Compilation

The UNIX commands for compiling the programs using the GNU `gcc` compiler are as follows and are listed in a file named `paml.gcc`.

```
gcc -c -O2 tools.c eigen.c
gcc -o baseml -O2 baseml.c tools.o eigen.o
gcc -o basemlg -O2 basemlg.c tools.o
gcc -o codeml -O2 codeml.c tools.o eigen.o
gcc -o pamp -O2 pamp.c tools.o eigen.o
gcc -o mcmctree -O2 mcmctree.c tools.o
gcc -o listtree -O2 listtree.c tools.o
```

Type `source paml.gcc` at the UNIX command prompt to compile. If you have the SUN ANSI C compiler acc, you should type `source paml.acc` for compilation. Some compilers do not link to the math library by default; in such case, the option `-lm` should be added (Insert `-lm` before `-o` in each of the above lines). If you want to run the programs using other ANSI C compilers or operating systems, you should get help from your local C experts. Make sure that optimization is turned on during the compilation.

3.2 Execution

You need to prepare a sequence data file (*e.g.*, `brown.nuc`) and modify the options in the appropriate control file (the defaults are `baseml.ctl` for `baseml` or `basemlg`, `codeml.ctl` for `codeml`, `mcmctree.ctl` for `mcmctree`). If you have chosen `runmode` = 0 or 1 in the control file, which means that the tree topologies are specified, you also need to prepare a *tree structure file* (*e.g.*, `trees.4s`). The sequence data format and the control files are explained in §4. The syntax for running the programs is

ProgramName

or

ProgramName ControlFileName

3.3 Guidelines for running `basemlg`

Because of the intensive computation requierd by `basemlg`, the discrete-gamma model implemented in `baseml` is recommended for data analysis. If you choose to use `basemlg`, you should run `baseml` first, and then run `basemlg`, This allows `baseml` to collect initial values, into a file named `InitVal`, for use by `basemlg`. Note that `basemlg` can fit only a subset of models that `baseml` can (see §4.3 for details).

The iteration process, *i.e.*, the minimization of the negative log-likelihood, is recorded in `rub`. For each round of iteration, the size (norm) of the gradient or search direction (h), the negative log likelihood (`-l`), and the current values of parameters are listed. A healthy iteration is indicated by the decrease of both h and `-l`, and h is particularly sensitive. This file should be checked to ensure the convergence of the algorithm. A typical symptom of failure of the algorithm is that estimates of parameters are at the preset boundaries, with values like 2.00000, 5.00000.

4 Input File Format

4.1 Sequence data format

The format is similar to, but less stringent than, the format used in PHYLIP (Felsenstein, 1993). For example, the following data set has 4 sequences, each of 60 nucleotides.

```
4  60
sequence 1
AAGCTTCACCGGCGCAGTCATTCTCATAATCGCCCACGGACTTACATCCTCATTACTATT
sequence 2
AAGCTTCACCGGCGCAATTATCCTCATAATCGCCCACGGACTTACATCCTCATTATTATT
sequence 3
AAGCTTCACCGGCGCAGTTGTTCTTATAATTGCCCACGGACTTACATCATCATTATTATT
sequence 4
AAGCTTCACCGGCGCAACCACCCTCATGATTGCCCATGGACTCACATCCTCCCTACTGTT
```

The first line contains the number of species and the sequence length (possibly followed by option characters). With `codonml` (`codeml` with `seqtype` = 1), the sequence length in the sequence file refers to the number of nucleotides rather than the number of codons. The only options allowed in the sequence file are I, S or G. The sequences may be in either *interleaved* format (option I, example data file `abglobin.nuc`), or *sequential* format (option S, example data file `brown.nuc`). The default option is S. The option G is used for combined analyses of multiple data sets such as those of multiple genes (see below). Comments may be placed at the end of the sequence file and will be ignored.

The maximum number of characters in a species name is, by default, 30. This may be changed (see section 8.1). Spaces are allowed in a name. In a sequence, three special characters ".", "-", and "?" may be used: a dot means the same character as in the first sequence, a dash means an alignment gap, and a question mark means an undetermined site. Sites at which at least one sequence involves a "-" or "?"are excluded from all sequences before analysis, with the sequence length adjusted. For codon sequences, the whole codon is removed. Characters T, C, A, G, U, t, c, a, g, u are recognized as nucleotides (for `baseml`, `basemlg` and `codonml`), while the standard one-letter codes (A, R, N, D, C, Q, E, G, H, I, L, K, M, F, P, S, T, W, Y, V or their lowercase equivalents) are used for amino acids. Other alphabetic characters cause errors. Non-alphabetic symbols such as ><!"£$%^0123456789 are simply ignored and can be freely used as landmarks. Lines do not have to be equally long and you can put the whole sequence on one line.

Option G: This option is for combined analyses of heterogeneous data sets such as data of multiple genes or data of the three codon positions. The sequences must be concatenated and the option is used to specify which gene or codon position each site is from. There are two data formats with this option. The first is illustrated by an excerpt of a sequence file listed below. The example data of Brown *et al.* (1982) are an 895-bp segment from the mitochondrial genome, which codes for parts of two proteins (ND4 and ND5) at the two ends and three tRNAs in the middle. Sites in the sequence fall naturally into 4 classes: the three codon positions and the tRNA coding region. The first line of the file contains the option character G. The second line begins with G and is followed by the number of site classes. The following lines contain the site marks, one for each site in the sequence (or each codon in the case of `codonml`). The site mark specifies which class each site is from. If there are g classes, the marks should be 1, 2, ..., g, and if g > 9, the marks need to be separated by spaces. The total number of marks must be equal to the total number of sites in each sequence.

```
   5   895    G
G   4
3
123123123123123123123123123123123123123123123123123123123123
123123123123123123123123123123123123123123123123123123123123
123123123123123123123123123123123123123123123123123123123123
123123123123123123123123123123123123123123123123123123123123
123123123123123123123123123123123123123123123123123123123123
123123123123123123123123123123123123123123123123123123123123
123123123123123123123123123123123123123123123123123123123123
1231231231231231231231231231231231231
444444444444444444444444444444444444444444444444444444444444
444444444444444444444444444444444444444444444444444444444444
444444444444444444444444444444444444444444444444444444444444
444444444444444444
123123123123123123123123123123123123123123123123123123123123
123123123123123123123123123123123123123123123123123123123123
123123123123123123123123123123123123123123123123123123123123
12312312312312312312312312312312312312312312312312312312312
Human
AAGCTTCACCGGCGCAGTCATTCTCATAATCGCCCACGGACTTACATCCTCATTACTATT
CTGCCTAGCAAACTCAAACTACGAACGCACTCACAGTCGCATCATAATC........
Chimpanzee
.........
```

If the data are concatenated sequences of multiple genes, a simpler format, shown below for an example data set, may be used. This sequence has 1000 nucleotides from 4 genes, obtained from concatenating four genes with 100, 200, 300, and 400 nucleotides from genes 1, 2, 3, and 4, respectively. The "lengths" for the genes must be on the line that starts with G, *i.e.*, on the second line of the sequence file. (This requirement allows the program to determine which of the two formats is being used.) The sum of the lengths for the genes should be equal to the number of nucleotides, amino acids, or codons in the combined sequence for `baseml` (or `basemlg`), `aaml`, and `codonml`, respectively.

```
5   1000   G
G    4    100 200 300 400
Sequence 1
TCGATAGATAGGTTTTAGGGGGGGGGGTAAAAAAAAA.......
```

4.2 The tree structure file and representations of the tree topology

A file which specifies tree topologies is used when `runmode` = 0 or 1. The file name is specified in the appropriate control file. By default, the parenthesis representation of the tree is used, with the numbering of species corresponding to the order in the sequence data file. If the species names are used, these have to match exactly those in the sequence data file (including spaces or strange characters in the names). Branch lengths (if any) are ignored. The following is a possible `trees.4s` file.

```
4   4          // 4 species, 3 trees
(1234)         // the star tree
((12)34)       // species 1 and 2 are clustered together
((1,2),3,4)    // Commas are needed with more than 9 species
((human,chimpanzee),gorilla,orangutan)
((human:.1,chimpanzee:.2):.05,gorilla:.3,orangutan:.5)
```

Another way of representing the tree topology is by means of its branches and is mainly used in the result files for outputting the estimated branch lengths (see §5). For example, the tree ((12)34) is specified by its 5 branches:

$$5 \to 6,\ 6 \to 1,\ 6 \to 2,\ 5 \to 3,\ 5 \to 4$$

The nodes in a tree are marked with consecutive natural numbers, with 1, 2, ..., s representing the s known sequences, in the same order as in the data. A number larger than s means an interior node, at which the sequence is unknown. This representation saves space in the output in comparison with a tree diagram but may make it difficult to discern the topology when there are many species. In case some sequences in the data are ancestral to some others, this method is convenient. To use this format in the tree structure file, begin the line with $, followed by the number of branches, and then the branches as specified by the end nodes. For example, the third tree in the following `trees.5s` file has 4 branches, with taxon 5 to be the common ancestor of taxa 1, 2, 3, and 4:

```
5   3
(((12)3)45)
$ 4    5 1  5 2  5 3  5 4
(((13)2)45)
```

Whether the tree will be considered rooted or unrooted depends on whether a molecular clock is assumed. Without the clock, the trees are unrooted, and so ((12)34) is the same as (12(34)); with the clock, the trees are rooted and these two trees are different and both are different from (((12)3)4).

4.3 The control file for programs `baseml` *and* `basemlg`

The default name of the control file for `baseml` and `basemlg` is `baseml.ctl`, and an example is shown below. Note that spaces are required on both sides of the equal sign, and blank lines or lines beginning with '*' are treated as comments.

```
      seqfile  = brown.nuc  * sequence data file name
      outfile  = mlb        * main result file name
     treefile  = trees.5s   * tree structure file name
        noisy  = 2  * 0,1,2,3: how much rubbish on the screen
      verbose  = 1  * 1: detailed output, 0: concise output
      runmode  = 0  * 0:user tree; 1:semi-automatic; 2:automatic tree search
        model  = 3  * 0:JC69, 1:K80, 2:F81, 3:F84, 4:HKY85, 5:TN93, 6:REV
        Mgene  = 0  * 0:rates, 1:separate; 2:pi, 3:kappa, 4:all

  given_kappa  = 0
        kappa  = 2. * initial or given kappa
```

```
    given_alpha   = 1
          alpha   = 0  * initial or given alpha, 0:infinity for a single rate
         Malpha   = 0  * 1: different alpha's for genes, 0: one alpha
          ncatG   = 8  * No. categories in the dG, AdG, or nparK models of rates

      given_rho   = 1
            rho   = 0 * initial or given rho, 0:no correlation
          nparK   = 0 * nonparametric models of variable rates for sites
          clock   = 0  * 0:no clock, unrooted tree, 1:clock, rooted tree
          nhomo   = 0  * 0 & 1:homogeneous, 2:N1, 3:N2
          getSE   = 0  * S.E.s of parameter estimates (1:yes, 0:no)
   RateAncestor   = 0* (1/0) calculate rates or ancestral nucleotides
```

The descriptions below refer to the program `baseml`, with the differences of `basemlg` discussed afterwards.

`seqfile`, `outfile`, and `treefile`, specifies the names of the sequence data file, main result file, and the tree structure file, respectively.

`noisy` controls how much output you want on the screen. If the model being fitted involves much computation (such as in the case of `basemlg` or `codonml`), you can choose a large number for `noisy` to avoid loneliness. `verbose` controls how much output in the result file.

Three run modes are possible: `runmode` = 0 means evaluation of the tree topologies specified in the tree structure file only. With `runmode` = 2, an algorithm is used for heuristic tree search. With `runmode` = 1, the program will read a multifurcating tree from the tree structure file and try to estimate the best bifurcating tree compatible with it.

`model` specifies the model of nucleotide substitution (§6.1).

`Mgene` is used with the option G in the data file, and is explained in §6.5.

`given_kappa` specifies whether κ in K80, F84, or HKY85 is given at a fixed value or is to be estimated by iteration from the data. If `given_kappa` = 1, the value of another variable, `kappa`, is the given value, and otherwise the value of `kappa` is used as the initial estimate for iteration. The variables `given_kappa` and `kappa` have no effect with JC69 or F81 which does not involve such a parameter, or with TN93 and REV which have two and five rate parameters respectively and all of them are estimated from the data.

`given_alpha` and `alpha` work in the same way, where `alpha` refers to the shape parameter α of the gamma distribution for variable substitution rates across sites. The model of a single rate for all sites is specified as `given_alpha` = 1 and `alpha` = 0 (0 means infinity), while the (discrete-) gamma model is specified by a positive value for `alpha`, and `ncatG` is then the number of categories for the discrete-gamma model (`baseml`).

`given_rho` and `rho` work in a similar way and concern independence or correlation of rates at adjacent sites, where ρ (rho) is the correlation parameter of the auto-discrete-gamma model. The model of independent rates for sites is specified as `given_rho` = 1 and `rho` = 0; choosing `alpha` = 0 further means a constant rate for all sites. The auto-discrete-gamma model is specified by positive values for both `alpha` and `rho`. The model of a constant rate for sites is a special case of the (discrete) gamma model with $\alpha = \infty$ (`alpha` = 0), and the model of independent rates for sites is a special case of the auto-discrete-gamma model with $\rho = 0$ (`rho` = 0).

`nparK` specifies nonparametric models for variable and Markov-dependent rates across sites: `nparK` = 1 or 2 means several (`ncatG`) categories of independent rates for sites, while `nparK` = 3 or 4 means the rates are Markov-dependent at adjacent sites; `nparK` = 1 and 3 have the restriction that each rate category has equal probability while `nparK` = 2 and 4 do not have this restriction (Yang, 1995).

`clock` specifies whether we assume rate constancy along different branches in the tree (`clock` = 1) or variable rates among lineages (`clock` = 0).

`nhomo` relates to the non-homogeneous process models (`nhomo` = 2 or 3). Choosing `nhomo` = 1 will produce the proper maximum likelihood estimates of the frequency parameters (π_T, π_C and π_A; π_G is not a free parameter as the sum is 1) in the F81, F84, HKY85, TN93, or REV models, while normally (`nhomo` = 0) these are estimated by the averages of the observed frequencies.

`getSE` tells whether we want estimates of the standard errors of estimated parameters. These are calculated by the curvature method, *i.e.*, by inverting the matrix of second derivatives of the log likelihood with respect to parameters; the option is not available when tree-search is performed (`runmode` = 1 or 2).

`RateAncestor` also works with `runmode` = 0 only, and if given the value 1, will lead to calculation of rates for sites along the sequence for models of gamma rates across sites or reconstruction of ancestral sequences for models of a constant rate at sites.

With `basemlg`, tree-search or the assumption of a molecular clock are not allowed and so choose `runmode` = 0 and `clock` = 0. Substitution models available for `basemlg` are JC69, F81, K80, F84 and HKY85, and a continuous gamma is always assumed for rates at sites. The variables `ncatG`, `given_rho`, `rho`, `nhomo` have no effect. The S.E.'s of parameter estimates are always printed out because they are calculated during the iteration, and so `getSE` has no effect. See §3.3 for running `basemlg`.

4.4 The control file for the program `codeml` *(*`codonml` *and* `aaml`*)*

The default file name is `codeml.ctl`, and below is a sample copy.

```
      seqfile  =  stewart.aa     * sequence data file name
      outfile  =  mlc            * main result file name
     treefile  =  trees.6s       * tree structure file name
        noisy  =  2   * 0,1,2,3: how much rubbish on the screen
      runmode  =  0   * 0:user tree; 1:semi-automatic; 2:automatic

      seqtype  =  2   * 1:codons; 2:AAs; 3:codon-translated AAs
        model  =  2   * models for AAs or codon-translated AAs:
                      *   0:1/20 each, 1:pi[aai], 2:Dayhoff(KYM),
                      *   3:JTT(KYM), 4:dayhoff(mine), 6:FromCodon
                      * models for codons:
                          0:1/61 each, 1:1X4 2:3X4, 3:codon usage table
        icode  =  0   * 0:standard genetic code; 1:mammalian mt code
        Mgene  =  0   * 0:rates, 1:separate; 2:pi, 3:kappa, 4:all

  given_kappa  =  0   * 1: kappa given, 2: omega given, 3: both given
        kappa  =  2.  * initial or given kappa and omega in GY94.
        omega  =  2.

  given_alpha  =  1
        alpha  =  0   * initial or given alpha, 0:infinity
```

```
      Malpha  =  0   * different alpha for genes
       ncatG  =  8   * # of categories in the dG or AdG models of rates

   given_rho  =  1
         rho  =  0   * initial or given rho, 0:no correlation

       clock  =  0   * 0:no clock, unrooted tree, 1:clock, rooted tree
       getSE  =  0   * S.E.s of parameter estimates (1:yes, 0:no)
RateAncestor  =  0   * rates for sites or ancestral reconstructions (1:yes, 0:no)
```

The variables `seqfile`, `outfile`, `treefile`, `noisy`, `Mgene`, `given_alpha`, `alpha`, `Malpha`, `given_rho`, `rho`, `clock`, `getSE` and `RateAncestor` are used in the same way as in `baseml.ctl` and are described in the previous section (§4.3). The variable `seqtype` specifies the type of sequences in the data; `seqtype` = 1 means codon sequences (the program is then effectively `codonml`); 2 means amino acid sequences (the program is then `aaml`); and 3 means codon sequences which are to be translated into proteins for analysis.

The variable `model` depends on `seqtype`. For codon sequences (`seqtype` = 1), it concerns assumptions about the equilibrium codon frequencies in the model of Goldman and Yang (1994a); 3 means using all codon frequencies as free parameters (61 - 1 free parameters for the general genetic code), while 2 means using the nucleotide frequencies at the three codon positions to calculate the expected codon frequencies [$9 = 3 \times (4 - 1)$ free parameters]; in both cases the average observed frequencies in the data are used as estimates. The genetic code table is specified by `icode`. About a dozen genetic code tables are implemented, with 0 standing for the standard code and 1 for the mammalian mitochondrial code (the rest are noted in the `codeml.ctl`). The variable `given_kappa` controls both `kappa` and `omega`; these parameters are described in §6.2.

For amino acid sequences (`seqtype` = 2 or 3), the variable `model` specifies the model of amino acid substitution: `model` = 0 means the Poisson process model assuming equal rates for any amino acid substitutions (Bishop and Friday, 1987); `model` = 2 is based on the empirical substitution matrix of Dayhoff *et al.* (1978) (Kishino *et al.*, 1990); and `model` = 3 is based on the updated PAM matrix by Jones *et al.* (1992); `model` = 6 uses the model of Goldman and Yang (1994) for codon substitution to describe protein evolution.

`runmode` also works in the same way as in `baseml.ctl`, but an exception is that `runmode`=-2 for `codonml` forces the program to calculate the numbers of synonymous and nonsynonymous substitution rates in all pairwise sequence comparisons.

4.5 The control file for the program `mcmctree`

The default control file name is `mcmctree.ctl`, and a sample copy is shown below.

```
  seqfile  = mtprim9.nuc   * sequence data file name
  outfile  = mcmctree.out  * main result file name
 treefile  = trees.9s      * tree structure file name
   LHfile  = Lhs    * LH file name. read (MCMC=0) or overwritten (MCMC=1)
     MCMC  = 0      *0: read LHs from LHfile, 1: use MCMC to generate Lhs
     beta  = 0.15   * prob{change labeled history}, used if MCMC=1 only
   delta0  = 1      * small number for MCMC, used if MCMC=1 only
   delta1  = .2     * smaller number for comparing candidate Lhs

    model  = 2      * 0:JC69, 1:K80, 2:F81, 3:F84, 4:HKY85
```

```
      kappa  = 2.     * given kappa and omega in GY94.
      alpha  = 0      * given alpha, 0:infinity
      ncatG  = 8      * # of categories in the dG or AdG models of rates

   hierarch  = 0      * 1:hierarchical; 0:empirical Bayes analysis
      birth  = 6.7    * lineage birth rate
      death  = 2.5    * lineage death rate
     sample  = .06    * sampling proportion
     mutate  = .24    * mutation rate (# of mutations from root to present)
```

The variables `seqfile`, `outfile`, `treefile`, `model`, `kappa`, `alpha`, and `ncatG` are defined as in `baseml.ctl` for program `baseml`. The program first collects a set of candidate labeled histories. If `MCMC`=0, the candidate labeled histories are read from the file `LHfile`, while if `MCMC`=1, they are generated from the Markov chain Monte Carlo, with the Monte Carlo integration over ancestral speciation times evaluated at a low accuracy level (controled by `delta0`). The program then calculate the (relative) posterior probabilities of these candidate labeled histories, with the Monte Carlo integration evaluated at a higher accuracy level (controled by `delta1`). The Markov chain moves with probability `beta` to another labeled history of the same tree topology, or to a labeled history of a different tree topology through a nearest neighbor interchange. `hierarch` controls whether an empirical Bayes analysis or a hierarchical Bayes analysis is to be performed. `birth`, `death`, and `sample` are paramters of the prior distribution specified by the birth-death process with species sampling.

5 Interpreting the Output

5.1 Output file for `baseml` *and* `basemlg`

This file is self-explanatory and part of an example output file (from `baseml`) is listed below. Descriptive statistics are always listed. The observed *site patterns* and their frequencies are listed, together with the proportions of *constant patterns*. Nucleotide frequencies for each species (and for each gene in case of multiple gene data) are counted and listed. $l_{max} = \ln(L_{max})$ is the upper limit of the log likelihood and may be compared with the likelihood for the best (or true) tree under the substitution model to test the model's goodness of fit to data (Goldman, 1993a).

```
Baseml.c  brown.nuc  HKY85  (4 genes, different rates)
human
chimapnzee
gorilla
orangutan
gibbon

ns = 5   ls = 895
ngene= 4:  lengths: 233  232  232  198
# of site patterns = 153
Average T:0.25318  C:0.32894  A:0.31196  G:0.10592
# constant sites: 613 ( 68.49%)

TREE # 1: (12345)
lnL(ntime: 5 np: 9): -2587.506845    +0.000000
  6..1    6..2    6..3    6..4    6..5
0.03882 0.05012 0.06074 0.14634 0.18633 0.33989 3.40714 0.58077 11.46578

TREE # 2: ((123)45)
lnL(ntime: 6 np: 10): -2550.884230   +36.622616
  6..7    7..1    7..2    7..3    6..4    6..5
0.05586 0.03290 0.04135 0.05501 0.08516 0.12082 0.39024 3.93286 0.64946 11.72917

TREE # 3: (((12)3)45)
lnL(ntime: 7 np: 11): -2542.948041   +44.558804
  6..7    7..8    8..1    8..2    7..3    6..4    6..5
0.05081 0.01323 0.03219 0.04059 0.04502 0.08774 0.12418 0.37752 3.83740 0.61889 12.05461
```

Pairwise distances are calculated based on the same model specified for the likelihood analysis, with either a single rate for all sites or (continuous) gamma rates for sites assumed; in the latter case the initial value for the α parameter is used in the calculation. If the specified model is HKY85 or REV, the distances are calculated using the model of Tamura and Nei (1993). The transition/transversion rate ratio (κ) for the K80, F84 models are also estimated for each pairwise comparison, and with the model of Tamura and Nei (1993) which has two such ratios, the average is calculated.

For `runmode` = 0 (user tree), the programs fit branch lengths by the (ordinary) least squares method for each tree topology based on pairwise distance estimates. These are then used as initial values for iterative likelihood estimation; initial values chosen in this way are actually quite poor because the two methods normally produce quite different branch length estimates, especially for "bad" trees (Yang 1994b). For each tree, the log likelihood is listed, together with the likelihood difference between the current tree and the first tree (usually a star-like tree). This likelihood difference is listed for easy comparison but not for evaluating the reliability of the bifurcating tree. The listed value for `ntime` in the bracket is the number of branches when the clock is not assumed or the number of interior nodes when the clock is assumed, while `np` is the number of all parameters being estimated. Theoretically, a bifurcating tree should not be poorer (have lower likelihood value) than the star-like tree, but with my programs, the bifurcating tree may look slightly worse. The reason is that parameters are bounded during the iteration; for example, branch lengths may be restricted to be, say, > 0.0001. If this happens, one or more interior-branch lengths should be very close to zero, and the bifurcating tree should be taken as converging to the multifurcating tree.

For the tree-search modes (`runmode` = 1 or 2, unavailable for `basemlg`), two values are listed for the likelihood: the first is the log likelihood of the current tree and the second is the likelihood increment relative to the best tree of the previous search stage. (See §7 for an example.)

Below the line of likelihood values are the maximum likelihood estimates of parameters. These include branch lengths in an unrooted tree or divergence times in a rooted tree (both called time parameters), and other parameters which are common to tree topologies. Parameters are arranged in a fixed order in the output (and in the iterative algorithm):

[1] t: branch lengths (`clock` = 0) or divergence times (`clock` = 1);

[2] c_j: rate parameters for genes (g - 1 parameters for g genes; $c_1 = 1$);

[3] rate ratio parameters: κ in K80, F84, HKY85, κ_1 and κ_2 in TN93, or a, b, c, d, e in REV (TN93 and REV for `baseml` only);

[4] frequency parameters π_T, π_C, π_A (for `nhomo` > 0 in `baseml` only);

[5] α of the (discrete-) gamma distribution;

[6] ρ of the auto-discrete-gamma model (unavailable for `basemlg`);

The above list does not include parameters that do not exist in the model or are not estimated by iteration. For example, when JC69 is assumed or when K80 is assumed with `given_kappa` = 1, κ will not be on the above list in either model as it is not a parameter to be estimated from the data. With data of multiple genes (option G in the data file and `Mgene` in the control file), the above order of parameters is maintained. For example, if different transition/transversion rate parameters and gamma distributions are assumed for different genes in the TN93 model, the order of the parameters will be κ_1 and κ_2 for the first gene, κ_1 and κ_2 for the second gene, and so on, and α for the first gene, followed by α for the second gene, and so forth.

Without the assumption of a molecular clock, the model will involve as many time parameters as the number of branchs; estimates of these are listed right below the branches. When a clock is assumed, the model will involve as many time parameters as the number of interior nodes, which are the times before present of divergence at the nodes. For example, the rooted tree ((13)24), represented with its branches as

$$5 \to 6,\ 6 \to 1,\ 6 \to 3,\ 5 \to 2,\ 5 \to 4$$

has two interior nodes (5 and 6) and has two time parameters; these are arranged in the increasing order of the node numbers (*i.e.*, that for node 5 comes first and that for node 6 second).

In the example shown above, the HKY85 model was fitted to the data of Brown *et al*. (1982). The sites are grouped into four classes (4 genes) as discussed in §4.1. Tree #3, represented as (((12)3)45), is (((human, chimpanzee), gorilla), orangutan, gibbon), and estimates of the common parameters from this tree are c_2 = 0.37752, c_3 = 3.83740, c_4 = 0.61889, and κ = 12.05461.

For `basemlg` or for `baseml` with `getSE` = 1, the S.E.'s, which are the square roots of the large sample variances, are listed exactly below the parameter estimates. Zeros on this line mean errors, either caused by divergence of the algorithm or zero branch lengths. The S.E.'s of the common parameters measure the reliability of the estimates. For example, $(\kappa - 1)/\mathrm{SE}(\kappa)$, when κ is estimated under K80, can be compared with a normal distribution to see whether there is real difference between K80 and JC69. The test can be more reliably performed by comparing the log-likelihood values under the two models, using the likelihood ratio test. It has to be stressed that the S.E.'s of the estimated branch lengths should not be misinterpreted as an evaluation of the reliability of the estimated tree topology (*e.g.*, Yang, 1994b, Goldman and Yang, 1994).

5.2 Output file for `codeml` *(*`codonml` *and* `aaml`*)*

5.2.1 For `codonml` *(*`codeml` *with* `seqtype` = 1*)*

The codon frequencies in each sequence are counted and listed in a genetic code table, together with their sums across species. Each table contains up to six species, and is about the width of a page when the 8-point courier font is used. For data of multiple genes (option G in the sequence file), codon frequencies in each gene (summed over species) are also listed. The base distributions at the three codon positions are also listed. The method of Nei and Gojobori (1986) is used to calculate the number of synonymous substitutions per synonymous site (K_s) and the number of nonsynonymous substitutions per nonsynonymous site (K_a) and their ratio (K_s/K_a). These are used to construct initial estimates of branch lengths for the likelihood analysis. Note that the estimates of these quantities for the α- and β-globin genes shown in Table 2 of Goldman and Yang (1994a), calculated using the MEGA package (Kumar *et al.*, 1993), are not accurate.

The program `codonml` implements the model of Goldman and Yang (1994a). Parameters in the model are arranged in the following order

[1] t: branch lengths (clock = 0) or divergence times (clock = 1);

[2] c_j: rate parameters for genes (c_2, c_3, …, with $c_1 = 1$);

[3] κ and ω in the rate matrix of equation 9 (§6.2);

[4] α of the discrete-gamma model for rates at codon sites;

[5] ρ of the auto-discrete-gamma model for rates at codon sites.

The number of rate parameters (item [3]) is given as `nrate` at the beginning of the file. The ratio of the number of synonymous substitutions per synonymous site (K_s) to the number of nonsynonymous substitutions per nonsynonymous site (K_a) is calculated from the codon-based model following Goldman and Yang (1994), and is listed below parameter estimates (and their S.E.'s if any).

5.2.2 For `aaml` *(*`codeml` *with* `seqtype` *= 2)*

The file `mlc` produced for `aaml` is self-explanatory and very similar to the result files for the nucleotide- and codon-based models. The models for amino acid substitution based on the PAM matrices of Dayhoff *et al.* (1978) or Jones *et al*. (1992) do not involve any parameters in the substitution rate matrix (nrate = 0).

5.3 Secondary result file `rst`

When `print` = 1, the programs `baseml` and `aaml` also generate results in the file `rst`. If the model assumes a single rate for all sites, the program reconstructs ancestral sequences, while if the discrete-gamma mode is assumed for rates at sites, the substitution rates at sites are calculated from the model. Note that for both problems, the models produce identical results for sites with the same data, so that some of the analyses are listed once for the site pattern. The file can be very large, so that you should be cautious when directing it to the printer.

6 Models and methods

6.1 Substitution models for nucleotides (`baseml` and `basemlg`)

Nucleotide substitution is described by a Markov process. Different restrictions on the structure of the rate matrix lead to different models. These include JC69 (Jukes and Cantor, 1969), K80 (Kimura, 1980), F81 (Felsenstein, 1981), F84 (Felsenstein, DNAML program since 1984, PHYLIP Version 2.6), HKY85 (Hasegawa *et al.*, 1985), Tamura and Nei (1993), and REV (Yang, 1994a). The rate matrices of these models are given below

JC69:
$$\mathbf{Q} = \begin{pmatrix} . & 1 & 1 & 1 \\ 1 & . & 1 & 1 \\ 1 & 1 & . & 1 \\ 1 & 1 & 1 & . \end{pmatrix} \tag{1}$$

K80:
$$\mathbf{Q} = \begin{pmatrix} . & \kappa & 1 & 1 \\ \kappa & . & 1 & 1 \\ 1 & 1 & . & \kappa \\ 1 & 1 & \kappa & . \end{pmatrix} \tag{2}$$

F81:
$$\mathbf{Q} = \begin{pmatrix} . & \pi_C & \pi_A & \pi_G \\ \pi_T & . & \pi_A & \pi_G \\ \pi_T & \pi_C & . & \pi_G \\ \pi_T & \pi_C & \pi_A & . \end{pmatrix} \tag{3}$$

F84:
$$\mathbf{Q} = \begin{pmatrix} . & (1+\kappa/\pi_Y)\pi_C & \pi_A & \pi_G \\ (1+\kappa/\pi_Y)\pi_T & . & \pi_A & \pi_G \\ \pi_T & \pi_C & . & (1+\kappa/\pi_R)\pi_G \\ \pi_T & \pi_C & (1+\kappa/\pi_R)\pi_A & . \end{pmatrix} \tag{4}$$

with $\pi_Y = \pi_T + \pi_C$ and $\pi_R = \pi_A + \pi_G$.

HKY85:
$$\mathbf{Q} = \begin{pmatrix} . & \kappa\pi_C & \pi_A & \pi_G \\ \kappa\pi_T & . & \pi_A & \pi_G \\ \pi_T & \pi_C & . & \kappa\pi_G \\ \pi_T & \pi_C & \kappa\pi_A & . \end{pmatrix} \tag{5}$$

TN93:
$$\mathbf{Q} = \begin{pmatrix} . & \kappa_1\pi_C & \pi_A & \pi_G \\ \kappa_1\pi_T & . & \pi_A & \pi_G \\ \pi_T & \pi_C & . & \kappa_2\pi_G \\ \pi_T & \pi_C & \kappa_2\pi_A & . \end{pmatrix} \tag{6}$$

REV:

$$\mathbf{Q} = \begin{pmatrix} \cdot & a\pi_C & b\pi_A & c\pi_G \\ a\pi_T & \cdot & d\pi_A & e\pi_G \\ b\pi_T & d\pi_C & \cdot & \pi_G \\ c\pi_T & e\pi_C & \pi_A & \cdot \end{pmatrix} \quad (7)$$

The element Q_{ij} ($i \neq j$) represents the rate of substitution from nucleotide i to j, with the nucleotides ordered T, C, A, G. The diagonals Q_{ii} are specified by the mathematical requirement that row sums of $\mathbf{Q}$ are zero. The transition probability matrix over time t is then $\mathbf{P}(t) = \{P_{ij}(t)\} = \exp(\mathbf{Q}t)$, where $P_{ij}(t)$ is the probability that nucleotide i will become nucleotide j after time t. The nature of the problem does not permit separation of rate ($\mathbf{Q}$) and time (t), and $\mathbf{Q}$ is multiplied by a constant so that the average rate of substitution is 1 when the process is in equilibrium. This scaling means that time t, or the branch length in a tree, is measured by the expected number of nucleotide substitutions per site. $\mathbf{Q}$ thus represents the *pattern of substitution*, while the amount of evolution is reflected in time or the branch length. The *frequency parameters* π_T, π_C, π_A, π_G (with sum to be 1) give the equilibrium distribution of the process for the F81, F84, HKY85, TN93 and REV models; the equalibrium distribution under the JC69 and K80 models has equal frequencies (1/4) for the four nucleotides, which is rarely true for real data. Parameters a, b, c, d, e in REV, κ in F84 or HKY85, and κ_1 and κ_2 in TN93 may be termed *rate ratio parameters.* So the JC69, K80, F81, F84, HKY85, TN93 and REV models contain 0, 1, 0, 1, 1, 2, 5 rate ratio parameters respectively, and 0, 0, 3, 3, 3, 3, 3 frequency parameters respectively. Normally the frequency parameters are estimated using the averages of the observed frequencies, which should be very close to the true maximum likelihood estimates if the assumptions of homogeneity and stationarity are not too wrong.

Parameter κ in the K80 and HKY85 models is equivalent to α/β in the notation of Kimura (1980) and Hasegawa *et al.* (1985), who used α and β to refer to rates of transitional and transversional substitutions, repectively. Tamura and Nei (1993) used α_2, α_1, and β instead of κ_1 and κ_2. The present notation is more convenient in a maximum likelihood analysis as the ratio is assumed to be constant for different branches of the tree. F84 is the model implemented in J. Felsenstein's DNAML program. The rate matrix for this model was given by Hasegawa and Kishino (1989), Kishino and Hasegawa (1989), Yang (1994b, 1994c) and Tateno *et al*. (1994). Thorne *et al.* (1992) described the transition probability matrix, and Yang (1994c) and Tateno *et al.* (1994) derived formulae for estimating sequence distances under the model. REV is the general reversible process model (Yang, 1994a; see also Tavare, 1986; Zharkikh, 1994) and is used in `baseml` only. It seems sufficiently general to enable accurate estimation of the substitution pattern from real data. See Gillespie (1986), Tavaré (1986), Rodriguez *et al*. (1990), Yang (1994a), Zharkikh (1994), and Wakeley (1996) for reviews of substitution models.

Note that there are several different definitions of the "transition/transversion rate ratio". My programs estimate the parameters specified in the above formulas. To make estimates comparable among models, one can calculate the transition/transversion rate ratio averaged by base frequencies:

$R = (\pi_T Q_{TC} + \pi_C Q_{CT} + \pi_A Q_{AG} + \pi_G Q_{GA})$

$/ (\pi_T Q_{TA} + \pi_T Q_{TG} + \pi_C Q_{CA} + \pi_C Q_{CG} + \pi_A Q_{AT} + \pi_A Q_{AC} + \pi_G Q_{GT} + \pi_G Q_{GC})$.

Formulas for such calculations are listed below.

Model	Average transition/transversion rate ratio (R)
JC69	1/2
K80	$\kappa/2$
F81	$(\pi_T\pi_C + \pi_A\pi_G) / (\pi_Y\pi_R)$
F84	$[\pi_T\pi_C(1 + \kappa/\pi_Y) + \pi_A\pi_G(1 + \kappa/\pi_R)] / (\pi_Y\pi_R)$
HKY85	$(\pi_T\pi_C + \pi_A\pi_G)\kappa / (\pi_Y\pi_R)$
TN93	$(\pi_T\pi_C\kappa_1 + \pi_A\pi_G\kappa_2) / (\pi_Y\pi_R)$
REV	$(\pi_T\pi_C a + \pi_A\pi_G) / (\pi_T\pi_A b + \pi_T\pi_G c + \pi_C\pi_A d + \pi_C\pi_G e)$

6.2 Substitution models for codons (`codonml: codeml.c` *with* `seqtype` = 1)

Compared with models for nucleotide substitution, models for codon substitution are not well developed. Goldman and Yang (1994a) suggested a Markov process model for describing the probability of ssubstitution between the sense codons. Stop codons are not allowed in the process. The rate matrix for this model is given as $\mathbf{Q} = \{Q_{ij}\}$, where

$$Q_{ij} = \begin{cases} 0, & \text{if codons } i \text{ and } j \text{ differ by more than one nucleotide;} \\ \pi_j \exp(-0.01 d_{aa_i,aa_j}\omega), & \text{if the only difference is a transversion;} \\ \kappa\pi_j \exp(-0.01 d_{aa_i,aa_j}\omega), & \text{if the only difference is a transition.} \end{cases} \quad (9)$$

Q_{ij} is the rate of substitution from codon i to j. The equalibrium frequency of codon j (π_j) can be considered a free parameter, but can also be calculated from the nucleotide frequencies at the three codon positions. d_{aa_i,aa_j} is the physico-chemical distance between the two amino acids (aa_i and aa_j) encoded by codons i and j (Grantham, 1974). Grantham's distances have mean 100, and are scaled such that the mean is 1. We set $d_{aa_i,aa_j} = 0$ if the two amino acids are identical. Parameter ω is positively related to the ratio of synonymous rates to nonsynonymous rates and often to the conservedness of the protein. As an oversimplified interpretation, $\omega = 0$ means amino acid changes are neutral, $\omega > 0$ means they are deleterious, and $\omega < 0$ means they are favorable. Goldman and Yang (1994a) used a different parameter V instead of ω; the relationships are $\omega = 100/V$, and $\mathrm{SE}(\omega) = 100\cdot\mathrm{SE}(V)/V^2$ or $V = 100/\omega$ and $\mathrm{SE}(V) = 100\cdot\mathrm{SE}(\omega)/\omega^2$.

Similar to the case of nucleotide-based models, the matrix $\mathbf{Q}$ of equation (9) is scaled so that the average rate of substitution at equilibrium is one nucleotide substitution per codon; the branch length or time is then measured by the numbers of nucleotide substitutions per codon. Note that rate differences at the three codon positions are implied in the model, and are accommodated by the parameter ω.

Muse and Gaut (1994) have described a similar codon-based model, which may be obtained by setting $\kappa = 1$, and $d_{aa_i,aa_j} = d$ for all $aa_i \neq aa_j$ in equation 9, and by restricting the πs to be equal to values expected from the average nucleotide fequencies.

Goldman and Yang (1994) also described the method for estimating the number of synonymous substitutions per synonymous site (K_s) and the number of nonsynonymous substitutions per nonsynonymous site (K_a). The program `codonml` with option `runmode`=-2 calculates these statistics for all pairwise comparisons.

6.3 Substitution models for amino acids (`aaml: codeml.c` *with* `seqtype` **= 2)**

Empirical models based on the Dayhoff substitution matrix (model = 2) or its updated version of Jones *et al.* (1992) are constructed using the same strategy. The transition probability matrix over a very short time period such as one PAM, *i.e.*, **P**(0.01), is used to approximate the matrix of instantaneous rates (**Q**). The empirical matrices of Dayhoff *et al.* (1978) and Jones *et al.* (1992) were made to satisfy the reversibility condition, that is,

$$\pi_i Q_{ij} = \pi_j Q_{ji} \tag{10}$$

for any i and j, so that my implimentations may be slightly different from that of Kishino *et al.* (1990). These models assume a fixed pattern of amino acid substitution, and cannot identify possible differences in the evolutionary dynamics of different proteins. "Mechanistic" models of amino acid substitution would require consideration of both the mutational distance between the amino acids as determined by the locations of their encoding codons in the genetic code table, and the effects that the potential change may have on the structure and function of the protein, which may be related to the physical, chemical and structural differences between amino acids. It seems natural that such a model should be formulated at the level of codons. Currently the model of Goldman and Yang (1994a) is the only such model implemented in `aaml`.

6.4 Variable substitution rates across sites

Recent work on this topic was reviewed by Yang (1996b). Two approaches have been taken to account for variable rates among sites in models used in phylogenetic analysis. The first is to use several categories of rates. Such models are implemented in `baseml` only (the control variable `nparK`). The difficulty is that there will be too many parameters if more than two categories are assumed. The second approach is to use a continuous distribution to approximate variable rates at sites, and the most commonly used continuous distribution is the gamma. This distribution has a shape parameter α and a scale parameter β with mean α/β and variance α/β^2. We set $\beta = \alpha$ so that the mean of the distribution is 1 (with variance $1/\alpha$). The single parameter α is then inversely related to the extent of rate variation at sites. Maximum llikelihood estimation under the gamma model of rates for sites (implemented in `basemlg` for nucleotide sequences only) is very slow. A more practical approach is the discrete-gamma model suggested by Yang (1994c) and used in `baseml`, `codonml`, and `aaml` to approximate the rate variation among nucleotide, codon and amino acid sites. Several categories, chosen such that each category has equal probability of occurrence, are used to approximate the gamma distribution, with the mean of the category used to represent all rates in the category (Yang, 1994c). So the discrete-gamma model, like the continuous gamma, involves a single parameter α. Four such categories (`ncatG`=4) seems to be enough for a good approximation (Yang, 1994c). However, if the α estimate is very small (say, <0.2) or if the purpose of the analysis is to estimate the α parameter of the (continuous) gamma for use in estimating pairwise distances, it may be better to use more categories (say 8). The auto-discrete-gamma model was initiated by the observation of the existence of *variable* and *conserved* regions in a gene, from which we expect substitution rates at neighboring sites to be positively correlated. The model, described by Yang (1995), has an extra parameter ρ that measures the correlation of rates at two adjacent sites. The model is implemented in `baseml`, `codonml` and `aaml`.

Substitution rates for sites are calculated using the conditional mean of the rates given the data (Yang and Wang, 1995; Yang, 1995). The calculated rates for sites, after some smoothing, can be plotted along the sequence to help identify variable and conserved regions.

6.5 Models for combined analyses of heterogeneous data (multiple genes or codon positions)

Several models are described by Yang (1996a) and implemented in programs `baseml` and `codeml` (`codonml` and `aaml`) for analyzing heterogeneous data sets (such as those of multiple genes or different codon positions). The implimentation and description below refer to the case of multiple genes, but in the case of nucleotide-based models (`baseml`), the method can be used to analyze data of different codon positions. These models account for different aspects of heterogeneity among the different data sets and are useful for testing hypotheses concerning the similarities and differences in the evolutionary process of different data sets. The data format (option G) is explained in §4.1.

The simplest model which assumes complete homogeneity among genes can be fitted by concatenating different genes into one sequence without using the option G (and by specifying `Mgene` = 0 in the control file). The most general model is equavilent to a separate analysis. This can be done by fitting the same model to each data set (each gene), but can also be done by specifying `Mgene` = 1 with the option G in the combined data file. The sum of the log-likelihood values over different genes is then the log likelihood of the most general model considered here. Models accounting for some aspects of the heterogeneity of multiple genes are fitted by specifying `Mgene` in combination with the option G in the sequence data file. `Mgene` = 0 means a model that asumes different substitution rates but the same pattern of nucleotide substitution for different genes. `Mgene` = 2 means different frequency parameters for different genes but the same rate ratio parameters (κ in the K80, F84, HKY85 models or the rate parameters in the TN93 and REV models). `Mgene` = 3 means different rate ratio parameters and the same frequency parameters. `Mgene` = 4 means both different rate ratio parameters and different frequency parameters for different genes. Parameters and assumptions made in these models are summarized in the following table, with the HKY85 model used as an example. When substitution rates are assumed to vary from site to site, the control variable Malpha specifies whether one gamma distribution will be applied across all sites (`Malpha`=0) or a different gamma distribution is used for each gene (or codon position).

Sequence file	Control file	Parameters across genes
Without option G	`Mgene` = 0	everything equal
Option G	`Mgene` = 0	the same κ and π, but different cs (proportional branch lengths)

Option G	`Mgene = 2`	the same κ, but different πs and cs
Option G	`Mgene = 3`	the same π, but different κs and cs
Option G	`Mgene = 4`	different κ, πs, and cs
Option G	`Mgene = 1`	different κ, πs, and different (unproportional) branch lengths

The different cs for different genes mean that branch lengths estimated for different genes are proportional. Parameters π represent the equilibrium nucleotide frequencies, which are estimated using the observed frequencies (`nhomo`=0). The transition/transversion rate ratio κ in HKY85 can be replaced by the two or five rate ratio parameters under the TN93 or REV models, respectively. The likelihood ratio test can be used to compare these models, using an approach called the analysis of deviance, which is very similar to the more familiar analysis of variance.

The following two tables list results obtained from analyzing the mtDNA data of Brown *et al.* (1982). The HKY85 model is used. The first table is for separate analyses of the four site classes (`Mgene`=1). The second is from combined analyses, and the six rows of results correspond to the six models specified in the table above. More results for these data are listed in Section 9.

	# sites	π_T	π_C	π_A	π_G	$\hat{\kappa}$	l
Position 1	233	0.2009	0.2858	0.3811	0.1322	7.809	-640.46
Position 2	232	0.3922	0.3241	0.1793	0.1043	16.583	-465.00
Position 3	232	0.1543	0.4664	0.3371	0.0422	50.538	-864.10
tRNA region	198	0.2677	0.2242	0.3566	0.1515	10.461	-477.91
Total	895	0.2532	0.3289	0.3120	0.1059		-2447.46

	np	l	$\hat{c}_2$	$\hat{c}_3$	$\hat{c}_4$	$\hat{\kappa}_1$	$\hat{\kappa}_2$	$\hat{\kappa}_3$	$\hat{\kappa}_4$
No G, `Mgene=0`	11	-2665.42		1				9.390	
G, `Mgene=0`	14	-2542.95	0.378	3.837	0.619			12.055	
G, `Mgene=2`	23	-2468.08	0.431	4.832	0.637			16.995	
G, `Mgene=3`	17	-2540.91	0.384	4.055	0.630	7.841	20.669	14.099	10.546
G, `Mgene=4`	26	-2461.14	0.441	6.230	0.646	8.016	16.345	29.046	10.621
G, `Mgene=1`	44	-2447.46							

np: number of parameters in the model

6.6 The molecular clock

The assumption of a molecular clock means that substitution rates are constant among lineages. It is specified as `clock` = 1 for `baseml`, `codonml`, and `aaml`, and is not available for `basemlg`. With this assumption, the trees are rooted and there are as many time parameters as the number of interior nodes in the tree; without this assumption (`clock` = 0), the number of time parameters is equal to the number of branches. Currently my implementation of the clock model is not satisfactory (see §8.2).

6.7 The nonhomogeneous models (`nhomo` *for* `baseml`*)*

The option `nhomo` for `baseml` concerns the frequency parameters in the F81, F84, HKY85, TN93, or REV models. The option `nhomo` = 1 fits a homogeneous model, but solves for the frequency parameters. Normally (`nhomo` = 0) they are estimated by the averages of the observed frequencies. The options `nhomo` = 2 or 3, in combination with F84 or HKY85, fit nonhomogeneous models of Yang and Roberts (1995). Substitutions are assumed to follow the pattern of F84 or HKY85, but with different frequency parameters assigned for different branches in the tree, to allow for unequal base frequencies in different sequences. The position of the root then makes a difference to the likelihood, and rooted trees are used. Because of the parameter-richness, the model may only be used with data of 4 species when base frequencies are drastically different in different sequences. Convergence is hard to achieve, especially if the sequences are short, and different initial values need to be tried. Choose `given_kappa` = 1, which means one common κ is assumed for all branches. The option `nhomo` = 3 assigns one set of frequency parameters for each node, which are the parameters for the rate matrix along the branch leading to the node or are the initial distribution if the node is the root of the tree. In the output, estimates of the frequency parameters are shown in the order of nodes $s + 1$, $s + 2$, ..., where s is the number of sequences.

6.8 Reconstruction of ancestral sequences

A set of nucleotides or amino acids assigned to the interior nodes of the tree is termed a reconstruction at the site. Parsimony reconstructs ancestral character states by using the criterion that the number of changes along the tree at the site is minimized. Algorithms based on this criterion were developed by Fitch (1971) and Hartigan (1973), and are implemented in the program `pamp`. The likelihood analysis uses more information contained in the data, especially concerning the branch lengths and the substitution pattern. Method based on the likelihood analysis was developed by Yang *et al.* (1995) and is implemented in `baseml` for nucleotide sequences and in `aaml` (`codeml.c` with `seqtype`=2) for amino acid sequences. After maximum likelihood estimates of parameters (such as branch lengths and other parameters in the model) are obtained, the method evaluates the posterior probabilities of different pathways at a site given the data at the site, and the pathway that has the highest posterior probability is the best reconstruction. There are typically a great many pathways at a site, and the program evaluates only the "likely" ones. With fewer than 15 interior nodes in the tree for nucleotide sequences (or 7 for amino acid sequences), the program evaluates all pathways that can be generated by assigning to each interior node all nucleotides (or amino acids) that are observed at the site in any of the sequences. For even larger trees, the program uses the parsimony algorithm to generate the equivocal character states at each interior node, and then evaluates pathways that can be generated from those equivocal states.

Currently, character reconstruction is performed only if a constant substitution rate is assumed for all sites (although models accounting for gene-specific or codon position-specific rates are allowed). When the option `print`=1 is specified, results are collected in the file rst. The analysis and results follow closely Yang *et al.* (1995). Reconstructions are first performed with characters for all interior nodes at a site considered as a whole. Then each node is considered separately, and this provides another measure of the accuracy. The reconstructed sequences are then listed together with the observed data.

6.9 Bayesian estimation of phylogenetic trees

The program `mcmctree` performs Bayesian estimation of phylogenies (Rannala and Yang, 1996; Yang and Rannala, submitted). The birth-death process with species sampling is used to specify the prior distribution of phylogenies ("labeled histories" which are rooted tree topologies with the interior nodes ordered according their associated speciaiton times), and the posterior probabilities of the labeled histories are compared to select the maximum posterior probability tree. The program implementing the method of Rannala and Yang (1996) is distributed as the algorithm involves extensive computation. Instead, `mcmctree` implements a refined method (Yang and Rannala submitted), which uses Markov chain Monte Carlo to select candidate labeled histories, and Monte Carlo integration to integrate over the distribution of the ancestral speciation times.

7 The Tree Search Algorithm

The tree search algorithm implemented in `baseml`, `codonml` and `aaml` is a divisive algorithm, termed "star-decomposition" by Adachi and Hasegawa (1992). The algorithm starts from either the star tree (`runmode` = 2) or a multifurcation tree read from the tree structure file (`runmode` = 1). The algorithm joins two taxa to achieve the greatest increase in log-likelihood over the star-like tree. This will reduce the number of OTUs by one. The process is repeated to reduce the number of OTUs by one at each stage, until no multifurcation exists in the tree. The algorithm is not guaranteed to find the maximum likelihood tree, and using multifurcating trees (when the large structure of the phylogeny is known) to perform several runs with `runmode = 1` is a good idea.

An example is shown below, with the concatenated α- and β-globin data (amino acid sequences) analyzed assuming the empirical model based of Dayhoff *et al.* (1978), combined with an auto-discrete-gamma model for rates at sites (`seqtype` = 2 and `runmode` = 2 in `codeml.ctl`). What is listed here is essentially the file `rst` which records the process of tree search, and is similar to what is seen on the screen (with `noisy` = 2). The number after the parenthesis representation of the topology, is the likelihood increment relative to the star-like tree for the current search stage. The integer number before the cross is the number of times that the function which calculates the log-likelihood has been called during the iteration. A cross (X) means that the iterative algorithm probably has converged and a question mark in this place (?) means failure is suspected. The two numbers after the cross (X), not seen on the screen but listed in the file `rst`, are estimates of the common parameters (α and ρ in this case). The model is used here to show the insensitivity of estimates of common parameters to tree topologies, and for tree search in real data analysis, it is unnecessary to use such a delicate model.

```
AAML   abglobin.aa    Model: Dayhoff  (nrate:0)   Auto-dGamma  (ncatG=4)
ns = 5     ls = 285
# of site patterns =     126

Average
```

```
0.1144 0.0211 0.0358 0.0554 0.0119 0.0182 0.0414 0.0730 0.0625 0.0133
0.1200 0.0828 0.0119 0.0533 0.0393 0.0646 0.0554 0.0112 0.0204 0.0940

# constant sites:    170 ( 59.65%)
ln{Lmax}:    -1189.106658

distances

human
goat-cow         0.203
rabbit           0.190  0.287
rat              0.331  0.433  0.340
marsupial        0.468  0.504  0.630  0.745

stage 0: (12345)
lnL:  -1693.742585   241

stage 1:    10 trees, ntime:  6  np:  8
star tree: (12345)  lnL:-1693.742585

S=1:  1/10  T=   1  ((12)345)    +1.168   285 X  0.614  0.393
S=1:  2/10  T=   2  ((13)245)    +2.502   321 X  0.639  0.401
S=1:  3/10  T=   3  ((14)235)    +1.435   321 X  0.630  0.420
S=1:  4/10  T=   4  ((15)234)    +0.782   288 X  0.612  0.368
S=1:  5/10  T=   5  (1(23)45)    +1.328   315 X  0.621  0.431
S=1:  6/10  T=   6  (1(24)35)    +2.401   281 X  0.650  0.435
S=1:  7/10  T=   7  (1(25)34)    +2.035   278 X  0.617  0.358
S=1:  8/10  T=   8  (12(34)5)    +6.643   307 X  0.636  0.368
S=1:  9/10  T=   9  (12(35)4)    +0.096   259 X  0.606  0.383
S=1: 10/10  T=  10  (123(45))    +0.374   268 X  0.616  0.393

stage 2:     3 trees, ntime:  7  np:  9
star tree: (12(34)5)  lnL:-1687.099761

S=2:  1/3  T=  11  ((12)(34)5)    +0.071   290 X  0.640  0.371
S=2:  2/3  T=  12  ((1(34))25)    +1.802   290 X  0.654  0.353
S=2:  3/3  T=  13  ((15)2(34))    +0.358   288 X  0.645  0.360

best tree: ((1(34))25)   lnL:  -1685.298046
```

8 Technical Details and Limitations

8.1 Adjustable variables in the source codes

The maximum values of certain variables are listed as constants in uppercase at the beginning of the main programs (`baseml.c`, `basemlg.c`, `codeml.c`). These values can be raised without increasing the memory requirement by too much.

`NS`	maximum number of sequences (species)
`LSPNAME`	maximum number of characters in a species name
`NGENE`	maximum number of "genes" in data of multiple genes (option G)
`NCATG`	maximum number of rate categories in the (auto-) discrete-gamma model (`baseml.c`, `codeml.c`)

You may need to change the value of LSPNAME to suit your need.

The real limit of the programs is the computation involved, which increases very rapidly with the number of species even for one given tree topology, not to mention the explosion of the number of tree topologies. The programs `baseml`, `aaml` and `codonml` use essentially the same algorithm, the difference being that they work with 4×4, 20×20 and 61×61 matrices respectively. The algorithm used by `basemlg` for the (continuous) gamma model is quite different, and the calculation required increases much faster with the number of species than that for `baseml`. The computational requirement is also related to the complexity of the model used. The discrete-gamma models need roughly `ncatG` (the number of rate categories) times the computation for a single-rate model, and the auto-discrete-gamma models need slightly more. For nucleotide based models (`baseml` and `basemlg`), the calculation is related to the number of distinct eigenvalues in the rate matrix (see equations 1-7), which is 2 for JC69 and F81, 3 for K80 and F84, and 4 for HKY85, TN93 and REV. So the calculation required for JC69 and F81 is the same and that for K80 and F84 is the same. However, TN93 needs more calculation than HKY85 while REV needs the most as these three models involve different numbers of parameters to be estimated.

With a workstation, the practical limits for the size of the data may be 20 - 100 sequences for `baseml`, 10 - 50 sequences for `aaml`, 5 - 8 sequences for `codonml`, and 4 - 6 sequences for `basemlg`. The parsimony program (`pamp`) can manage much larger data sets, but you need to use other programs to prepare the tree structure file. Long sequences may not be a problem, but too many sequences are.

Other variables that may be changed include the bounds for parameters, specified at the beginning of the function `testx` in the main programs (`baseml.c` and `codeml.c`).

8.2 Known problems

Under the molecular clock model (`clock`=1), the divergence times for the interior nodes of the tree are specified as parameters. They should satisfy the requirement that the time for an ancestral node is no smaller than that for a descedent node. When the optimum values of such two time parameters are equal, that is, when the tree converges to a multifurcating tree with the length of the branch separating the two nodes approaching zero, my program may not converge to the optimum values. What happens in the algorithm is that the time parameter that is supposed to be smaller wants to increase and the parameter that is supposed to be larger wants to decrease when they are already equal. In such a case the algorithm will give up, and fail to find the optimum values that could be obtained if the two time paramters were hold equal and changed together. I hope to fix this problem soon.

The problem, however, is not difficult to detect and get around. If estimates of two time parameters are equal under the clock model, this problem probably has occurred. To get around this problem, you can fit the same model to the multifurcating tree that that tree seems to be converging to.

9 Results from an example data set

Mainly for error-checking, the 895-bp mtDNA data of Brown *et al.* (1982) (`brown.nuc` in the package) is analyzed using a subset of models implemented in `baseml` and `basemlg`. The results are summarized in the table below. Part of the output file running the HKY85+C model were listed in §5.1. The five species are human (H), chimpanzee (C), gorilla (G), orangutan (O) and gibbon (Gb), and there are 895 nucleotides in the sequence (one site is omitted from all sequences).

Model	Tree	l	κ	α	ρ	c_2	c_3	c_4
JC69 (0)	T_3	-2913.739						
	T_1	-2914.115						
F81 (3)	T_3	-2842.670						
	T_1	-2843.877						
K80 (1)	T_1	-2748.411	8.651					
F84 (4)	T_1	-2667.076	4.344					
HKY85 (4)	T_1	-2665.422	9.390					
TN93 (5)	T_1	-2665.358	9.253					
			9.718					
REV (8)	T_1	-2658.221	*					
JC69+dG (1)	T_1	-2902.410		1.002				
F81+dG (4)	T_3	-2826.763		0.768				
	T_1	-2827.238		0.766				
K80+dG (2)	T_1	-2726.377	11.004	0.543				
F84+dG (5)	T_1	-2622.381	11.037	0.247				
HKY85+dG (5)	T_1	-2622.377	21.978	0.252				
F84+Γ (5)	T_1	-2623.059	11.127	0.248				
HKY85+Γ (5)	T_1	-2623.078	22.270	0.252				
JC69+C (3)	T_3	-2819.343				0.497	3.186	0.701
	T_1	-2821.303				0.458	2.983	0.664
F81+C (6)	T_3	-2739.744				0.490	3.430	0.689
	T_1	-2742.479				0.452	3.214	0.652
K80+C (4)	T_1	-2650.209	9.491			0.446	3.217	0.653
F84+C (7)	T_1	-2542.993	5.799			0.396	3.985	0.619
HKY85+C (7)	T_1	-2542.948	12.055			0.378	3.837	0.619
JC69+C+dG (4)	T_3	-2819.110		9.568		0.495	3.233	0.700
	T_1	-2820.916		7.583		0.456	3.033	0.662
F81+C+dG (7)	T_3	-2739.040		5.058		0.486	3.545	0.686
	T_1	-2741.506		4.376		0.447	3.327	0.648
K80+C+dG (5)	T_1	-2645.048	11.319	1.462		0.421	3.786	0.633
F84+C+dG (8)	T_1	-2532.850	9.239	0.855		0.338	5.589	0.579
HKY85+C+dG (8)	T_1	-2532.864	18.850	0.849		0.318	5.387	0.573
F84+C+Γ (8)	T_1	-2532.581	9.369	0.844		0.337	5.675	0.577
HKY85+C+Γ (8)	T_1	-2532.626	19.059	0.842		0.317	5.465	0.572
JC69+C+AdG (5)	T_3	-2811.485		2.838	0.815	0.490	3.342	0.718
	T_1	-2812.813		2.741	0.817	0.451	3.117	0.684
F81+C+AdG (8)	T_3	-2730.711		2.354	0.788	0.481	3.674	0.711
	T_1	-2732.805		2.280	0.788	0.444	3.428	0.675
K80+C+AdG (6)	T_1	-2635.483	11.239	1.424	0.721	0.424	3.751	0.674
F84+C+AdG (9)	T_1	-2524.244	8.744	0.899	0.603	0.343	5.418	0.639
HKY85+C+AdG (9)	T_1	-2524.655	17.658	0.918	0.606	0.320	5.168	0.630

[1] Γ: the continuous gamma model for rates at sites (Yang, 1993) (`basemlg`);

[2] dG: the discrete-gamma model (Yang, 1994c) for variable rates at sites. Eight categories (`ncatG` = 8) of rates are assumed;

[3] AdG: the auto-discrete-gamma model for variable and dependent rates at sites (Yang, 1995);

[4] C: Sites in the sequence are grouped into four classes with different rate parameters $c_1 = 1$, c_2, c_3, and c_4 assigned to them. These include sites at the first, second, and third codon positions within the protein-coding regions, and sites within the tRNA-coding region respectively. The substitution pattern is assumed to be the same at codon positions (`Mgene`=2).

The nucleotide frequencies are more or less homogeneous across species and the average frequencies are 0.2532 (T), 0.3289 (C), 0.3120 (A) and 0.1059 (G); these are taken as estimates of the frequency parameters in the F81, F84, HKY85, TN93 and REV models. There are 613 sites (68.5%) that have identical nucleotides across species.

All models except the continuous gamma models (Γ) are fitted using `baseml` with `runmode` = 2. The "maximum likelihood" tree is T_1 = (((H,C),G),O,Gb), except for a few cases involving the JC69 and F81 models where the best tree is T_3 = (((C,G),H),O,Gb). Other trees including T_0 = ((H,C,G),O,Gb) and T_2 = (((H,G),C),O,Gb) are never the best for the models considered here. The molecular clock is not assumed and so all trees are unrooted. The number in parenthesis is the number of common parameters in the model, not including the seven branch lengths in the tree which are not shown. The continuous gamma analyses (Γ) are performed using `basemlg`; only the four trees T_0, T_1, T_2, T_3 were evaluated using `runmode` = 0, with initial values obtained from the dG models (see §3.3). The results show the similarity in the likelihood values and parameter estimates between the discrete and continuous gamma models. (HKY85+Γ and HKY85+C+Γ are quite slow with these data of five sequences.) The transition/transversion rate bias is very severe for these mtDNA sequences and so the K80 models have higher likelihood values than the F81 models although the latter involve more parameters. F84 involves less calculation than HKY85, but its fit is very similar to HKY85; κ in the two models has different interpretations (see equations 4 and 5 in §6.1). Adding the gamma model of rates across sites does not improve the JC69+C or F81+C models by much, and so there is no point in introducing serial correlation of rates for sites (JC69+C+AdG or F81+C+AdG). The REV model involve five rate parameters and their estimates are a = 0.950, b = 0.083, c = 0.002, d = 0.150, e = 0.079 (equation 7); TN93 has two rate parameters, with estimates κ_1 = 9.253 and κ_2 = 9.718 (equation 6).

The log-likelihood values can be compared to test different models, using the χ^2 distribution for the likelihood ratio statistic. For example, the following table lists results for testing the rate constancy among sites with different substitution models assumed. The differences are significant by any substitution model, but simple models like JC69 or F81 tend to overestimate α and to underestimate the likelihood difference ($\Delta\ell$), giving the wrong impression that the rate variation among sites is not very important for the data.

Models compared	Tree	$\hat{\alpha} \pm$ SE	$2\Delta\ell$
JC69+dG vs. JC69	T_1	1.002 ± 0.270	23.41
F81+dG vs. F81	T_3	0.766 ± 0.182	31.81
K80+dG vs. K80	T_1	0.543 ± 0.126	44.07
HKY85+dG vs. HKY85	T_1	0.252 ± 0.056	86.09

Acknowledgments

I thank Nick Goldman, Adrian Friday, and Sudhir Kumar for many useful comments on versions of this document which accompanied different versions of the program package. I thank Tianlin Wang for allowing me to use his eigen routine in the package. Development of the program package has been partially supported by a grant from the National Natural Science Foundation (China) to Z. Y., NSF and NIH grants to M. Nei, and NIH grant to M. Slatkin.

References

Brown, W. M., E. M. Prager, A. Wang, and A. C. Wilson. 1982. Mitochondrial DNA sequences of primates, tempo and mode of evolution. *Journal of Molecular Evolution* **18**:225-239.

Dayhoff, M. O., R. M. Schwartz, and B. C. Orcutt. 1978. A model of evolutionary change in proteins. In *Atlas of Protein Sequence and Structure*, Vol 5, Suppl. 3 (ed M. O. Dayhoff), National Biomedical Research Foundation, Washington D.C., pp. 345-352.

Edwards, A. W. F. 1970. Estimation of the branch points of a branching diffusion process (with Discussion). *Journal of Royal Statistical Society*, *B* **32**:155- 174.

Felsenstein, J. 1981. Evolutionary trees from DNA sequences: a maximum likelihood approach. *Journal of Molecular Evolution* **17**:368-376.

Felsenstein, J. 1985. Confidence limits on phylogenies: an approach using the bootstrap. *Evolution* **39**:783-791.

Felsenstein, J. 1993. *Phylogenetic Inference Package (PHYLIP), Version 3.5*. University of Washington, Seattle.

Gillespie, J. H. 1986. Rates of molecular evolution. *Annual Review of Ecology and Systematics* **17**:637-65.

Goldman, N. 1990. Maximum likelihood inference of phylogenetic trees, with special reference to a Poisson process model of DNA substitution and to parsimony analysis. *Systematic Zoology* **39**:345-361.

Goldman, N. 1993a. Statistical tests of models of DNA substitution. *Journal of Molecular Evolution* **36**:182-198.

Goldman, N. 1993b. Simple diagnostic statistical tests of models for DNA substitution. *Journal of Molecular Evolution* **37**:650-661.

Goldman, N., and Z. Yang. 1994a. A codon-based model of nucleotide substitution for protein-coding DNA sequences. *Molecular Biology and Evolution* **11**:725- 736.

Goldman, N., and Z. Yang. 1994b. Models of DNA substitution and the discrimination of evolutionary parameters. In: *Proceedings of the XVII*th *International Biometrics Conference, Vol. I: Invited Papers.* Biometrics Society, Hamilton, Ontario, Canada, pp. 407-420.

Grantham, R. 1974. Amino acid difference formula to help explain protein evolution. *Science* **185**:862-864.

Hartigan, J. A. 1973. Minimum evolution fits to a given tree. *Biometrics* **29**: 53-65.

Hasegawa, M., and H. Kishino. 1989. Confidence limits on the maximum likelihood estimation of the hominoid tree from mitochondrial DNA sequences. *Evolution* **43**:672-677.

Hasegawa, M., H. Kishino, and T. Yano. 1985. Dating the human-ape splitting by a molecular clock of mitochondrial DNA. *Journal of Molecular Evolution* **22**:160-174.

Jones, D.T., W. R. Taylor, and J. M. Thornton. 1992. The rapid generation of mutation data matrices from protein sequences. *Computer Application in Biosciences* **8**:275-282.

Kimura, M. 1980. A simple method for estimating evolutionary rates of base substitutions through comparative studies of nucleotide sequences. *Journal of Molecular Evolution* **16**:111-120.

Kishino, H., and M. Hasegawa. 1989. Evaluation of maximum likelihood estimate of the evolutionary tree topologies from DNA sequence data, and the branching order in Hominoidea. *Journal of Molecular*

Evolution **29**:170-179.

Kishino, H., and M. Hasegawa. 1990. Converting distance to time: an application to human evolution. *Methods in Enzymology* **183**:550-570.

Kishino, H., T. Miyata, and M. Hasegawa. 1990. Maximum likelihood inference of protein phylogeny and the origin of chloroplasts. *Journal of Molecular Evolution* **31**:151-160.

Kumar, S., K. Tamura, and M. Nei. 1993. *MEGA: Molecular Evolutionary Genetics Analysis.* The Pennsylvania State University, University Park, PA 16802.

Muse, S. V., and B. S. Gaut. 1994. A likelihood approach for comparing synonymous and nonsynonymous nucleotide substitution rates, with application to chloroplast genome. *Molecular Biology and Evolution* **11**:715-724.

Nei, M., and T. Gojobori. 1986. Simple methods for estimating the numbers of synonymous and nonsynonymous nucleotide substitutions. *Molecular Biology and Evolution* **3**:418-426.

Rannala, B. and Z. Yang. 1996. Probability distributions of molecular evolutionary trees: a new method of phylogenetic inference. *Journal of Molecular Evolution* **43**:304-311.

Rodriguez, F., J. F. Oliver, A. Marin, and J. R. Medina. 1990. The general stochastic model of nucleotide substituions. *Journal of Theoretical Biology* **142**:485-501.

Swofford, D. L. 1993. *Phylogenetic Analysis Using Parsimony (PAUP), Version 3.2.* University of Illinois, Champaign.

Swofford, D. L., and G. J. Olsen. 1990. Phylogeny reconstruction. In *Molecular systematics* (eds. D. M. Hillis and C. Moritz), Sinauer Associates Inc. Massachusetts. pp. 411-501.

Swofford, D. L., G. J. Olsen, P. J. Waddel, and D. M. Hillis. 1996. Phylogeny Inference. Pp. 411-501 in D. M. Hillis, C. Moritz, and B. K. Mable eds. Molecular Systematics, 2nd ed. Sinauer Associates, Sunderland, Massachusetts.

Tamura, K., and M. Nei. 1993. Estimation of the number of nucleotide substitutions in the control region of mitochondrial DNA in humans and chimpanzees. *Molecular Biology and Evolution* **10**:512-526.

Tavaré, S. 1986. Some probabilistic and statistical problems on the analysis of DNA sequences. In *Lectures in Mathematics in the Life Sciences*, Vol. 17, pp. 57-86.

Thorne, J. L., H. Kishino, and J. Felsenstein. 1991. An evolutionary model for maximum likelihood alignment of DNA sequences. *Journal of Molecular Evolution* **33**:114-124. (Erratum: *Journal of Molecular Evolution* **34:**91 [1992].)

Thorne, J. L., H. Kishino, and J. Felsenstein. 1992. Inching toward reliability: An improved likelihood model of sequence evolution. *Journal of Molecular Evolution* **34**:3-16.

Wakeley, J. 1993. Substitution rate variation among sites in hypervariable region 1 of human mitochondrial DNA. *Journal of Molecular Evolution* **37**:613-623.

Wakeley, J. 1996. The excess of transitions among nucleotide substitutions: new methods of estimating transition bias underscore its significance. *Trends in Ecology and Evolution* **11**:158-163.

Yang, Z. 1993. Maximum likelihood estimation of phylogeny from DNA sequences when substitution rates differ over sites. *Molecular Biology and Evolution* **10**:1396-1401.

Yang, Z. 1994a. Estimating the pattern of nucleotide substitution. *Journal of Molecular Evolution* **39**:105-111.

Yang, Z. 1994b. Statistical properties of the maximum likelihood method of phylogenetic estimation and comparison with distance matrix methods. *Systematic Biology* **43**:329-342.

Yang, Z. 1994c. Maximum likelihood phylogenetic estimation from DNA sequences with variable rates over sites: approximate methods. *Journal of Molecular Evolution* **39:**306-314.

Yang, Z. 1995. A space-time process model for the evolution of DNA sequences. *Genetics* **139**:993-1005.

Yang, Z. 1996*a*. Maximum likelihood models for combined analyses of multiple sequence data. *Journal of Molecular Evolution* **42**:587-596.

Yang, Z. 1996*b*. The among-site rate variation and its impact on phylogenetic analyses. *Trends in Ecology and Evolution* **11**:367-372.

Yang, Z., and S. Kumar. 1996. New parsimony-based methods for estimating the pattern of nucleotide substitution and the variation of substitution rates among sites and comparison with likelihood methods. *Molecular Biology and Evolution* **13**:650-659.

Yang, Z., and B. Rannala. submitted. Bayesian phylogenetic inference using DNA sequences: Markov chain Monte Carlo methods. *Molecular Biology and Evolution*

Yang, Z., and D. Roberts. 1995. On the use of nucleic acid sequences to infer early branchings in the tree of life. *Molecular Biology and Evolution* **12**:451-458.

Yang, Z., and T. Wang. 1995. Mixed model analysis of DNA sequence evolution. *Biometrics* **51**:552-561.

Yang, Z., N. Goldman, and A. E. Friday. 1994. Comparison of models for nucleotide substitution used in maximum likelihood phylogenetic estimation. *Molecular Biology and Evolution* **11**:316-324.

Yang, Z., N. Goldman, and A. E. Friday. 1995. Maximum likelihood trees from DNA sequences: a peculiar statistical estimation problem. *Systematic Biology* **44**:384-399.

Yang, Z., S. Kumar, and M. Nei. 1995. A new method of inference of ancestral nucleotide and amino acid sequences. *Genetics* **141**:1641-1650.

Zharkikh, A. 1994. Estimation of evolutionary distances between nucleotide sequences. *Journal of Molecular Evolution* **39**:315-329.