Estimating selection pressures on alignments of coding sequences

Analyses using *HyPhy*

Edited by
Sergei L. Kosakovsky Pond, Art F.Y. Poon, and Simon D.W. Frost

1

Theory

1.1 Introduction

Understanding the selective pressures that have shaped genetic variation is a central goal in the study of evolutionary biology. As nonsynonymous mutations can directly affect protein function, they are more likely to influence the fitness of an organism than mutations that leave the amino acid sequence unchanged (*i.e.* synonymous mutations). Under negative or purifying selection, less 'fit' nonsynonymous substitutions accumulate more slowly than synonymous substitutions, and under diversifying or positive selection, the converse is true. Therefore, an important concept in the analysis of coding sequences is that the comparison of relative rates of nonsynonymous (β) and synonymous (α) substitutions can provide information on the type of selection that has acted on a given set of protein-coding sequences. The ratio $\omega = \beta/\alpha$ (also referred to as dN/dS or K_A/K_S) has become a standard measure of selective pressure [1]; $\omega \approx 1$ signifies neutral evolution, $\omega < 1$ - negative selection and $\omega > 1$ - positive selection.

There are five fundamental questions which can be answered with existing methods and software tools that estimate such substitution rates.

- Is there evidence of selection operating on a gene? We can address this question by testing whether $\beta \neq \alpha$ for some regions in the sequence.
- Where did selection happen? We can identify sequence regions or individual codons under selection, and determine the level of statistical significance.
- When did selection happen? We can estimate at what point in the evolutionary past (i.e. along which branch of the phylogenetic tree) non-neutral evolution occurred.
- What types of substitutions were selected for or against? We can clas-

sify amino-acid substitutions (*e.g.* Leucine↔Valine) into those which were preferred and those which were suppressed.

- Are selective pressures different between genes/samples? Given two genes from the same set of taxa, or two samples of taxa covering the same gene, we can determine whether they evolved under similar or different selective pressures.

Our ability to address these questions depends on the accurate estimation of nonsynonymous and synonymous rates, which was recently facilitated by the adoption of codon models of evolution within a phylogenetic maximum likelihood framework. We begin by providing a simple justification for why such complex models are necessary, despite the steep computational cost.

The unit of evolution. The structure of the genetic code forces realistic evolutionary models to consider triplets of nucleotides, *i.e.* codons, to be the basic unit of evolution. For example, the common assumption that the rates of evolution of the third codon position can serve as a proxy for synonymous substitution rates is a rather crude approximation. If x_i and y_i denote nucleotides in the i^{th} position in a codon, then among all possible substitutions at the third codon position starting from codon $n_1n_2n_3$ and ending in codon $n_1n_2m_3$ ($n_3 \neq m_3$ and $n_1n_2m_3$ is not a stop codon), 50 are nonsynonymous and 126 are synonymous, based on the universal genetic code †. More importantly, whether or not $x_3 \rightarrow y_3$ is a synonymous or a nonsynonymous substitution, depends on the *context* of n_1n_2. For example $GGx_3 \rightarrow GGy_3$ is always synonymous, whereas $CAx_3 \rightarrow CAy_3$ is synonymous if $x_3 \rightarrow y_3$ is a *transition* ($A \leftrightarrow G$ or $C \leftrightarrow T$), and nonsynonymous otherwise. Probabilistic codon substitution models [2, 3] offer a natural and formal way to take into account such context dependency.

Estimating the neutral expectation. Before one can begin testing for selection, it is necessary to correctly assess what would happen under the neutral scenario. In the simplest case, one would like to know what proportion of random substitutions (measured per codon, to correct for the length of the alignment) would be synonymous and what proportion would be nonsynonymous. Simply estimating the raw numbers of synonymous and nonsynonymous substitutions and comparing them to detect selection will almost always fail because: (i) the structure of the genetic code is such that a larger proportion of random substitutions are nonsynonymous rather than synonymous; (ii) some kinds of substitutions (*e.g.* transitions) are more

† Run the *HyPhy* script `CountSubstitutions.bf` to tabulate various kinds of substitutions given a genetic code. You will need to download and install HyPhy (`http://www.hyphy.org/`), execute the `Analysis>Standard Analyses` menu option, choose `Phylohandbook.bf` from `Miscellaneous` rubrik and finally select the appropriate analysis from the list.

frequent than others; and (iii) some codons are more frequent than others. The effect of each factor can be significant. For example, using the universal genetic code, assuming neutral evolution, equal codon frequencies (1/61 for each non-stop codon) and no biases in nucleotide substitution rates (*i.e.* the Jukes-Cantor [4] model), 25.5% substitutions are expected to synonymous and 74.5% - nonsynonymous (`NeutralExpectation.bf`). If transitions are assumed to happen at the rate 5 times that of transversion (such ratios are quite common in biological data), these numbers change to 30.9% and 69.1% respectively. Furthermore, taking codon frequencies to be unequal (*e.g.* the distribution estimated from the HIV-1 *pol* gene) alters the counts to 27.1% and 72.9%. Hence, if one were to infer that a given sequence sample contained on average 2 nonsynonymous and 1 synonymous substitutions per codon, one would have to account for all the factors influencing the neutral expectation before making a deduction about what selection mode - neutral, positive or negative - acted on the sample.

There are relatively sophisticated methods based on *codon distances*, which attempt to estimate the neutral expectation and evolutionary rates by comparing pairs of homologous sequences [5], and in certain cases, these estimates can be quite accurate. However, there are several statistical issues inherent to all such methods [6], in particular they are difficult to generalize to comparing more than two sequences at a time. The use of codon substitution models can easily account for all the above confounding factors when estimating substitution rates, and can represent neutral evolution simply by setting the synonymous and nonsynonymous rates to be the same ($\alpha = \beta$), or, equivalently, setting their ratio $\omega = \beta/\alpha$ to one.

Taking phylogenies into account. Genetic variation found in extant sequences is a combination of independent substitutions in different lineages and substitutions inherited by related sequences from a common ancestor. To estimate substitution rates correctly one must be able to separate these two effects. Consider a simple example shown in Figure 1.1, where the same five extant codons at site 142 in Influenza A/H5N1 haemagglutinin are analyzed using two different phylogenies, one a maximum likelihood tree and the other a star topology, which is equivalent to the assumption made for naïve pairwise sequence comparison (*e.g.* no substitutions are shared by descent). Even for such a small sample, some of the conclusions (*e.g.* the minimal number of substitutions needed to explain the observed pattern) clearly may be affected by which phylogeny was used to relate the sequences. Without going into much detail, however, we note that in practice, the inference of substitution rates using codon substitution models tends to be "robust" to some errors in the phylogenetic tree, *i.e.* so long as the tree is not "too

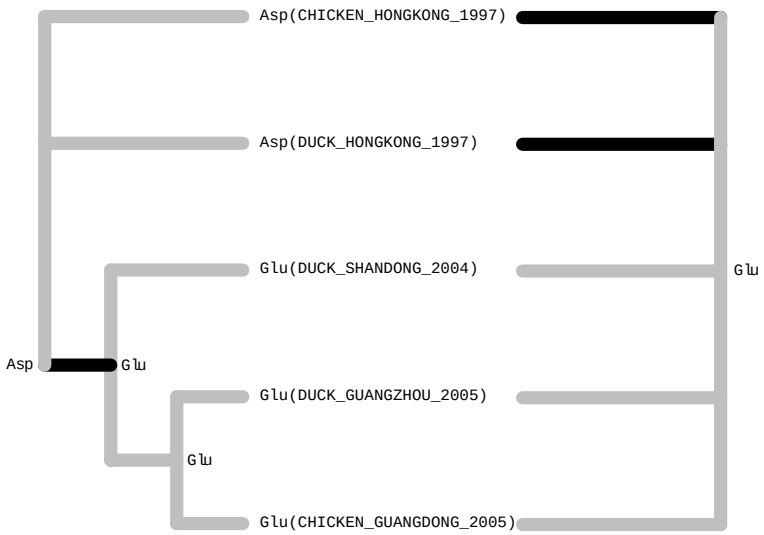

Fig. 1.1. Effect of phylogeny on estimating synonymous and nonsynonymous substitution counts in a dataset of Influenza A/H5N1 haemagglutinin sequences. Using the maximum likelihood tree on the left, the observed variation can be parsimoniously explained with one nonsynonymous substitution along the darker branch, whereas the star tree on the right involves at least two.

wrong", rate estimates will not change much from tree to tree. As a simple illustration, `EffectOfTopology.bf` examines how the estimate of ω changes over all 15 possible trees relating the 5 influenza sequences (other small data sets can also be examined with this file). The likelihood scores differ by over 100 points between the best- and the worst-fitting trees, yet the ω estimate ranges only from 0.22 to 0.26. Applying grossly incorrect phylogenies (or assuming the lack of phylogenetic relatedness via pairwise comparisons) may have a much more profound effect on other types of inference, or larger datasets, as we will show later.

Different types of selection. The term 'positive selection' encompasses several different evolutionary processes. It is critically important to distinguish between the two primary ones: directional and diversifying selection, because specific comparative methods must be used to identify each kind. Directional selection operating at a given position in a gene is manifested by concerted substitution towards a particular residue, which, given enough time, will result in a selective sweep, *i.e.* fixation of the new allele in the population. For example, when wildtype HIV-1 infects a number of different patients receiving the same antiretroviral drug, there will be strong selective pressure on the virus to independently acquire those mutations that

confer drug resistance. For many early antiretroviral drugs, a single non-synonymous substitution was often sufficient for the acquisition of strong drug resistance, explaining why early HIV treatments were ineffectual [7]. If one were to sample these viruses after a complete selective sweep, when the virus population in each host has fixed the resistance mutation, there would be no remaining evidence of any selection having taken place. Diversifying selection, on the other hand, results from a selective regime whereby amino-acid diversity at a given site is maintained in the population [8]. In HIV-1, this might occur at those codon positions that are the targets of host immune response. As immune systems in different hosts generally vary in their ability to recognize and target specific viral antigens, some viruses may be under selective pressure to evolve immune escape, while others may maintain wildtype residues.

In the rest of the chapter, we show how probabilistic models of codon substitution can be used to identify various types of selection pressures. While these methods are powerful and represent the current state-of-the-art in coding sequence analysis, it is important to realize that these models are able to recapitulate only some of the actual evolutionary processes that shape sequence evolution. There are many remaining assumptions and simplifications, which are often made to retain computational feasibility, and in certain cases the methods can be misled by recombination, small sample size, or biological processes not included in the model. We take great care to highlight such possible shortcomings, because it is our firm belief that the knowledge of a methods' limitations is as important as the knowledge of their power.

1.2 Prerequisites

In order to conduct an analysis of selection on a gene, one needs a multiple sequence alignment (Section II of this book), and an underlying phylogenetic tree (Section III), or in the case of recombination (Section VI), multiple phylogenetic trees (one for each non-recombinant segment).

When preparing alignments for codon analyses, one should ensure that the alignment process does not introduce frameshifts and preserves codons (*i.e.* by only inserting/deleting nucleotides in multiples of three). Hence it is advisable to carry out sequence alignment on *translated* protein sequences, and then map aligned residues to codons.

A number of algorithms that have been proposed in order to estimate a phylogenetic tree, including distance-based methods such as neighbor-joining [9], maximum likelihood based methods [10], and Bayesian appro-

aches. Most of the time, rate estimates derived with a substitution model are robust to the details of the phylogenetic tree. An important exception to this occurs when recombinant sequences are present. Recombination has relatively little impact on estimates of global rates, but can have a profound effect on estimates of site-to-site and branch-to-branch variation in selection pressure. In order to accommodate recombination in a phylogenetic context, it is necessary to split the alignment into non-recombinant sequence fragments first. We have implemented a method called Genetic Algorithms for Recombination Detection (GARD [11] `http://www.datamonkey.org/GARD`), that uses a genetic algorithm to identify nonrecombinant fragments; several other programs can be employed to do the same. Once these have been identified, selection analyses can be run separately on each fragment, or jointly by assuming that some parameters are shared between fragments.

1.3 Codon substitution models

The first tractable codon models were proposed independently by Goldman and Yang [3] and Muse and Gaut [2], and published in the same issue of *Molecular Biology and Evolution*. The process of substituting a non-stop codon $x = n_1n_2n_3$ with another non-stop codon $y = m_1m_2m_3$ over a time interval $t > 0$ is described by a continuous time, homogeneous, stationary and time-reversible Markov process, described by the *transition matrix*: $T(t)$, whose (i, j) entry contains the probability of replacing codon i with codon j over time interval $t \geq 0$. Stop codons are disallowed as evolutionary states since their random introduction in an active gene is overwhelmingly likely to destroy the function of the translated protein.

The (i, j) element or the rate matrix (Q^{MG94}) for the generalized Muse-Gaut (MG94) model defines the instantaneous **rate** of replacing codon i with codon j ($i \neq j$).

$$q_{ij}^{MG94} = \begin{cases} \theta_{mn}\alpha_s^b\pi_n^p, & i \rightarrow j \text{is a one-nucleotide synonymous substitution} \\ & \text{from nucleotide } m \text{ to nucleotide } n \text{ in codon position } p, \\ \theta_{mn}\beta_s^b\pi_n^p, & i \rightarrow j \text{is a one-nucleotide nonsynonymous substitution} \\ & \text{from nucleotide } m \text{ to nucleotide } n \text{ in codon position } p, \\ 0, & \text{otherwise.} \end{cases}$$

Mathematical properties of codon substitution models. A random process $X(t)$ (in our case, taking values in the space of all non-stop codons in a genetic code) is **Markov** if the behavior of the process in the future is determined entirely by its current state. Formally, $Pr\{X(t_0+s)=x|X(t_0)\} = Pr\{X(t_0+s)=x|\{X(t)\}, t \leq t_0\}$ for all $t_0, s \geq 0$ and x. Markov processes are memoryless, and this property is critically important for efficient evaluation of phylogenetic likelihood [10]. $X(t)$ is **time-homogeneous**, if the behavior of the process does not depend on the starting time, *i.e.* $Pr\{X(t_0+s)=x|X(t_0)=y\} = Pr\{X(t_1+s)=x|X(t_1)=y\}$ for all $t_0, t_1, s \geq 0$, x, y. This assumption allows the process to be described by a single rate matrix - the derivative of the transition matrix T as time approaches 0: $Q = \lim_{t \downarrow 0}(T(t)-I)/t$, where I is an identity matrix. In order to recover $T(t)$ from Q, one computes the matrix exponential $\exp(Qt)$. Because the computational complexity of matrix exponentiation scales as the cube of the matrix dimension, codon based models require roughly $(61/4)^3 \approx 3500$ more operations than nucleotide models. The process is **stationary** if the expected distribution of states (codons) is the same throughout the tree. Formally, the stationary distribution π satisfies the matrix equation $\pi T(t) = \pi$ (or, equivalently $\pi Q = 0$) for all $t \geq 0$, and is subject to $\sum_i \pi_i = 1$. In a phylogenetic context, stationarity means that the average codon composition of sequences does not change throughout the tree. Hence comparing homologous sequences which may have substantially different codon usage biases, for example, may require the use of a model which allows frequencies to evolve throughout the tree. In particular, it assumes that all extant sequences have statistically indistinguishable codon compositions. Lastly, a process is **time-reversible** if it is stationary and, for all codons i, j and times $t \geq 0$, if the "detailed balance" relationship holds: $\pi_i Pr\{i \rightarrow j|t\} = \pi_j Pr\{j \rightarrow i|t\}$. Intuitively, in a reversible process, the "flow" from i to j is balanced by the reciprocal "flow" from j to i. This assumption is effectively saying that we know nothing about the direction of time - the origin (root) of the phylogenetic tree can be placed anywhere in the tree without altering the evolutionary process. Time-reversibility greatly reduces the number of estimable parameters, and allows one to consider only unrooted trees, thus removing one of the branches, and reducing the computational burden.

π_n^p denotes the frequency of nucleotide $n \in \{A, C, G, T\}$ in codon position $p = 1, 2, 3$. Synonymous and nonsynonymous substitution rates α_s^b and β_s^b may depend both on the alignment site (s) and the branch of the tree (b), as denoted by the sub/superscript. For example, the synonymous $ACG \rightarrow ACT$ substitution involves the change $G \rightarrow T$ in the third codon position, and its corresponding rate is $q_{ACG,ACT} = \theta_{GT}\alpha_s^b \pi_T^3$. For the remainder of the chapter, we assume that these frequencies are estimated by counts from the data. Although it is easy to estimate these frequencies by maximum likelihood, in most practical situations the observed frequencies are used, as this approximation (which is usually very good) saves computational time. θ_{mn} corrects for the nucleotide substitution bias, and because of time

reversibility $\theta_{mn} = \theta_{nm}$. In the simplest case, all $\theta_{mn} = 1$, reducing to the original Muse-Gaut model, and in the most general, six rates can be specified; however, because the phylogenetic likelihood function depends only on *products* of rates and evolutionary times $q_{xy}t$, only five of those can be estimated, hence we arbitrarily set one of the rates (we choose θ_{AG}) to one, and all other nucleotide rates are estimated *relative* to the θ_{AG} rate. Diagonal entries of the rate matrix are defined by $q_{ii} = -\sum_{j \neq i} q_{ij}$, ensuring that each row of the transition matrix $T(t)$ forms a valid probability distribution.

The model assumes that point mutations alter one nucleotide at a time, hence most of the instantaneous rates (3134/3761 or 84.2% in the case of the universal genetic code) are 0. This restriction, however, does not mean that the model disallows any substitutions that involve multiple nucleotides (e.g $ACT \rightarrow AGG$). Such substitutions must simply be realized via several single nucleotide steps. In fact the (i, j) element of $T(t) = \exp(Qt)$ sums the probabilities of all such possible pathways of length t. For a model which does allows multiple nucleotides to be substituted at once, we refer the reader to a recent paper by Whelan and Goldman [12].

Stationary codon frequencies for the MG94 model are given by $\pi(x = n_1 n_2 n_3) = \pi^1_{n_1} \pi^2_{n_2} \pi^3_{n_3} / N$ and include 9 parameters (also referred to as the $F3 \times 4$ estimator for those familiar with the PAML [13] package). N is the normalizing constant, which accounts for the absence of stop codons and is defined as $N = 1 - \sum_{(m_1 m_2 m_3 \text{ is a stop codon})} \pi^1_{m_1} \pi^2_{m_2} \pi^3_{m_3}$. In the original MG94 paper [2], nucleotide frequencies were pooled from all three codon positions, *i.e.* $\pi^1_n = \pi^2_n = \pi^3_n$ for all four nucleotides ($F1 \times 4$ estimator), yielding three frequency parameters.

The GY94 model, first implemented in the PAML package [13] is similar to MG94, but it differs in two choices of model parameterization. Firstly, the synonymous evolutionary rate is set to 1, making $\omega = \beta/\alpha = \beta$. As we will point out later, it is important to tease apart the effects of both α and β on the estimates of their ratio (or difference). From a statistical perspective, ratios are notoriously difficult to estimate, especially when the denominator is small. Secondly, in GY94, rates are proportional not to the frequency of target nucleotides (π^p_n), but rather to the frequency of target *codons*. In most practical cases, this distinction has a minor effect on the estimation of substitution rates α and β, and for the remainder of the chapter we will primarily focus on the MG94 model, albeit nearly everything we discuss can be run with the GY94 model instead.

The rest of the section concerns itself mostly with describing different

methods for estimating α_s^b and β_s^b. There is no biological reason to assume that the selective pressures are the same for any two branches or any two sites, however, one's ability to estimate these quantities from finite data demands that we consider simplified models. Indeed, for N sequences on S sites, there would be $2S(2N-3)$ parameters to estimate, if each branch/site combination has its own rate - a number greater than the most liberally counted number of samples (each branch/site combination: $S(2N-3)$) available in the alignment. Clearly, a model with more parameters than available data points is going to grossly overfit the data, and no measure of statistical sophistication can do away with this fundamental issue.

1.4 Simulated data: how and why

Biological data are very complex, and even the most sophisticated models that we can currently propose are at best scraping the surface of biological realism. In order to evaluate the statistical properties of parameter estimates and various testing procedures, it is therefore imperative to be able to generate sequence data that evolved according to a pre-specified Markov process with known rates. At the very basic level, if enough data with known rates are generated, the inference procedure (with the same model as the one used to generate the data) should be able to return correct parameter estimates on average (consistency) and render accurate estimates of rate parameters (efficiency). The powerful statistical technique of *bootstrapping* is dependent on simulated data generated either by resampling the original data (nonparametric bootstrapping [14, 15]) or by simulating a substitution model on a given tree (parametric bootstrapping [16]). Very large codon data sets can be simulated quickly, enabling the study and validation of various statistical properties of different estimation procedures.

Simulating data using a codon substitution model. Let there be a tree τ, where each branch b is endowed with an evolutionary model, described (as in the previous section), by a rate matrix Q^b. As each branch has an associated evolutionary time t^b, one can compute the transition matrix for branch b, $T^b = \exp Q^b t^b$. Suppose that S codon sites must be simulated and (potentially) each site may have its own collection of Q^b attached to branches of τ. To generate an alignment column for codon position s, the following steps are followed.

- Select the state at the root of the tree, by randomly drawing a codon from the equilibrium distribution π. This can be done by first sampling a random number r uniformly from $[0, 1]$ (using a pseudo-random number generator). Next, one computes the cumulative probability of codon i, defined as $f(i) = \sum_{j=1}^{i} \pi_j$ and choosing codon i_0 so that $f(i_0) \geq r$ and $f(i_0 - 1) \leq r$ with the convention that $f(0) = 0$.
- Traverse the tree τ using the *pre-order* algorithm (which guarantees that branch b is visited after all of its parent branches)
- At each branch b the state of the parent node is known (call it p^b) when it is visited, hence one generates the codon at the downstream end of branch b by sampling a codon randomly from the conditional distribution defined by p^b-th row of the transition matrix T^b, *i.e.* given a starting state of the Markov process at branch b, one randomly picks the ending state (after time t^b), based on how likely each one is. The same sampling procedure as described in step 1 can be used.
- Stop when all the leaves of the tree (extant sequences) have been labeled.

1.4.1 Distance-based approaches

A number of distance-based approaches have been proposed to estimate the relative rate of nonsynonymous to synonymous substitution, some of which incorporate biased nucleotide frequencies and substitution rates. We use a heavily sited (over 1600 references) method proposed by Nei and Gojobori in 1986 [17] to illustrate the concepts. The cornerstone idea has already been mentioned in Introduction: one estimates the expected ratio of non-synonymous/synonymous substitutions under the neutral model and compares it to the one inferred from the data (we reframe several of the concepts using stochastic models in the following sections). Consider an alignment of two homologous sequences on C codons. For each codon $c = 1 \dots C$ in every sequence, we consider how many of the nine single-nucleotide substitutions leading away from the codon are synonymous (f_c^s), and how many are non-synonymous (f_c^n). For example, $f_{GAA}^n = 8$, because 8/9 one nucleotide substitutions (AAA, CAA, GCA, GGA, GTA, GAC, GAG, GAT, TAA) are non-synonymous (compare to section 1.4.3, for a note on substitutions to a stop codon) and $f_{GAA}^s = 1$ (GAG is the only synonymous change). For every position in the alignment, we average these quantities for the corresponding codon in each sequence, and sum over all positions to arrive at the estimates S (number of synonymous sites) and N (number of non-synonymous sites). N/S provides an estimate of the expected ratio of non-synonymous to synonymous substitutions under the neutral model for the given codon composition of the two sequences. The actual number of synonymous (D_s) and non-synonymous (D_n) substitutions between two sequences is estimated by counting the differences codon by codon, assuming the shortest evolutionary path between the two. If the codons differ by a more than one nucleotide, all shortest paths, obtained by considering all possible orderings of the substitutions (2 if two substitutions are needed, 6 - if three are needed) and averaging the numbers of synonymous and non-synonymous substitutions over all pathways (see 1.6.3 for further insight). One can now estimate mean $dS = D_s/S$ and $dN = D_n/N$ for the entire sequence, and the corresponding ratio $dN/dS = (D_n/D_s)/(N/S)$. The effect of multiple substitutions at a site can be approximated by setting $dS_c = -3/4 \log(1 - 4/3dS)$, and applying an analogous correction to dN. A careful reader will recognize this as the standard Jukes-Cantor [4] estimate of the genetic distance between two sequences, assuming a single substitution rate between all synonymous (and non-synonymous) codons, but it is merely an approximation. For example, it cannot, in principle, handle the case when all evolutionary paths between two observed codons x

and y involves both synonymous and non-synonymous substitutions, since this would imply that different substitution rates apply in parts of the evolutionary past of the sequences. To decide if dN/dS is statistically different from 1, one can, for example, obtain a confidence interval around dN/dS (by bootstrap or using a variance estimator) and examine whether or not the confidence interval overlaps with 1.

While these approaches are useful exploratory tools, especially since they can be run very quickly, they are poorly suited to hypothesis testing, because statistical significance may be difficult to assess, and the effect of phylogenetic relatedness on estimated rates can be strong when rate estimates are based on pairwise sequence comparisons (see section 1.6.3 regarding the adaptation of these ideas to account for phylogenetic relatedness). The *de facto* standard package for distance-based sequence analysis is MEGA (`http://www.megasoftware.net/`), a mature and feature-rich program whose main limitation is that the software can only be run under Microsoft Windows. The multi-platform HyPhy package discussed in the Practice section also provides a number features for distance based estimation.

1.4.2 Maximum likelihood approaches

The very basic - global, or single rate - model posits that α and β do not vary from site-to-site or branch to branch. Clearly, this assumption can not be expected to hold in most biological systems and the model must, therefore, be treated as a rough approximation. As with all phylogenetic maximum likelihood methods, a variant of Felsenstein's pruning algorithm [10] is used to evaluate the probability of extant codon sequences given all model parameters, *i.e.* the likelihood function, and independent parameters are adjusted using a numerical optimization technique to obtain their maximum likelihood estimates (MLEs). There are numerous statistical and philosophical reasons to use maximum likelihood methods for parameter estimation (*e.g.* [18]). For example, assuming that the model which generated the data is the same as the one being fitted, and given enough data (alignment columns), MLEs will be consistent (*i.e.* converge to the true values) and efficient (have minimum variance among all unbiased estimators). For example, Rogers [19] has demonstrated the consistency of maximum likelihood phylogenetic tree estimation for any reversible substitution model with a finite number of states.

Global α and β models are the simplest computationally and contain the fewest number of estimable parameters, hence they are suitable for coarse

data characterization (exploratory analysis), analyses of small samples (a few sequences or very short alignments) or when substitution rates are a nuisance parameter (*i.e.* there are used as a means to estimate something else, *e.g.* phylogeny or ancestral sequences), although more complex models may provide a better result in the latter case. Lastly, the global model serves as a useful null hypothesis to form the basic of testing for spatial or temporal rate heterogeneity.

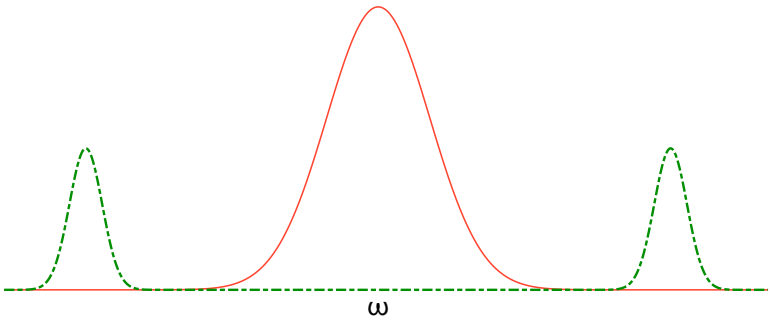

Fig. 1.2. Two different rate distributions (solid and dashed) which have the same mean ω.

When α and β are the primary object of evolutionary analysis, the global model is nearly always a poor choice for inference. Selective regimes may differ from site to site in a gene due to varying functional and structural constraints, selective pressures and other biological processes. Since the global model is only able to estimate the mean, it reveals remarkably little about the unknown distribution of rates. Two genes with mean $\omega = 0.5$ may, for example, have dramatically different distributions of ω across sites (Fig. 1.2), hence it might be erroneous to state, based solely on the equality of the means, that two genes are evolving under similar selective pressures. `WhatsInTheMean.bf` shows how simulated alignments with vastly different distributions of ω yield very similar estimates of mean ω.

1.4.3 Estimating dS and dN

Quite often one may be interested in estimating *evolutionary distances* between coding sequences, usually measured as the expected substitutions per site per unit time, or % divergence. As codon-substitution Markov processes are time-homogeneous, one can use the property that the distribution of waiting times (*i.e.* the time for a process to change its state from codon

i to some other codon) is an exponential distribution with rate parameter defined by the off-diagonal entries of the rate matrix Q, as $r_i = \sum_{j \neq i} q_{ij}$. Recalling that the diagonal elements of the rate matrix Q were defined as $q_{ii} = -r_i$, the expected time to change from i to some other state is $-1/q_{ii}$, *i.e.* an average of q_{ii} changes from i to some other state given over a unit length of time. The total expected number of changes per codon per unit time can be obtained by taking a weighted average over all possible codons

$$E[subs] = -\sum_i \pi_i \hat{q}_{ii},$$

where $\hat{q}$ denotes that the rate matrix Q is evaluated using maximum likelihood estimates for all model parameters. To make codon based distances directly comparable with those obtained from nucleotide models, it is customary to divide the estimates by 3, reflecting the fact that there are three nucleotides in a codon.

The total expected number of substitutions can be decomposed into the sum of synonymous and non-synonymous changes per codon, by summing rate matrix entries which correspond to synonymous and non-synonymous substitutions only as follows:

$$q_{ii} = q_{ii}^s + q_{ii}^{ns} = \sum_{j \neq i,\ \text{j and i are synonymous}} q_{ij} + \sum_{j \neq i,\ \text{j and i are nonsynonymous}} q_{ij},$$

and

$$E[subs] = E[syn] + E[nonsyn] = -\sum_i \pi_i \hat{q}_{ii}^s - \sum_i \pi_i \hat{q}_{ii}^{ns}.$$

In order to convert the expected numbers of substitutions *per codon* to a more customary dN and dS, one must normalize the raw counts by the proportions of synonymous and non-synonymous sites (see below), allowing us to compensate for unequal codon compositions in different alignments.

It is important to realize that $\omega = \beta/\alpha$ is in general *not equal* to dN/dS as defined above, although the two quantities are proportional, with the constant dependent upon base frequencies and other model parameters, such as nucleotide substitution biases. When more than two sequences are involved in an analysis, the computation of genetic distances between any pair of sequences can be carried out by summing the desired quantities (*e.g.* dS and dN) over all the branches in the phylogenetic tree which lie on the shortest path connecting the two sequences in question. An alternative approach is to estimate the quantities directly from a two-sequence analysis, which implicitly assumes that the sequences are unrelated (*e.g.* conform to the star topology). Depending on the strength of phylogenetic signal and the

assumptions of the model (*e.g.* variable selective pressure over the tree), the estimates obtained by the two methods can vary a great deal, and, generally, phylogeny based estimates should be preferred. `dSdN.bf` provides a numerical example of generating dN and dS estimates from biological sequence data.

Calculating the number of nonsynonymous and synonymous sites. The calculation of the number of nonsynonymous and synonymous sites is performed as described previously [6].

- Given a genetic code, for each codon i compute three numbers: T_i - the total number of one-nucleotide substitutions that do not lead to a stop codon, S_i - those substitution which are synonymous and N_i^t - those which are nonsynonymous. Clearly, $T_i = S_i + N_i$. For example, $T_{GAA} = 8$, because 8/9 one nucleotide substitutions (AAA, CAA, GCA, GGA, GTA, GAC, GAG, GAT) do not lead to a stop codon, but one (TAA) does, $S_{GAA} = 1$ (GAG is the only synonymous change) and $N_{GAA} = 7$. This step depends only on the genetic code, and not on the alignment being analyzed.
- Compute the expected values of the three quantities for a given alignment, by averaging over the stationary distribution of codons π:

$$T = \sum_i \pi_i T_i, \quad S = \sum_i \pi_i S_i, \quad N = \sum_i \pi_i N_i.$$

  Note, that $T = S + N$.
- Define $dS = E[syn]\frac{S}{T}$, $dN = E[nonsyn]\frac{N}{T}$, which can now be interpreted as the expected number of synonymous substitutions per synonymous site, and the nonsynonymous analog, respectively.

1.4.4 Correcting for nucleotide substitution biases

As we noted in the introduction, biased nucleotide substitutions, *e.g.* the preponderance of transitions over transversions, can have a significant effect on the proportions of synonymous and nonsynonymous substitutions, and, by extension, they can affect the estimates of α and β. The MG94 model incorporates parameters (θ_{mn}) to correct for such biases. These parameters, in most cases, are not of primary interest to selection analyses, and, often they are indeed nuisance parameters.

There are several possible strategies for selecting one of the 203 possible nucleotide bias models; having chosen a 'nuisance' nucleotide substitution model (or models), we can incorporate (or 'cross') this model with a codon substitution model in order to estimate β/α. The `NucleotideBiases.bf` example evaluates the effect of nucleotide biases on the β/α estimate. For five H5N1 influenza sequences, β/α ranges from 0.148 to 0.233. The estimate for REV is 0.216, and that for the best fitting model (which happens to be 010023 as determined by the lowest Akaike's information criterion score, see below) is 0.214. Lastly, a model averaged estimate for β/α is 0.221.

1.4.4.1 Hypothesis testing

Hypothesis testing concerns itself with selecting, from a number of *a priori* available models, or *hypotheses*, the one that explains the observed data best, or minimizes a loss function (*e.g.* squared distance). For example, one might test for evidence of non-neutral evolution across the gene on average by comparing an MG94 model which enforces $\beta = \alpha$ (neutral evolution) with one that estimates both β and α independently.

In the likelihood framework, all the information about how well the data D support any given model H is contained in the likelihood function $L(H|D) = Pr\{D|H\}$, *i.e.* the probability of generating the data given the model. When comparing two models H_0 (null) and H_A (alternative), the strength of support for model H_A relative to H_0 is often assessed using the likelihood ratio statistic (often abbreviated as LR), defined as $LR = 2(\log L(H_A|D) - \log L(H_0|D))$. A classical likelihood ratio test decides between H_0 and H_A by comparing the LR to a fixed number c, selecting H_A if $LR > c$, and selecting H_0 otherwise. The Neyman-Pearson lemma gives theoretical grounds to prefer the likelihood ratio test to all other procedures for simple hypothesis testing, because for all tests of given *size* α (defined as the probability of selecting H_A when H_0 is true, *i.e.* making a false positive/Type I error), the likelihood ratio test is the most powerful test, *i.e.* it has the highest probability of selecting H_A when it is true, and hence the lowest Type II/false negative error rate.

An important particular case of the likelihood ratio test arises when H_0 and H_A are *nested*. In phylogenetics, H_0 and H_A can almost always be defined as a parametric model family (*e.g.* MG94), where some of the parameters are free to vary (*e.g.* branch lengths), and some may be constrained (*e.g.* $\beta = \alpha$). When H_0 can be obtained from H_A by adding B well-behaved constraints on model parameters, then the distribution of the likelihood ratio test statistic LR when the data are generated under H_0 follows the χ^2 distribution with B degrees of freedom, if the data sample is sufficiently large†. Given a significance level α, which describes the willingness to tolerate false positive results, one computes the critical level c which solves $Pr\{\chi^2_B \geq c\} = \alpha$, and rejects H_0 if $LR \geq c$. Otherwise, one *fails to reject* H_0, which may be either because H_0 is true, or because the data sample is not sufficiently large to distinguish the two (lack of power), thus the hypothesis testing framework is geared towards rejecting the null hypothesis.

† If some parameters are bounded *e.g.* by 0, then the distribution of the LR follows a mixture of χ^2 distributions and a point mass at 0, with the proportion of each generally being dependent on the data at hand. In the simple case of a single one-sided constraint, the appropriate mixture is 50% χ^2_1 and 50% point mass at 0

Choosing a nucleotide model. Consider the symmetric matrix form of nucleotide substitution biases:

$$B = \begin{pmatrix} - & \theta_{AC} & \theta_{AG}(=1) & \theta_{AT} \\ - & - & \theta_{CG} & \theta_{CT} \\ - & - & - & \theta_{GT} \\ - & - & - & - \end{pmatrix}$$

Reading this matrix left to right and top to bottom arranges the six rates as $\theta_{AC}, \theta_{AG}(=1), \theta_{AT}, \theta_{CG}, \theta_{CT}, \theta_{GT}$. Any of the models of nucleotide biases can be defined by specifying some constraints of the form $\theta_{ab} = \theta_{cd}$ (*e.g.* $\theta_{AG} = \theta_{CT} = 1, \theta_{AC} = \theta_{AT} = \theta_{CG} = \theta_{GT}$ for HKY85). A convenient shorthand (adapted from PAUP* and the first exhaustive model search publication [20]) for defining such constraints is a string of six characters where each character corresponds to a θ rate in the above ordering, and if two characters are equal, then the two corresponding rates are constrained to be equal as well. The shorthand for HKY85 is 010010, and the model specified by 010121 defines the constraints $\theta_{AC} = \theta_{AT}, \theta_{CG} = \theta_{GT} = \theta_{AG}(=1)$.

- A "named model", such as HKY85 (the "hard-wired" choice in GY94). Generally, this is a poor choice, because many organisms/genes seem to have non-standard substitution biases [21], unless the alignment is small or a model selection procedure suggests that a "named" model is appropriate.
- The general reversible model (REV), which estimates five biases from the data as a part of the model. While this is a good 'default' model, some of the biases may be difficult to estimate from small data sets, and, as is the case with overly parameter rich models, overfitting is a danger. Overfitted models can actually increase the error in some parameter estimates, because instead of reflecting a real biological process, they may be fitting the noise.
- A nucleotide bias correction based on a model selection procedure (*e.g.* ModelTest [22] or as described in Practice). Generally, this is the preferred approach, because it allows the correction of substitution biases without overfitting. This approach has two drawbacks: additional computation expense (although it is usually small compared to the cost of fitting a codon model), and the fact that most model selection procedures are based on nucleotide inference, and may incorrectly deduce nucleotide biases because they fail to account for codon constraints and selective pressures.
- Model averaged estimates. The most robust, yet computationally expensive, process is to fit all 203 models, obtain α and β estimates from each model, and then compute a weighted sum, where the contribution from each model is determined by its Akaike Weight (defined in section 1.5.3), which can be interpreted as the probability that the model is the "best-fitting" model given the data. Such exhaustive model fitting is an overkill except for small and/or short data sets, where several models may provide nearly equally good fits to the data. Fortunately, fitting all 203 models is practical precisely for smaller data sets, where model uncertainty is likely to occur.

Some intuition for how hypothesis testing using this framework is justified can be gained by means of this simple example. Consider some sequence data that evolves according to the neutral MG94 model (*i.e.* $\alpha = \beta$). Here $H_0 : \beta = \alpha$ and H_A estimates β and α separately. H_0 is nested in H_A, with one fewer degree of freedom. The log-likelihood of H_0 can always be matched or improved by H_A because it contains H_0 and one finds $LR \geq 0$ in this case. If one were to consider a large number of independent data sets (of the same size) for which the correct model was H_0, compute the LR for each one and tabulate a histogram, then the histogram of LR would approximate the density function of χ^2, if the size of each sample was large enough. It is possible for the LR to exceed any arbitrarily large cutoff by chance, but with vanishing probabilities, hence one settles for the value large enough that only in α proportion of the cases does one falsely reject H_0 by estimating the location of the appropriate tail of the χ^2_1 distribution.

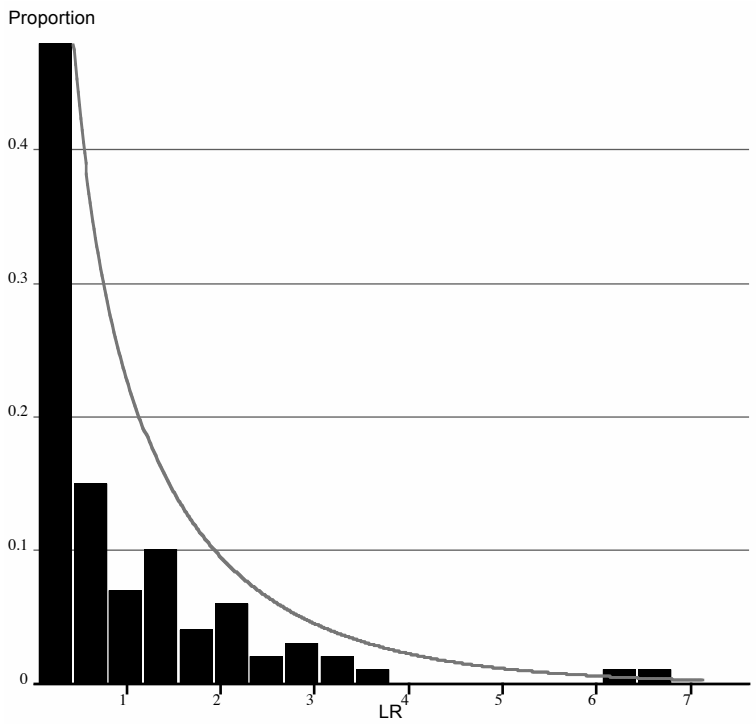

Fig. 1.3. Simulated distribution of the likelihood ratio test statistic based on 100 iterates, and the asymptotic χ^2_1 distribution density (solid line). Note that the simulated distribution is skewed towards smaller values compared to χ^2_1, suggesting that the data sample is too small to have achieved the asymptotic approximation, and tests based on χ^2_1 are likely to be slightly conservative.

The example `LRT.bf` tests the hypothesis $H_0 : \alpha = \beta$ (neutral evolution) versus the alternative of non-neutral evolution on average across an entire gene. For instance, in the Influenza A/H5N1 HA dataset, the hypothesis of neutrality is rejected ($LR = 65.44, p \ll 0.001$), with $\omega = 0.23$, suggesting overall purifying selection. Based on 100 random parametric replicates under H_0, we found the distribution of LR (Fig. 1.3) to range from 0.0004 to 6.81.

Hypothesis testing is a powerful tool if applied judiciously, but its findings can easily be over-interpreted. Indeed, it is crucially important to remember that *when H_0 is rejected in favor of H_A, all we can be certain of is that the data being analyzed are unlikely to have been produced by H_0* ***not*** *that H_A is the best explanation for the data.* For example, it is often tempting to reject an overly simplistic H_0 in favor of an overly complicated H_A, when in fact, a model intermediate in complexity may be the best choice. We will

return to this topic in a later section when we describe a general procedure for model selection.

1.4.4.2 Confidence intervals on parameter estimates

As most sequence samples have relatively few independent observations (alignment columns) per model parameter, there tends to be a fair amount of stochastic 'noise' in model parameter estimates. This noise is usually referred to as *sampling variance*, and it gives one an idea of how variable the estimate of a particular parameter would have been, had independent data samples of the same size been available. Sometimes, it may also be beneficial to estimate the entire *sampling distribution* of one or more model parameters, especially if maximum likelihood estimates are being used for *post hoc* inference, and ignoring the errors in such estimates can lead to patently erroneous conclusions [23]. We note that very few studies that employ codon substitution models (or indeed, any substitution model) report confidence intervals for the model parameters, hence we devote a section to how these intervals can be obtained, in order to encourage their use. Within a maximum likelihood framework, there are at least three different approaches to deducing how much error there is in a given parameter estimate.

Asymptotic normality. For sufficiently large samples, the joint sampling distribution of all model parameters approaches a multivariate normal distribution with the variance-covariance matrix given by the inverse of Fisher information matrix, *i.e.* the log likelihood surface is quadratic in the vicinity of the maximum. Briefly, if $l(\theta_1, ...\theta_n) = \log L(H(\theta_1, ...\theta_n)|D)$ is the log likelihood function of all estimable model parameters, the information matrix I is defined as

$$I(\theta_1, ...\theta_n) = \begin{pmatrix} \frac{\partial^2 l}{\partial \theta_1^2} & \frac{\partial^2 l}{\partial \theta_1 \partial \theta_2} & \cdots & \frac{\partial^2 l}{\partial \theta_1 \partial \theta_n} \\ \vdots & & & \vdots \\ \frac{\partial^2 l}{\partial \theta_n^2} & \frac{\partial^1 l}{\partial \theta_n \partial \theta_2} & \cdots & \frac{\partial^2 l}{\partial \theta_n^2} \end{pmatrix},$$

evaluated at the maximum likelihood values of the parameters. The advantage of using this method is that it is able, given enough sample data, to correctly model possible codependencies between model parameter estimates in addition to yielding parameter errors, because the entire joint distribution of model parameters is approximated. However, it may be difficult to check whether the sample is large enough to allow the normal approximation to hold, especially if one or more of the parameter values are near the boundary of a parameter space. In addition, the information matrix has to

be estimated numerically, which turns out to be computationally expensive and numerically challenging, because one has to estimate $\approx n^2/2$ partial derivatives with sufficient accuracy to obtain an accurate matrix inverse. As in phylogenetic models, n grows linearly with the number of sequences, the only reliable way to achieve asymptotic normality is to analyze very long sequences, which may be impossible due to biological constraints (*e.g.* gene lengths).

Profile likelihood. If only confidence intervals around a parameter estimate, or a collection of parameters, is desired, especially for smaller samples and if asymptotic normality may in doubt, then component-wise profile likelihood intervals may be used instead. A $1-\alpha$ level confidence interval around a maximum likelihood estimate $\hat{\theta}_i$ is defined as all those values of t for which the hypothesis $\theta_i = t$ cannot be rejected against in favor of the hypothesis $\theta_i = \hat{\theta}_i$ at significance level α, when all other model parameters are fixed at their maximum likelihood estimates. Profile confidence intervals are easy to understand graphically: if one plots the log-likelihood function as a function of parameter θ_i only, then (assuming the likelihood function is unimodal) a confidence interval is obtained simply by bracketing the $\hat{\theta}_i$ using the line c_α units below the maximum, where c_α solves is a critical value for the χ^2 distribution: $Pr\{\chi^2_1(x) \geq c_\alpha\} = \alpha$ (Fig. 1.4). Profile likelihood intervals can be found quickly, and can handle cases when the likelihood surface is not well approximated by a quadratic surface over a long range of parameter values. For instance, profile likelihood can produce asymmetric confidence intervals, and handle values near the boundary of the parameter space . However, because profile likelihood works with only one parameter at a time, the intervals it finds may be too small.

Sampling techniques If MLE values of model parameters are used for subsequent inference (*e.g.* in the REL method for site-by-site selection analysis, section 1.6.1), it may be desirable to incorporate parameter estimation errors and see how they might influence the conclusions of an analysis. One possible way to accomplish this is to average over values from an approximate joint distribution of parameter estimates. The *sampling importance resampling* (SIR) algorithm is a simple technique for sampling from probability distributions. Firstly, parameter values are sampled from the likelihood surface. In order to sample parameter values that have high likelihood, one could estimate 95% profile confidence intervals for each model parameter: (θ_i^l, θ_i^u) (perhaps enlarging them by an inflation factor > 1), arriving at an n-dimensional rectangle from which one draws a large number $N \gg n$ (*e.g.*

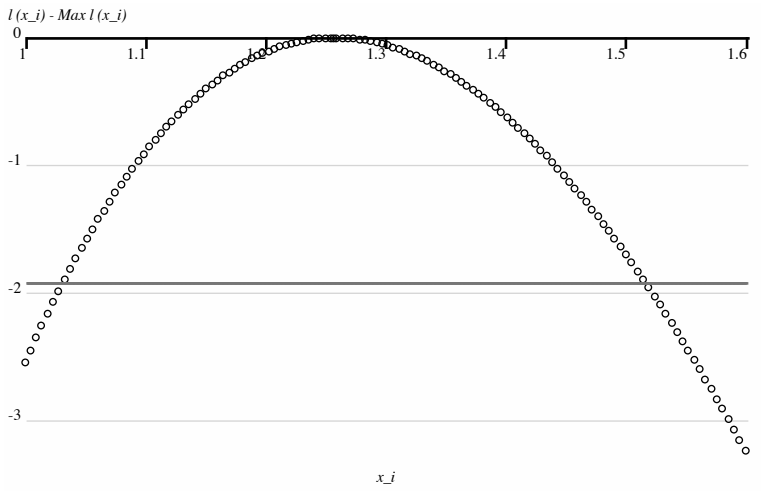

Fig. 1.4. Graphical interpretation for profile likelihood confidence intervals. The maximum of the log-likelihood function l is set at 0 and as the value of the x_i parameter is taken further away from the MLE, we seek the points where the likelihood curve intersects the $c_{0.05} = -1.92$ line. For this parameter, the MLE is 1.26, and the 95% confidence interval turns out to be $(1.03, 1.52)$. Note that the curve is slightly asymmetric, suggesting that asymptotic normality has not yet been achieved.

1000) of samples. A technique called Latin Hypercube Sampling (LHS) can sample this space very efficiently. For each parameter i, LHS draws points by sequentially picking a random index r_i from 0 to N, with the constraint that all r_i are distinct and forming a sample point $\theta_i = \theta_i^l + (\theta_i^u - \theta_i^l)r_i/N$. Lastly, a resampling step is applied, where a much smaller number of points $M < N$ are drawn from the original sample, but now in proportion to their likelihood score. Statements about the sampling properties of all θ_i, and derivative quantities can now be made based on the resampled set of points. The closer the original distribution is to the true distribution, the larger the *effective sample size* will be.

`ErrorEstimates.bf` applies all three methods to estimating the sampling properties of ω ratios for the MG94 model with a separate ω for each branch. Note that short branches tend to provide highly unreliable estimates of ω when compared to longer branches.

1.4.5 Bayesian approaches

Instead of a maximum likelihood approach, which works with $L(H|D) = Pr\{D|H\}$, *i.e.* the probability of generating the data given the model, a

Bayesian approach works with $Pr\{H|D\}$, *i.e.* the probability of the model given the data. In this way, a Bayesian approach is much more similar to the way that many people interpret statistical results, and the output of a Bayesian model, called a posterior distribution, can be interpreted as true probabilities. However, a Bayesian approach requires the assumption of prior distributions for the parameters. Not only may the choice of priors affect the results, the use of certain prior distributions necessitates the use of computationally-intensive Markov Chain Monte Carlo (MCMC) techniques to sample from the posterior distribution. Although different from a philosophical point, Bayesian approaches and maximum likelihood approaches should arrive at the same results, given sufficient data. The two major phylogenetic software packages based on the Bayesian paradigm are MrBayes (`http://mrbayes.csit.fsu.edu/`) and BEAST (`http://evolve.zoo.ox.ac.uk/beast/`), and we refer interested readers to their documentation pages for further details.

1.5 Estimating branch-by-branch variation in rates

As selection pressures almost certainly fluctuate over time, it may be unreasonable to use models that assume a constant selective pressure for all branches in the phylogenetic tree. For instance, in a tree of influenza sequences from various hosts, one might expect to find elevated selection pressure on branches separating sequences from different hosts, because they are reflective of adaptation to different evolutionary environments.

We have already mentioned the model which allows a separate ω in every branch of the tree - the 'local' model, or to follow the nomenclature of the original paper [24], the 'free ratio' model. Other possibilities are the global (single-ratio) model, which posits the same ω for all branches and a large array of intermediate complexity models, where some branches are assigned to one of several classes, with all branches within a single class sharing one ω value. Formally, this model can be described as

$$\beta^b = \omega^{I(b)}\alpha^b,$$

where $I(b)$ is the assignment of branch b to an ω class. For the global model $I(b) = 1$ and for the local model $I(b) = b$.

1.5.1 Local versus global model

A naive approach to test for branch-to-branch rate variation is to fit the global model as H_0, the local model as H_A, and declare that there is evidence

of branch-by-branch rate heterogeneity if H_0 can be rejected. Since the models are nested, one can use the likelihood ratio test with $B-1$ (B is the total number of branches in the tree) degrees of freedom. `LocalvsGlobal.bf` performs this test using the MG94 model.

This procedure, however is lacking in two critical areas. Firstly, it may lack power if only a few branches in a large tree are under strong selective pressure, because the signal from a few branches may be drowned out by the "background". Secondly, it lacks specificity, in that the real question a biologist may want to ask is "Where in the tree did selection occur?" and not "Did selection occur somewhere in the tree?". A rough answer to this question may be gleaned from examining the confidence intervals on branch by branch ω and saying that two branches are under different selective pressures if their confidence intervals do not overlap. However, these confidence intervals are suitable only for data exploration, as they may not achieve appropriate coverage. For instance, there is an implicit multiple comparison problem and the intervals may be too narrow in some cases, and they may be difficult to interpret for large trees where many pairwise comparisons would have to be made.

1.5.2 Specifying branches a priori

The first likelihood-based procedure for identifying different selective regimes on tree branches [24] relied on an *a priori* specification of some branches of interest. For example, if a branch separates taxa from different evolutionary environments (*e.g.* virus in different hosts, geographically separate habitats, taxa with and without a specific phenotype), one may be interested in studying the strength of selection on that branch. The *a priori* branch model separates all B branches into a few ($F < B$) of interest (foreground), for which the ω parameter is estimated individually, and all other branches (background), which share a common ω_b - background selection regime. To test for significance, one conducts a LRT with F degrees of freedom. This analysis boosts the detection power because the number of model parameters is significantly reduced, and focuses on specific branches.

The main drawback of such a test is that it assumes that the rest of the branches have a uniform selective pressure. This assumption is less likely to hold as the number of taxa (and tree branches) is increased, and the model can be easily misled into claiming selection on a 'foreground' branch if the background is strongly non-uniform. A simple example in Fig. 1.5 shows that the likelihood ratio test can perform very poorly if the assumptions of the model are violated. The assumption of neutrality along a given branch

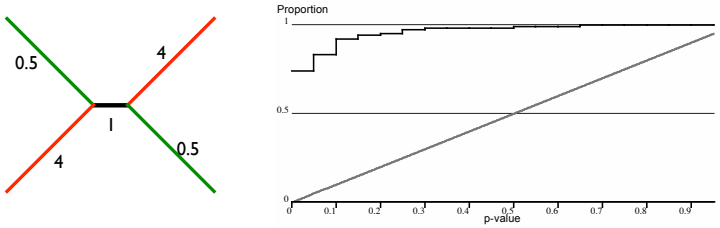

Fig. 1.5. The effect of model mis-specification on the likelihood ratio test. We simulated 100 long datasets (with 5000 codons each) using the tree on the left, which branch ω shown for every branch and then tested whether $\omega = 1$ along the short middle branch using the likelihood ratio test which makes an incorrect assumption that all other branches of the tree have the same ω. The panel on the right tabulates the cumulative distribution of p-values for rejecting the null hypothesis $\omega = 1$ along the internal branch. A correct test for this model would reject neutrality at level p in approximately $p \times 100$ cases (the solid line), whereas this test has uncontrollable rates of false positives. It rejects neutrality 74 times at $p = 0.05$, for example. In addition, the point estimate of ω along the internal branch is strongly biased by the incorrect model (mean of 1.94, instead of the correct value of 1).

was rejected at $p = 0.05$, *i.e.* very strongly, for 74/100 datasets simulated with the neutral "foreground" branch.

A test which is more robust to a non-uniform background may be the following: to decide whether a given branch b_0 is under positive selection (*i.e.* has $\omega^{b_0} > 1$), one fits $H_0 : \omega^{b_0} \leq 1$ and $H_A : \omega^{b_0}$ is unconstrained, allowing all other branches to have their own ω and conducts an LRT. This is a one-sided test (*e.g.* the constraint is an inequality, rather than an assignment), and the appropriate distribution of the LR test statistic to check against is a mixture of χ^2_1 and a point mass at 0 [25].

`BranchAPriori.bf` can be used to conduct both types of tests. To conduct one of the tests of [24], select Node1 and Node5 as foreground branches in the Primate Lysozyme data set.

1.5.3 Data-driven branch selection

However, in many instances there may exist no *a priori* evidence to suspect selection at a specific branch in the tree. There are several naive approaches which we would like to caution against. Firstly, it may be tempting to begin by fitting a local model, inspecting the values of ω estimated for each branch, and then selecting some branches which appear to be under selection for further testing. Using the data to formulate a hypothesis to test is referred to as "data-dredging" and this should generally be avoided.

Hypotheses formulated using a data set and then tested with the same data set will nearly always be highly biased, *i.e.* appear significant when in fact they are not. Secondly, one might attempt to test every branch one at a time, and declare all those which appear under selection in individual tests to be significant. However, this method consists of multiple tests, and as such, requires a multiple test correction (*e.g.* Bonferroni or false discovery rate [26]). Intuitively, if each test has the probability of making a Type I error 5% of the time, at least one of the 17 tests (each branch in a 10 sequence tree) would, in the worst case scenario make a Type I error in $1 - (1 - 0.05)^{17} = 58.2\%$ of the time, *i.e.* every other significant finding could be a false positive! There is another issue, in that each individual test assumes that every branch, except the foreground branch being tested, is under a uniform (background), presumably negative, selective pressure. Hence, if two of the tests return positive results (*i.e.* two foreground branches are positively selected), these results are incompatible.

Since the model of rate variation is a nuisance parameter in this case, we advocate the idea of *searching* the space of many possible models, selecting those which fit well, and averaging over models to achieve robust inference. For full details of the methods we refer the reader to [27], but the basic idea is intuitive. Let us consider models with up to C different ω assigned to branches. A model like this is completely specified by assigning each branch in a phylogenetic tree to one of C classes, with the total number of models on B branches given by the Stirling numbers of the second kind - the number of unique ways to assign B objects to C bins:

$$S(C;B) = \frac{1}{C!}\sum_{k=1}^{C}(-1)^{C-k}\frac{C!}{k!(C-k)!}k^B.$$

The number of models grows combinatorially fast with B, even if C is small (*e.g.* 3). The models considered during the search will no longer always be nested, hence a new model comparison technique is called for. We chose a small sample Akaike information criterion score of each model, defined as

$$AIC_c(M) = -2\log L + 2p\frac{s}{s-p-1},$$

where L is the maximum log-likelihood score of the model, p is the number of model parameters and s is the number of independent samples available for inference (the number of sites in an alignment). AIC_c rewards a model for a good likelihood score and penalizes it for the number of parameters, progressively more so as the number of parameters approaches the number of independent samples. AIC_c minimizes the expected Kullback-Liebler di-

vergence between model M and the true model that generated the data, and there exist fundamental results supporting its use. We use a genetic algorithm (GA) to search the space of possible models, measuring the fitness of each by its AIC_c score. GAs have proven very adept at rapidly finding good solution in complex, poorly understood optimization problems. As an added benefit, the availability of AIC_c scores allows one to compute Akaike weights for each model, defined as $w_M = \exp(\min_M AIC_c(M) - AIC_c(M))/2$, normalized to sum to one. w_M can be interpreted as the probability that model M is the best (in the Kullback-Liebler divergence sense) of all those considered given, the data. Now, instead of basing inference solely on the best fitting model, one can compute the model averaged probability of finding $\beta^b > \alpha^b$ for every branch in the tree. The GA search is a computationally expensive procedure, and we recommend that the reader first try our web interface (`http://www.hyphy.org/gabranch`).

1.6 Estimating site-by-site variation in rates

Often, we are most interested in positive or diversifying selection, which may be restricted to a small number of sites. Several methods to detect site-specific selection pressure have been proposed; for a review and detailed discussion of these methods, see Kosakovsky Pond and Frost [28]. There are two fundamentally different approaches to estimating site-specific rates. The first approach, first proposed by Nielsen and Yang [1] assumes that the true distribution g of α_s and β_s can be well represented by some predefined, preferably simple distribution f, uses the data to infer the parameters of f, integrates the values of unobserved model parameters out of the likelihood function, and then assigns each site to a rate class from f to make inference about what occurs at a single site. This class of models - random effects likelihood (REL) models - was first implemented in the PAML [13] package and has since been widely adopted in the field of molecular evolution. Bayesian versions of these models have been developed by Huelsenbeck and colleagues [29, 30]. The second approach, first proposed by Suzuki and Gojobori [31] estimates site-specific rates directly from each site independently, either using maximum likelihood methods (fixed effects likelihood, FEL), or counting heuristics. We now describe how these methods differ in their approach to estimating site-specific rates.

1.6.1 Random Effects Likelihood (REL)

When the objective of an analysis is to estimate, for each codon $c = 1 \dots S$ in the alignment, a pair of *unobserved* rates α_s and β_s, the random effects approach posits than there exists a distribution of rates that is almost always assumed to be discrete with D categories for computational feasibility, with values (α^d, β^d), and the probability of drawing each pair of values is (p^d), subject to $\sum_d p^d = 1$. Examples of such distributions might be the general discrete distribution (all parameters are estimated), or the M8 model of PAML, where $D = 11$, $\alpha_d = 1$ and β is sampled from a discrete beta distribution (10 bins) or a point mass $\omega > 1$. The value D is fixed *a priori*, and all other distribution parameters are usually estimated by maximum likelihood. To compute the likelihood function at codon site c, one now has to compute an expectation over the distribution of rates

$$L(\text{site c}) = \sum_{d=1}^{D} L(\text{site c}|\alpha_c = \alpha_d, \beta_c = \beta_d)p^d,$$

where each of the conditional probabilities in the sum can be computed using the standard pruning algorithm [10]. Finally, the likelihood of the entire alignment, making the usual assumption that sites evolve independently, can be found as the product of site-by-site likelihoods.

The REL approach can be used to test whether positive selection operated on a proportion of sites in an alignment. To that end, two nested REL models are fitted to the data: one which allows $\beta > \alpha$, and one that does not. The most cited test involves the M8a (or M7) and M8b models of Yang and colleagues [32], which each assume a constant synonymous rate $\alpha \equiv 1$ and use a beta distribution discretized into 10 equiprobable bins to model negative selection ($\beta_i \leq 1$ for $i \leq 10$), but M8a forces $\beta_{11} = 1$ while M8b allows $\beta_{11} \geq 1$. If M8a can be rejected in favor of M8b using the likelihood ratio test with a one sided constraint, this provides evidence that a p_{11} proportion of sites are evolving under positive selection. Another test, allowing for variable synonymous rates has been proposed by Sorhannus and Kosakovsky Pond [33] and involves fitting two D (*e.g.* $D = 4$ or $D = 9$) bin general discrete distributions with one of them constraining $\beta_d \leq \alpha_d$ for all rate classes.

To find individual codon sites under positive selective pressure, REL methods can use an empirical Bayes approach, whereby the posterior probability of a rate class at every site is computed. A simple application of Bayes rule,

treating the inferred distribution of rates as a prior shows that

$$Pr(\alpha_s = \alpha_d, \beta_s = \beta_d | \text{site c}) = \frac{L(\text{site c} | \alpha_s = \alpha_d, \beta_s = \beta_d) p^d}{L(\text{site c})}.$$

A site can be classified as selected if the posterior probabilities for all those rate classes with $\beta_d > \alpha_d$ exceed a fixed threshold (*e.g.* 0.95). While it may be tempting to interpret this value as an analog of a p-value, it is not one. The Bayesian analog of a p-value is the Bayes factor for some event E, defined as

$$BF(E) = \frac{\text{posterior odds } E}{\text{prior odds } E} = \frac{Pr_{posterior}(E)/(1 - Pr_{posterior}(E))}{Pr_{prior}(E)/(1 - Pr_{prior}(E))}.$$

A Bayes factor measures how much more confident (in terms of odds) one becomes about proposition E having seen the data. If $BF(\beta > \alpha)$ at site c exceeds some predefined value (*e.g.* 20), a site can be called selected. Our simulations [28] indicate that in phylogenetic REL methods $1/BF$ is approximately numerically equivalent to a standard p-value.

REL methods are powerful tools if applied properly, and can detect selection when direct site-by-site estimation is likely to fail. For example, if there are 100 sites in an alignment with $\omega = 1.1$, unless the alignment consists of hundreds of sequences, methods which estimate rates from individual sites (discussed below) are unlikely to identify any *given* site with $\omega = 1.1$ to be positively selected, because selection is very weak. Hence they may miss evidence for positive selection altogether. On the other hand, REL methods can pool small likelihood contributions from all 100 sites to a single class with $\omega > 1$ and use *cumulative* evidence to conclude that there is selection somewhere in the sequence. However, REL methods may also fail to identify any individual site with any degree of confidence. The ability to pool information across multiple sites is a key advantage of REL.

However, one needs to be keenly aware of two major shortcomings of REL. Firstly, there is a danger that the distribution of rates chosen *a priori* to model α and β is inadequate. For example, there is *no compelling biological reason* to support the mixture of a beta and a point mass distribution (PAML's M8). In extreme cases (see section 1.6.5) this can lead to positively misleading inference, whereas in others "smoothing" - the underestimation of high rates and overestimation of low rates - can occur, and result in loss of power. Secondly, posterior empirical Bayes inference assumes that rate estimates are exact. For example, an $\omega = 1.05$ may be estimated and used to compute Bayes factors, but if the confidence interval on that estimate is $(0.5, 1.5)$, then one cannot be certain whether or not the contribution from

this rate class should be counted for or against evidence of positive selection at a given site. REL methods have received somewhat undeserved criticisms (*e.g.* [23]), mostly arising of the application of these methods to very sparse (small and low divergence data), where all inferred parameter values had large associated errors. Yang and colleagues [34] partially addressed the issue of incorporating parameter errors into REL analyses using a Bayes empirical Bayes (BEB) technique, that uses a series of approximations to integrate over those parameters which influence the distribution of α_d, β_d. More generally, sampling methods (such as the SIR algorithm described previously) can also be drawn upon to average over parameter uncertainty, by drawing a sample of parameter values, computing a Bayes factor for positive selection at every site, and then reporting a site as positively selected if a large (*e.g.* $> 95\%$) proportion of sampled Bayes factors exceeded the preset threshold.

1.6.1.1 Comparing distributions of rates in different genes

One of the advantages of using a random effects approach is that it is straightforward to test whether two sequence alignments (which may be totally unrelated) have the same distribution of substitution rates across sites. The easiest way to test for this is to select a distribution family (*e.g.* M8 or a general discrete distribution), $f(\nu)$, where ν denotes the full vector of distribution parameters. Several tests can be readily conducted.

Are the rate distributions different between two datasets? To answer this question, we fit the null model H_0, which forces the ν parameter to be the same on two datasets (letting all other model parameters, *e.g.* base frequencies, branch lengths and nucleotide biases to vary freely in both data sets), and the alternative model H_A, where each dataset is endowed with its own vector of distribution parameters (ν_1 and ν_2). A likelihood ratio test with the degrees of freedom equal to the number of estimable parameters in ν can then be used to decide whether the distributions are different. This test is better than simply comparing the means (see an earlier section), but it is effectively qualitative, in that no insight can be gleaned about *how* the distributions are different if the test is statistically significant.

Is the extent or strength of selection the same in two datasets? If the choice of distribution f is fixed, it may be possible to pose a more focused series of questions regarding how the distributions might be different between two data sets. Let us consider, for instance, a 4 bin general discrete distribution (GDD_4) with rates $(\alpha_1, \beta_1), (\alpha_2, \beta_2), (\alpha_3, \beta_3), (\alpha_4, \beta_4)$,

probabilities p_1, p_2, p_3 and $p4 = 1 - p_1 - p_2 - p_3$, with the further restriction (explained in 1.6.5) that $\sum \alpha_i p_i = 1$. Furthermore, let us assume that the first two bins represent negative selection ($\alpha_1 > \beta_1$ and $\alpha_2 > \beta_2$), bin three reflects neutral evolution ($\alpha_3 = \beta_3$), and bin four - positive selection ($\beta_4 > \alpha_4$). Distributions with more bins, or a different allocation of selective pressures between bins can be readily substituted. Using the most general model, where two independent GDD_4 distributions are fitted to each data set as the alternative hypothesis, the following likelihood ratio tests can be carried out: (i) are the proportions of positively selected sites the same in both datasets ($H_0 : p_4^1 = p_4^2$); (ii) are the strengths of selection the same in both datastes ($H_0 : \beta_4^1/\alpha_4^1 = \beta_4^2/\alpha_4^2$); (iii) are the proportions and strengths of selection the same in both datasets (both constraints).

1.6.2 Fixed Effects Likelihood (FEL)

With a FEL approach, instead of drawing (α, β) from a distribution of rates, one instead estimates them directly at each site. Firstly, the entire data set is used to infer global alignment parameters, such as nucleotide substitution biases, codon frequencies and branch lengths; those values are fixed afterwards for all individual site fits. Secondly, FEL considers each codon site c as a number of independent realizations of the substitution process (now depending only on the site specific rates (α_c, β_c)) operating on the tree - roughly one realization per branch, yielding a sample size on the order of N - the number of sequences in the alignment. This observation underscores the need to have a substantial number of sequences (*e.g.* $N \geq 20$) before reliable estimates of (α_c, β_c) can be obtained. Two models are fitted to every codon site c: $H_0 : \alpha_c = \beta_c$ (the neutral model) and H_A, where (α_c, β_c) are estimated independently (the selection model), and, because the models are nested, the standard LRT can be applied to decide if site c evolves non-neutrally. When the LRT is significant, if $\alpha_c < \beta_c$, site c is declared to be under positive selection, otherwise site c is under negative selection.

FEL has several advantages over REL. Firstly, no assumptions about the underlying distribution of rates are made, making the estimation of (α, β) potentially more accurate. Secondly, a traditional p-value is derived as a level of significance at every site, automatically taking into account errors in the estimates of α_c and β_c. Thirdly, FEL can be trivially parallelized to run quickly on a cluster of computers, because each site is processed independently of others.

The drawbacks of FEL are that it may require a substantial number (*e.g.* $20 - 30$) of sequences to gain power, that it cannot pool information across

sites to look for evidence of alignment-wide selection, and that it is not well suited for rate comparisons between unrelated data sets. In addition, nuisance parameters such as nucleotide substitution biases, codon frequencies and branch lengths are treated as known rather than subject to error, however simulations suggest that this assumption does not lead to high type I (*i.e.* false positives) errors, and FEL appears to be extremely good at capturing 'true' substitution rates, at least in simulated data [28].

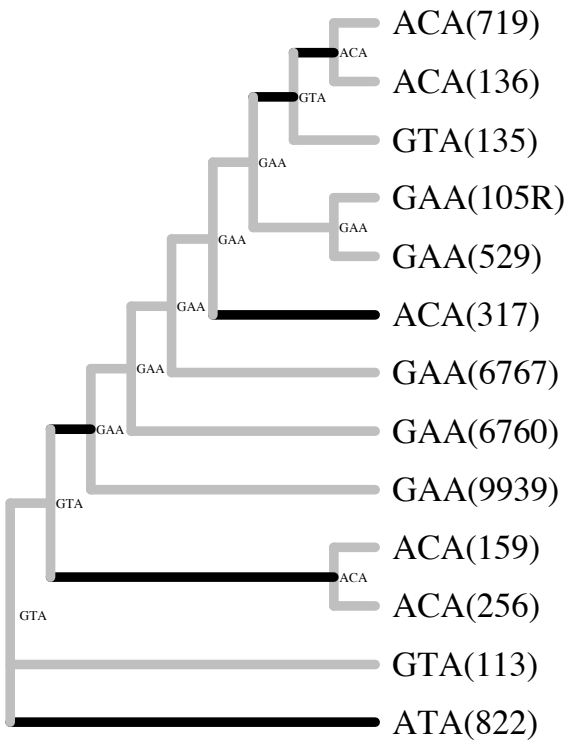

Fig. 1.6. An illustration of SLAC method, applied to a small HIV-1 envelope V3 loop alignment. Sequence names are shown in parentheses. Likelihood state ancestral reconstruction is shown at internal nodes. The parsimonious count yields 0 synonymous and 9 non-synonymous substitutions (highlighted with a dark shade) at that site. Based on the codon composition of the site and branch lengths (not shown), the expected proportion of synonymous substitutions is $p_e = 0.25$. An extended binomial distribution on 9 substitutions with the probability of success of 0.25, the probability of observing 0 synonymous substitutions is 0.07, hence the site is borderline significant for positive selection.

1.6.3 Counting methods

Counting methods provide a 'quick-and-dirty' alternative to FEL, and perform nearly as well for sufficiently large (*e.g.* 50 or more sequences) [28]. A counting method consists of the following steps.

(i) Unobserved ancestral codons, *i.e.* those which reside at internal tree branches, are reconstructed. The original counting method of Suzuki and Gojobori [31] used nucleotide-based parsimony reconstruction, which led to problems such as the possibility of inferring stop codons at internal branches or hundreds of equally good reconstructions. In our Single Likelihood Ancestor Counting (SLAC) method [28], a global MG94 model is fitted to the entire alignment, and used for maximum likelihood reconstruction of ancestral codons.

(ii) Based on a given ancestral reconstruction, the number of observed synonymous and non-synonymous substitutions (NS and NN) are counted, averaging over all possible shortest paths when multiple substitutions are required, *e.g.* to handle the $ACT \rightarrow AGA$ substitution one would average over $ACT \rightarrow ACA, ACA \rightarrow AGA$ and $ACT \rightarrow AGT, AGT \rightarrow AGA$ pathways.

(iii) Using the same ancestral state reconstruction, one computes the mean (over all branches) proportion of synonymous and non-synonymous sites (see section 1.4.3) at a given site, ES and EN. $p_e = ES/(ES + EN)$ can then serve as the *expected proportion* of synonymous changes at that site under neutrality.

(iv) One then tests whether the 'observed' (strictly speaking, inferred) proportion of synonymous substitutions $p_o = NS/(NS + NN)$ is significantly different from p_e, using a extended (to deal with non-integer counts) binomial distribution with p_e probability of success and $NS + NN$ outcomes to determine the level of significance. If $p_e < p_o$ and the result is statistically significant then a site is called negatively selected, and if $p_e > p_o$ - positively selected.

The method is very fast and intuitively attractive (Fig. 1.6) and performs nearly identically with FEL and REL on large simulated data sets. However, there are many assumptions implicit in the method, which may mislead it in certain cases. Firstly, the ancestral state reconstruction is treated as known, where in fact it is inferred (with possible errors) from the data. This shortcoming can be dealt with by averaging over all possible ancestral states or using a sampling procedure to study the effect of ancestral uncertainty on inference of selection [28]. Secondly, parsimonious evolution is assumed,

hence multiple possible substitutions along a branch are discounted. This is generally not a major issue, unless the phylogeny possesses many long branches. Thirdly, the binomial distribution is only a rough approximation to neutral evolution. For example, even though an "average" codon may have a p_e proportion of random substitutions turn out synonymous, codon usage variation throughout the tree may bias this proportion quite strongly.

1.6.4 Which method to use?

The availability of multiple methods to estimate site-specific nonsynonymous and synonymous substitution rates has led to discussion about which method is most appropriate. Fortunately, given enough sequence data, we have shown that all methods (properly applied) tend to perform similarly [28], hence the choice of a method is mostly a matter of preference or computational expediency. Difficult cases arise when inference must be made from limited sequence data, in which case we advocate the use of every available method for a consensus-based inference.

1.6.5 The importance of synonymous rate variation

When REL methods were first introduced in 1998 [1], the assumption that synonymous rates α were constant for the entire length of the gene (proportional to the underlying neutral mutation rate) was made. If this assumption is incorrect, however, then an elevated ω could be due to either lowered α, that could be a result of a functional constraint on synonymous substitutions, such as selection on exon splicing enhancement elements [35] or secondary RNA structure [36]. In addition, when a site is hypervariable, *e.g.* has both high α and β (but $\beta \leq \alpha$), models which assume a constant α are likely to misclassify such sites as strongly selected.

SLAC, FEL and REL (with an appropriate rate distribution) are all able to model variation in both synonymous and non-synonymous rates, and we cannot emphasise enough how important it is to do so. We strongly encourage routine testing for synonymous rate variation, using the procedure described by Kosakovsky Pond and Muse [37]. In the same paper, it was demonstrated that a wide variety of sequence alignments, sampled from different genes and different types of organisms show a near universal strong support for variation in synonymous substitution rates. The test fits two models: H_A - a 3 bin general discrete distribution to β and a separate 3 bin general discrete distribution to α (constrained to have mean one, see [37] for

technical details), and H_0, which imposes the restriction $\alpha = 1$. The models are nested, with 4 constraints, hence the usual likelihood ratio test applies.

1.7 Comparing rates at a site in different branches

One of the most complex models considers both site-to-site variation and branch-to-branch variation in selection pressures. Since rates α_s^b, β_s^b now depend both on the branch and the site, the models must adopt some restrictions on how this dependency is modeled, otherwise too many parameters will be fitted to too few data points leading to unreliable inferences.

If one is interested in selection in a *group* of branches, for example, a specific subtree of the phylogeny, or internal branches of the tree, then FEL methods can be readily adapted. Branches in the tree are split into the group of interest (B_I) and the rest of the branches (B_B). The alternative model fits three rates to each codon c in the alignment: $\alpha_c, \beta_c^I, \beta_c^B$, where β_c^I operates on branches of interest, and β_c^B - on the rest of tree branches. The null model forces neutral evolution on the branches of interest $\beta_c^I = \alpha_c$, and to test for significance, a one degree of freedom LRT is employed. This method has been previously applied to the study of population level selective forces on the HIV-1 virus, where each sequence was sampled from a different individual, which fell into one of two genetically distinct populations [38]. The branches of interest (all internal branches), represent the evolution of successfully transmitted viral variants, and can be used as a proxy for population level selective forces. Based on simulation studies, this approach has low error rates, but needs many sequences to gain power, and would not be advisable with a group of interest branches that comprised of only a few branches.

The REL take on the problem was advanced in 2002 by Yang and his colleagues [39], who proposed what has since become known as 'branch-site' methods. The most recent version of the method [40] requires that a branch or branches of interest be specified *a priori*, assumes a constant synonymous rate for all sites, and samples β_s^b from a four bin distribution defined as follows:

(i) Class 0. Negative selection with $\beta = \omega_0 < 1$ on all tree branches. Weight p_0.
(ii) Class 1. Neutral evolution with $\beta = 1$ on all tree branches. Weight p_1.
(iii) Class 2a. Negatively selected background $\beta^B = \omega_0 < 1$, positively selected foreground $\beta^I = \omega_2 \geq 1$. Weight $(1 - p_0 - p_1)p_0/(p_0 + p_1)$.

(iv) Class 2b. Neutrally evolving background $\beta^B = 1$, positively selected foreground $\beta^I = \omega_2 \geq 1$. Weight $(1 - p_0 - p_1)p_1/(p_0 + p_1)$.

The alternative model fits all parameters independently, whereas the null model fixes $\omega_2 = 1$. A one sided LRT can be used for evidence of selection on foreground branches somewhere in the sequence, and empirical Bayes inference is used to detect sites with strong Bayes factors (or posterior probabilities) of being in Classes 2a or 2b. As with other REL methods, the main advantage gained from adopting a heavily parameterized form of possible rate classes is the pooling of information from many sites. Hence, it may be possible to use a 'branch-site' REL to look for evidence of episodic selection along a single branch. One has to be aware of issues similar to those highlighted in section 1.5.2, when the assumption of uniform background selection may be violated, leading to incorrect inference. In addition, it is unclear what effect synonymous rate variability would have on this method.

1.8 Discussion and further directions

To conclude, currently available methodology allows the fitting of codon substitution models to characterize patterns of nonsynonymous and synonymous substitution in multiple sequence alignments of coding sequences. Extensions of these models allow us to identify particular sites or branches in a phylogenetic tree that are evolving under positive or negative selection. These models can incorporate biological features such as biased nucleotide substitution patterns, and recombination through the assumption of different phylogenies for different parts of the sequence.

While the codon models discussed in this chapter represent the current state-of-the-art, it is important to remember that they still make simplifying assumptions. Perhaps the most important assumption is that sites are assumed to evolve independently from one another. We cannot simply look for associations between codons in extant sequences. Simple 'counting based' methods have been applied to look for associations while taking the shared evolutionary history into account [41], but ideally, a process-based model is more desirable. While 'covarion' models of codon substitution have been proposed [42], strictly speaking, these are heterotachy models, that allow the substitution rate to vary over time. Other models allow correlations between the evolutionary rates at nucleotide sites [43, 44, 45], which can be extended to consider codon substitution models; however, these do not consider how the rates may change at one site depending on a particular state at another site. Recently, models have been proposed that explicitly

consider interactions between sites by considering the entire sequence as a single state [46, 47]; these approaches are highly computationally intensive, due to the large number of possible states. Another approach would be to consider interactions between a small number of sites (2-3).

2

Practice

In the practice section, we briefly review software packages for the analysis of selection pressures on coding sequences, before embarking on a detailed walk-through of some of the analyses available within our HyPhy software package.

2.1 Software for estimating selection

There is currently a wide variety of software packages available for inferring patterns of selection from protein-coding sequences, and the majority are freely-downloadable from the web. In most cases, there are compiled binaries available for 'conventional' operating systems (*i.e.* Macintosh and Windows), and can otherwise be compiled from publicly-released source code. As is often the case with public-domain software, however, the source code and documentation is often unavailable or poorly-maintained. Here, we will review some of the software packages that have been used to estimate selection.

2.1.1 PAML

PAML (an abbreviation of "Phylogenetic Analysis by Maximum Likelihood") was developed by Ziheng Yang and originally released in 1997 [13], providing the first publicly-available implementation of codon model-based methods of selection inference. PAML can estimate selection on a site-by-site basis, modeling variation by random effects likelihood (REL). It has since become widely adopted as the gold standard for estimating selection from sequence alignments, having reached over 1000 citations in the literature. A large number of nested models are available within PAML for likelihood-based model selection. However, the programs that make

up PAML are command-line executable binaries, meaning that no graphical user interface is provided. Program settings are modified by editing plain-text (*i.e.* ASCII) files in which various analytical options are listed. Furthermore, the programs in PAML cannot be easily customized to implement other related models. Source code and pre-compiled binaries for Macintosh and Windows can be obtained for free at the PAML website (`http://abacus.gene.ucl.ac.uk/software/paml.html`). A substantial manual written by Ziheng Yang is currently included in distributions of PAML as a PDF file.

2.1.2 ADAPTSITE

ADAPTSITE was developed by Yoshiyuki Suzuki, Takashi Gojobori, and Masatoshi Nei and was originally released in 2001 [48]. This program was the first implementation of a method for estimating selection by counting the number of inferred nonsynonymous and synonymous substitutions throughout the tree [31]. The source code for ADAPTSITE is distributed for free from the website (`http://www.cib.nig.ac.jp/dda/yossuzuk`), but there are no pre-compiled binaries available. With the exception of basic instructions for compiling and installing ADAPTSITE, there is no additional documentation provided.

2.1.3 MEGA

MEGA (an abbreviation of "Molecular Evolutionary Genetic Analysis") was developed by Sudhir Kumar, Koichiro Tamura, Masatoshi Nei, and Ingrid Jacobsen, and originally released in 1993 [49]; it is currently at version 3.1. This software package provides a comprehensive array of methods for sequence alignment, reconstructing phylogenetic trees, and hypothesis testing. However, it is distributed as a Windows executable binary only (from the MEGA homepage, `http://www.megasoftware.net`). For estimating selection, MEGA implements distance-based methods for estimating the number of nonsynonymous and synonymous substitutions [17], and then evaluates the hypothesis $\beta = \alpha$ using one of several available statistical tests.

2.1.4 HyPhy

HyPhy (an abbreviation of the phrase "HYpothesis testing using PHYlogenies") was developed by Sergei Kosakovsky Pond, Spencer Muse, and Simon Frost, and was first released publicly in 2000 [50]. It is a free and

actively-maintained software package that can be downloaded from the web page (`http://www.hyphy.org`) as either a pre-compiled binary executable for Macintosh or Windows, or as source code. *HyPhy* provides a broad array of tools for the analysis of genetic sequences by maximum likelihood, and features an intuitive graphical user interface. It is uniquely flexible in that all of the models and statistical tests implemented in the software package are scripted in a custom batch language (instead of compiled source code), which can be freely modified to suit one's needs. Furthermore, many of the default methods can be executed in parallel, allowing *HyPhy* to handle exceptionally difficult problems by parallel computing, using either multiple threads (useful for multiple-core processors) or multiple processes (for clusters of computers). A user manual and batch language reference guide are included with distributions of *HyPhy*.

2.1.5 Datamonkey

Datamonkey (`http://www.datamonkey.org`) is actually a web-server application of specific programs in the *HyPhy* package, hosted on a high-performance computing cluster that currently consists of 80 processors. As the *HyPhy* programs for estimating selection on a site-by-site basis are designed to take advantage of parallel computing, *Datamonkey* provides a very fast method for analyzing large sequence data sets within an easy-to-use interface. In the first three years of its existence, *Datamonkey* has been used to analyze over 15,000 sequence alignments. *Datamonkey* also provides a parallelized implementation of genetic algorithms for rapid detection of recombination breakpoints in a sequence alignment [11], which, if unaccounted for, could lead to spurious results in a selection analysis.

2.2 Influenza A as a case study

We will demonstrate the various methods in *HyPhy* for measuring selection pressures in protein-coding sequences, by applying these methods to a series of data sets based on 357 sequences of the influenza A virus (serotype H3) haemagglutinin gene that was originally analyzed by Robin Bush and colleagues [51]. We will henceforth refer to this data set as 'Influenza/A/H3(HA)'. Influenza A can infect a broad range of animal hosts (*e.g.* waterfowl, swine, and horses) and currently accounts for about 30,000 human deaths a year in the United States alone [52, 53]. The influenza A virus possesses a single-stranded RNA genome and lacks proof-reading upon replication, leading to an exceptionally rapid rate of evolution and abundant genetic variation [54].

The hemagglutinin gene (HA) encodes several antigenic sites that can elicit an immune response and is therefore likely to be responsible for much of the virus' adaptability. This data set also provides an example of branch-by-branch variation in substitution rates, which was attributed by Bush *et al.* [51] to selection in a novel environment caused by the serial passaging of influenza A virus isolates in chicken eggs in the laboratory. Furthermore, it demonstrates the importance of modeling site-by-site variation in rates, due to the immunological roles of specific amino acids in the HA protein.

We have also prepared several example data sets containing a smaller number of influenza A virus (H3) HA gene sequences. Although sample sizes in this range are only marginally sufficient for measuring selection with confidence, such samples are more representative what investigators tend to employ for this purpose. They are also more manageable and can be analyzed using the methods described below on a conventional desktop computer quickly. An archive with example alignments can be downloaded from `http://www.hyphy.org/phylohandbook/data.zip`, and some of the analytic results from `http://www.hyphy.org/phylohandbook/results.zip`

2.3 Prerequisites

2.3.1 Getting acquainted with HyPhy

HyPhy is a software package for molecular evolutionary analysis that consists of three major components:

(i) a formal scripting language (HyPhy Batch Language or HBL) designed to implement powerful statistical tools for phylogenetic inference by maximum likelihood;
(ii) a pre-packaged set of templates in HBL that implement a comprehensive array of standard phylogenetic analyses; and
(iii) a graphical interface for Mac OS, Windows and the GTK toolkit for X11 which runs on nearly all Unix and Linux distribution, that provides quick and intuitive access to these templates alongside visual displays of sequences, trees, and analytical results.

As a result, there is frequently more than one way to do things in *HyPhy*. In most cases, we will use the template files (*i.e.* 'standard analyses') to demonstrate how to carry out various analyses of selection. A standard analysis is activated by selecting the menu option '`Analysis` > `Standard Analyses...`'.

Installing the *HyPhy* package will create a folder containing the executable file and a number of subfolders. Running the executable from your desktop

will activate the graphical user interface (GUI), while running the executable from a command-line will activate a text-based analog. Upon activation of the *HyPhy* GUI, a console window will appear on your desktop, which may be accompanied by an optional greeting window (which can be dismissed by clicking '`OK`').

The console window consists of two parts: the log window, in which most results from standard analyses will be displayed; and the input window, in which you may be prompted by *HyPhy* to specify various options during an analysis. When a standard analysis is being executed, the name of the corresponding template batch file is displayed in the lower-left corner of the console window. There is also a status indicator at the base of the console window, which is most useful for tracking the progress of a long-term analysis (*i.e.* optimization of a likelihood function). At the bottom-right corner, there are icons that activate various functions such as web-updating of the software package; however, most users will not need to use these functions in general. Depending on which platform you use, there may be menu options at the top of the console window.

Any object (*i.e.* sequence alignment, tree, likelihood function) in *HyPhy* can be viewed in a type-specific window. The most important of these is the data window, which displays sequences and provides a number of tools for setting up an analysis of selection. To open a window for any object stored in memory, access the object inspector window by selecting the menu option '`Windows` > `Object Inspector`'. We will describe the basic aspects of each window type as we proceed in the following sections. In versions of *HyPhy* compiled for Microsoft Windows and the GTK toolkit for various flavors of Unix and Linux, different menu options are distributed between the various types of windows. For Macintosh versions of *HyPhy*, a single set of menu options that change in response to the active window type is displayed only at the top of the screen. For command-line versions of *HyPhy*, most menu options in the GUI are either irrelevant or have an analogous batch language command.

2.3.2 Importing alignments and trees

No matter what software package you choose, any implementation of a codon substitution model for measuring selection will require both an alignment of sequences and a corresponding tree. There are many programs available for generating either an alignment or inferring a tree. Although its main purpose is for analyzing alignments and trees provided by the user, *HyPhy* has limited capabilities built-in for preparing both an alignment and a tree

from any given set of related sequences, which will be discussed in sections 2.3.5 and 2.3.6, respectively. Here, we will demonstrate how to import an alignment and a tree from the respective files into *HyPhy*. We will also discuss how to use the graphical user interface (GUI) of *HyPhy* to visually inspect and manipulate alignments and trees. Although this is not usually a necessary step in a selection analysis, we consider it prudent to confirm that your data is being imported correctly, regardless of whichever software you use.

To import a file containing sequences in *HyPhy*, select the menu option '`File > Open > Open Data File...`' to bring up a window displaying the contents of the current working directory. By default, *HyPhy* does not necessarily assume that every file in the directory is readable, and might not allow your file to be selected as a result. For example on Mac OS you may need to set the '`Enable`' tab located above the directory display to the option '`All Documents`'. At this setting, *HyPhy* will attempt to open any file that you select, irrespective of type. In contrast to other packages, *HyPhy* is actually quite versatile in being able to handle FASTA, GDE, NEXUS, and PHYLIP-formatted sequences from files generated in Windows, Unix, or Macintosh-based systems equally well.

Nevertheless, like any other software package, *HyPhy* makes certain assumptions about how a file containing sequences or trees is formatted. For example, some non-standard characters that are occasionally inserted into sequences (*e.g.* the tilde character ~ which is used by BioEdit) will be ignored by *HyPhy* and can induce a frame-shift upon importing the sequences (but this does not affect standard IUPAC symbols for ambiguous nucleotides, *e.g.* 'R' or 'Y', or gaps, *e.g.* '–' or '.'). In addition, *HyPhy* has some restrictions on how sequences can be named. Every sequence name must begin with a letter or a number, and cannot contain any punctuation marks or spaces (*i.e.* non-alphanumeric characters) with the exception of the underscore character, '_' (which is conventionally used to replace whitespace). Selecting the menu option '`Data > Name Display > Clean up sequence names`' will automatically modify the names of imported sequences to conform to these requirements, and also renames sequences with identical names. There is no restriction on the length of a sequence name. Having incompatible names in the sequence file will not prevent you from importing the file or displaying the sequences in the GUI, but may cause problems in subsequent analyses.

The procedure for opening a tree file in *HyPhy* is virtually identical to that for opening a sequence file, except that you select the menu option '`File > Open > Open Tree File...`'. *HyPhy* will import trees generated in other software packages such as *PAUP** [55] or *PHYLIP* [56], but may

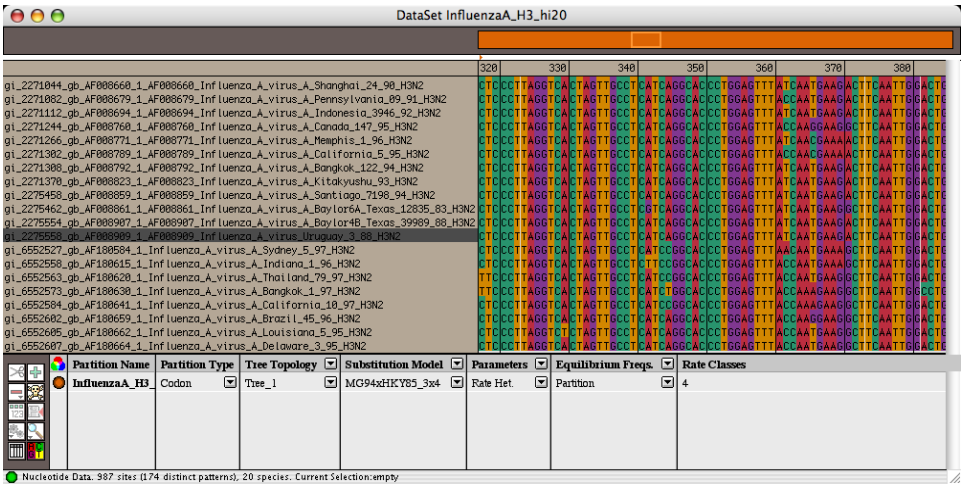

Fig. 2.1. Example of a data window in *HyPhy*

fail to load a tree from a PHYLIP-formatted file (*i.e.* a Newick string) if the tree contains incompatible sequence names containing punctuation marks or spaces. If a tree containing incompatible sequence names is imported from a NEXUS-formatted file, however, then the tree will be accepted and displayed properly (but it is highly recommended to clean up the sequence names). Trees recovered from a NEXUS-formatted file will not be automatically displayed in *HyPhy*, but are available for subsequent analyses.

2.3.3 Previewing sequences in HyPhy

Importing sequences from a file will automatically spawn a data window in the *HyPhy* GUI (Fig. 2.1). A data window consists of several fields:

(i) **Name display.** Sequence names are listed in the leftmost field, and can be highlighted individually or in sequence. You can rename a sequence by double-clicking on the corresponding name in this field. A number of options affecting sequence names can be accessed in the menu '`Data > Name Display`' or by right-clicking on the field.

(ii) **Sequence display.** The nucleotide or amino acid sequence corresponding to each name is displayed in the field immediately to the right of the name display. Directly above this field, there is a modified horizontal scroll bar indicating what interval of the sequence alignment is currently being displayed in the window.

(iii) **Partition display.** Located at the base of the data window, the partition display consists of a chart in which a new row is displayed for every data partition that is defined on the alignment. This display can be used for setting up a limited number of analyses.

A data partition is a fundamental object in *HyPhy* that defines which columns in the alignment will be passed onto an analysis. By specifying a data partition, for instance, we can perform a selection analysis on a specific region of a gene. You can create a partition from the entire alignment by selecting the menu option '`Edit` > `Select All`', followed by '`Data` > `Selection->Partition`'. Every time a new partition is defined on a data set, a colored field will appear in the horizontal scroll bar at the top of the data window to indicate the range of the alignment in the partition, and the partition display will be updated.

Having an open data window provides a good opportunity to visually inspect your alignment. To make it easier to see discrepancies in the alignment, *HyPhy* can display nucleotide and amino acid alignments in a block-color mode that is activated by the '`Toggle Coloring Mode`' (colored AC/GT) icon located to the left of the partition display . If one of your sequences acts as a reference for aligning other sequences, you can click and drag it to the desired location in the name display window.

The Influenza/A/H3(HA) alignment extends beyond the region coding for the hemagglutinin gene to include non-coding nucleotides. In order to fit a codon substitution model to our data, we first must isolate the coding region in our alignment. You can select a range of columns in the sequence display by clicking at one location and dragging across the desired interval (this automatically selects every sequence in the alignment). Alternatively, you can click on one column of the alignment, and while holding the 'shift'-key click on another column to select the interval contained in-between. Once the desired interval has been selected, create a partition as above so that a new data partition appears in the list. You can also specify a data partition by selecting '`Data` > `Input Partition`' from the menu and entering a partition specification string; *e.g.* "1-30" creates a partition of the first 30 nucleotide positions in the alignment.

One advantage of creating a data partition is that a number of additional tools specific to partitions can be activated using the icons to the left of the partition display in the data window. For example, you can search for identical sequences in the data partition by selecting the '`Data Operations`' (magnifying glass) icon. Removing identical sequences from the partition can provide a considerable savings in computational time for complex selec-

tion analyses. In our example, this operation identifies eight sequences that are identical to another sequence in the partition, which become highlighted in the name display field. To filter these sequences from the alignment, select '`Data > Invert Selection`' from the menu and right-click on the name display field to spawn a new data window. The contents of the new window can be saved to a NEXUS-formatted file (by selecting the menu option '`File > Save > Save As...`' and setting `Format` to "`Include Sequence Data, NEXUS`" in the window that appears). Any data partition can be written to a file by selecting the 'Save Partition to Disk' (floppy disk) icon, which can then be re-imported for subsequent analyses in *HyPhy*.

Exercise. Import sequences from `InfluenzaA_H3.fasta` into the data viewer using `File>Open>Open Data File` (on Mac OS X you need to enable `All Documents` in the file dialog to make the file 'selectable'). 357 sequences with 987 nucleotides each will be displayed. As is often common with GenBank sequence tags, sequence names are very long and contain repetitive information. For example, the first sequence in the file is named `gi|2271036|gb|AF008656.1|AF008656 Influenza A virus (A/Perth/01/92(H3N2)) hemagglutinin (HA) gene, partial cds`. Invoke `Edit>Search and Replace` from the data viewer to access a search and replace dialog for sequence names. *HyPhy* incorporates support for *regular expressions* - a powerful mechanism to specify text patterns. Enter the search pattern `.+Influenza A virus \(` - a regular expression that will match an arbitrary number of characters followed by the substring `Influenza A virus` and an opening parenthesis '(' - and an empty pattern to replace with. Next, repeat this with the search pattern `\(H3N2\).+`, specifying (H3N2) followed by an arbitrary number of any characters. Now, the first sequence name is a lot more manageable `A/Perth/01/92`, with others following in kind. Scroll down to examine other sequence names, and notice that two of the names are still long. One of them is `gi|2271150|gb|AF008713.1|AF008713 Influenza A virus (A/Fukushima/114/96`. This is because two sequences would have been marked as `A/Fukushima/114/96` had the second one being processed, thus HyPhy skipped that sequence during renaming. Double click on the long sequence name, and manually edit it to `A/Fukushima/114/96_2`. Finally, manually edit the last remaining long sequence name. Select `Data>Name Display>Clean up sequence names` to enforce valid HyPhy sequence names. Execute `Edit>Select All` followed with `Data>Selection→Partition` to create a partition with the all alignment columns. Click on the looking glass button, and choose `Identical Sequences Matching Ambiguities` to identify all sequences which are an identical copy (where ambiguities are treated as a match if they overlap in at least one resolution) of another sequence in the data set (8 sequences will be selected). Execute `Data>Invert Selection` followed by a right click (or Control-click on Macintosh computers with a single button mouse), in the sequence name part of the panel to display a context sensitive menu, from which you should select `Spawn a new datapanel with selected sequences`. This will create a new panel with 349 (unique) sequences. Create a new partition with all sequence data, and click on the disk button to save it to disk for later use (*e.g.* as a NEXUS file). The file generated by these manipulations can be found in `InfluenzaA_H3.nex`.

2.3.4 Previewing trees in HyPhy

Importing a tree from a file will automatically open a tree display window (unless you have unflagged this option in the *HyPhy* preferences menu). Any tree that is stored in memory can also be viewed at any time by opening the object inspector window, selecting the '`Trees`' setting, and double-clicking the tree object in the list. An example of a tree viewing window is shown in Fig. 2.2. The main viewer window depicts a portion of the tree, which corresponds to the section indicated in the summary window in the upper left

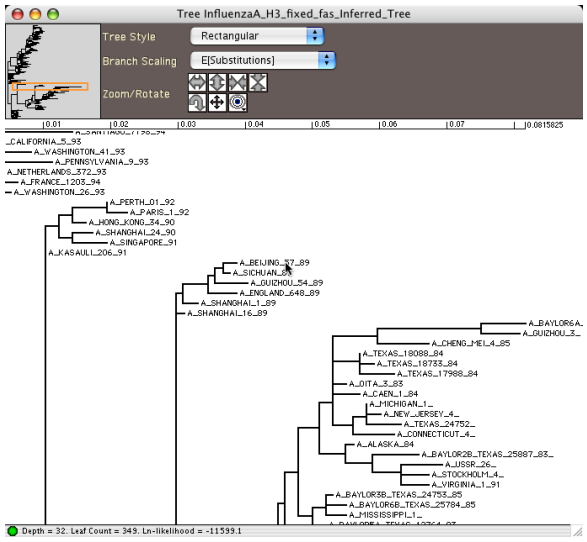

Fig. 2.2. Previewing a tree object in *HyPhy*

corner. By click-dragging the box in the summary window, you can quickly select what region of the tree to display in the main window. There are also a number of tools for modifying the main viewer perspective activated by various icons grouped under the label '`Zoom/Rotate`'.

HyPhy can plot the tree in several conventional styles (*e.g.* slanted, radial) that are selected from the '`Tree Style`' tab, but defaults to plotting 'rectangular' trees as shown in Fig. 2.2. The '`Branch Scaling`' tab allows you to select unscaled branches (*i.e.* cladogram) or branches scaled according to the length values (phylogram) either provided in the original file, or estimated *de novo* by *HyPhy*. If the tree is associated with a substitution model, then the branch lengths can also be scaled to specific model parameters.

2.3.5 Making an alignment

Generating a multiple sequence alignment is a computationally-demanding problem that can require a long period of time to complete on conventional desktop computers. For alignment methods and tools we refer the reader to the relevant chapters of this book. *HyPhy* has a limited built-in capability for sequence alignment that is faster than the algorithms implemented in most alignment software, by making assumptions about genetic distance that are appropriate under certain circumstances. This alignment algorithm is implemented in a batch file called `SeqAlignment.bf`, which can be accessed in the

`Standard Analyses` menu under the heading `Data File Tools`. The most common procedures for multiple sequence alignment are based on progressive algorithms, in which the addition of each sequence requires a pairwise comparison to every sequence in the alignment (*i.e.* with a time complexity $O(n^2)$ or greater for n sequences). In contrast, `SeqAlignment.bf` performs pairwise alignments of each sequence to a user-defined reference sequence, requiring linear time $O(n)$. This can provide a sufficient alignment when the sequences can be assumed to be related by a star phylogeny, as is often the case with human immunodeficiency virus type 1 (HIV-1) sequences from within-patient isolates. `SeqAlignment.bf` translates codons sequences to amino-acids, aligns amino-acid sequences and then maps them back to nucleotides, enforcing the maintenance of reading frame.

Regardless of which procedure is employed to generate an alignment, it is always recommended to visually inspect your alignment before attempting a selection analysis. For example, misaligned codons in a sequence may be interpreted as spurious nonsynonymous substitutions in the alignment that may cause your analysis to overestimate β at that position.

Exercise. Open `InfluenzaA_H3.nex` in a data viewer and create a single partition with all sequence data. Change partition type from `Nucleotide` to `Codon`, selecting `Universal` genetic code. Note that instead of creating a solid partition (shown in the upper scroll bar), HyPhy skipped a number of positions, because they contained premature stop codons in some of the sequences. Select `Data>Partition→Selection` to highlight sites included in the alignment, and scroll around to see which sequences appear to have premature stop codons. A cursory examination will show that some of the sequences are simply missing a few starting nucleotides, and hence, are in another reading frame. Execute `StandardAnalyses>Data File Tools>SeqAlignment.bf` to perform a simple clean-up alignment on `InfluenzaA_H3.nex`. Use `BLOSUM62` scoring matrix, with `No penatly` for Prefix/Suffix Indels and `First in file` reference sequence, `No` reference sequence and `Universal` genetic code. The analysis finds 347 sequences in reading frame 1 and 2 sequences in frame 3. The output of `SeqAlignment.bf` consists of both a protein and a nucleotide alignment (`.nuc` extension) saved to a file of your choice. Import the two cleaned up alignments and check that the reading frame is now preserved throughout. For your reference, the output is available in `InfluenzaA_H3_cleaned.fas` and `InfluenzaA_H3_cleaned.fas.nuc`. Note that if `SeqAlignment.bf` does not work well, another alignment program or manual editing may be needed to prepare an alignment for codon model analyses.

2.3.6 Estimating a tree

Once an alignment has been prepared and inspected, it can be applied towards the reconstruction of the evolutionary relationships among the sequences. Again, there are many programs available for estimating the tree from an alignment. *HyPhy* has a reasonably diverse suite of procedures built-

in for phylogenetic reconstruction, including clustering methods (*e.g.* UPGMA) [57], neighbor-joining [9], and maximum-likelihood based tree search algorithms (*e.g.* star decomposition). Details on any of these procedures can be found in previous chapters on phylogenetic reconstruction. These procedures can be accessed from the `Standard Analyses` menu under the heading `Phylogeny Reconstruction`.

To generate a tree from an alignment by the neighbor-joining method in *HyPhy*, select the batch file `NeighborJoining.bf` from the menu. You will be prompted to specify the following:

(i) `Distance Computation` — Quantify the genetic distance between two related sequences by using predefined distance measure (*e.g.* Jukes-Cantor); estimating branch lengths by maximum likelihood given a standard substitution model; or using a user-defined matrix of pairwise distances†.

(ii) `Data type` — Whether genetic distances are to be defined at the nucleotide, amino acid, or codon level.

(iii) `Data file` — Select a file containing the sequences from a window displaying the current working directory.

(iv) `Negative Branch Lengths` — Whether to allow branch lengths with negative values, or to reset these to zero.

(v) `Distance formula/standard model`— Select a distance formula or substitution model from a list of standard formulae/models. If using a model to compute distances, specify whether to estimate the model parameters locally or globally, and whether to model site-by-site rate variation (see sections below).

Once *HyPhy* has finished reconstructing a tree by the neighbor joining method, the corresponding Newick tree string can be exported to a user-specified file for later use in a selection analysis. *HyPhy* will always ask whether a newly inferred tree should be saved to a file.

In practice, it turns out that the outcome of a selection analysis is usually fairly robust to the method used to reconstruct the tree; nevertheless, it is always a good idea to make a reasonable effort to provide an accurate reconstruction. Moreover, many tree search algorithms can require a long time on most desktop computers, especially when dealing with large alignments, although there exist very fast heuristic algorithms, which can handle even large alignments quickly,*e.g.* as implemented in PhyML (`http://atgc.lirmm.fr/phyml/`) and GARLi (`http://www.bio.utexas.edu/faculty/antisense/garli/Garli.html`). In most cases, the neighbor-joining method with an appropriate distance formula (*e.g.* Tamura-Nei (TN93) distance

† You will be prompted to select a file containing a *HyPhy*-formatted $n \times n$ matrix, where n is the number of sequences; *e.g.* "{{0, 0.1}, {0.1, 0}}" specifies a pairwise distance of 0.1 between two sequences.

[58]) will provide a sufficiently accurate reconstruction for an analysis of selection. Estimating a neighbor-joining tree using these settings for the Influenza/A/H3(HA) alignment, for example, should only require about half a minute on a conventional desktop computer.

Exercise. Open `InfluenzaA_H3_fixed.fas.nuc` in a data viewer and create a single partition with all sequence data. Set `Tree Topology` to 'Infer Tree', `Substitution Model` to 'REV', `Parameters` to 'Global' and `Equilibrium Frequencies` to 'Partition'. Select `Likelihood>Inference>Infer Topology` choosing `Neighbor_Joining`, `Force Zero` and `TN93`. HyPhy will infer the tree in about 15 seconds and then proceed to fit the 'REV' model to the alignment, which will take a few minutes. Inspect the fitted tree (`Window->Object Inspector`). A part of the resulting tree with branch lengths derived from the REV model is shown in Fig. 2.2. Use `MeanPairwiseDivergence` accessible from `User Actions` (the gears icon) button in the bottom right corner of the *console* window to compute mean pairwise nucleotide divergence of the Influenza tree (4.25% with the 95% confidence interval of $4.03 - 4.48\%$). Save the results via `File>Save` choosing `Include sequence data, NEXUS` option from the pull down menu in the file dialog - this will create a self-contained document with the alignment, tree and the code needed to define the models and store parameter estimates. Finally, export the partition to a file (this will also include the inferred tree in the file).

2.3.7 Estimating nucleotide biases

Different types of nucleotide substitutions rarely occur at exactly the same rate over time. For example, there is a well-known bias favoring transitions (*e.g.* G→A) over transversions (*e.g.* G→C). Failing to account for such biases can severely affect the accuracy of estimating non-synonymous and synonymous substitution rates, and the reliability of a selection analysis thereby. Biases in the nucleotide substitution rates can be estimated by fitting a general time-reversible model (GTR) of nucleotide substitution to the alignment given a tree [59]. By definition, this is the most complex time-reversible model that requires the estimation of six parameters. However, simplified versions of the GTR model may often provide a sufficient approximation of the nucleotide substitution rates with fewer parameters, improving the efficiency of subsequent analyses and preventing over-fitting of the data. Although there are several standard or 'named' models of nucleotide substitution (*e.g.* Hasegawa-Kishino-Yano or HKY85 [60]), the evolution of nucleotide sequences is often best explained by nonstandard models [61, 21]. The GTR model contains six parameters, corresponding to the substitution rates between $A \leftrightarrow C/G/T$, $C \leftrightarrow G/T$, and $G \leftrightarrow T$. The parameters can be grouped in 203 different ways - with each group sharing a common rate, defining a range of models from F81 [10] (a single group, all

rates are equal), to HKY85 [60] (2 groups, one for transitions and one for transversions) and ultimately GTR itself (6 groups, one parameter in each).

An iterative procedure that evaluates every possible time-reversible model has been implemented in *HyPhy* in the `Standard Analyses` menu under the heading `Model Comparison`, called "`NucModelCompare.bf`". This procedure determines which model accomplishes the most accurate fit to the data with the fewest number of parameters according to Akaike's Information Criterion (AIC). Other than requiring you to select files containing sequences and the tree, the batch file will prompt for the following:

(i) `Model Options` — Whether to fit the model with local or global parameters and to model site-by-site variation in rates (see section 2.4.2)

(ii) `Estimate Branch Lengths` — whether branch lengths are to be re-estimated for every model, or to re-use estimates from the GTR model;

(iii) whether to create a NEXUS-formatted file for every model fit (please note that this will create 203 files);

(iv) and the significance level at which models are rejected (in console window).

Fitting the models using global parameters and re-using branch length estimates can substantially reduce the amount of time required for the procedure to run, and should yield a sufficiently accurate assessment in almost all cases. Because a large number of models are being fit, this procedure can require a substantial amount of time. For example, the nucleotide model comparison for our Influenza/A/H3(HA) alignment required about 35 minutes to complete on a desktop computer. (`NucModelCompare.bf` contains a parallel-computing implementation for running on a distributed cluster, and a version of it has also been implemented in *http://www.datamonkey.org*.)

It is usually a good idea to export the best fitting nucleotide model by writing the likelihood function object to a NEXUS-formatted file, which is carried out in *HyPhy* by selecting the menu option '`Analysis > Results > Save Results > Export Analysis`'. In section 2.6, we will describe how to import the fitted nucleotide model from this file into a selection analysis.

Exercise. Use `Standard Analyses>Model Comparison>NucModelCompare.bf` to find the best nucleotide model for `InfluenzaA_H3_final.nex`. Use `Global` model options, estimate branch lengths `Once`, model rejection level of 0.0002 (a number this small ensures that even after the maximal possible number of tests, the overall error rate does not exceed 0.05, based on the conservative Bonferroni correction) and forego the saving of each of the 203 model fits. Based on AIC, the best fitting model should be `(012312)`. Repeat the analysis with a random subset of 35 sequences from the master alignment (`InfluenzaA_H3_Random35.nex`), to investigate how much of the signal is lost when only 10% of the sequences are included. Not surprisingly, the best fitting model is a slight simplification of the one we found from the large alignment: `(012212)`.

2.3.8 Detecting recombination

Many phylogenetic methods implicitly assume that all sites in a sequence share a common evolutionary history. However, recombination can violate this assumption by allowing sites to move freely between different genetic backgrounds, which may cause different sections of an alignment to lead to contradictory estimates of the tree [62] and subsequently confuse model inferences [63]. For example, failing to account for recombination can elevate the false positive error rate in positive selection inference [64, 65]. This problem in model inference can be accommodated by allowing different parts of an alignment to evolve independently, each according to their own unique tree. However, we are still required to specify the positions in the alignment at which recombination events have taken place (*i.e.* breakpoints). There is a large number of methods available for accomplishing this task [62]. A simple approach that performs at least as well as any other method available [11] specifies a given number of breakpoints ($B \geq 1$) at which recombination events have occured between some number of sequences in the alignment. This method is implemented in *HyPhy* in the batch file `SingleBreakpoint-Recomb.bf` for the case $B = 1$, which can be accessed from the `Standard Analyses` menu under the heading `Recombination`.

The batch file will prompt you for the following options:

(i) `Data type` — Should *HyPhy* fit a nucleotide or codon model of substitution to the data? You will be prompted to select a file containing the sequence data.

(ii) `KH Testing` — Evaluate the fit of models using AIC [66]; or use Kishino-Hasegawa resampling of sequences [67] to estimate the confidence interval for the improvement of log-likelihood from the null model (either the model fitted without a recombination breakpoint, or by swapping topologies between sites on either side of the breakpoint).

(iii) `Standard model` — To fit to the data for evaluating incongruence of phylogenies on either side of the breakpoint. There may also be the usual additional options for model fitting (*i.e.* local vs. global, site-by-site variation);

(iv) and a file to write output from the analysis to.

The batch file will iterate through every potential recombination breakpoint in the alignment, and fit independent models to the sequences on either side of the breakpoint. Consequently, the detection of recombination breakpoints can require a long time to compute for a large alignment (*i.e.* over 50 sequences), although other methods of recombination detection can be run more quickly.

Exercise. Run `Standard Analyses>Recombination>SingleBreakpointRecomb.bf` on `InfluenzaA_H3_Random35.nex` using `Nucleotide` model, `Skip` KH testing, use `CUSTOM` model with `(012212)` correction found in an earlier exercise, `Global` model options, with `Estimated` rate parameters, using `Observed` equilibrium frequencies. The analysis will examine 189 potential breakpoint locations (all variable sites in the alignment) using three information criteria (AIC_c is the default one) and fail to find evidence of recombination in these sequences, because models with two trees have worse scores (negative improvements) than the model without recombination. The analysis completes in about $5-10$ minutes on a desktop. You can also run this analysis via Datamonkey at `http://www.datamonkey.org/gard` [11].

2.4 Estimating global rates

As mentioned in section 1.3, comparing the global estimates of α and β averaged over the entire alignment can provide a crude measure of the overall strength of selection on the coding region. Global estimates of α and β can be obtained by fitting a codon substitution model to a given alignment and corresponding tree [2]. There are several procedures that are available in the *HyPhy* package for fitting codon models. We will describe two methods of fitting a codon model to the Influenza/A/H3(HA) alignment, first using a basic implementation using the graphical interface, and then a more comprehensive implementation that is available as under the `Standard Analyses` menu.

2.4.1 Fitting a global model in the HyPhy GUI

Importing an alignment in the *HyPhy* graphical user interface (GUI) will automatically spawn a data window, containing a basic array of analytical tools. We have already described how to create a data partition from an alignment in section 2.3.2, which will appear as a new row in the list of data partitions located at the base of the data window. To set up an analysis on the data partition, you will need to specify a number of options under each column of the partition list, which can be modified by clicking on the corresponding button. (Note that each partition row has an associated set of buttons, distinct from the buttons located beside each column heading that specify options for all partitions in the list.) For example, under the column heading '`Partition Type`' there are three options available: nucleotide, dinucleotide, and codon. To obtain global estimates of nonsynonymous and synonymous substitution rates, we specify that the partition contains codon data, which requires a user-specified reading frame and genetic code to be selected in the window that appears.

A partition must also associated with a tree topology and substitution model, which are specified in the next two columns of the partition list. A tree can be imported from a file by selecting '`Read Tree From File...`'

from the pop-up menu under the column heading '`Tree Topology`'. (If a tree was included in the imported NEXUS file, then it will also appear as a selectable option.) Selecting the '`Infer Tree`' option allows you to access one of the many available methods in *HyPhy* to estimate a tree *de novo* (see section 2.3.6). Finally, selecting the '`Create New Tree...`'option allows you to specify an arbitrary tree topology by either choosing one of several template topologies, or inputting a Newick tree string. In most cases, however, you will want to select one of the other two options.

A limited number of standard substitution models are available for each type of partition under the column heading '`Substitution Model`'. For example, the substitution models that are available for a codon partition are either based on Goldman-Yang (GY94) [3] or Muse-Gaut (MG94) [2] rate matrices. Many models based on MG94 matrices are further 'crossed' by one of several standard nucleotide rate matrices, as indicated in *HyPhy* by the concatenation of "MG94" with a standard nucleotide model abbreviation (*e.g.* `MG94xHKY85`). Equilibrium codon frequencies are either estimated from the overall frequency of nucleotides, or from nucleotide frequencies specific to each codon position (indicated in the model specification string by the suffix "3x4"). The standard codon model generated by crossing MG94 with the REV nucleotide model (`MG94xREV_3x4`) provides a general model which can be constrained down to a simpler version as needed (see the next Exercise).

For any model that depends on more than one rate parameter, *HyPhy* can either obtain local parameter estimates for each branch in the tree, or global parameter estimates that are shared by all branches the tree. Moreover, *HyPhy* can allow global parameter estimates to vary across codon positions in the data partition (*i.e.* rate heterogeneity). If this option is selected, then a field will appear under the column heading `Rate Classes`, which can be edited by the user). Model parameter options can be specified for a data partition under the column heading `Parameters`. Whichever option is selected for analyzing a codon partition will determine the total number of parameters to be estimated, and thereby the length of time required to optimize the likelihood function. For example, estimating a local `MG94` model effectively doubles the number of parameters because the rates α and β are being estimated for every branch in the tree.

After an analysis has been set up for the codon partition, a likelihood function can be constructed by selecting the menu option '`Likelihood > Build Function`'. When building a likelihood function, *HyPhy* will apply an algorithm in order to improve the efficiency of likelihood evalutation, which can substantially reduce the amount of time required to estimate the model parameters [68].

Exercise. Estimate the global ω for `InfluenzaA_H3_Random35.nex` using the GUI. Import the alignment into a data panel viewer via `File>Open>Open Data File` and create a partition with the entire length of the sequence. Change data type of the partition to `Codon`. In the dialog which appears, ensure that `Universal` genetic code is selected and rename the partition to `HA` (for brevity). Select `InfluenzaA_H3_Random35_tree` for the tree topology (this tree was read automatically from the alignment file), `MG94xREV_3x4` substitution model with `Global` parameters and `Partition` based equilibrium frequencies. Note that the light icon in the bottom left corner of the data panel window has changed to yellow from red, indicating that we have provided enough information for HyPhy to build a likelihood function. Execute `Likelihood>Build Function` and examine console output for confirmation that the function has been built. Note that the `HA` partition name has been converted to boldface, to indicate that this partition is a part of an active likelihood function and the light icon has turned to green. Before we optimize the likelihood function, we need to constrain the REV model down to (012212) - a best fitting nucleotide model. Open the parameter viewer (the table button in the bottom left panel of the data viewer). In the parameter table, select the rows for global rate parameters `HA_Shared_AT` (see `http://www.hyphy.org/docs/HyphyDocs.pdf`) for a HyPhy primer including variable naming conventions and many other useful tips), `HA_Shared_CG` and `HA_Shared_GT` (hold the Shift key down and click to select multiple rows), and click on the constrain button, forcing all parameters to shared the value of `HA_Shared_AT`. Note how the display for two of the tree variables has changed to show the constraints. Also, double-click in the constraints window for `HA_Shared_CT` and enter "1" (to enforce $\theta_{AG} = \theta_{CT} = 1$). Finally, select `Likelihood>Optimize LF` to obtain MLE of parameter values. When the optimization has finished ($1 - 3$ minutes on a desktop), HyPhy will refresh the parameter table with derived parameter estimates (see Fig. 2.3). The global ω estimate, represented by `HA_Shared_R` has been estimated at 0.495. Select the row with this variable and execute `Likelihood>Covariance, Sampler and CI` to obtain profile likelihood error estimates on ω (`Likelihood Profile [chi2]` with significance of 0.95). The reported 95% confidence interval bounds are 0.419 and 0.579. Once a likelihood function has been fitted to the data, it is immediately possible to simulate data under the model(s) in the function via the `Data>Simulation` menu. Finally, save the likelihood function with all parameter estimates by switching back to the data panel, executing `File>Save>Save`, and choosing `Include sequence data, NEXUS` as a format option in the save dialog.

2.4.2 Fitting a global model with a HyPhy batch file

There are also a number of template batch files listed in the `Standard Analyses` menu that will fit a codon substitution model to a data partition. The most appropriate procedure for estimating the global rates α and β, however, has been implemented in the batch file `AnalyzeCodonData.bf`, which can be selected from the `Basic Analyses` submenu. A number of codon models are available in this batch file that have not been implemented in the *HyPhy* GUI.

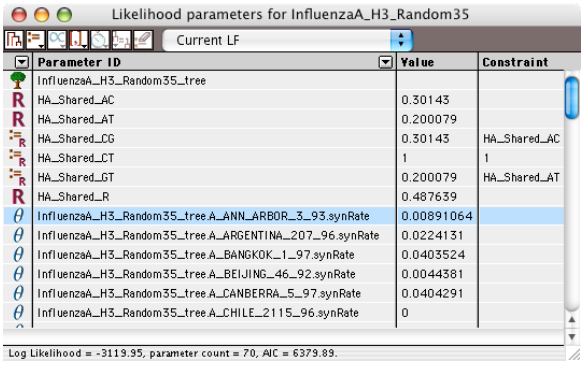

Fig. 2.3. Inspecting a likelihood function in *HyPhy*

As before, we have the option of fitting the model locally or globally, and can allow global parameter estimates to vary across codon positions in the data partition. In addition, this batch file can model variation across codon positions according to a hidden Markov model, which assumes that the evolutionary rates at adjacent positions are more similar, according to an autocorrelation parameter (λ) that is estimated by the model. There are also many more distributions available for modeling rate variation across codon positions in `AnalyzeCodonData.bf` (*e.g.* beta, log-normal, and mixture distributions); not all of these distributions are available via *HyPhy* GUI. If the data partition was imported from a NEXUS file containing a tree, *HyPhy* will ask if you wish to use this tree in fitting the model. If not, then you will be prompted to select a tree file.

Exercise. Estimate the global ω for `InfluenzaA_H3_Random35.nex` using `Standard Analyses>Basic Analyses>AnalyzeCodonData.bf`. Select `MG94CUSTOM`, `Global`, `012212`, and use the tree provided in the file. When the analysis is finished, HyPhy will report global ω as $R = 0.495$. This value may be slightly different from the one found through the GUI, because of how initial guesses for the optimization procedure are obtained. Once an analysis has finished, look at the `Analysis>Results` menu - it contains a list of *post processors* which HyPhy can execute after most standard analyses. For instance, the `Syn and non-syn trees` option can be used to display the trees scaled on the expected numbers of synonymous and non-synonymous substitution per codon site, and on dS and dN. Note that because the model has a shared ω for all branches, all four trees are have proportional branch lengths. Finally, many of the GUI tools can be applied to models fitted by standard analysis. To do this, open the likelihood function parameter table, using the `Object Inspector`.

2.5 Estimating branch-by-branch variation in rates

As noted in section 1.5, it is unreasonable to assume that evolutionary rates remain constant over time. For example, a sudden change in the environment affecting one lineage may be manifested as episodic selection [69], which may become averaged out over the entire tree to an undetectable level. In *HyPhy*, it is possible to assign a unique set of substitution rates to every branch in a tree, by locally fitting a codon substitution model. However, because the number of parameters in a local model is approximately proportional to the number of sequences, this may require an exceedingly long time to compute. It also does not provide a robust framework for hypothesis testing (*e.g.* whether a specific branch evolves at a significantly different rate than the rest of the tree). Hence, many procedures for selection inference that allow branch-by-branch variation in rates require the investigator to pre-specify which branches evolve under a particular model [70, 24], as discussed in section 1.5. Several procedures of this type are implemented as batch files in *HyPhy*. In the batch file `SelectionLRT.bf`, a single branch is chosen to partition the other branches of the tree into two clades, for which the model parameters are estimated separately. Another batch file called `TestBranchDNDS.bf` allows you to test whether the strength of selection is significantly different for an arbitrary selection of branches in contrast to the rest of the tree.

To demonstrate the use of these methods in *HyPhy*, we will use a data set containing 20 sequences of influenza A serotype H3 viruses that have been isolated from waterfowl and mammals (mostly equine). Severe outbreaks in horse populations have been caused by equine influenza A virus, with mortality rates as high as 20%. The equine and waterfowl-isolated sequences form two distinct clades in a reconstructed phylogeny. As waterfowl are a main reservoir of influenza A virus infection, the transmission of virus populations to equine hosts may be accompanied by strong positive selection.

2.5.1 Fitting a local codon model in HyPhy

Fitting a local codon model is very similar to the procedure for fitting a global model, as demonstrated in section 2.4. Whether using the partition list menus to setup an analysis in the *HyPhy* GUI, or while executing a batch file, you will almost always have the option to specify whether to fit a model globally or locally. Local models provide a useful exploratory tool, identifying branches in the tree in which strong selection has occurred and providing an alternative model for hypotheses testing. Unless a panel displaying the model parameter estimates was automatically spawned af-

ter optimization of the likelihood function, the most convenient means for viewing local parameter estimates is to access the Object Inspector panel (see section 2.3.3) and select the corresponding likelihood function object. Figure 2.3 depicts a likelihood function panel in which all global and local parameters are displayed.

Each branch in the tree is associated with local parameter estimates of α and β, labeled as "`synRate`" and "`nonSynRate`", respectively. A quick procedure for calculating the ratio β/α for each branch in the tree requires you to open a tree viewer window by 'double-clicking' on the first row in the likelihood function panel, which is labeled by a tree icon. Select all branches by choosing the menu option '`Edit` > `Select All`', and then create a new parameter for each branch by selecting '`Tree` > `Edit Properties`', clicking on the 'plus' icon in the panel that appears, and entering the string "`nonSynRate/synRate`" in the `Formula` window. Once this panel is closed, this new variable will be available for scaling branch lengths in the tree viewer window, in the `Branch Scaling` menu.

It is not unusual for estimates of α or β to converge to zero for one or more branches in a tree. Clearly, this outcome implies that there is insufficient sequence variation to support a branch of non-zero length in that part of the tree. If this occurs, then local estimates of the ratio β/α may assume the undefined value $0/0$ or ∞. As a result, the tree will not be displayed properly in the *HyPhy* tree viewer window when it is scaled to β/α. Selection analyses performed on such trees, however, will not be affected by these poorly-defined branch lengths.

Although locally fitting a codon substitution model can provide a detailed picture of branch-by-branch variation, we strongly caution against re-applying the outcome from this method to other analyses. For example, it is tempting to use a local fit as an initial screen for branches with accelerated evolution, and then specify those branches in a hypothesis-testing procedure on the same data (*e.g.* to obtain a 'P-value'). However, this unfairly biases the analysis towards obtaining a significant result.

Exercise. Fit a local model to the sample of 12 bird and 8 mammalian Influenza/A/H3(HA) sequences (file `MammalsBirds.nex`). Import the alignment (`File>Open>Open Data File`) into a data viewer, create a partition with all sequence data, convert it to a codon partition, select the topology included with file, apply `MG94xREV_3x4` substitution model with `Local` parameters, and frequencies estimates from `Partition`. Fit the model `Likelihood>Build Function` followed by `Likelihood>Optimize`. The procedure completes in a few minutes on a desktop, and should yield a likelihood score of -4837.85. Next we will demonstrate how to test hypotheses with HyPhy GUI by establishing that this tree has variable dN/dS along its branches by comparing the local model fit with the global (single rate model fit). Firstly, we save the likelihood function state describing the local model, by switching to the parameter table display, and choosing `Save LF state` from the pull-down menu at the top of that window. Enter `Local` when prompted to name the likelihood function state. Next, select this model as the Alternative hypothesis, by choosing `Select as alternative` from the pulldown menu. Proceed to define the global model by constraining the β/α ratio for every branch to be the same for all branches in the tree. This is equivalent to setting $treeName.branchName.nonSynRate := globalRatio * treeName.branchName.synRate$ for every branch in the tree. The fastest way to apply this constraint is to invoke `Likelihood>Enter Command` to bring up an interface with the HyPhy batch language command parser, and type in `global globalRatio=1;ReplicateConstraint(‘‘this1.?.nonSynRate:= globalRatio*this2.?.synRate’’,MammalsBirds_tree,MammalsBirds_tree);`. This code instructs HyPhy to traverse the tree MammalsBirds_tree and apply the constraint to every branch ('?' matches all names). Note how the parameter display table changes to reflect newly imposed constraints. Optimize the likelihood function (`Likelihood>Optimize`) to obtain the global model fit. `Save LF state`, naming it `Global` and `Select as null`. Now that we have defined two hypotheses to compare, one can use the LRT (for nested models), and parametric or non-parametric bootstrap to evaluate support for the alternative hypothesis. Select `LRT` from the parameter table pull-down menu. The likelihood ratio test statistic in this case is 100.05, and there are 36 constraints, resulting in a very strong ($p \approx 10^{-7}$) support in favor of the local model. If you now save the likelihood function from the data panel window, all hypotheses defined in the parameter table will be preserved for later.

2.5.2 Interclade variation in substitution rates

We have *a priori* reasons to expect that the ratio β/α could vary significantly between the waterfowl and equine clades, because the virus is evolving in two distinct environments. We have previously investigated these phenomena in epidemiologically linked patient pairs of HIV patients [71]. To evaluate the support for this hypothesis, use the batch file `SelectionLRT.bf` specifying the internal branch which separates the clades of interest. This batch file is executed through the `Standard Analyses` menu, under the submenu

`Compartmentalization`. Upon execution of this batch file, *HyPhy* prompts the user to specify:

(i) a codon translation table (genetic code);

(ii) the file containing the protein-coding sequences;

(iii) a *PAUP** nucleotide model specification string (*e.g.* `010020` for TN93 [58]);

(iv) the file containing the corresponding tree;

(v) and the branch separating two clades in the tree.

Subsequently, *HyPhy* will globally fit the codon model (`MG94` crossed by the specified nucleotide model) to the data, before iteratively evaluating models in which estimates of β for each clade and the internal branch are either independent or constrained to be equal (for a total of five phases of model fitting). To determine whether a nested model provides a significant improvement of fit to the data, *HyPhy* reports p-values from the likelihood ratio test [24] and the Akaike information criterion (AIC) values, which adjust likelihood ratios for the difference in the number of model parameters [66]. *HyPhy* also provides 95% confidence bounds, derived from the likelihood profile (see 1.4.4.2), for estimates of the ratio β/α for each clade and the internal branch.

Exercise. Compare selective pressures on waterfowl and equine clade Influenza sequences from the file `MammalsBirds.nex`. Execute `StandardAnalyses > Compartmentalization > SelectionLRT.bf` using the best fitting (012343) nucleotide model, and selecting `Node1` - the root of the waterfowl clade to define one of the clades (Clade A) of interest. To verify that this is indeed the correct node, you can use the tree viewer. The equine-waterfowl data is best explained (based on AIC) by a fully unconstrained model (`MG94x012343`) in which β is estimated independently for each clade and the separating branch. For the equine clade, β/α was estimated to be 0.223 (95% CI: 0.158, 0.302). In contrast, β/α for the waterfowl clade was estimated to be 0.058 (95% CI: 0.038, 0.083), and the estimate for the internal branch separating the two clades was lower still ($\beta/\alpha = 0.033$; 95% CI: 0.026 0.041).

2.5.3 Comparing internal and terminal branches

A useful hypothesis that can be tested in viral sequences is whether the relative rate of nonsynonymous substitution varies between terminal and internal branches of the tree. Finding a significant difference between these classes would suggest that selection on a virus population within a host was distinct from selection for transmission among hosts. Support for this hypothesis can be evaluated by the batch file `TestBranchDNDS.bf`, which is found in the `Standard Analyses` menu, under the heading `Positive Selection`. The model selection procedure implemented in this file is similar to the previous file, attempting to fit a global model in which β is

constrained to the same value (`SharedNS1`) on all branches, before relaxing this constraint for an arbitrary selection of branches.

However, there are several additional options that are available in `TestBranchDNDS.bf`. For example, you can also model site-by-site variation in rates (see section 2.6), either in β only, or under a 'complete' model in which α and β are both sampled from independent distributions†. As always, *HyPhy* will prompt you to specify the number of rate classes when modeling rate variation. In addition, you can weight the codon substitution rates according to amino acid classifications, such that substitutions between similar amino acids are assumed to be more common.

Exercise. Use `TestBranchDNDS.bf` from the `Positive Selection` rubrik of `Standard Analyses` to test whether terminal branches in `MammalsBirds.nex` evolve with dN/dS different from the rest of the tree (and each other). Use the (012343) nucleotide model, and begin by choosing `None` for site-to-site rate variation and `Default` for amino-acid bias models. When prompted to select branches of interest, use shift-click (or control-click to highlight a range) to select all 20 terminal branches. Because there are 20 terminal branches, the alternative model has 20 additional parameters compared to the null model. Despite the addition of so many parameters, this model was significantly favored over the null model by a likelihood ratio test ($\chi^2_{20} = 68.87$, $p \approx 2 \times 10^{-7}$). The examination of ω values for each branch reported to the console indicate great variability from tip to tip, with several showing accelerated non-synonymous rates (but note wide 95% confidence intervals which highlight the limitations in our ability to infer tight ω estimates for a single branch). As an additional exercise, examine the effect of allowing site-to-site rate variation on the conclusions of the model.

2.6 Estimating site-by-site variation in rates

So far, estimates of β/α have represented an average over all codon positions in the gene sequence. But there are many reasons to expect substantial site-by-site variation in these rates, as discussed in section 1.6. Counting and fixed-effects likelihood (FEL) methods for evaluating site-specific levels of selection have been implemented in *HyPhy* as a single batch file named `QuickSelectionDetection.bf` (some of the options lead to methods that are not terribly quick!), which is found in the `Standard Analyses` menu under the heading `Positive Selection`. Random-effects likelihood (REL) methods have also implemented in *HyPhy* and are described in the next section. In practice, however, all three classes of methods for site-by-site inference of selection converge to very similar results [28].

Counting methods (*e.g.* single likelihood ancestor counting, SLAC) are the most efficient and are well-suited to analyzing large data sets (*i.e.* over

† In either case, rate variation for β and/or α is being modeled by separate gamma distributions, each partitioned into discrete classes according to a beta distribution whose parameters are also estimated [37].

40 sequences), but can be more conservative than the other methods. On the other hand, FEL methods are far more time-consuming but more sensitive, and may be more successful at detecting selection in smaller data sets. REL methods are even slower and may suffer from a high false-positive rate for small data sets.

2.6.1 Preliminary analysis set-up

The batch file `QuickSelectionDetection.bf` was designed to provide a versatile and powerful array of methods for detecting site-by-site variation in selection. As a result, there are several options that need to be specified before an analysis can be carried out. This batch file proceeds in four phases: (*i*) fitting a nucleotide model; (*ii*) generating a codon model approximation; (*iii*) fitting the approximate codon model; and (*iv*) ancestral state reconstruction and substitution counting. A general outline for the preliminary set-up of an analysis follows:

(i) `Choose Genetic Code` — Select a genetic code for codon translation. A comprehensive list of codon translation tables covering a broad range of taxa has been built into *HyPhy*.

(ii) `New/Restore` — Restore nucleotide model parameter estimates from a previous analysis. Choosing '`Restore`' requires you to select a file that contains a previously exported likelihood function, which will also contain the sequence data and tree. This option allows you to iteratively run the batch file under different settings without having to re-optimize the same nucleotide model for every session. It also provides a convenient means of importing the best-fitting nucleotide model from the automated model selection procedure implemented in the batch file `NucModelCompare.bf` (see section 2.3.7). Choosing '`New Analysis`' will prompt you to select a file containing the protein-coding sequences.

(iii) `Model Options` — (New Analysis only) Select a nucleotide model to fit to your data . Choose '`Default`' to fit an HKY85 nucleotide model. Otherwise, you will be prompted to enter a 'custom' *PAUP** model specification string. *HyPhy* subsequently prompts you to select a file containing a tree corresponding to your sequences, and second file for exporting the fitted nucleotide model.

2.6.2 Estimating β/α

Because a codon model contains a large number of parameters, it is impractical to optimize all of them for a large number of sequences. To accelerate estimation under codon models, *HyPhy* applies the branch lengths and nucleotide substitution rate parameter estimates from the nucleotide model to approximate the analogous parameters of the codon model [28]. As a result,

it is potentially important to select a nucleotide model that can provide a reasonably accurate fit to the data (see section 2.3.7).

This approximation scheme introduces a global scaling parameter, called `rConstr`, that is shared by all branches in the tree. This scaling approximation is based on the observation that the joint distributions of branch lengths in nucleotide and codon models tend to be highly correlated [72], such that all branch lengths from the nucleotide model can be adjusted upwards by a given factor to approximate codon branch lengths. The scaling factor will be reported in the *HyPhy* console window during the analysis.

Thus, the next step in setting up our analysis is to specify how the global parameters β/α and `rConstr` are handled during optimization of the approximate codon model. These methods are listed under the heading '`dN/dS bias parameter options`' as:

- '`Neutral`', to constrain β/α to 1.
- '`User`', to prompt for a constant value (> 0) for constraining β/α, if for instance it had been estimated in a previous analysis.
- '`Estimate`', to estimate β/α and `rConstr` from the data.
- '`Estimate + CI`', to estimate β/α and `rConstr`, and calculate confidence intervals for β/α.
- '`Estimate dN/dS only`', to estimate β/α and constrain `rConstr` to be calculated directly from β/α and nucleotide model parameters.

The first two options are provided for model selection, *i.e.* calculating the improvement of fit from incorporating β/α into the model as a global parameter.

At this point, it remains only to specify which counting or FEL method to use before proceeding with the selection analysis. The available methods are listed under the heading '`Ancestor Counting Options`'. In the following sections, we will discuss the practical aspects of applying two of these methods; namely, single ancestor likelihood counting (SLAC), and two-rate FEL.

2.6.3 Single-likelihood ancestor counting (SLAC)

As discussed in section 1.6.3, the nucleotide and codon model parameter estimates are used to reconstruct the ancestral codon sequences at internal nodes of the tree. The single-most likely ancestral sequences are then fixed as known variables, and applied to inferring the expected number of nonsynonymous or synonymous substitutions that have occurred along each branch, for each codon position. This procedure requires you to specify the following:

(i) `SLAC Options` — Apply ancestral reconstruction and counting to the entire tree at once (`Full tree`), or as two separate analyses for terminal and internal branches of the tree, respectively (`Tips vs. internals`).

(ii) `Treatment of Ambiguities` — Ambiguous reconstructions of ancestral codons are averaged over all possible codon states (`Averaged`), or resolved into the most frequent codon (`Resolved`). The latter is more appropriate when ambiguous codons may have been due to sequencing errors, as opposed to being representative of sequence polymorphism (*e.g.* as in bulk viral sequences).

(iii) `Test Statistic` — Use the continuous extension of the binomial distribution (`Approximate`), or simulate a null distribution from the data (`Simulated Null`) - this is an experimental and very slow option. For assigning a p-value to whether β is significantly different from α at a given site, you will be prompted for a significance level in the *HyPhy* console window.

(iv) `Output Options` — Spool the output to the *HyPhy* console window (`ASCII Table`); write the output as a tab-separated table into a file (`Export to File`); or display the output as a graphical chart in a separate window (`Chart`). The last option is only available in GUI versions of *HyPhy*. You will be prompted for a file name if you select `Export to File`.

(v) `Rate class estimator` — Skip the estimation of the number of β and α rate classes (`Skip`), or approximate the number of classes from the data (`Count`).

The output of a SLAC analysis consists of 12 columns (see Exercise below for an explanation), and as many rows as there are codon positions in the alignment. All codon positions with significant positive or negative selection, according to the user-defined significance level are automatically reported to the *HyPhy* console window.

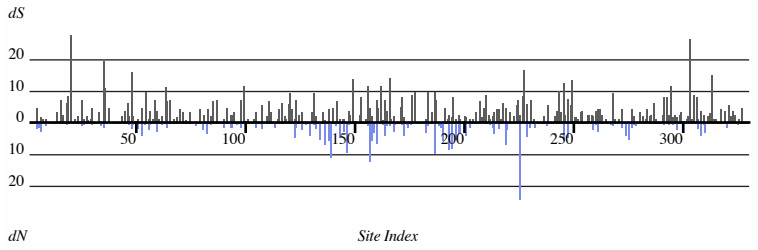

Fig. 2.4. dS and dN estimated from the 349 Influenza sequences using SLAC.

Exercise. Conduct a SLAC analysis on the `InfluenzaA_H3_final.nex` file selecting the `(012321)` nucleotide model, `Estimate dN/dS only` option for the estimation of dN/dS, `Single likelihood ancestor` for the 'Ancestor options' dialog, selection analysis over the `Full tree`, `Averaged` for the treatment of ambiguities, `Approximate` test statistic, p-value of 0.05, `Chart window` for result display and `Skip` the *post-hoc* rate class counter. The entire analysis should take $5-10$ minutes on a desktop computer and identify 8 (128, 135, 138, 145, 194, 226, 275 and 276) positively and 75 negatively selected codons. A chart displayed at the end of a SLAC analysis shows detailed inference for each site, and a dS vs dN plot codon by codon (Fig. 2.4). HyPhy charts can be saved and reloaded (`File>Save>Save Chart`), and their contents can be exported in a variety of formats (`SaveTable`).

. To understand how to read the output for a given codon, look at the entries corresponding to codon 226. Reading the columns left to right, we find out that SLAC inferred 2.5 synonymous and 44.5 non-synonymous substitutions at codon 226, the codon has an expected 1.1 synonymous sites (out of 3) and 1.8 non-synonymous sites. The observed proportion of synonymous substitutions was $2.5/47 = 0.05$, whereas the expectation was much higher $1.10/(1.10+1.82) = 0.37$. $dS = 2.5/1.1 = 2.3$, $dN = 44.5/1.82 = 24.4$ and $dN - dS = 22.14$ are reported next (dN/dS would often be undefined or infinite because of $dS = 0$ - hence the difference is reported). Based on the binomial distribution on 47 substitutions with the expected proportion of synonymous substitutions under neutrality of 0.37, we find that the probability of observing 2.5 or fewer synonymous substitutions is very low (*i.e.* the p-value is 5×10^{-7}), suggesting that the site is under strong positive selection. The probability of observing 2.5 or more synonymous substitutions is ≈ 1 (arguing against negative selection). Lastly, $dN - dS$ scaled by the total length of the tree is 15.7 (this scaling enables the comparison across different datasets). Finally, experiment with the built-in data processing tools in the chart window. For example, click on the 'dN-dS' column heading to select the entire column, then execute `Chart>Data Processing>Descriptive Statistics`. This command will report a number of standard descriptive statistics computed from the selected values to the console, including the mean of -1.805.

2.6.4 Fixed effects likelihood (FEL)

The procedure '`Two rate FEL`' fits the rate parameters α and β to each codon position independently in order to accommodate site-by-site variation. User configurable options specific to FEL include the significance level (p-

value) for the likelihood ratio test and which branches should be tested for selection.

(i) `All`. The entire tree is tested for evidence of non-neutral evolution.

(ii) `Internal Only`. Only interior branches are tested for evidence of non-neutral evolution, while terminal branches share an arbitrary β/α ratio.

(iii) `A subtree only`. Only branches in a given subtree (clade) are tested, while the rest of the branches share an arbitrary β/α ratio.

(iv) `Custom subset`. User selects a set of branches to be tested, while the rest of the branches share an arbitrary β/α ratio.

While the FEL analysis iterates through every codon position in the alignment, it will report estimates of the ratio β/α and the corresponding log-likelihood, likelihood ratio test statistic, and p-value to the console window for each position. Positions under significant positive or negative selection are identified on the right margin by the symbol '`*P`' or '`*N`'. At the end of the analysis, GUI versions of HyPhy will show a chart window displaying analysis results will be displayed, and in all cases the user will be prompted to specify a file to save a comma separated output that is suitable for subsequent analyses in a spreadsheet program, HyPhy GUI - via the `Open>Open Table` menu, or a statistical package.

For every codon, the output from a FEL analysis consists of 8 columns (or 7 if the entire tree is used for testing): `dN/dS` - the ratio of β/α for the branches of interest (could be undefined or infinite if α is estimated to be 0); `dN` - the β estimate for the branches of interest; `dN` - the α estimate for the entire tree; `dS(=dN)` - the $\alpha = \beta$ estimate for the branches of interest under the hypothesis of neutral evolution; `Log(L)` - the likelihood score of a given site (only reported for variable sites); `LRT` - the likelihood ratio test statistic for non-neutral evolution at a given site; `p` - the p-value for the likelihood ratio test; `dN_other` - the β estimate for the background branches (except if the `All` branches option is selected).

You can also use *HyPhy* to determine whether the site-by-site patterns of selection vary from the internal branches to the tips of the tree. A procedure for performing this analysis has been implemented in the batch file `Subtree-SelectionComparison.bf`, under the heading `Positive Selection` in the `Standard Analyses` menu. This analysis performs a site-by-site FEL estimation, but instead of testing whether $\alpha \neq \beta$ for some branches at a given site, it tests whether $\beta_1 \neq \beta_2$ at a given branch, where β_1 is applied to a set of selected branches, and β_2 - to the rest of the tree.

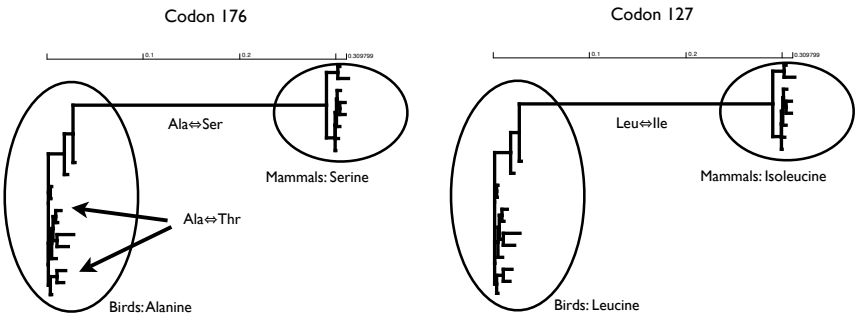

Fig. 2.5. The tree of sequences contained in `MammalsBirds.nex`, with 8 samples from mammals and 12 from birds. Codon 176 is an example of ongoing diversifying selection (in birds), while codon 127 shows what looks like a selective sweep in one of the populations, which can only be detected with a 'branch-site' type method.

Exercise. Conduct a FEL analysis on a sample of 12 bird and 8 mammalian Influenza/A/H3(HA) sequences (file `MammalsBirds.nex`). Confirm that the best nucleotide model is (012343) and then use it to obtain the nucleotide model fit. First, run the `Two rate FEL` analysis on the entire tree using $p = 0.1$. Sample output for two of the codons is shown below.

```
| Codon:  175| dN/dS:  0.54| dN:  0.57| dS:  1.05| dS(=dN):  0.75| Log(L):     -10.90| LRT:  0.16| p:  0.69
| Codon:  176| dN/dS:   inf| dN:  2.35| dS:  0.00| dS(=dN):  1.32| Log(L):     -16.67| LRT:  3.62| p:  0.06 *P
```

The analysis should finish in about 30 minutes on a modern desktop. Two codons (23 and 176) should be found under selection with $p < 0.1$. Repeat the FEL analysis on our cluster using http://www.datamonkey.org. Now the analysis completes in about a minute. Next, conduct a FEL analysis with the same settings except test for selection only within the mammalian clade, rooted at internal node `Node4` (see Fig. 2.5); this test will also include the long branch separating viral samples from birds and mammals. Codons $13, 15, 205, 277, 344$ should be found under selection. Lastly, use FEL to do a 'branch-site' type analysis on `Node4` (the separating branch). Codons $9, 11, 13, 15, 127, 176, 277, 458$ are found to be under selection. Run the SLAC analysis using http://www.datamonkey.org, and use the `Inferred Substitutions` link from the results page to visualize the evolutionary history of each of the selected codons (see Fig. 2.5 for two stylized examples).

2.6.5 REL methods in HyPhy

Like the previous batch file, `dNdSRateAnalysis.bf` (located in the `Codon Selection Analyses` rubrik) requires the user to import files containing the sequences and a tree, and to specify a genetic code. (Recall that dN and dS are alternative notations for β and α, respectively.) In addition, it prompts the user to specify the following options (for complete details, see [37]):

(i) `Branch Lengths` — Re-estimate branch lengths jointly with the codon rate parameters (`Codon Model`); or use the values proportional to those estimated in the nucleotide model before optimizing the codon model (`Nucleotide Model`), taking advantage of the strong correlation between nucleotide and codon branch lengths. For large datasets (e.g > 25 sequences), the `Nucleotide Model` option can cut computational time by several orders of magnitude.

(ii) `Rate Matrix Options` — Select one of several standard models of codon substitution rates to estimate from the data. `MG94Multi` permits multiple classes of β, adjusted according to one of several amino acid similarity matrices (selected at a later stage, under the heading `Amino Acid Class Model`).

(iii) `Rate Variation Options` — Evaluate all available models of rate variation (`Run All`), or an arbitrary selection from a subsequent list of models (`Run Custom`).

(iv) `Rate Variation Models` (Run Custom) — `Constant` assumes that neither α nor β vary across sites (simple global mean model); `Proportional` allows α and β to vary, but constrains β/α to be constant, assuming that the variability in substitution rates can be well explained by local mutation rates (this model is not expected to fit well); `Nonsynonymous` constrains $\alpha = 1$ while allowing β to vary, making it very similar to the models implemented in PAML; `Dual` draws values of β and α from independent or correlated distributions. You can hold 'shift' while clicking to select more than one option, so that multiple models will be evaluated in sequence. To run REL, select `Dual`. In addition, to test for evidence of synonymous rate variation, select `Non-synonymous` (together with `Dual`. A likelihood ratio test (section 1.6.5) will then be used to derive a p-value to decide whether or not α is variable in a given data set.

(v) `Distribution Option` — Values of β and/or α are drawn from independent gamma distributions (`Syn:Gamma, Non-syn:Gamma`); β is instead drawn from a mixed gamma distribution with a class for invariant sites (`Syn:Gamma, Non-syn:Inv+Gamma`); β and α are drawn from independent general discrete distributions (GDDs†, `Independent Discrete`), correlated GDDs (`Correlated Discrete`), or constrained GDDs such that $\beta \leq \alpha$ (`Non-positive Discrete`).

A good default setting is to use independent GDD distributions with 3 bins each (or 2 bins each for very small, or low divergence data sets). One can try several runs with various numbers of bins, staring with 2 each, and then increasing the number of bins by 1 (β followed by α) until the AIC score for the `Dual` model no longer decreases.

(vi) `Initial Value Options` — Use default or randomized initial values for

† GDDs are useful non-parametric distributions that make a minimal number of assumptions about the true underlying distribution of rate variation across sites.

parameters of the rate distributions. Repeated runs that make use of randomized initial values can be used to verify that the maximum likelihood found by HyPhy is indeed a global maximum.

The batch file will prompt you to specify a file (call it `ResultFile`) for saving the analysis results. The analysis will generate two summary files: `ResultFile` containing a copy of console output and `ResultFile.distributions` listing maximum likelihood estimates for $\alpha, \beta, \beta/\alpha$ rate distributions inferred by each model. In addition, for each rate variation model a `ResultFile.model` (likelihood function fit with that model) and a `ResultFile.model.marginals` (a file containing rate distributions conditional probabilities of each codon being in a given rate class) will be generated. The `.marginals` can be used as input to `dNdSResultProcessor.bf` (located in the `Codon Selection Analyses` rubrik) - a template file which can be used, in particular, to compute Bayes Factors (or posterior probabilities) of a site being under positive selection.

Exercise. Conduct a REL analysis on a sample of 35 randomly chosen Influenza/A/H3(HA) sequences (file `InfluenzaA_H3_Random35.nex`) using nucleotide-based branch lengths, MG94×CUSTOM with (012212) corection model, Nonsynonymous and Dual rate variation model, GDD distributions with 3 classes for α and 3 for β. Sample console output is shown below.

```
RUNNING MG94x012212 MODEL COMPARISONS on /home/sergei/Latest/HYPHY_Source/data/InfluenzaA_H3_Random35.nex


########## 3x3 CLASSES ##########

+--------------------+--------------------+---------------+--------------------+--------------------+---------------+-----+-----------+
|       Model        |   Log Likelihood   | Synonymous CV |  NS Exp and CV     |  N/S Exp and CV    |    P-Value    | Prm |    AIC    |
+--------------------+--------------------+---------------+--------------------+--------------------+---------------+-----+-----------+
| Var. N.Syn. Rates  |        -3053.54189 |      N/A      |  0.50446, 1.76495  |  0.50446, 1.76495  |      N/A      |  74 |   6255.08 |
+--------------------+--------------------+---------------+--------------------+--------------------+---------------+-----+-----------+
| Dual Variable Rates|        -3044.93374 |   1.04164118  |  0.49665, 1.81033  |  1.02586, 2.08251  |   0.00175455  |  78 |   6245.87 |
+--------------------+--------------------+---------------+--------------------+--------------------+---------------+-----+-----------+
```

A large CV (coefficient of variation, defined as mean/standard deviation), for synonymous rates α and a low (0.002) p-value for the test that $CV(\alpha) = 0$ indicate that synonymous rates vary from codon to codon in this alignment. Use `dNdSResultProcessor.bf` analysis to find positively selected sites under both models, based a Bayes Factor of 50, locating the appropriate `.marginals` file written by the REL analysis when prompted. The list from the Dual model should contain 13 codons (in particular, codons $135, 145, 194, 226$ and 275, previously identified by SLAC on the complete alignment), and 20 codons under Nonsynonymous model. Additional exercises might include: (i) investigating the effect of the number of rate classes on model fit and inference of sites under selection; (ii) identifying the list of sites which are found to be under selection by the Nonsynonymous model, but not the Dual model and vice versa; (iii) generating various plots (*e.g.* α, β by codon position) using `dNdSResultProcessor.bf`.

2.7 Estimating gene-by-gene variation in rates

Ultimately, the estimation of selection pressures on gene sequences is a comparative study, with the implicit objective of finding divergent patterns among genes of branch-by-branch or site-by-site variation in selection. For instance, copies of the same gene may experience strong positive selection at a codon position in one population, but negligible levels of selection in a second population. Drastic changes can easily be caused by environmental differences between populations. Different genes will also undergo different patterns of selection. However, it is more difficult to compare variation in selection at this level without having an overall sequence configuration in common between the two groups. We will discuss how this issue can be resolved through the application of data-driven model inference.

2.7.1 Comparing selection in different populations

Divergence in the site-by-site patterns of selection can be evaluated in *HyPhy* using batch files `CompareSelectivePressure.bf` and `CompareSelectivePressureIVL.bf` , under the heading `Positive Selection` in the `Standard Analyses` menu. These batch files are set up in much the same way as the preceding examples, except that it requires two sets of files containing sequences and trees from different populations. The former analysis tests the hypothesis that at a given site, β/α differs between the two samples along the entire tree, whereas the latter concerns itself only with *i*nterior tree branches. Please note, that both alignments must encode homologous sequences (*i.e.* two samples of the same gene from different populations or environments), and they must be aligned in the same codon coordinates, otherwise the comparisons are meaningless. Finally, large samples (at least 50 sequences in each sample) are required to gain power to discriminate differential evolution.

The output of these analyses will be displayed in a chart (GUI versions) and written to a comma separated file. `CompareSelectivePressure.bf` will report $dS, dN, dN/dS$ estimates for both samples and those under the null model (joint, *i.e.* when dN/dS is shared between two samples), the LR statistic and the asymptotic p-value. `CompareSelectivePressureIVL.bf` will print the estimates of $dS, dN_{leaves}, dN_{internal}$ for each sample, and the estimates under the shared $dN_{internal}/dS$ model, the LR statistic and the asymptotic p-value.

Neither analysis prompts for a specific significance level; sites of interest can be culled after the run is complete by selecting all those sites for which p is less than a desired value.

Compare selective pressures on Influenza neuraminidase gene from two different serotypes: H5N1 ('avian' flu) and H1N1 (the same serotype of as the 1918 pandemic). Even though the antigenic properties of neuaraminidase are similar, as it belongs to the N1 subtype in both samples, the evolutionary environments are different, because of the effect of other viral genes, and different host environments. Execute `Standard Analyses>Positive Selection>CompareSelectivePressure.bf` on `H1N1_NA.nex` - an alignment of 186 H1N1 sequences and 96 `H5N1_NA.nex` - an alignment of 186 H1N1 sequences 96 H5N1 sequences. Select the `(012345)` model for each alignment. The analysis takes can take a few hours on a desktop or about 15 minutes on a small computer cluster. 9 sites are evolving differentially at $p \leq 0.01$: $3, 28, 44, 74, 135, 257, 259, 412$ and 429 and in all 9 cases, stronger positive selection is indicated for the H5N1 sample.

A complementary method that can be used to compare `distributions` of rates, including proportions of sites under selection (but not their location) and the strength of selection is outlined in section 2.7.2.

2.7.2 Comparing selection between different genes

As we noted in the section 1.4.2, comparing the means of β/α rate distributions between two datasets to determine whether or not they are under similar selective pressures can be misleading. `dNdSDistributionComparison.bf` (under the `Codon Selection Analyses` rubrik in `Standard Analyses`) implements a more general procedure which fits an independent 4 bin distribution of (α_s^i, β_s^i) (with bin weights p_s^i) to data set i, with two negatively selected ($\alpha > \beta$) bins, one neutral ($\alpha = \beta$) bin and one positively selected ($\alpha < \beta$) bin (the 'Independent' model). To test for differences between the distributions, four simpler models are also fitted: (a) same selection strength model with $\beta_4^1/\alpha_4^1 = \beta_4^2/\alpha_4^2$ ('SharedStrength' model) ; (b) same selected proportion of sites: $p_4^1 = p_4^2$ ('SharedProportion' model); (c) same selective regime: (a) and (b) combined ('SharedPositiveSelection' model); (d) same distributions - all distribution parameters are shared between the data sets ('JointAll' model).

The analysis prompts for two datasets, a nucleotide bias model for each. The user can select whether relative branch lengths should be approximated with a nucleotide model, or estimated directly. The former option greatly accelerates convergence and, in most cases, has very little effect on the estimates of rates. Lastly, starting parameter values can be set to default values, or chosen randomly. Optimizing parameter rich distributions can be numerically unstable, and to alleviate convergence problems, `dNdSDistributionComparison.bf` incorporates a series of steps to ensure reliable convergence to a good maximum.

At the end of the run, *HyPhy* will report the results of 4 tests discussed above, write 5 files with models fits, which can be examined and reused later, print a summary of model fits to the console and echo it to a file.

Exercise. Compare selective pressures on two 16 sequence simulated datasets `Sim1.nex` and `Sim2.nex`. The alignments each consist of 500 codons and were generated with different rate profiles.

	Sim1.nex				Sim2.nex		
dN/dS	dS	dN	Prob	dN/dS	dS	dN	Prob
0.5	1	0.5	0.4	0.5	1	0.5	0.3
0.1	0.5	0.05	0.2	0.2	0.5	0.1	0.3
1.0	1.5	1.5	0.2	1.0	1.5	1.5	0.3
4.0	1	4.0	0.2	10	1	10	0.1

Execute `dNdSDistributionComparison.bf` on files `Sim1.nex` and `Sim2.nex`, specifying `Nucleotide` branch lengths, `(010010)` models for both data sets, and `Default` starting values. The analysis takes about an hour to run on a desktop. The inferred distributions under the independent rates model are listed below:

```
Inferred rates for data set 1:
   dN/dS      dS      dN      Prob
   3.167   1.005   3.184     0.260
   1.000   0.038   0.038     0.003
   0.133   0.758   0.101     0.326
   0.684   1.195   0.818     0.411
Inferred rates for data set 2:
   dN/dS      dS      dN      Prob
  11.947   0.714   8.531     0.104
   1.000   0.000   0.000     0.019
   0.252   0.750   0.189     0.470
   0.847   1.410   1.193     0.407
```

Note that because of the small sample size, inferred distributions are not quite the same as those used for data generation. However, all 4 tests correctly report that the distributions are different, in general, and in the positive selection regime. We also note that because of the complexity of the models being fitted, it may be advisable to run the analysis several times (using both Default and Random starting values) to verify convergence.

```
Are the distributions different?
        LR =     89.843 DF = 10 p =    0.000
Are selective regimes (dN/dS and proportions) different?
        LR =     25.421 DF = 2 p =    0.000
Are selection strengths (dN/dS) different?
        LR =     15.554 DF = 1 p =    0.000
Are the proportions of codons under selection different?
        LR =     18.457 DF = 1 p =    0.000
```

2.8 Automating choices for *HyPhy* analyses

Interactive, dialog driven analyses are useful for learning, exploring new options and running analyses infrequently. However, if a large number of sequence files must be subjected to the same analysis flow, then a mechanism to automate making the same choices over and over again is desirable.

HyPhy provides a mechanism for scripting any standard analysis using *input redirection*. To instruct *HyPhy* to make any given set of selections automatically, one must first execute the standard analysis for which the selections must be made and make notes of the actual choices being made. For instance, to use the standard analysis `AnalyzeCodonData.bf` with a local $MG94 \times 012232$ substitution model, 6 choices must be made: genetic code to use (Universal), alignment file to import, substitution model

(MG94CUSTOM), model options (Local), nucleotide bias (012232) and the tree to use. Having made these choices, one next creates a text file with a script in the HyPhy batch language which may look like this (assuming that the tree included in the file is to be used for step 6).

```
inputRedirect = {};
inputRedirect[''01'']=''Universal'';
inputRedirect[''02'']=''/Users/sergei/Desktop/MyFiles/somealignment.nex'';
inputRedirect[''03'']=''MG94CUSTOM'';
inputRedirect[''04'']=''Local'';
inputRedirect[''05'']=''012232'';
inputRedirect[''06'']=''y';
ExecuteAFile (HYPHY_BASE_DIRECTORY + ''TemplateBatchFiles''+
DIRECTORY_SEPARATOR+''AnalyzeCodonData.bf'', inputRedirect);
```

`inputRedirect` is a data structure (an associative array) which stores a succession of inputs, indexed by the order in which they will be used, and the `ExecuteAFile` command executes the needed standard analysis using some predefined variables to specify the path to that file, using `inputRedirect` to fetch user responses from. All standard analyses reside in the same directory, so this command can be easily adjusted for other analyses. The input for step "02" must, of course, refer to an existing file. Another option is to leave that option blank (""), and have *HyPhy* prompt just for the file, keeping other options as specified. To execute a file like this, invoke `File> Open> Open Batch File`.

2.9 Simulations

HyPhy has versatile data simulation capabilities. In particular, any combination of site and branch specific α_s^b, β_s^b can be employed to generate codon data.

There are two primary ways for specifying a model to simulate under. If a likelihood function has been defined and optimized then it can be simulated from with a single command. GUI based analyses should use `Data>Simulate` menu from the data panel. Likelihood functions defined otherwise, *e.g.* via a standard analysis, can be simulated from using the `SimulateFromLF` tool accessible via the `User Actions` button in the console window.

Often times, however, it may be easier to specify all the model parameters, such as base frequencies, branch lengths and selection profiles and simulate replicates from those parameters. We have written a customizable script for simulating both site-by-site rate variation and branch-and-site rate variation, available as *HyPhy* scripts from `http://www.hyphy.org/pubs/dNdS_`

`Simulator.tgz`. Instructions on how to use and modify simulation scripts are included in configuration files.

2.10 Summary of standard analyses

Here we list all *HyPhy* standard analyses which can be used to conduct codon-based rate estimates and test selection hypotheses. Please note that the collection of *HyPhy* analyses is always growing, so new analyses may have been added since this chapter was written.

Analysis	Primary Use	Ref.
Miscellaneous Phylohandbook.bf	Run example/tutorial analyses referenced in the theory section.	
Basic Analyses AnalyzeCodonData.bf	Estimate mean ω for an alignment; fit a local model where each branch has a separate omega.	[3, 2, 24]
Codon Selection Analyses dNdSRateAnalysis.bf	REL site-by-site selection. Test for synonymous rate variation. Test for global evidence of positive selection ($\beta > \alpha$) when synonymous rates are variable.	[28, 37, 33]
Codon Selection Analyses dNdSResultProcessor.bf	Process and visualize results generated with dNdSRateAnalysis.bf	[37]
Codon Selection Analyses dNdSDistributionComparison.bf	Fit a series of REL models to two alignments and compare whether or not the distribution of rates differ between the alignments. Also compares the proportion of sites under and/or the strength of positive selection	
Compartmentalization SelectionLRT.bf	Splits the tree into a clade of interest and the rest of the tree. Compare mean ω between the clade, the rest of the tree, and the branch separating the two and test for significant differences.	[71]
Miscellaneous SlidingWindowAnalysis.bf	Fit a global (or local) model to a series of sliding windows. This is an obsolete distribution-free way to estimate spatial rate variation, which has been superseded by FEL to a large extent.	
Miscellaneous MGvsGY.bf	Investigate whether Muse-Gaut or Goldman-Yang codon model parameterization fits the data better.	[3, 2]
Miscellaneous NucModelCompare.bf	Select the best fitting nucleotide model which is an input option to most other analyses.	[73]
Molecular Clock Multiple files	Carry out a molecular clock (or dated molecular clock) test using a codon model. The clock can be imposed on branch lengths (global models), or only on synonymous or non-synonymous distances (local models).	
Positive Selection TestBranchDNDS.bf	Test whether selected branches (foreground) have different ω when compared to the rest of the tree (uniform background). Site-to-site rate variation can also be handled.	[24]
Positive Selection CompareSelectivePressure.bf CompareSelectivePressureIVL.bf	Use FEL to test for differential selection at a site between two samples of the same gene. The entire tree, or only internal branches can be tested.	[38]

Positive Selection SubtreeSelectionComparison.bf	Use FEL to check whether β/α differ significantly between a user-specified clade and the rest of the tree. Can also test internal versus terminal branches.	
Positive Selection NielsenYang.bf	Use a series of REL models (with constant α) to test for selection. This analysis (with the GY94 model) is identical to PAML analyses, *e.g.* M8a v M8b tests.	[74, 32]
Positive Selection QuickSelectionDetection.bf	SLAC (and other counting based methods) and FEL site-by-site analyses. FEL also supports the analysis of a part of the tree, enabling branch-site type analyses.	[28]
Positive Selection YangNielsenBranchSite2005.bf	Use the improved branch-site REL method of Yang and Nielsen (2005) to look for episodic selection in sequences.	[40]
Recombination SingleBreakpointRecomb.bf	Screen an alignment for evidence of recombination using the single breakpoint method	[11]
`http://www.datamonkey.org`	A web interface to run model selection, SLAC, FEL and REL analyses on our cluster.	[28]
`http://www.datamonkey.org/GARD`	A web interface to run recombination detection tools on our cluster.	[11]
`http://www.hyphy.org/gabranch`	Run a genetic algorithm to find good fitting models of temporal (branch-to-branch) variation in selection pressure	[27]

2.11 Discussion

While we have made every attempt to be thorough, this practice section only touches on what can be done in terms of selection analyses. HyPhy and Datamonkey have active user communities, and specific questions, can be posted on our bulletin board (`http://www.hyphy.org/cgi-bin/yabb/yabb.pl`). We make every effort to address these questions in a timely and informative manner. We also encourage users to develop their own batch files to implement specific analyses, and share them with the rest of the community. As further funding is secured, we hope to expand the capabilities of Datamonkey in order to offer a wider range of computationally-intensive analyses to the research community.

Bibliography

R. Nielsen and Z. H. Yang. Likelihood models for detecting positively selected amino acid sites and applications to the HIV-1 envelope gene. *Genetics*, 148:929–936, 1998.

S. V. Muse and B. S. Gaut. A likelihood approach for comparing synonymous and nonsynonymous nucleotide substitution rates, with application to the chloroplast genome. *Mol Biol Evol*, 11:715–724, 1994.

N. Goldman and Z. Yang. A codon-based model of nucleotide substitution for protein-coding DNA sequences. *Mol Biol Evol*, 11(5):725–736, Sep 1994.

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

Yun-Huei Tzeng, Runsun Pan, and Wen-Hsiung Li. Comparison of three methods for estimating rates of synonymous and nonsynonymous nucleotide substitutions. *Mol Biol Evol*, 21(12):2290–2298, 2004.

SV Muse. Estimating synonymous and nonsynonymous substitution rates. *Mol Biol Evol*, 13(1):105–114, 1996.

Simon D. W. Frost, Monique Nijhuis, Rob Schuurman, Charles A. B. Boucher, and Andrew J. Leigh Brown. Evolution of lamivudine resistance in human immunodeficiency virus type 1-infected individuals: the relative roles of drift and selection. *J. Virol.*, 74(14):6262–6268, 2000.

Corey B. Moore, Mina John, Ian R. James, Frank T. Christiansen, Campbell S. Witt, and Simon A. Mallal. Evidence of HIV-1 Adaptation to HLA-Restricted Immune Responses at a Population Level. *Science*, 296(5572):1439–1443, 2002.

N Saitou and M Nei. The neighbor-joining method: a new method for reconstructing phylogenetic trees. *Mol Biol Evol*, 4(4):406–25, Jul 1987.

J. Felsenstein. Evolutionary trees from DNA-sequences – a maximum-likelihood approach. *J Mol Evol*, 17:368–376, 1981.

S.L. Kosakovsky Pond, D Posada, M.B. Gravenor, C.H. Woelk, and S. D. W. Frost. Automated phylogenetic detection of recombination using a genetic algorithm. *Mol Biol Evol*, 23(10):1891–1901, 2006.

S. Whelan and N. Goldman. Estimating the frequency of events that cause multiple-nucleotide changes. *Genetics*, 167:2027–2043, 2004.

Z. H. Yang. PAML: a program package for phylogenetic analysis by maximum likelihood. *Computer Applications In The Biosciences*, 13:555–556, 1997.

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

B. Efron, E. Halloran, and S. Holmes. Bootstrap confidence levels for phylogenetic trees. *Proc Natl Acad Sci U S A*, 93(23):13429–13434, Nov 1996.

N Goldman. Statistical tests of models of DNA substitution. *J Mol Evol*, 36(2):182–98, Feb 1993.

M. Nei and T. Gojobori. Simple methods for estimating the numbers of synonymous and nonsynonymous nucleotide substitutions. *Mol Biol Evol*, 3:418–426, 1986.

A.W.F. Edwards. *Likelihood*. Cambridge University Press, 1972.

J. S. Rogers. On the consistency of maximum likelihood estimation of phylogenetic trees from nucleotide sequences. *Syst Biol*, 46(2):354–357, Jun 1997.

S. V. Muse. Modeling the molecular evolution of HIV sequences. In K. A. Crandall, editor, *The Evolution of HIV*, chapter 4, pages 122–152. The Johns Hopkins University Press, 1999.

John P. Huelsenbeck, Bret Larget, and Michael E. Alfaro. Bayesian phylogenetic model selection using reversible jump Markov chain Monte Carlo. *Mol Biol Evol*, 21(6):1123–1133, 2004.

D Posada and KA Crandall. Modeltest: testing the model of dna substitution. *Bioinformatics*, 14(9):817–818, 1998.

Y. Suzuki and M. Nei. False-positive selection identified by ML-based methods: Examples from the *s*ig1 gene of the diatom *t*halassiosira weissflogii and the *t*ax gene of a human T-cell lymphotropic virus. *Molecular Biology And Evolution*,

21:914–921, 2004.

Z. Yang. Likelihood ratio tests for detecting positive selection and application to primate lysozyme evolution. *Mol Biol Evol*, 15:568–573, 1998.

S.G. Self and K-Y. Liang. Asymptotic properties of maximum likelihood estimators and likelihood ratio tests under nonstandard conditions. *J Am Stat Assoc*, 82(398):605–610, 1987.

Y. Benjamini and Hochberg Y. Controlling the false discovery rate: a practical and powerful approach to multiple testing. *J Royal Stat Soc B*, 57(1):289–300, 1995.

Sergei L Kosakovsky Pond and Simon D W Frost. A genetic algorithm approach to detecting lineage-specific variation in selection pressure. *Mol Biol Evol*, 22(3):478–485, Mar 2005.

S.L. Kosakovsky Pond and S.D.W. Frost. Not so different after all: a comparison of methods for detecting amino-acid sites under selection. *Mol Biol Evol*, 22:1208–1222, 2005.

John P Huelsenbeck and Kelly A Dyer. Bayesian estimation of positively selected sites. *J Mol Evol*, 58(6):661–672, Jun 2004.

John P Huelsenbeck, Sonia Jain, Simon W D Frost, and Sergei L Kosakovsky Pond. A Dirichlet process model for detecting positive selection in protein-coding DNA sequences. *Proc Natl Acad Sci U S A*, 103(16):6263–6268, Apr 2006.

Y. Suzuki and T. Gojobori. A method for detecting positive selection at single amino acid sites. *Mol Biol Evol*, 16:1315–1328, 1999.

Willie J. Swanson, Rasmus Nielsen, and Qiaofeng Yang. Pervasive adaptive evolution in mammalian fertilization proteins. *Mol Biol Evol*, 20(1):18–20, 2003.

Ulf Sorhannus and Sergei L. Kosakovsky Pond. Evidence for positive selection on a sexual reproduction gene in the diatom genus Thalassiosira (Bacillariophyta). *J Mol Evol*, V63:231–239, 2006.

Ziheng Yang, Wendy S.W. Wong, and Rasmus Nielsen. Bayes Empirical Bayes inference of amino acid sites under positive selection. *Mol Biol Evol*, 22(4):1107–1118, 2005.

J. V. Chamary, Joanna L. Parmley, and Laurence D. Hurst. Hearing silence: non-neutral evolution at synonymous sites in mammals. *Nat Rev Genet*, 7:98–108, 2006.

A. Tuplin, J. Wood, D.J. Evans, A.H. Patel, and P. Simmonds. Thermodynamic and phylogenetic prediction of RNA secondary structures in the coding region of hepatitis C virus. *RNA*, 8:824–841, 2002.

S. L. Kosakovsky Pond and S. V. Muse. Site-to-site variation of synonymous substitution rates. *Mol Biol Evol*, 22(12):2375–2385, Dec 2005.

Sergei L. Kosakovsky Pond, Simon D. W. Frost, Zehava Grossman, Michael B. Gravenor, Douglas D. Richman, and A. J. Leigh Brown. Adaptation to different human populations by HIV-1 revealed by codon-based analyses. *PLoS Comp Biol*, 2, 2006.

Z. Yang and R. Nielsen. Codon-substitution models for detecting molecular adaptation at individual sites along specific lineages. *Mol Biol Evol*, 19(6):908–17, 2002.

Jianzhi Zhang, Rasmus Nielsen, and Ziheng Yang. Evaluation of an improved branch-site likelihood method for detecting positive selection at the molecular level. *Mol Biol Evol*, 22(12):2472–2479, 2005.

Julien Dutheil, Tal Pupko, Alain Jean-Marie, and Nicolas Galtier. A model-based

approach for detecting coevolving positions in a molecule. *Mol Biol Evol*, 22(9):1919–1928, Sep 2005.

Stéphane Guindon, Allen G Rodrigo, Kelly A Dyer, and John P Huelsenbeck. Modeling the site-specific variation of selection patterns along lineages. *Proc Natl Acad Sci USA*, 101(35):12957–62, Aug 2004.

Z. Yang. A space-time process model for the evolution of dna sequences. *Genetics*, 139(2):993–1005, Feb 1995.

J. Felsenstein and G. A. Churchill. A Hidden Markov model approach to variation among sites in rate of evolution. *Mol Biol Evol*, 13(1):93–104, Jan 1996.

Adi Stern and Tal Pupko. An evolutionary space-time model with varying among-site dependencies. *Mol Biol Evol*, 23(2):392–400, Feb 2006.

Douglas M Robinson, David T Jones, Hirohisa Kishino, Nick Goldman, and Jeffrey L Thorne. Protein evolution with dependence among codons due to tertiary structure. *Mol Biol Evol*, 20(10):1692–1704, Oct 2003.

Nicolas Rodrigue, Nicolas Lartillot, David Bryant, and HervÈ Philippe. Site interdependence attributed to tertiary structure in amino acid sequence evolution. *Gene*, 347(2):207–217, Mar 2005.

Y. Suzuki, T. Gojobori, and M. Nei. ADAPTSITE: detecting natural selection at single amino acid sites. *Bioinformatics*, 17:660–661, 2001.

S Kumar, K Tamura, and M Nei. Mega: Molecular evolutionary genetics analysis software for microcomputers. *Bioinformatics*, 10(2):189–191, 1994.

Sergei L Kosakovsky Pond, Simon D W Frost, and Spencer V Muse. HyPhy: Hypothesis testing using phylogenies. *Bioinformatics*, 21(5):676–9, Mar 2005.

Robin M. Bush, Catherine A. Bender, Kanta Subbarao, Nancy J. Cox, and Walter M. Fitch. Predicting the evolution of human Influenza A. *Science*, 286(5446):1921–1925, 1999.

R. G. Webster, W. J. Bean, O. T. Gorman, T. M. Chambers, and Y. Kawaoka. Evolution and ecology of influenza A viruses. *Microbiol Rev*, 56:152–179, Mar 1992.

William W. Thompson, David K. Shay, Eric Weintraub, Lynnette Brammer, Nancy Cox, Larry J. Anderson, and Keiji Fukuda. Mortality associated with influenza and respiratory syncytial virus in the united states. *JAMA*, 289:179–186, Jan 2003.

RG Webster, WG Laver, GM Air, and GC Schild. Molecular mechanisms of variation in influenza viruses. *Nature*, 296(5853):115–121, 1982.

J Felsenstein. Phylip - phylogeny inference package (version 3.2). *Cladistics*, 5:164–166, 1989.

DL Swofford. *PAUP* Phylogenetic Analysis Using Parsimony (*and Other Methods)*. Sinauer Associates, Sunderland, Massachusetts, 2003.

RR Sokal and CD Michener. A statistical method for evaluating systematic relationships. *Univ. of Kans. Sci. Bull.*, 38:1409–1438, 1958.

K Tamura and M Nei. Estimation of the number of nucleotide substitutions in the control region of mitochondrial DNA in humans and chimpanzees. *Mol Biol Evol*, 10:512–526, 1993.

C. Lanave, G. Preparata, C. Saccone, and G. Serio. A new method for calculating evolutionary substitution rates. *J Mol Evol*, 20:86–93, 1984.

M. Hasegawa, H. Kishino, and T. A. Yano. Dating of the human ape splitting by a molecular clock of mitochondrial-DNA. *J Mol Evol*, 22:160–174, 1985.

S. V. Muse. Modeling the molecular evolution of HIV sequences. In K. A. Crandall, editor, *The Evolution of HIV*, chapter 4, pages 122–152. The Johns Hopkins

University Press, 1999.

D. Posada and K. A. Crandall. The effect of recombination on the accuracy of phylogeny estimation. *J Mol Evol*, 54:396–402, 2002.

M. H. Schierup and J. Hein. Consequences of recombination on traditional phylogenetic analysis. *Genetics*, 156:879–891, 2000.

M. Anisimova, R. Nielsen, and Z. H. Yang. Effect of recombination on the accuracy of the likelihood method for detecting positive selection at amino acid sites. *Genetics*, 164:1229–1236, 2003.

D. Shriner, D. C. Nickle, M. A. Jensen, and J. I. Mullins. Potential impact of recombination on sitewise approaches for detecting positive natural selection. *Genet Res*, 81:115–121, 2003.

H. Akaike. A new look at the statistical model identification. *IEEE Trans Aut Control*, 119:716–723, 1974.

M Hasegawa and H Kishino. Confidence limits on the maximum-likelihood estimate of the hominid tree from mitochondrial-dna sequences. *Evolution*, 43(3):672–677, 1989.

S.L. Kosakovsky Pond and S. V. Muse. Column sorting: Rapid calculation of the phylogenetic likelihood function. *Syst Biol*, 53(5):685–692, 2004.

JH Gillespie. The molecular clock may be an episodic clock. *Proc. Natl. Acad. Sci. USA*, 81(24):8009–8013, 1984.

W Messier and C-B Stewart. Episodic adaptive evolution of primate lysozymes. *Nature*, 385:151–154, 1997.

Simon D W Frost, Yang Liu, Sergei L Kosakovsky Pond, Colombe Chappey, Terri Wrin, Christos J Petropoulos, Susan J Little, and Douglas D Richman. Characterization of human immunodeficiency virus type 1 (HIV-1) envelope variation and neutralizing antibody responses during transmission of HIV-1 subtype B. *J Virol*, 79(10):6523–6527, May 2005.

Z. H. Yang. Maximum likelihood estimation on large phylogenies and analysis of adaptive evolution in human influenza virus A. *J Mol Evol*, 51:423–432, 2000.

S.L. Kosakovsky Pond and S.D.W. Frost. A simple hierarchical approach to modeling distributions of substitution rates. *Mol Biol Evol*, 22:223–234, 2005.

Z. H. Yang, R. Nielsen, N. Goldman, and A. M. K. Pedersen. Codon-substitution models for heterogeneous selection pressure at amino acid sites. *Genetics*, 155:431–449, 2000.