UNIVERSITY OF CALIFORNIA

Santa Barbara

# Testing Trait Evolution Models on Phylogenetic Trees

A Thesis submitted in partial satisfaction of the

requirements for the degree Master of Science

in Computer Science

by

Chunghau Lee

Committee in charge:

Professor Ambuj Singh, Co-Chair

Professor Todd Oakley, Co-Chair

Professor Yuan-Fang Wang

March 2005

The thesis of Chunghau Lee is approved.

_____

Yuan-Fang Wang

_____

Todd Oakley, Committee Co-Chair

_____

Ambuj Singh, Committee Co-Chair

March 2005

ABSTRACT

Testing Trait Evolution Models on Phylogenetic Trees

by

Chunghau Lee

Tracing the ancestry of current species, whether they be whole organisms or features of organisms, by constructing phylogenetic trees may yield vital information surrounding the relationships between species. In addition, practical applications of phylogenetic trees include identifying related species to serve as new drug test targets. Since the construction of these trees are based on genetic similarity, the phylogeny of actual physical traits may not be well represented. However, by applying different models of trait evolution, one can test the evolution of observed data. The testing is provided by the novel tool CoMET, short for Continuous-character Model Evaluation and Testing, which ranks the results of these models using maximum likelihood. CoMET's approach includes maximizing discontinuous non-linear functions with an approximative algorithm and reducing exponentially-growing data structures by simplifying tree redundancies. Furthermore, simulation experiments provide suggestions for improvement by showing that one model variant is ineffective while some models requiring trees to have zero-length branches unnecessarily add complexity. A follow-up experiment then showed that for some models, user input can improve the usefulness of the CoMET tests. In conclusion, the end-product CoMET complements phylogenetic trees through model testing for continuously-varying character data, addressing the need to infer the phylogeny of non-genomic characteristics.

# Table of Contents

# Chapter 1: Background

## 1.1 Introduction

Phylogeny is the study of ancestral relationships between current, observed species. One example application of phylogenetic techniques is the well-known Tree of Life (http://tolweb.org). This project's ongoing purpose is to arrange all known organisms onto a tree showing all the ancestral relationships. One application of phylogenetics is the mapping of gene expression data (Shulze and Downward 2001) of an organism onto a tree to study the evolution of expression (Oakley, et al 2004, Gu 2004). With the construction of phylogenetic trees, researchers may gain insight that may be difficult to elucidate otherwise about the history and relationships between compared data.

This first chapter discusses the background behind phylogenetic trees and covers techniques measuring the quality of a hypothesis tree for given data. In addition, it also discusses the background on the main project of this thesis: CoMET, short for Continuous-character Model Evaluation and Testing. CoMET is a general-purpose tool that applies phylogenetic techniques and recently developed models by Dr. Todd Oakley to evaluate the phylogenetic history of observed data. CoMET calculates the maximum likelihood (ML) (Hulsenbeck and Crandall 1997) of the input data and hypothesis tree, and presents to the user the likelihoods of different trait evolutionary models. The second chapter covers in detail the design and implementation of CoMET. The third chapter discusses the simulation experiments on CoMET for comparing the different Oakley models. In addition to answering existing questions about the models, the results of the experiments also produced new insights on using these models.

## 1.2 Trees, Taxa, and Characters

A phylogenetic tree describes how its taxa are related ancestrally. Each taxon , located at the tips of the tree, represent the current, observed species. A rooted phylogenetic tree starts at a root node, representing the common ancestor of all taxa, and branches upward toward the taxa terminal nodes at the tree tips. Between the root and the tips may be several internal nodes representing ancestors of the taxa. For CoMET, only binary trees are used: binary in the sense that every parent node has two and only two daughter nodes. Therefore, for *n* taxa nodes, there are *n*-1 internal nodes including the root. Feature data belonging to the taxa are called characters, and they may be discrete values like DNA sequences, or, in the case of CoMET, continuously-varying values representing various states. Character data is collected for each taxon, and the whole character matrix may include several rows of characters for each, representing different states for different conditions.

## 1.3 Tree Evaluation

To measure the quality of a tree for a given set of taxa and character data, parsimony score and likelihood are two well known methods. Parsimony scores are generally related to trees built using the maximum parsimony (MP) method (Eck and Dayhoff 1966, Fitch 1971), in which trees are first built and the taxa are arranged at the tips. Then, the number of mutations required to change from daughter states to parent states for all branches are summed together into the MP score. Under MP, simpler hypotheses are preferred to more complicated ones, and therefore, the configurations with the lowest scores are deemed as the best trees.

Like MP, ML also looks for the best fit trees, but its statistical approach produces probabilities instead of scores. ML calculates the probability of data given a hypothesis tree and model by accruing the individual probabilities of all branching events according to a random

distribution. CoMET calculates likelihood based on the work of J. Felsenstein (Felsenstein 1981), who developed a system for calculating ML over continuous character data in his Restricted Evolution Maximum Likelihood (REML) algorithm. In this algorithm, the Brownian motion diffusion model is used to predict character states given a time from a parent state. The main advantages of this model are its simplicity and its applicability to a wide range of data. If the root state is zero, then the mean of all the taxa states is also zero because of how Brownian motion is defined. This, in turn, becomes a limitation because it restricts the overall evolutionary growth as non-directional.

In Felsenstein's algorithm, the overall likelihood can be expressed with the following equation, reflecting the underlying assumption of Brownian motion as the model for character evolution:

$$L = \prod_{ii'} \frac{1}{\sqrt{2\pi\beta v_{i'}}} \exp\left(\frac{-(x_i - x_{i'})^2}{2\beta v_{i'}}\right) \qquad \text{(Equation 1)}$$

In this formula, $i$ and $i'$ represent two nodes in the tree and $v_{i'}$ is the time distance, usually depicted as branch lengths, between these two nodes. Their states are represented by $x_i$ and $x_{i'}$ while the parameter $\beta$ represents the rate of change. With REML, Felsenstein simplified Equation 1 and introduced the recursive approach to compute likelihood.

The basic ML element in a tree is the *contrast*, which involves a parent node $p$ with two daughter branches $b_1$ and $b_2$, and two daughter states $s_1$ and $s_2$:

$$contrast_p = -0.5 * \ln(b_1 + b_2) - 0.5 * (s_1 - s_2)^2 / (b_1 + b_2) \qquad \text{(Equation 2)}$$

The contrast value is the natural log of the likelihood of this branching event. As an example, let the

parent node have two daughter nodes that are terminal nodes, which contain character data. If there are a total of three characters to examine, then the contrast operation would be calculated three times and summed, once for each character and using the states from just one character at a time. If a daughter node for a contrast is an internal node of the tree, then the state information for that internal node is inferred from its two daughters based on Brownian motion. The equation below shows how the state and branch length of $p$, a parent node, are calculated based on two daughter branches $b_1$ and $b_2$, and two daughter states $s_1$ and $s_2$:

$$s_p = ((s_1 / b_1) + (s_2 / b_2)) / (b_1^{-1} + b_2^{-1}) \qquad \text{(Equation 3a)}$$

$$b_p = b_p + (b_1 b_2) / (b_1 + b_2) \qquad \text{(Equation 4)}$$

In the case that the length of one of the daughter branches is zero, then Equation 3a would have a division-by-zero problem in either $(s_1 / b_1)$ or $(s_2 / b_2)$. But if the parent node $p$ and a daughter node $d$ node have no distance between them, then their states are assumed equal:

$$s_p = s_d \mid b_d = 0 \qquad \text{(Equation 3b)}$$

The reason branch lengths are adjusted in Equation 4 is to allow for proper state inferences later. The tree's actual branches are not changed, just the temporary representation. Again, given the nature of these equations, the overall likelihood of a tree is computed recursively. Contrast values are first collected from the tips of the tree and then the computation progresses downward toward the root, calculating branch length adjustments and inferring states along the way. At the end of the process, the contrasts from the internal nodes are summed together to form the likelihood.

## 1.4 Oakley's Nine Models of Trait Evolution

Using the ML equations, CoMET tests the tree and character data to see which of the nine Oakley's models of trait evolution fits best. These models represent a combination of different styles and characteristics in trait evolution, and are grouped into three general classes and three model types.

The three general classes are *phylogenetic*, *non-phylogenetic*, and *punctuated*. Models under *phylogenetic* assume that character traits are directly affected by speciation events: a split in the tree means a divergence in character states. In *non-phylogenetic* models, close genetic relatives are assumed to be just as similar as distant relatives when compared by phenotype. In *punctuated* models, phylogeny layout also correlates with the resulting character states, as in *phylogenetic* models, but has one key difference: at each branching event in the tree, one daughter node retains the same phenotype as the parent while the other daughter does not.

The three model types are *distance*, *equal*, and *free*. Models of the *distance* type assume that branch lengths represent the amount of divergence a daughter state has from its parent. The *equal* models assume equal divergence for all nodes by modeling all branches to have equal lengths. Lastly, the *free* models allow for independent and different rates of divergence by individually scaling the branch lengths.

Crossing three general classes with the three model types results in nine different models, each combining different models of phylogeny and divergence rates:

1. Phylogenetic / Distance
2. Phylogenetic / Equal
3. Phylogenetic / Free
4. Non-phylogenetic / Distance
5. Non-phylogenetic / Equal
6. Non-phylogenetic / Free
7. Punctuated / Distance
8. Punctuated / Equal
9. Punctuated / Free

Again, using these models, CoMET examines the input data to see which evolutionary model fits the data with the greatest likelihood.

## 1.5 The Akaike Information Criterion

The degrees of freedom (DOF) involved in the nine models are dependent upon the number of parameters in the models. Consequently, when character data are tested against these models, the ML results alone do not factor in choosing the best model. By using the Akaike Information Criterion (AIC) (Akaike 1973) and defining the DOF for each model, the results of each model can be ranked. The general formula is as follows:

$$AIC = -2 \ln L + 2 d \qquad \text{(Equation 5)}$$

In this equation, $L$ is the likelihood, and $d$ is the DOF used in a particular model. The model with the lowest AIC is deemed the best fit model for the data. Values of $d$ used for the models are discussed in the CoMET chapter.

## 1.6 The Mesquite Project

CoMET is built upon the Mesquite platform,(Maddison and Maddison 2004) a pervasively modular graphical program written in Java. The CoMET module contains three packages accessible for Mesquite: the CoMET evaluator for character matrices, the CoMET evaluator for single characters, and the CoMET simulation program. Mesquite benefits CoMET by providing a graphical user interface; data structures for trees, taxa, and characters; an implementation of the Brownian motion model; and also scientific routines from the Phylogenetic Analysis Library (PAL) (Drummond and Strimmer 2001). The particular PAL classes used by CoMET are separately

packaged using the latest version from the official PAL website because CoMET requires monitoring features not found in Mesquite's PAL.

# Chapter 2: CoMET

## 2.1 CoMET Introduction

As mentioned earlier, CoMET's purpose is to determine the most likely model of trait evolution for a given set of continuously-varying data. Since the only limitation for the data is to be continuously-varying, CoMET can be used for problems of many types. For example, physical characteristics like body sizes (Mooers, et al 1999) and experimental data like microarray expression levels can all be used as input for CoMET. A hypothesis tree is also needed from the user, which can be constructed using the same data or a different data source. One example setup (Oakley, et al 2004) analyzed the phylogeny of a set of related genes by first constructing a tree using DNA comparison. The character data at the taxa tips contained the log-based-two of expression levels for each gene at different time points in different microarray experiments. This, in effect, tested how gene expression for this group of genes evolved.

## 2.2 Program Flow

Again, the input parameters for CoMET consist of a character data matrix and a hypothesis tree. CoMET begins by making copies of the hypothesis tree and altering these copies to represent the different models (Figure 1). Then, the character matrix is used in the likelihood calculations for these altered trees.

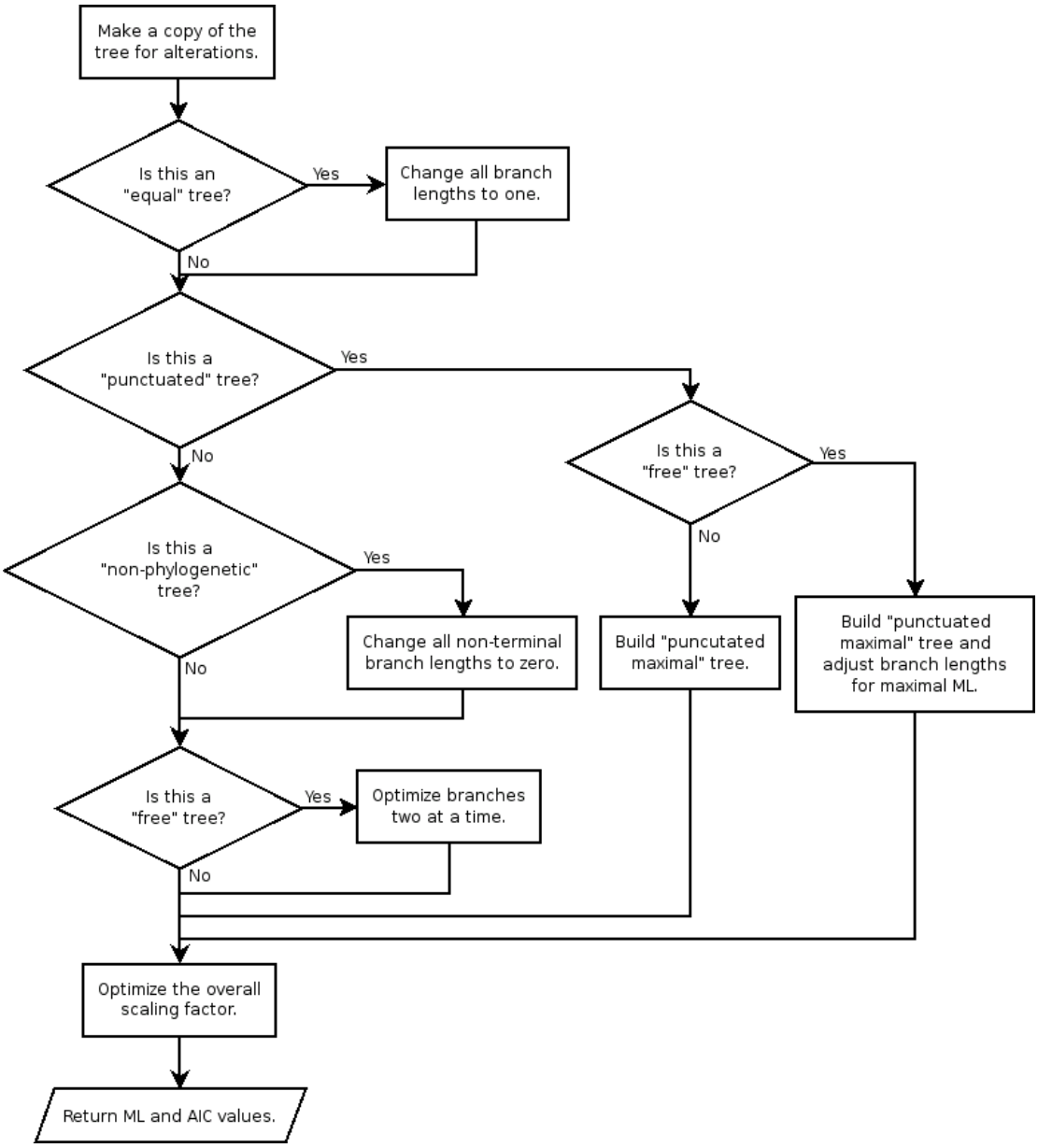



*Figure 1: CoMET program flow chart.*

How CoMET alters a tree depends on which two model combinations the tree represents. For the *distance* models, the branch lengths are left unchanged. For the *equal* models, all branch lengths are changed to one to represent equal rates of divergence. In the *free* models, each branch length is allowed to grow or shrink to achieve the greatest likelihood to represent varying divergence rates.

In addition to the changes according to model types, CoMET applies changes according to

the general class to which the tree is assigned. For *phylogenetic*, the branches are once again left unchanged. For *non-phylogenetic*, every branch not directly connected to the taxa nodes has its length reduced to zero, modeling the situation in which all taxa nodes are directly connected to the root and allowing for close relatives to be just as similar or dissimilar to each other as distant ones. For *punctuated*, CoMET presents two different approaches, a *punctuated average* and a *punctuated maximal* version. The finer details about tree alterations for the *punctuated* models are discussed in a later section. However, just as a brief description, CoMET models punctuated evolution by changing the branch length of one daughter branch at every internal parent node to zero. This effectively models one daughter state to remain the same as the parent state, while the state of the daughter with the non-zero branch length would diverge.

After the construction of these altered trees, likelihood can be calculated using character data. One additional parameter is optimized at this stage: $\beta$, from Equation 1, representing the rate of phenotypic change. This value is multiplied to all branch lengths of the tree to further maximize the likelihood. CoMET finds the optimal scaling factor through PAL's UnivariateMinimum class, which employs Brent's numerical method without calculating derivatives. CoMET implements an extension of the UnivariateFunction class to be used with UnivariateMinimum. This function takes in the scalar to use, scales all the branches of the tree using this value, and returns the likelihood of the resulting tree with the given character data. This process quickly finds the scalar yielding maximal results.

## 2.4 Degrees of Freedom for AIC Values

As mentioned earlier, the use of AIC over ML facilitates direct comparison between the results from these altered trees because AIC values account for the different DOF's used by the models. For most models, the DOF $d$ is 1 because the only parameter is the final scaling factor.

The *free* models have different values for $d$ because each branch length may have a parameter. In the case of *phylogenetic / free*, $d$ equals the total number of branches, which for a tree with $n$ taxa is $2n - 2$. For *non-phylogenetic / free*, there are only as many scalable branches as there are taxa nodes, so this model's DOF is $n$. For the case of *punctuated / free*, only half of the branches remain non-zero and are thus parameters; so for this model, $d = n - 1$. After adjusting ML into AIC values, CoMET outputs the results for each model to the user. The model with the lowest AIC is the best fit for the data.

## 2.3 Free Models

Trees under the *free* model type have each branch length adjusted toward greater overall likelihood. The simplest way to achieve this is to provide the basis tree and character data, and let one of PAL's MultivariateMinimum classes optimize every branch. This idea was tried but abandoned for two reasons. One, the running time is long and does not scale well as the number of taxa increases. The second reason is more difficult to ignore: the function to minimize is not continuous, which in turn causes PAL's methods to fail to produce useful results. The reason why the function is not continuous is because branch lengths must be allowed to scale to as low as zero. However, only one daughter branch is allowed to do so at every node due to the division-by-zero in Equation 4. In addition, the state of the parent node cannot be determined since both daughter states are of a distance zero away, there is no way to choose one state over another. PAL's methods do not know of this limitation and will test combinations that result in these situations, which are called *zero forks*. When the ML calculator comes to evaluate the choices made by the optimizer and encounters these *zero forks*, it returns a low likelihood value to discourage the choice. When multiple variables are involved, these spikes confuse the optimizer, which surrenders with poor results.

CoMET's solution is a greedy algorithm that recursively optimizes two sister branches at a time, resulting in a maximal value as the ML. With just two variables, PAL's conjugate direction search has no problems dealing with the function spike in the *zero fork* situation and can quickly reach an answer. Even though only two branches at a time are optimized, the quality metric for the choices made by the conjugate direction search is the ML over the entire tree and not just the single contrast value of the parent node involved. This choice helps guide the subsequent pair optimizations toward the highest likelihood in the end. Tree recursion also plays a helpful role to improve the quality of this algorithm. By processing the tips of the tree before the inner branches, optimizations in the later stages would not affect the results from earlier. An additional assumption is that the quality of earlier optimizations is less sensitive to later optimizations because the inference of states starts from tips and move toward root, as described by Equations 3a, 3b, and 4. In all, this greedy algorithm represents a best-effort, two-at-a-time maximizer and runs significantly faster than simultaneously optimizing all the branch lengths.

## 2.4 Punctuated Models

CoMET presents two variations of the *punctuated* class: *punctuated maximal* and *punctuated average*. In the first case, a greedy algorithm recursively visits parent nodes and sets one of the daughter branch lengths to zero. The daughter branch that gets chosen is the one that will result in a higher overall likelihood. Like in the greedy algorithm used for the *free* models, the quality metric is the overall likelihood and not just the local likelihood at the parent node.

The *punctuated average* model presents a more conservative model by considering all combinations of punctuation and averaging the results. The computation challenge here is the exponentially growing number of combinations possible as the number of taxa increases. For a tree with $t$ taxa, there are $t - 1$ internal nodes, each having two choices of punctuation as to which

daughter branch length to set to zero. This results in a total of $2^{t-1}$ different versions of the tree under the *punctuated average* model, as shown in Figure 2. However, within these $2^{t-1}$ trees are redundancies that can be exploited to reduce running time. The below algorithm describes how CoMET handles the *punctuated average* calculations:

1. If the current node's daughters are taxa nodes, sum the two combinations' contrasts and return.
2. Let $T_A$ be the subtree of daughter $A$, and $T_B$ be the subtree of daughter $B$. Let $I$ and $J$ be the number of internal nodes of each respective subtree.
3. Calculate $ML_A$ and $ML_B$ as the total of ML of the combinations in $T_A$ and $T_B$, respectively.
4. Let $ML_A' = ML_A * 2^{J+1}$, because $T_A$ is repeated $2^{J+1}$ times. $2^{J+1}$ is also the number of different arrangements for $T_B$. The extra 1 added to $J$ accounts for the parent node of $T_A$ and $T_B$.
5. Likewise, $ML_B' = ML_B * 2^{I+1}$
6. Efficiently compile all possible state-pairs at this node $P$ and calculate the contrasts using these pairs and the daughter branch lengths. Let $ML_P$ be the sum of these contrasts.
7. The total ML at current parent node $P$, covering all combinations, is $ML_P' = ML_P + ML_A' + ML_B'$.
8. The average ML is then $ML_P' / k$, where $k = 2^{n-1}$ and $n$ is the number of taxa and $n - 1$ is the number of internal nodes in the tree.

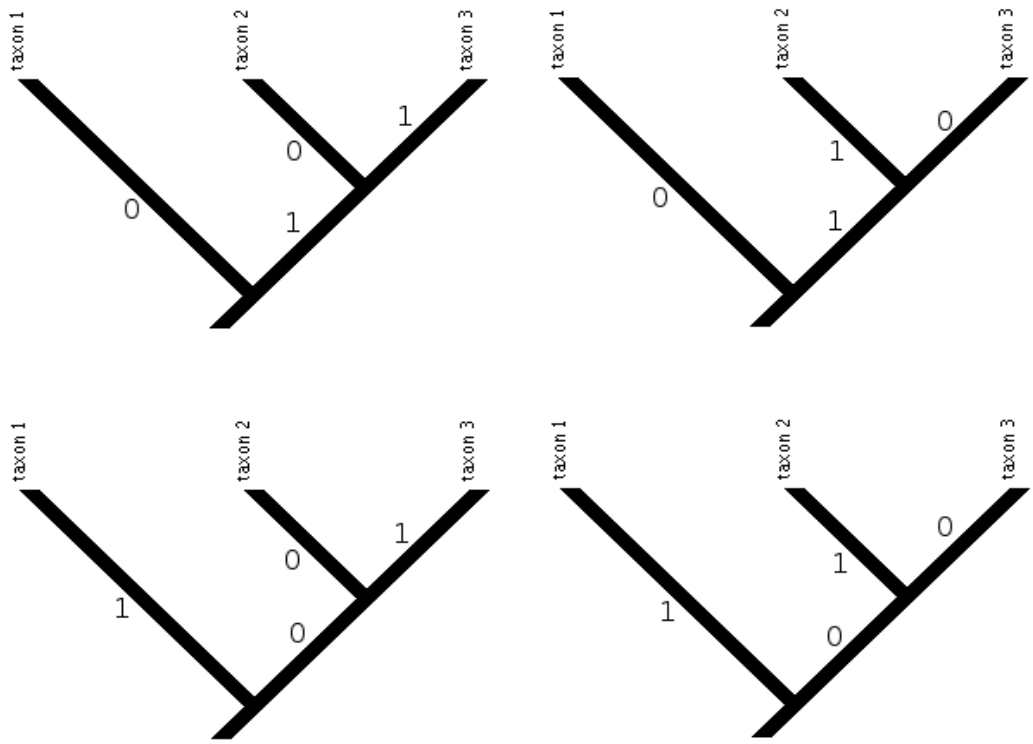



*Figure 2: A three-taxa tree has four punctuation possibilities.*

Even though significant redundancies have been reduced by steps 4 and 5, step 6 represents another challenge. In step 6, the sum of all possible contrasts at node *P* is calculated, so the need to find all possible states that occur at P's daughter nodes appears, and this collection of states must also be done as efficiently as possible due to the exponential nature in counting the number of state pairs. To illustrate, take Figure 2 again as example. In each punctuated version of the tree, the root node's contrast calculation requires the state at taxon 1 and the state at the internal node *p*, parent to taxa nodes 2 and 3. This translates to a pair of states per character per version of punctuation at the root node. The state at *p* is either the state at taxon 2 or taxon 3, depending on which daughter branch is zero due to Equation 3b. Therefore, instead of storing states, CoMET can store taxon numbers and lookup the states later.

In all, we have these pairs of taxa contributing their states for contrast calculation at the root node: {(1, 2), (1, 3), (1, 2), (1, 3)}. The size of this list is $2^{n-1}$ if $n$ is the number of internal nodes at a particular subtree. To save access time and memory, CoMET stores the taxa pair list as a taxa pair matrix (TPM). The TPM stores taxa pair counts and uses the counts as multipliers in the contrast total. Because there are two ways to punctuate two daughter branches, CoMET calculates the contrast twice using the two ways of punctuation and sums the two. This sum is then multiplied by the count value $c$ stored in the TPM. If the pair information were stored as lists with repeated elements, then CoMET would have to repeat the same contrast calculations by $c$ times . For the root node in the above example, its TPM would look like this:

| | ***1*** | ***2*** | ***3*** |
|---|---|---|---|
| ***1*** | | 1 | 1 |
| ***2*** | 1 | | |
| ***3*** | 1 | | |

*Figure 3: The root node TPM for trees in Figure 2.*

This small example does not show significant savings in memory, but for larger trees, the savings in memory and access time is clear. This TPM shows a repeat of counts because of the diagonal mirroring effect, but CoMET only reads half of this matrix when doing its calculations. There are two additional advantages for storing taxa pair information as TPM's. The first one is the ability to build TPM's recursively without requiring much more space. They begin as small tables at upper internal nodes, but merge together into larger TPM's as the recursion descends back toward the bottom of the tree. When the recursion returns to the root node, its TPM becomes an $n \times n$ matrix, where $n$ is the total number of taxa. Merging of TPM's is also fast due to indexed lookups for pair

counts. The second benefit of TPM's is reusability. The matrix uses taxa as rows and columns, and since the layout of the tree does not change with each punctuation version, the same set of matrices can be used. If instead of TPM's CoMET were to generate lists of propagated states at every parent node, then for every character, a new list would be required and generated, adding to running time significantly.

How CoMET merges TPM's is explained next. Every cell in the first TPM is compared with every cell in the second. If two cells are non-zero, the product of the two cells is added to the corresponding cells in the new TPM. As an example, Figures 5, 6, and 7 show a sample tree and the TPM's of the root's daughter nodes.

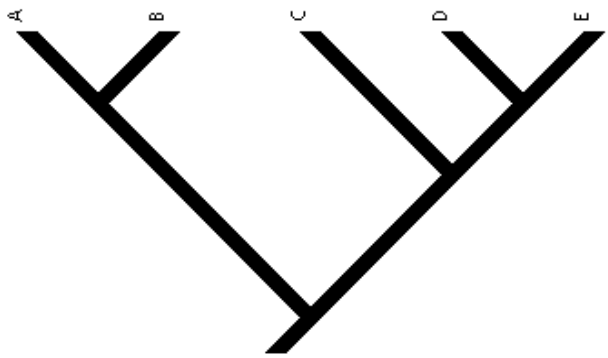



*Figure 4: A five-taxa tree.*

| | ***A*** | ***B*** |
|---|---|---|
| ***A*** | | 1 |
| ***B*** | 1 | |

*Figure 5: The TPM of the left daughter of the root in Figure 4.*

| | ***C*** | ***D*** | ***E*** |
|---|---|---|---|
| ***C*** | | 1 | 1 |
| ***D*** | 1 | | |

| | ***C*** | ***D*** | ***E*** |
|---|---|---|---|
| ***E*** | 1 | | |

*Figure 6: The TPM of the right daughter of the root in Figure 4.*

First, the cells *AB* from Figure 5 and *CD* from Figure 6 are merged. This adds 1 into the new matrix's cells *AC*, *AD*, *BC*, and *BD*. Next, cells *AB* and *CE* from the two older tables are processed, adding 1 to the new matrix's cells *AC*, *AE*, *BC*, and *BE* (Figure 8). Keep in mind that although CoMET stores the counts twice due to the mirroring in the TPM, the actual contrast calculation would only involve reading a diagonal half of this matrix.

| | ***A*** | ***B*** | ***C*** | ***D*** | ***E*** |
|---|---|---|---|---|---|
| ***A*** | | | 2 | 1 | 1 |
| ***B*** | | | 2 | 1 | 1 |
| ***C*** | 2 | 2 | | | |
| ***D*** | 1 | 1 | | | |
| ***E*** | 1 | 1 | | | |

*Figure 7: Merged TPM.*

## 2.5 Conclusion

CoMET is a general-purpose tool that matches character data with a best-fit model chosen from one of Oakley's nine models of trait evolution. As long as an experiment involves continuously-varying data, CoMET can be used to hypothesize the data's phylogeny. CoMET offers automatic optimization of the $\beta$ parameter, a fast algorithm for the *free* models, and an efficient process to handle *punctuated average* models. Through the Mesquite platform, CoMET is easily accessible through a graphical user interface for end users. In addition, Mesquite's framework allows CoMET to be easily extensible and integrated into other Mesquite modules.

# Chapter 3: Simulation Experiments

## 3.1 Introduction to Simulations

The purpose of the simulation experiments conducted was to answer questions concerning the *punctuated* models. First, the DOF for *punctuated maximal* models is unclear, because choices exist for deciding which branches are to remain unchanged and which to have lengths assumed to be zero. If these choices are considered as parameters, then the DOF would be as high as the number in *free* models, always resulting in poorer AIC values. The *punctuated average* avoids this uncertainty by averaging the ML over all versions of punctuation, although this model may be too conservative because its ML results have mostly been lower than other models. On the other hand, the *punctuated maximal* only processes the maximal version of the many ways to punctuate a tree, and the punctuation choices it makes may count as extra parameters. At this point, the DOF for *punctuated maximal* is unclear, and one of the goals of these experiments is to empirically determine its DOF.

The simulations first compared the two *punctuated* models with the *phylogenetic models* to test their sensitivities to varying levels of punctuation. The findings of these simulations lead to a greater understanding surrounding the DOF issue, suggesting an addition to CoMET for user-defined punctuation.

## 3.2 Experiments with Asymmetry Ratios

The experiments consisted of running a subset of the CoMET calculations on characters simulated over a range of punctuation levels. Specifically, the *phylogenetic*, *punctuated maximal*, and *punctuated average* classes ran ML and AIC calculations for each of their three model types:

*distance*, *equal*, and *free* for a total of nine models. The basis trees used in the CoMET simulations were the yeast kinases and HSP dnaK trees borrowed from Oakley's data. The program flow of the simulations is explained next.

Prior to simulating character data for CoMET, the simulator must first build trees of different punctuation levels with which to simulate characters. The measure of punctuation is determined by the length ratio between two sister branches. Trees with asymmetry ratios (AR's) ranging from 20 to 2000 were randomly constructed using a Gaussian distribution centered at the desired asymmetry ratio to provide exact multipliers for one of every two sister branches. These one-sided scaling actions turned the basis tree into variations with punctuated branching events.

After creating random punctuated trees, the simulator generated character states using Mesquite's built-in Brownian motion model. During the course of testing, it was discovered that generating states for more than one character leads to uninteresting results. This was due to the Brownian motion model producing tip states averaging toward zero in this restricted evolution model. With enough characters, even the states for a single taxon's multiple characters began to average toward zero due to the balanced mix of positive and negative states generated by Mesquite's Brownian motion model. Consequently, the characters did not show indications of punctuation, and the decision was made to generate only one character per random punctuated tree.

After simulating a character for a random tree, CoMET calculated the ML and AIC values for the aforementioned nine models. These calculations were repeated for ten iterations per AR. In each iteration, the best AIC from the three *phylogenetic* models was paired with the best AIC from *punctuated maximal*'s three models and the best AIC from *punctuated average*'s three models. The ratios between the best *phylogenetic* AIC with the other two best AIC's from the *punctuated* models were then computed and averaged over these ten iterations. The overall program flow for this simulation is show in Figure 8. This test showed how different the two variants of *punctuated* are to

*phylogenetic* when using simulated punctuated character data.

## 3.3 Experiments with Pure Punctuated Trees

Alongside the experiments using AR's to generate random punctuated trees were the experiments that use pure punctuated trees for character simulation. In this case, the AR was referred to as the non-zero length (NZL) because this random number was used for the exact branch length of one daughter branch per internal node while the other daughter branch was set to zero for ultimate punctuation. The results from these tests were compared against the results from the experiments with the more realistic punctuated trees discussed earlier.

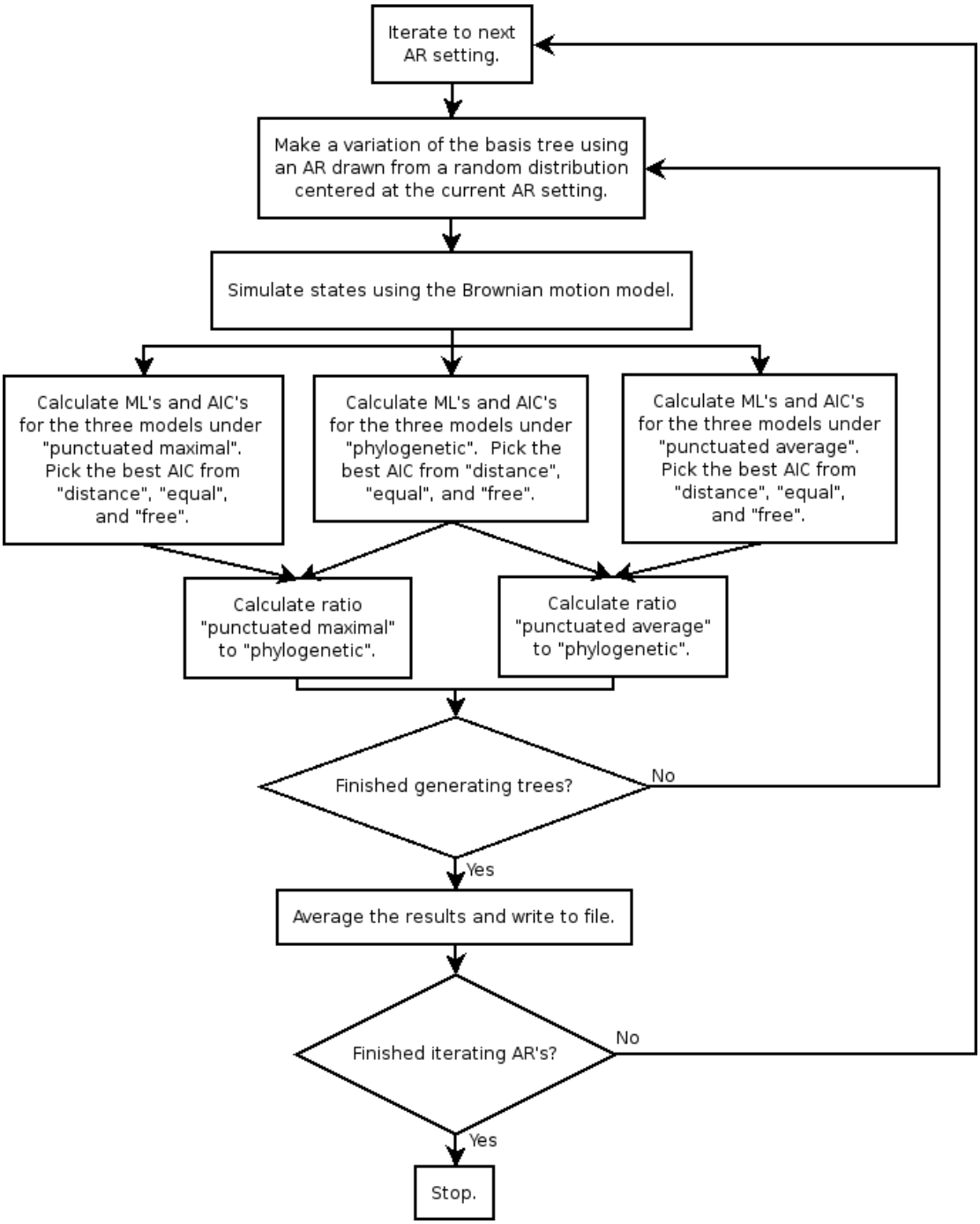
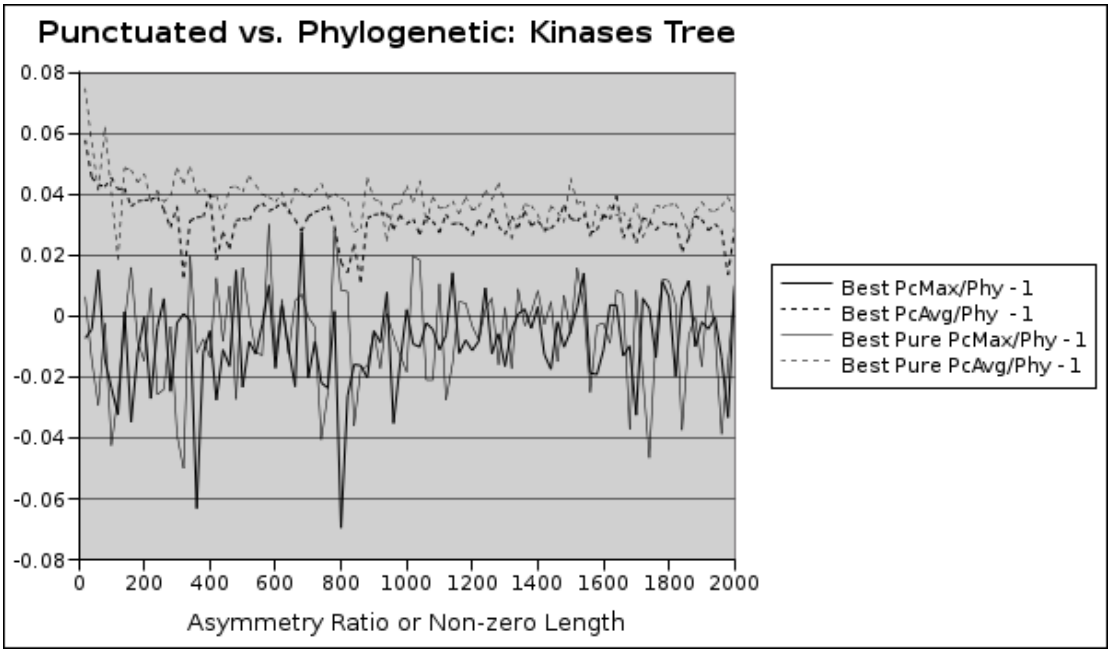



*Figure 8: Experiment program flow over a range of AR values.*

## 3.4 Results and Discussion

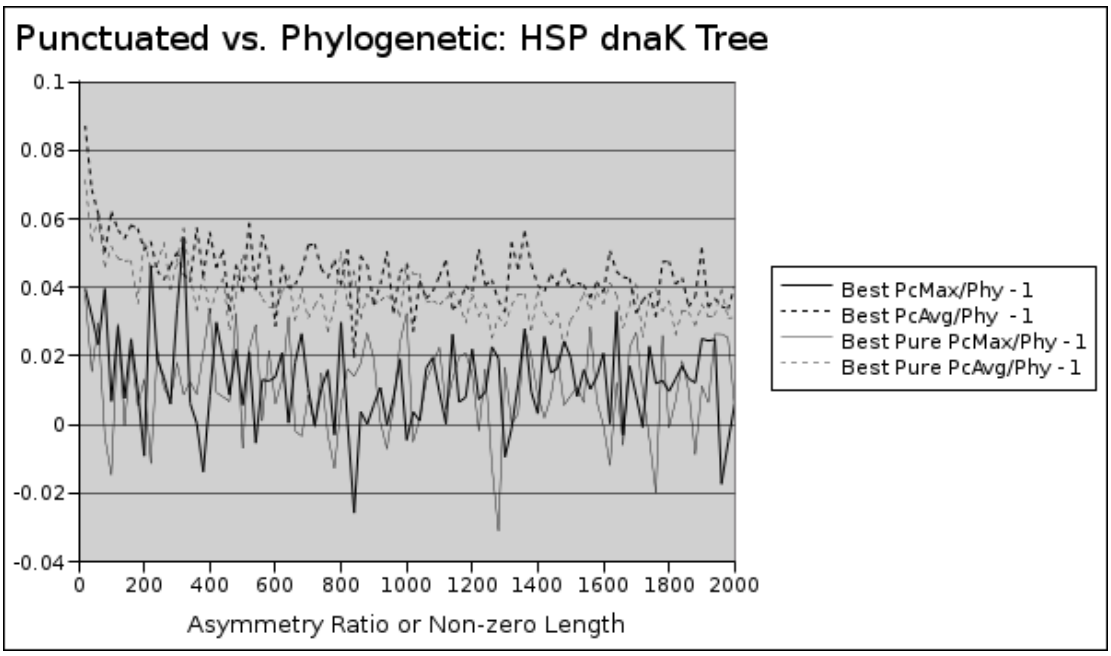
The graphs in Figure 9 and 10 below show how well *punctuated maximal* and *punctuated average* detect punctuation as compared to the standard *phylogenetic* models. In addition, the

results of the experiments generating pure punctuated trees using NZL settings are also plotted.




| ***AIC Results Comparison*** | ***Mean*** | ***Median*** |
|---|---|---|
| (Punctuated Max. / Phy.) - 1 | -0.00880 | -0.00850 |
| (Punctuated Avg. / Phy.) - 1 | 0.03144 | 0.03144 |
| Pure (Punctuated Max. / Phy.) - 1 | -0.00650 | -0.00500 |
| Pure (Punctuated Avg. / Phy.) - 1 | 0.03807 | 0.03746 |

*Figure 9: Results based on the kinases tree.*




| ***AIC Results Comparison*** | ***Mean*** | ***Median*** |
|---|---|---|
| (Punctuated Max. / Phy.) - 1 | 0.01277 | 0.01207 |
| (Punctuated Avg. / Phy.) - 1 | 0.04439 | 0.04317 |
| Pure (Punctuated Max. / Phy.) - 1 | 0.01047 | 0.00977 |
| Pure (Punctuated Avg. / Phy.) - 1 | 0.03790 | 0.03632 |

*Figure 10: Results based on the HSP dnaK tree.*

From these results, three main observations can be made. The first is the performance of the *punctuated maximal*. This model is a significantly better fit for the character data than *punctuated average*. In the case of the kinases tree, the *punctuated maximal* also beats *phylogenetic* rather consistently. The second observation finds the p*unctuated maximal* to be insensitive to the changing asymmetry, unlike the *punctuated average* model. This insensitivity plus the first observation together suggest that perhaps the *punctuated maximal* is too aggressive to model punctuation. In addition, these findings show that even the conservative *punctuated average* has the steady trend to

beat *phylogenetic* as asymmetry increases. As a result, *punctuated average* was accepted as the more useful model.

The third main observation is the similarity between results from simulating characters with pure and impure punctuated trees. Since pure punctuation is infinitely more punctuated than the impure, one expects results from those types of trees to be significantly different. However, the results say otherwise. In light of these findings, a likely explanation for the similarity is that even when the resulting characters show pure punctuation, the ML for pure punctuation would not be significantly different from the ML for impure punctuation due to the smoothing effect of randomness in the Brownian motion model.

## 3.5 Follow-up Experiment and Results

Using the lessons learned from the three observations from the first experiment, a follow-up experiment was conducted. This experiment's purpose was to test whether or not redefining punctuation as a user-specified variable is feasible. The graphs of the *punctuated average* models in Figures 9 and 10 show a steady trend toward favoring *punctuated average* over *phylogenetic,* so it would be interesting to see if the user of CoMET can decide what asymmetry ratio represents the threshold in defining a tree as punctuated or not. The graph would then be shifted downward and cross into the negatives at some point by multiplying a factor to the DOF to lower the AIC. This time, the *free* model was not used because it does not have the same DOF as the *distance* and *equal* models; therefore, an adjustment to the DOF would not account for all three model types properly. Also in this follow-up, results using *punctuated maximal* were no longer calculated, and characters were simulated using impure punctuated trees only. This is because, as previously discussed, the *punctuated maximal* is too aggressive to be useful and the pure punctuated trees offer no advantages over the more realistic impure punctuation. The chart in Figure 11 below shows the AIC ratio

comparison between *punctuated average* and *phylogenetic* using the best AIC from *distance* and *equal* only.

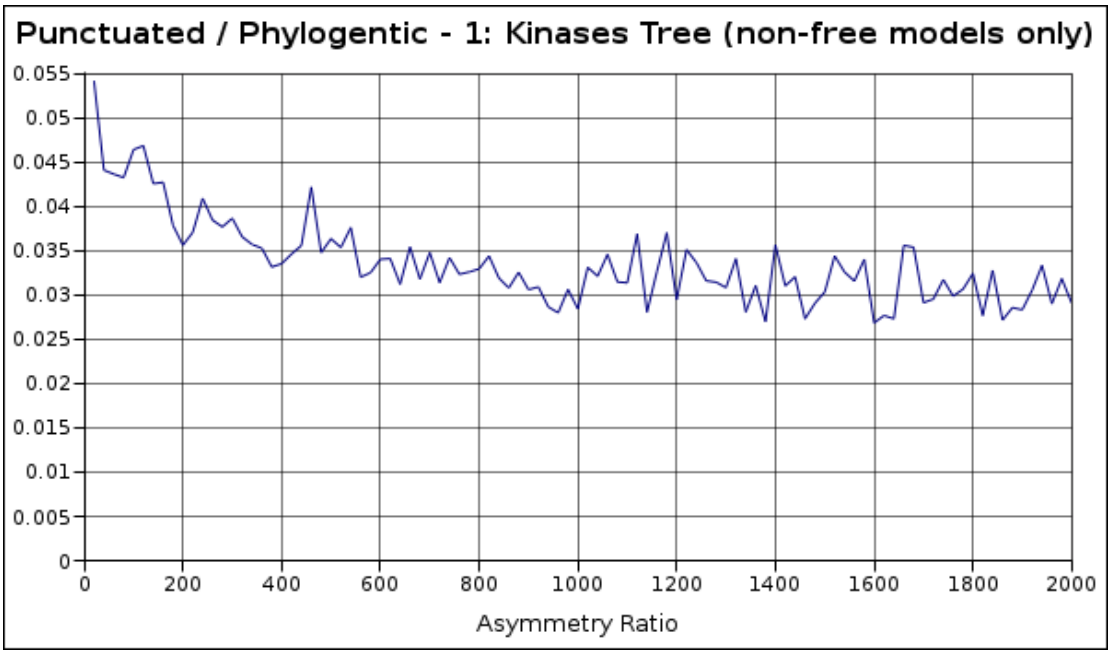



*Figure 11: AIC ratio results without influence from the* free *model.*

Suppose that a CoMET user runs a simulation like the one above, and wishes to set a punctuation threshold at AR = 500 so that when the full CoMET test is run, the *punctuated average* model would be favored for situations in which the AR is found to be over 500. CoMET could take the simulation data and find a DOF multiplier for the *punctuated average* to be used for later testing against character data. Using this experiment as an example, the average ML for *phylogenetic* at AR = 500 is -25.105. The average ML for *punctuated average* at AR = 500 is -26.053. If AR = 500 is the threshold, then for AR < 500, *phylogenetic* would have a lower AIC than *punctuated average*, and vice versa for AR > 500. Therefore, at AR = 500, the AIC's of both should be equal, resulting in the below expression using Equation 5:

$$-2(-25.105) + 2(1) = -2(-26.056) + 2(1)x$$

In this expression, *x* is the DOF multiplier necessary to establish the redefinition of punctuation. The value 1 is the default DOF for *punctuated* models. The value of *x* in this expression becomes 0.049. If this is multiplied to the DOF for *punctuated average / distance* and *punctuated average / equal*, then *punctuated* should be favored over *phylogenetic* when AR > 500 is detected. These custom DOF multipliers are tree-specific, so separate simulations to determine these multipliers are necessary for different trees.

As for the *free* model, a different simulation can be run that compares just the *punctuated average / free* with the best of the three *phylogenetic* models. A new DOF multiplier can be derived and reused for repeated testing with the same tree.

## 3.6 Conclusions

The simulation experiments initially began as a quest to find a way to deal with the DOF for the *punctuated maximal* models. After the initial findings, the usefulness of *punctuated maximal* was devalued so the DOF problem became a non-issue. Although *punctuated average* shows a good sensitivity to varying asymmetries, it is usually too conservative to beat *phylogenetic*. Therefore, the follow-up simulation shows how one can give *punctuated average* a handicap by lowering its DOF with a multiplier and allow the user to defined the threshold of punctuation.

Another insightful finding is the similarity of ML's between data simulated from pure and impure punctuated trees. This discovery suggests that zero-length branches may not be that different from short-length branches after all. A consequent reduction of some of CoMET's complexities in the future may follow, because modeling zero-length branches may no longer be necessary for the *free* models. In which case, confusing *zero fork* scenarios would not exist, and PAL's multivariate optimizers may be able to replace the current greedy algorithm.

# Chapter 4: Summary

## 4.1 Conclusions

Phylogenetic trees' graphical representations of ancestry can lead to a better understanding about current species as well as new insights concerning their evolutionary conditions. However, since most trees are based on genetic comparisons, they do not necessarily reflect the phylogeny of other features of the taxa. To address this issue, CoMET uses the Oakley models to test character data against phylogenetic trees and answers the question as to how the given data evolved.

The development of CoMET presented challenges in optimizing branch lengths of the *free* models and improving the efficiency in calculating the *punctuated average* models. The *free* models were accommodated by an approximative algorithm and the efficiency in calculating the ML for *punctuated average* was drastically improved through redundancy analysis and reduction..

Simulation experiments further increased the understanding on the models and the underlying algorithms in CoMET. After these experiments, the *punctuated maximal* is no longer considered a useful model due to its aggressive and insensitive results as well as its uncertainty in calculating AIC's. In addition, the comparison between pure and impure punctuated trees shows that the difference in ML results between using zero-length branches and non-zero-length branches is basically nonexistent.

Over 3600 lines of Java code are available to add to the Mesquite Project. This includes CoMET in whole-matrix and single-character versions as well as the simulation module used in the experiments. Binaries, source, and documentation are available at http://www.cs.ucsb.edu/~chunghau/comet, or by contacting Chunghau Lee or Dr. Todd Oakley at the University of California, Santa Barbara.

## 4.2 Open Problems and Future Work

Although CoMET was released even before simulations were done, the unsolved problems existing then still exist now. The major hurdle is still the *free* model. Since the simulations show that zero and non-zero branch lengths do not show much difference in ML results, we can model the *free* trees as continuous multivariable functions without *zero-fork* problems and use the conjugate gradient method.

Another idea for a new feature is the custom definition of punctuation through simulation training. After receiving the hypothesis tree and an AR from the user to use as the punctuation tolerance level, CoMET can run the simulation similar to the one shown in Figure 11 to determine a proper DOF multiplier for the *punctuated average* models. This would then allow character data suggesting punctuation exceeding the user-specified tolerance to be designated as punctuated.

# Appendix: References

Akaike, H. 1973. *Information theory and an extension of the maximum likelihood principal*. Proc. 2nd Int. Symp. Information Theory, Suppl. Problems of Control and Information Theory. 267-281.

Drummond, A., and K. Strimmer. 2001. *PAL: An object-oriented programming library for molecular evolution and phylogenetics*. Bioinformatics 17: 662-663.

Eck, R.V. and Margaret O. Dayhoff. 1966. *Evolution of the Structure of Ferredoxin Based on Living Relics of Primitive Amino Acid Sequences*. Science 152(3720): 363-366.

Felsenstein, J. 1981. *Evolutionary Trees From Gene Frequencies and Quantitative Characters: Finding Maximum Likelihood Estimates*. Evolution 35(6):1229-1242.

Felsenstein, J. 2003. *Inferring Phylogenies*. Sinauer Associates, Inc.

Fitch, W. M. 1971. *Toward defining the course of evolution: minimum change for a specified tree topology*. Systematic Zoology 20:406-416.

Gu, Xun. 2004. *Statistical framework for phylogenetic analysis of expression profiles*. Genetics 167:531-542.

Hulsenbeck, J. P. and K. A. Crandall. 1997. *Phylogeny Estimation and Hypothesis Testing Using Maximum Likelihood*.

Maddison, W. P. and D.R. Maddison. 2004. *Mesquite: a modular system for evolutionary analysis*. Version 1.05 http://www.mesquiteproject.org.

Mooers, A., S. Vamosi, and D. Schluter. 1999. *Using Phylogenies to Test Macroevolutionary Hypotheses of Trait Evolution in Cranes (Gruinae)*. The American Naturalist 154(2):249-259.

Oakley, T., Z. Gu, E. Abouheif, N. H. Patel, and W. H. Li. 2004. *Comparative Methods for the Analysis of Gene Expression Evolution: An Example Using Yeast Functional Genomic Data*. Molecular Biology and Evolution 22(1):40-50.

Saitou, N. and M. Nei. 1987. *The neighbor-joining method: a new method for reconstructing phylogenetic trees*. Molecular Biology and Evolution 4(4):406-425.

Schulze, A. and J. Downward. 2001. *Navigating gene expression using microarrays -- a technology review*. Nature Cell Biology 3:E190-E195.