UNIVERSITY OF CALIFORNIA
SANTA CRUZ

Using Phylogenetic Markov Trees to Detect Conserved Structure in RNA Multiple Alignments

A thesis submitted in partial satisfaction of the requirements for the degree of

MASTER OF SCIENCE

in

COMPUTER ENGINEERING

by

Bradford A. Gulko

March 1995

The thesis of Bradford A Gulko is approved:

_____
David Haussler
Professor of Computer and Information Sciences

_____
Kevin Karplus
Associate Professor of Computer Engineering

_____
Richard Hughey
Assistant Professor of Computer Engineering

_____
Dean of Graduate Studies and Research

Contents

Figures

Tables

Equations

ABSTRACT

This thesis discusses the need for more sophisticated techniques to determine the physical structure of ribonucleic acid molecules (RNA) *in vivo*. In particular, we emphasize several shortcomings in current techniques of secondary structure analysis. These shortcomings commonly stem from each technique's focus on individual nucleotide sequences. While the inclusion of phylogenetic sequence information in structure determination can alleviate such shortcomings, currently available phylogenetic techniques require substantial manual intervention. To automate RNA structure analysis, we develop a novel technique called the Tree Model that uses phylogenetic data to automatically model secondary structure evolution over entire families of related RNA sequences. We test the Tree Model by using it to find base pairing between multiple alignment columns.

The Tree Model employs Maximum Likelihood inference to generate a model for the evolution of multiple alignment column pairs. The course of this evolution is modeled through the use of a Markov Tree to represent the phylogenetic tree. The Markov Tree is developed as an extension of the Markov process to a tree-shaped graph. For a given multiple alignment column pair, each node of the graph represents a random variable over possible nucleotide pairs for an individual organism. Leaf nodes represent observed sequence data from each organism in the multiple alignment. Internal nodes represent "synthetic ancestors" whose sequence information must be inferred from its descendants. Edges of the graph represent local genetic relationships between direct descendants that are quantified through a point-mutation model. The mutation model's parameters represent the probability of a child having a specific nucleotide pair, given the parent's nucleotides for that column pair. We explore three methods of deriving these parameters from the multiple alignment data.

A Tree Model accepts a multiple alignment column pair and generates a probability distribution over the possible nucleotide pairs for each internal node of its tree. The probabilities of each possible evolutionary path through these nodes are then accumulated using Dynamic Programming to determine a total likelihood for the column pair. Such likelihoods can be generated for a given pair of columns under each of several Tree Models. These probabilities can then be compared to classify the novel column data, based on the set of multiple alignment columns used to generate each Tree Model's parameters.

As a test of the Tree Model, we use it to look for base pairings in a family of 1375 16S RNA. Multiple alignment column data is broken into a training set and a test set for cross validation purposes. The Tree Model parameters are configured on the training set and then applied to the validation set. The test set accuracy of this model in discriminating between base paired and non base paired column duos is shown to be in excess of 90%. Accuracy rises to more than 99% when highly conserved column duos are removed to reduce data degeneracies. This compares favorably with both the 85% accuracy provided by a simple frequency based model on the same data, and 60%-80% accuracies reported by other researchers using energy minimization and manual phylogenetic techniques on similar RNA data. Finally, we propose extensive directions for further research.

Acknowledgments

In the time that I have spent working on my thesis, I have had gracious help from a number of sources. I would first like to thank David Haussler for both the specific favor of providing me with the subject & direction for this work, as well as for introducing me to the field of Machine Learning. Dadiv's tutelage in this field has profoundly effected the way that I view the physical world as much, if not more, than any of my undergraduate studies in Physics. In addition, his clarity, flexibility and enthusiasm has more than once pulled this effort from the brink of oblivion.

Kevin Karplus's criticisms on my experimental technique and writing style helped elevate the drafts of this work from completely impenetrable, to largely legible. Many good ideas of his are featured in the future directions section of this work. Richard Hughey's last minute tutorial on grammar and the art of the "which" hunt was invaluable. Michelle Abram provided boundless support for my completion of this work, slogging through numerous early drafts and performing a several crucial remote procedures for me when I could not physically be on campus. I must reserve a special thanks for Michael Fleming whose tremendous patience, well timed sense of practicality and profound intellect provided well needed direction when my own was floundering. If any of this work is genuinely enjoyable to read, Michael is undoubtedly at the root of it. Finally, as my closest friend and housemate during the birth of this work, he generously put up with my ridiculous schedule and environmental cravings which were especially severe during the last few months of labor.

Several corporations also provided assistance on the completion of this work. Sun Microsystems and the Digital Equipment Corporation provided access to a substantial battery of computation facilities (during off hours) for running the compute intensive precursor the Tree Model. Silicon Graphics provided internet access and CPU months on their multiprocessor Onyx servers during the exploratory phase of this work. Lepton Incorporated provided personal computers, office facilities an occasional paycheck precisely when I was in most desperate in need of them. Without any of these facilities, the present work might have been delayed past the possibility of completion.

Finally, words are insufficient to express my thanks to Sandra Robbins and Robert Gulko. They both provided much needed encouragements and support far beyond any conceivable familial predisposition. In addition to cheering me along the astoundingly twisted path that lead to my graduate work at UCSC, their warmth, affection and confidence helped me through the most difficult parts of graduate school. Those which had little to do directly with academia.

1 Introduction

1. 1 General Motivation

The investigation of the structure and function of the human genome is one of the grand challenges of modern molecular biology [1][2]. As RNA & DNA are believed to embody the overwhelming majority of genomic functionality, a tremendous amount of research has been devoted to the investigation of nucleic acid structure and function. However, while DNA's role is largely limited to the storage of genetic information, RNA can self-replicate, store genetic information and build complex proteins. These characteristics bring RNA far closer to being a complete life form then DNA, and indicate that RNA is the more primal form of nucleic acid. This conception of RNA as the primal nucleic acid has motivated an acceleration in the effort to understand its function and structure. As the phenomenon of nucleotide pairing within biological RNA molecules is critical to both their structure and their function, nucleotide pairing research is of central importance in this effort [3]. The current work presents a novel method for investigating the evolution of such pairing structures in RNA multiple alignments. This method, which we call the Tree Model, uses a simple local mutation model to develop a phylogenetically global evolutionary model for RNA multiple alignment column duos[1]. The extension from the local statistics of point-mutation to a global evolutionary model is accomplished through Maximum Likelihood inference on the structure of a

[1] The term duo is used to refer to a 2-tuple of nucleotide columns in a multiple alignment. These columns may or may not interact. The term "pair" will be reserved for those duos that are believed to interact. Examples of such interactions include: Watson-Crick pairing, helix endcaps and, potentially, tertiary structure. For a more complete definition of interactions included in the term "paired", please see *3.1 Data Sources*, page *73*.

phylogenetic tree. Once developed, the Tree Model is used to infer the existence, location and evolutionary behavior of nucleotide column pairs.

Research in molecular biology has made it apparent that the *in vivo* 3-D structure of an RNA molecule plays a critical role in its function [4][5]. While a tremendous amount of nucleotide sequence data has recently become available through initiatives such as the Human Genome Project (HGP), information on the 3-D structure of RNA remains sparse. The techniques of Magnetic Resonance Imaging (MRI) [6], x-ray crystallography [7] and electron microscopy [8] have each provided some 3-D models of simple RNA structure. However, the process of sample preparation requires the dehydration of the sample, which is expected to have a significant effect on its structure. Some work has been done towards less destructive electron microscope measurement, but it is still nascent [9].

Due to the present technical barriers preventing the direct measurement of RNA 3-D structure *in vivo*, information regarding this structure must be inferred from available data such as nucleotide sequences [10]. Traditional efforts to derive 3-D structure from nucleotide data have been labor-intensive, involving a great number of researcher-hours spent pondering shared structure among a few available sequences. Automated computational tools that might assist in the process of structural modeling have become available but have shown only a modest potential. Such conventional tools for performing RNA modeling have been derived from principles of the physical chemistry of large molecules called macromolecules.

The most fundamental theoretical tools for the investigation of atomic interactions in a molecule are the equations governing quantum mechanics. While the smallest molecules might be computationally amenable to quantum modeling, biological RNA molecules grossly exceed this scale [11]. To reduce the complexity of macromolecular simulations, a complex molecule may be separated into a relatively small number of stable molecular groups, each group having only a few degrees of freedom [12]. However, even limiting each nucleotide in an RNA molecule to five or six degrees of freedom produces a computationally intractable assemblage for all but the smallest biological RNA molecules. Restricting the number of degrees of freedom per nucleotide further can cripple the ability of a simulation to accurately represent the complete RNA. As these classical analytic techniques have proven inadequate for the modeling of interesting RNA molecules, researchers have investigated simplified heuristics.

One such heuristic technique for the determination of RNA folding structure involves the definition and minimization of a global energy metric, such as Gibbs free energy [13][14]. First, a given molecule's 3-D structure is coarsely parameterized. Then, these parameters are iteratively altered to reduce the molecule's measure under the energy metric. Typical methods for finding optimal parameters for such constrained nonlinear optimization problems include simulated annealing and gradient descent. This type of numerical energy minimization has traditionally produced satisfactory solutions to systems that are intractable to exact analytic modeling. However, when this type of analysis was applied to the folding of a single RNA sequence into a single RNA

molecule, energy degeneracy problems arose. Under simply defined approximations to an RNA molecule's Gibbs free energy, numerous locally optimal solutions appear with similar energies but significantly differing folding patterns. The broad variety of foldings suggested by this technique requires heuristic, and generally manual, post-processing to produce acceptable results. This is not particularly surprising, as the interactions within the RNA molecule are sufficiently complex that computationally feasible energy potential approximations have a margin of error of approximately 10% [15].

The modeling techniques currently in use, energy minimization and macromolecular analysis, both concentrate on finding the 3-D structure for a single RNA sequence. It seems, however, that the underlying complexity of the folding process that determines the 3-D structure is sufficiently great so as to make these methods either indeterminate or prohibitively costly. We present a different paradigm for the design of structure modeling tools, in an effort to overcome these problems. It is hoped that this approach will yield more effective tools that will accurately and efficiently automate a large part of the modeling process. To avoid the previously discussed degeneracy problems, this novel approach abandons the detailed physical investigation of a small number of sequences in favor of statistical inference over thousands of samples. The present work seeks to harness the explosion in available primary structure data to provide folding parameters through the use of Maximum Likelihood inference on a phylogentic tree.

Since the base pairing structure of an RNA molecule in a multiple alignment has a profound effect on that molecule's 3-D structure, we apply the fore mentioned statistical methods to develop a tool for investigating this pairing. This tool constructs a complete probabilistic model for the evolution of multiple alignment column duos using a phylogenetic tree. The phylogentic tree is modeled as a Markov Tree, a novel extension of Markov processes to tree-shaped state relationships. The parameters of the Markov Tree are estimated over a large fraction of the entire multiple alignment, resulting in a compact, yet general, model for the evolution of RNA nucleotide duos[2] [16]. This model can provide interesting insights into the general process of RNA development over evolutionary time spans. In addition, the evaluation of a single column duo according to the trained parameters of this model produces a Maximum Likelihood distribution over all of the possible evolutionary paths for that column duo. *Dynamic programming* techniques [17] can then be applied to this distribution to calculate a posterior probability for the evolution of the evaluated column duo. When this posterior probability is compared to that produced by a null model for the same column duo, a simple yet powerful pairing discriminator is formed.

[2] The recent work of Han & Kim (1993) [16] has also used a technique involving a weighted summation over multiple homologous RNA molecules. However, their techniques were not probabilistic in nature. Though it did involve the construction of phylogenetic relationships between closely related sequences, it did not interpret this tree as a statistical process or use statistical inference to derive results. Han & Kim used arbitrarily constructed editing weights and produced variability coefficients that were not amenable to probabilistic interpretation. Their work automatically calculated secondary structure for sets of 20 to 40 tRNA molecules with approximately 70% accuracy as opposed to the 90%+ accuracy attained in this work by the Tree Model (*3.9 IO Model*).

1. 2 Technical Overview

1.2. 1 The Modeling Process

The primary result of this work is the development of a tool to detect paired columns in an RNA multiple alignment. However, the ultimate goal of the paradigm that was used to generate this tool is an automated process by which the complete structure of an RNA family can be automatically constructed from unaligned RNA sequences (*Figure 1-1*) [18]. As the secondary structure detection provided by the Tree Model occupies a position in this hypothetical modeling process, we briefly describe the

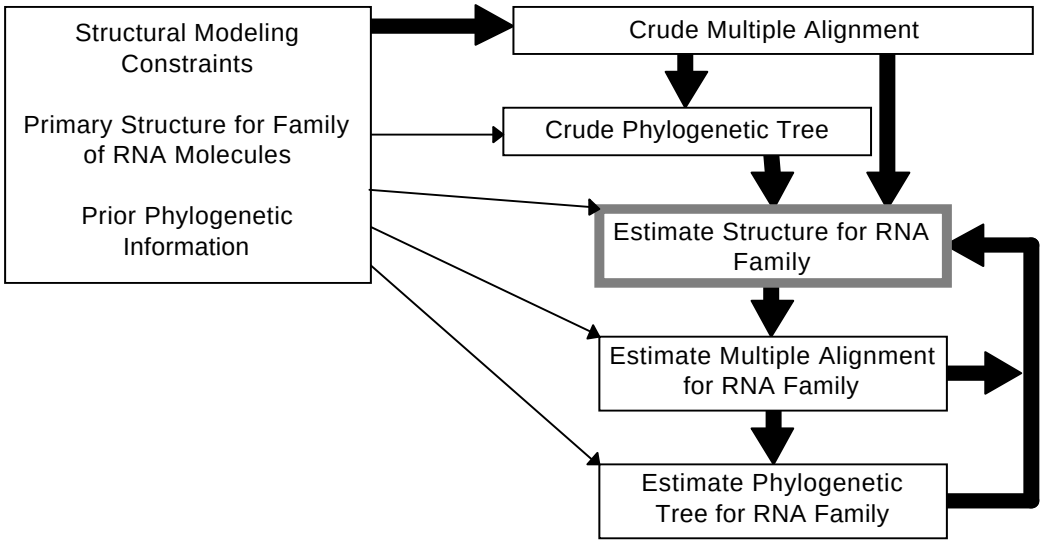

Figure 1-1: Generic Algorithm for RNA Structure Determination

This figure coarsely represents a process for the construction of an RNA structure model. Thick arrows represent the primary flow of the calculation, while the thin arrows represent data that also influences calculations. Currently, a large number of skilled researcher-hours is required for the construction of such a model. The current work aims to reduce these subjective factors by automating a piece of the Estimate Structure module (gray box). This is accomplished through the use statistical Maximum Likelihood inference applied to the tremendous amounts of newly available primary sequence information. It is hoped that the application of similar techniques to the other parts of the modeling process can fully automate this algorithm.

modeling process and show how the current work contributes to it.

Before discussing the details of our automated RNA structure modeler, it is imperative that we develop a consistent understanding of the formalisms presently used to represent RNA molecules. RNA molecules are composed of an ordered chain of nucleic acid molecules (nucleotides) which are covalently bonded to a linear phosphate/sugar backbone. Though the nucleotides themselves are fairly complex molecules, consisting of approximately 10 to 15 individual atoms [19], internal nucleotide structure will not play a significant role in the current work. The internal structure of the nucleotides may thus be neglected, and the individual nucleotides treated as unitary. The four nucleotides typically found in RNA are Adenine (A), Cytosine (C), Guanine (G) and Uracil (U). As the phosphate/sugar backbone to which these nucleotides are attached may be drawn into a linear form, an RNA's sequence of nucleotides may be compactly represent as a string over the alphabet (A, C, G, U). This string is referred to as the RNA molecule's sequence, or Primary Structure (*Figure 1-1*). Recent advances in automated sequencing, have provided a wealth of this type of data.

1.2. 2 The Mutiple Alignment

Certain structurally homologous[3] ribosomal RNA are present in all organisms that synthesize protein [20]. It is thus possible to find structurally homologous RNA molecules in vastly differing organisms. Groups of such related RNA molecules are

[3] Structurally homologous RNA molecules are those which have similar shapes. Evolutionary homologues share a common ancestry and functional homologues fulfill similar biological roles.

called families and include: Transfer RNA (tRNA), Small Subunit RNA (SSU or 16S RNA) and Large Subunit RNA (LSU or 23S RNA). The precise number of nucleotides in each of these molecules varies from organism to organism: Transfer RNA typically contains 60-130 nucleotides and is the most well understood; Large Subunit RNA typically contains 2500-5000 nucleotides; and Small Subunit RNA (used in the current work) typically contains 1200-2000 nucleotides per molecule. In order to highlight the similarities and differences within a given family of RNA sequences, a multiple alignment is constructed for the family. *Figure 1-2* serves as an example of the primary sequence information we given to work with and is discussed in the text that follows.

A multiple alignment is a template containing one column for each possible nucleotide position in a molecule. Each row of the alignment represents a single species' contribution (one sequence) to the RNA family. As the number of columns in the multiple alignment must be at least as large as the number of nucleotides in the largest molecule of a family, some spaces are inserted into the sequences of the smaller molecules from that family. These spaces are referred to as gaps or deletes. One primary purpose of the multiple alignment is to show structural correspondence by displaying corresponding nucleotides from differing organisms in the same column of the alignment. There is often heated debate as to which nucleotides are in "structural correspondence", and thus a strong subjective element in multiple alignment construction. Once a multiple alignment has been constructed for a given RNA family, the genetic similarity between aligned sequences may be used to construct a phylogenetic tree for the corresponding organisms (see *Figure 1-3*).

ID	Columns 33 to 60	Columns 2664 to 2682	Paired Duos 1	2	3	4	Random Duos 1	2	3	4
			576	577	578	579	1173	2221	1469	1066
			561	560	559	558	1211	2153	974	524
1	AUUCCGGUU-GAU-CCUG??GG	UG??????GAUCACCUCC?'	GA	GC	UA	CG	GU	CG	-A	CA
2	AUUCCGGUU-GAU-CCCGCCGG	UG??????GAUCACCUCC?'	GA	CG	CG	CG	GU	CG	-A	CA
3	AUUCCGGUU-GAU-CCCGCCGG	UGCGGCUGGAUCACCUCC?'	GA	CG	CG	CG	GU	CG	-A	CA
4	ACUCCGUUU-GAU-CCUGGCGG	UGCGGCUGGAUCACCUCCU'	GC	CG	CG	CG	GG	CG	-A	CA
5	AGUCCGUUU-GAU-CCUGGCGG	UGCGGCUGGAUCACCUCCU'	GC	UA	UA	CG	GA	CG	-A	CA
6	AAUCUGUUU-GAU-CCUGGCAG	UG??????GAUCACCUCCU'	GC	UA	UA	CG	GA	CG	-A	CA
7	AGUCCGUUU-GAU-CCUGGCGG	UGCGGCUGGAUCACCUCCU'	GC	UA	UA	CG	GA	CG	-A	CA
8	AUUCUG?UU-GAU-CCUGCCAG	UGCGGCUGGAUCACCUCCU'	AU	AU	UA	CG	GG	CC	-A	CA
9	AUUCUGGUU-GAU-CCUGCCAG	UG??????GAUCACCUCCU'	AU	UA	UA	CG	GG	CG	-A	CA
10	AUUCUGGUU-GAU-CCUGCCAG	UGCGGCUGGAUCACCUCCU'	AU	AU	UA	CG	GC	CG	-A	CA
11	AUUCCGGUU-GAU-CCUGCCGG	UGCGGCUGGAUCACCUCCU(	AU	UA	UA	CG	GC	CA	-A	CA
12	AUUCCGGUU-GAU-CCUGCCGG	UGCGGCUGGAUCACCUCCU'	AU	UA	UA	CG	GC	CA	-A	CA
13	AUUCCGGUU-GAU-CCUGCCGG	UGCGGCUGGAUCACCUCCU'	AU	UA	UA	CG	GC	CU	-A	CA
14	AUUCCGGUU-GAU-CCUGCCGG	UGCGGCUGGAUCACCUCCU'	AU	UA	UA	CG	GC	CU	-A	CA
15	AUUCCGGUU-GAU-CCUGCCGG	UGUGGCUGGAUCACCUCCU'	AU	UA	AU	CG	GC	CA	-A	CA
16	AUUCCGGUU-GAU-CCUGCCGG	UGCGGCUGGAUCACCUCCU'	AU	UA	AU	CG	GU	CG	-A	CA
17	ACUCCGGUC-GAU-CCUGCCGG	UGCGGAUGGAUCACCUCC?'	AU	GC	UA	CG	GU	CG	-A	GA
18	AUUCUGGUU-GAU-CCUGCCAG	UGCGGCUGGAUCACCUCCU'	GC	GC	UA	CG	GG	CG	-A	CA
19	AUUCCGGUU-GAU-CCUGCCGG	UACGGCUCGAUCACCUCCU'	GA	GC	UA	CG	GC	CG	-A	CA
20	ACUCCGGUU-GAU-CCUGCCGG	UGCGGCUGGAUCACCUCC?'	GA	CG	CG	CG	GG	CG	-A	CA
21	AAACCGGUU-GAU-CCUGCCGG	UGCGGCUGGAUCACCUCC?'	GA	GC	CG	CG	GC	CG	-A	CA
22	GGAGGGUUU-GAU-CCUGGCUC	UGCGGCUGGAUCACCUCCUI	GA	GC	CG	CG	GG	CG	AA	CA
23	UGAGAGUUU-GAU-CCUGGCUC	????????GAUCACCUCCUI	GA	CG	CG	CG	GC	CG	AA	CA
24	UGAGAGUUU-GAU-CCUGGCUC	????????GAUCACCUCCUI	GA	CG	CG	CG	GG	CG	AA	CA
25	UGAGAGUUU-GAU-CCUGGCUC	UG??????GAUCACCUCCUI	GA	CG	CG	CG	GA	CG	AA	CA
26	AUUCCGGUU-GAU-CCUG??GG	UG??????GAUCACCUCC?'	GA	GC	UA	CG	GU	CG	-A	CA
27	AUUCCGGUU-GAU-CCCGCCGG	UG??????GAUCACCUCC?'	GA	CG	CG	CG	GU	CG	-A	CA
28	ACUCCGUUU-GAU-CCUGGCGG	UGCGGCUGGAUCACCUCCU'	GC	CG	CG	CG	GG	CG	-A	CA
29	AGUCCGUUU-GAU-CCUGGCGG	UGCGGCUGGAUCACCUCCU'	GC	UA	UA	CG	GA	CG	-A	CA
30	AAUCCGUUU-GAU-CCUGGCGG	UG??????GAUCACCUCCU'	GC	UA	UA	CG	GA	CG	-A	CA
31	AAUCUGUUU-GAU-CCUGGCAG	UG??????GAUCACCUCCU'	GC	UA	UA	CG	GA	CG	-A	CA

ID	Organism	ID	Organism
1	Methanococcus jannaschii str. JAL-1 (DSM 2661)	17	Thermoplasma acidophilum str. 122-1B2
2	Methanococcus voltae str. PS (ATCC 33273)	18	Archaeoglobus fulgidus str. VC-16 (DSM 4304)
3	Methanococcus vannielii str. EY33	19	Thermococcus celer str. VU 13 (DSM 2476)
4	Methanothermus fervidus	20	Methanopyrus kandleri str. av19 (DSM 6324)
5	Methanobacterium formicicum (DSM 1312)	21	Thermoproteus tenax
6	Methanobrevibacter ruminantium str. M-1	22	Thermotoga maritima str. MSB8 (DSM 3109)
7	Methanobacterium thermoautotrophicum str. Marburg	23	Streptococcus bovis (ATCC 33317)
8	Methanospirillum hungatei str. JF1 (DSM 864)	24	Enterococcus faecalis
9	Methanogenium cariaci str. JR1 (ATCC 35093)	25	Leuconostoc mesenteroides subsp. mesenteroides
10	Methanosaeta concilii str. Opfikon (DSM 2139)	26	Lactobacillus delbrueckii subsp. lactis
11	Haloferax volcanii str. DS-2 (ATCC 29605)	27	Lactobacillus acidophilus (ATCC 4356; NCDO 1748)
12	Haloferax mediterranei (ATCC 33500)	28	Pediococcus pentosaceus (ATCC 33316; DSM 20336)
13	Halobacterium cutirubrum clone lambda-Hc4	29	Lactobacillus brevis (ATCC 14869; NCDO 1749)
14	Halobacterium halobium str. R1	30	Lactobacillus plantarum (ATCC 8014; DSM 20205)
15	Halobacterium marismortui [gene = rrnB]	31	Lactobacillus ruminis (ATCC 27780; DSM 20403)
16	Halococcus morrhuae (ATCC 17082)		

Figure 1-2: Sample of Multiple Alignment Data for 16S RNA

Data extracted from the first and last lines of the 16S RNA multiple alignment [40]. The multiple alignment contains 1380 organisms, each having 2688 columns of RNA data. Each organism contained in the alignment is represented by a single row of data. Most of the data is valid nucleotide data {A,C,G,U}. There are also gaps labeled "-". These are nucleotides are absent for a particular organism. There are also several "?" symbols. These represent positions in the alignment for which the data is uncertain, or for which the columnar structure of the alignment is questionable. The first and last few columns are removed as they contain nearly all ? symbols. In addition to the raw column data, there are four examples of column duos that are known to be paired as well as four column duos that were selected at random. It is noteworthy that column duo Paired-4 strongly resembles column duo Random-2. This is an example of two independently conserved columns seeming nearly indistinguishable from a conserved paired column duo. Such degeneracies limit secondary structure determination accuracy as discussed in sections *3.6 Results Format* and *4.1 Discussion.*

1.2. 3 The Phylogenetic Tree

A phylogenetic tree is a graph showing a set of evolutionary relationships between organisms. The graph consists of nodes, representing organisms, and directed edges, showing evolutionary relationships from each parent organism to its children. For purposes of organizational uniformity, the phylogentic tree is arranged as a binary tree with each node having either exactly two children or no children (*Figure 1-3*)). The

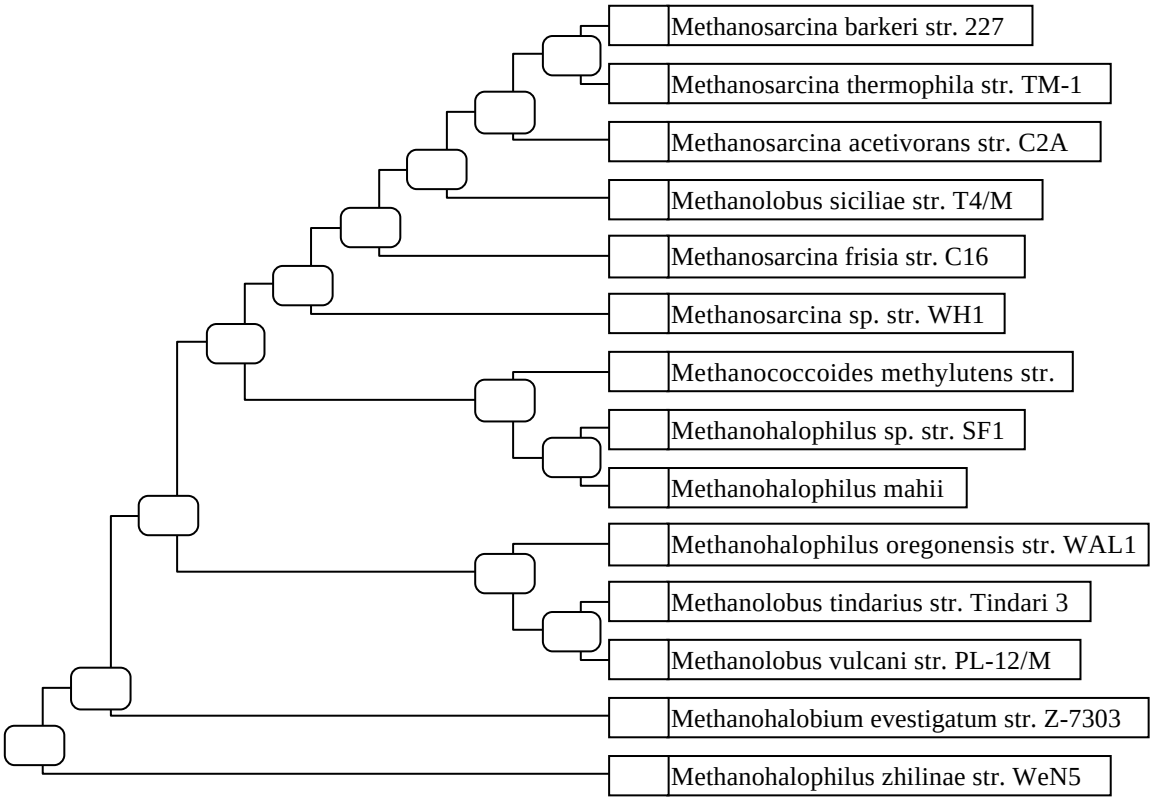

Figure 1-3: Section of a Phylogenetic Tree

This represents a sub-tree section of a full phylogenetic tree. Sequenced organisms included in the multiple alignment are found at the leaves and are identified by rectangular boxes. Internal nodes represent "synthetic ancestors" that have never been seen and are represented by rounded boxes. Due to technical constraints, the organisms in this tree do not correspond to those found in *Figure 1-2: Sample of Multiple Alignment Data for 16S RNA*. This graph was obtained from the RDP [41] and was constructed using the fastDNAML technique of [21]. The branch lengths shown here have no particular significance.

nodes with no children are leaf nodes and correspond to data from the multiple alignment. The internal nodes, which have two children each, represent "synthetic ancestors". The precise interpretation of these synthetic ancestors is not completely clear. While such internal nodes may be seen as representing progenitor organisms that are now extinct, their more general interpretation is that they simply serve to quantize genetic proximity.

Except in those rare cases where the course of an organism's evolution is known *a priori*, the internal elements of a phylogenetic tree are typically inferred from its multiple alignment by measuring the similarity of an alignment's sequences. While conflicting subjective processes in the construction of multiple alignments have led to a substantial amount of contention, the disagreements surrounding heuristic construction of phylogenetic trees can take on a truly internecine character.

1.2. 4 Folding Structure

The final step in the analysis of RNA structure, which is described in *Figure 1-1* *Estimate Structure*, is to look for multiple alignment columns that have statistically dependent nucleotides. For example, in column duo Paired-3 of *Figure 1-2*, we see that each Uracil nucleotide (U) in column 578 is accompanied by an Adenine nucleotide (A) in column 559. While this is some evidence that the two columns are statistically related, it is not strong evidence as this type of behavior could easily be found at random

in columns that are not found to change during evolution[4]. Stronger evidence for an inter-column relationship is found in the consistent covariance of the two columns. When U in column 578 changes to C (Cytosine), we always see a corresponding change of column 559 from A to G (Guanine). These correlated changes (UA→CG) are less likely to occur under an independent (and random) mutation process than if the corresponding columns are related through pair bonding in properly folded molecules. The correspondence of the mutations is therefore a strong indication of statistical dependence. If we examine the phylogenetic tree and find such parallel mutations are found within phylogenetically distal groups, then the evidence for a dependency is extremely strong. This statistical dependence between two columns in a multiple alignment, once established, is interpreted as strong evidence for some sort of structural dependence between the nucleotides inhabiting those columns. Chemical bonding between nucleotides is the primary source of such structural dependencies. As such chemical bonding can not occur between distal nucleotides, the presence of this bonding indicates that the nucleotides are proximal when the RNA molecule is folded *in vivo*. Through the combination the distance constraints imposed by this chemical bonding with *a priori* knowledge of RNA's physical structure, a 3-D structure for an RNA molecule can be formed [22]. The calculation of this 3-D structure completes the *Estimate Structure* module of the structuring process illustrated in *Figure 1-1: Generic Algorithm for RNA Structure Determination.*

[4] This could be the result of a critical structural dependency on the existence of a particular nucleotide in a particular column. If such a nucleotide were to mutate, the target of the mutation would quickly die out, making it vanishingly unlikely that the target's RNA would be contained in the multiple alignment.

The techniques described above summarize the process of RNA structure investigation as illustrated in *Figure 1-1*. As mentioned in *1.1 General Motivation* this process requires tremendous amounts of skilled labor, involves subjective and non-uniform methods and is not feasible for the large amounts of sequence information becoming available. To address these shortcomings, various techniques have been explored to help automate the process of determining: multiple alignments [23][24][25], phylogenetic trees [26][27][21][28] and nucleotide pairing [29][30][18]. While significant progress has been made for both of the first two processes, development of automated methods for the determination of pairing have met with only a limited success.

The present work was designed to fill the need for an effective pairing structure detector. As this method relies on the results from the *Estimate multiple alignment* and *Estimate Phylogentic Tree* modules of *Figure 1-1*, it fits clearly within the *Estimate Structure* module. As more sequence data has become available, it has become plausible to model RNA structure through a statistical description of the known samples. Statistical modeling can help circumvent complexity problems in physical modeling, by strictly limiting the total number of degrees of freedom to those supported by the data. Such statistical techniques model molecular physics indirectly by taking the physical laws as implicit in the evolution of a family of homologous RNA samples. Rather then having to decide which physical degrees of freedom are unimportant, *a priori*, the current work strives to build a statistical model that represents the population from which a data sample is drawn. If this task is successfully completed, then the

relevant physical laws are incorporated implicitly in the model, through the observed data. Impossible physical interactions are not generally seen in data samples, and thus are not incorporated into a model. Irrelevant physical interactions are eliminated as irreducible variance of structureless noise. Such modeling reduces the subjective and potentially dangerous burden of deciding which approximations to make *a priori*.

There have recently been several efforts to employ this type of statistical modeling in the analysis of RNA structure. Initially, such methods were based on the RNA version of nucleotide base pairing, as was posited for DNA by Watson & Crick[5] [19][31]. In RNA the hydrogen bonds which form nucleotides into Watson-Crick pairs[6] are known to have a profound effect on the 3-D structure and function of an RNA molecule. When such bonds are found in a helical configuration, they are referred to as the "secondary structure" of the RNA molecule. Unlike the primary structure (sequence) of an RNA molecule, the *in vivo* secondary structure can not currently be directly measured, only inferred. Early modeling efforts concentrated on maximizing the number of Watson-Crick pairings that could be formed in a multiple alignment. However, these efforts were not successful as numerous other interactions were found to have a dramatic effect on 3-D structure. Such features include: nucleotide loops, nonpaired end caps for helices, non-helical hydrogen nucleotide bonds, ionic bonding between nucleotides and ionic bonding between nucleotides and water. These features are referred to collectively

[5] It may be amusing for the reader to note that in the original 1953 article published by Watson & Crick, they explicitly discounted the possibility of helical structure made from ribose sugars rather than deoxyribose sugars as, "…the extra oxygen atom would make too close a van der Waals contact.".

[6] The hydrogen bonds in Watson Crick base pairs are found between Adenosine-Uracil and Guanine-Cytosine nucleotide pairs.

as the tertiary structure of an RNA molecule. This tertiary structure is believed to be critical to the 3-D structure of RNA, and embodies much of the complexity that is difficult to model *a priori* and has hampered previous efforts. It is precisely such subtle complexity which statistical models such as the Tree Model are designed to accommodate implicitly, rather than *a priori*.

While there have been some substantial results in the area of statistical RNA structure analysis, prior work has rested either on a purely columnar analysis of a multiple alignment, or on a small set of closely related molecules. Both of these types of modeling encounter difficulties because they fail to include important information. Phylogenetic analysis of a small number of closely related molecules does not consider enough of the entire homologous multiple alignment to provide a stable base of statistics. Neither does a small sample include enough evolutionary information to be able to derive a robust model for the general process of nucleic acid mutation, over evolutionary time spans. These limitations may evidence themselves as solution degeneracies, similar to those of the energy minimization technique, or an inability to generalize the model to larger samples of homologous RNA. While frequency analysis of complete multiple alignment columns need not suffer from the phylogentic generalization problem, it does exclude evolutionary relationships completely. The evolution of a nucleotide duo in a multiple alignment can be a critical factor in determining statistical dependency. Two column sets from a multiple alignment can have identical nucleotide frequency distributions, yet have radically differing evolutionary characteristics. These differing

evolutionary characteristics can be the key to discriminating between independent behavior and the dependent behavior that indicates base pairing.

1.2. 5 Tree Model for Structure Detection

The current work overcomes the limitations of algorithms based solely on a multiple alignment, as well those based on a small, genetically related sample. The Tree Model accomplishes this by performing inference on an entire phylogentic tree for each organism in a multiple alignment that consists of thousands of homologous RNA sequences. Thus, the Tree Model is superior to previous work in that it actively and automatically utilizes a far larger amount of the information generated by previous elements of the structure modeling process (*Estimate Multiple Alignment* and *Estimate Phylogentic Tree* modules of *Figure 1-1*).

The Tree Model constructs a complete probabilistic model for the evolution of multiple alignment column duos, using the phylogenetic tree. The phylogentic tree is modeled as a Markov Tree, an extension of Markov processes to tree-shaped state relationships. The parameters of the Markov Tree are estimated over a large fraction of an entire multiple alignment, resulting in a compact, yet general, model for the evolution of RNA nucleotide duos. Once the model parameters have been derived, column duos may be presented to the model to generate an evolutionary model conditioned on that duo. Through Maximum Likelihood inference, the novel column duo, along with the model parameters, serves to fix the nucleotide probability distributions for every leaf in the tree. Measurements of relevant evolutionary quantities can then be generated

through the calculation of an expectation value over the probability distributions. These expectation values can be investigated on a column by column basis, or aggregated across any subset of columns from the multiple alignment to form complete expectation values for the data set. Thus, the Tree Model can provide numerous measurements of the general process of RNA development over evolutionary time spans. In particular, *dynamic programming* is used to efficiently calculate the sum of the posterior probabilities of each possible evolutionary path. This produces a posterior probability for the column duo, given the trained Tree Model. As Tree Model parameters are extracted from a training set of multiple alignment column duos, Tree Models can be constructed to reflect any desired evolutionary characteristic for which a training set exists. Two training sets are thus constructed, one from column duos that are known to be paired (Pair) and another from randomly selected column duos with known column pairs excluded (Rand). For every novel column duo, posterior probabilities are calculated according to each model. These probabilities can then be compared to determine whether or not the column duo is paired. The current work demonstrates such a detector that is found to have a validation set misclassification rate of less than 10%, which declined to less than 1% on some filtered data. This represents a marked improvement over previous secondary structure detectors based on energy minimization and heuristic phylogeny comparison that have demonstrated nucleotide pair misclassification rates ranging from 20% to 30% [15][32].

2 Theory

In this chapter we derive the specific equations needed to implement the Tree Model. The chapter has four sections. In *2.1 Theoretic Overview* we provide a high-level description of the notation and techniques that will be derived in the remainder of this chapter. In *2.2 Frequency Model* we derive a simple null model called the Frequency Model. This derivation will further familiarize the reader with the notation and concepts employed in later sections. In *2.3 Tree Model Topology* we develop the Markov Tree, a Markov process on a tree shaped state structure, which is central to the probabilistic modeling of phylogenetic trees in the present work. In the final section, *2.4 Mutation Models*, we discuss three point-mutation models that are used to model local evolutionary relationships between states of the Markov Tree: Q, IO and IOM.

2. 1 Theoretic Overview

In this section we discuss the background required to understand and use the Tree Model. First, we develop a general description of the notation used to describe the data. Second, we present an overview of the sample Frequency Model. Third, we provide a high level description of the Tree Model algorithm. Finally we furnish a brief discussion of discrimination techniques and over-fitting concerns.

The multiple alignment from *Figure 1-2: Sample of Multiple Alignment Data for 16S RNA*, serves to show the kind of raw data that we have to work with. The RNA

sequence for each organism contributes one line to the multiple alignment. A column duo (d) may, thus, be viewed as a length-S vector of nucleotide duos (d^s), where S is the number of organisms represented in the multiple alignment and $1 \leq s \leq S$. Each column in an alignment is also numbered, allowing the terse representation of a column duo by a duo of column ID numbers. This representation is used to construct two similarly sized sets of column duos. The first set contains duos that are known to be paired. The second set contains duos that are selected independently and at random from the entirety of the multiple alignment. All column duos that are known column pairs are immediately removed from the second set. The specific question we are to answer is whether a column duo is more likely to have been drawn from the population that generated the paired sample, or the population that generated the nonpaired sample.

The simplest model evaluated here is the Frequency Model. This model does not make use of the phylogenetic tree. Since the Frequency Model only examines the distribution of nucleotides in a novel column duo d, and not the genetic relationships of their contributing organisms, this model is used as a null model against which the more sophisticated phylogentic tree based models are measured. The Frequency Model is developed using the same notation that is used for deriving the more complex models. It is hoped that the derivation of this relatively simple model will help the reader become more comfortable with our notation, facilitating the comprehension of the more demanding derivations of the IO Model.

The Tree Model is a general, statistical model of the evolution of RNA based on a phylogenetic tree. However, the formal derivation of Tree Model statistics is performed

within the context of base pairing detection. Though the resultant algorithms are generalizable to other purposes, this work will tailor them to the determination of base pairing in column duos of a multiple alignment.

The Tree Model uses a number of free parameters that control its statistical treatment of evolution. These parameters are collectively referred to as the mutation model, or simply as the Model. The parameters of the Model are extracted from a given set of training column duos D_{train}. The training process seeks to manipulate the Model parameters so as to maximize P(D_{train}|Model). Each Model is tuned to detect membership in a particular population of data. To perform discrimination between several populations of data, a sample is taken from each population, and a model tailored to that sample. As we are primarily concerned with discrimination between paired column duos and nonpaired column duos we need only choose two training samples. The first training sample is drawn from the population of column duos that are known to be paired (Pair), and the second at random from column duos not known to be paired (Rand). The models that are trained with these data sets are deferred to as $\text{Model}_{\text{Pair}}$ and $\text{Model}_{\text{Rand}}$. To classify a novel column duo *d*, P(*d*|$\text{Model}_{\text{Pair}}$) and P(*d*|$\text{Model}_{\text{Rand}}$) are compared. Column duo *d* is then assigned to the population whose corresponding model produced the higher likelihood[7] for *d*.

[7] A likelihood comparison is used here instead of a posterior probability comparison. This substitution is made because likelihoods are easier to calculate and the similarity in test set size for Rand and Pair produces a negligible difference between the two comparisons. See *2.2.2 Discrimination* for more details on this issue.

To evaluate the accuracy of this classifier, each complete set of Rand and Pair column duos is broken into two disjoint groups, D_{train} and D_{test} . The D_{train} group is used to establish a Model's parameters, while D_{test} is used as cross validation to evaluate the trained Model and to measure over-fitting.

2. 2 Frequency Model

This section provides a theoretical description of the Frequency Model. It is broken into three parts. In *2.2.1 Derivation of Frequency Model*, we introduce the notation, motivation and construction of the model. By the end of the first part we have derived a method for obtaining posterior probabilities for a given column duo according to both a random model and a paired model. In *2.2.2 Discrimination*, we discuss some considerations regarding the use of these probabilities to classify the duo. Finally, in *2.2.3 Motivation for Markov Trees*, some theoretical weaknesses of the Frequency Model are discussed. A desire to overcome these weaknesses provides a motivation for the development of a more complex model, the Tree Model. In addition, section *2.3.4 Notation Summary* provides a comprehensive listing of the notation used in the following derivations as a reference aid.

2.2. 1 Derivation of Frequency Model

The Frequency Model calculates column duo probabilities P(*d*|Model), based solely on the distribution of nucleotide duos in the column duo (*d*) [10]. It is thus insensitive to the ordering of the nucleotide duos within a given column duo. To train this model, a probability distribution over the possible nucleotide duos is generated from

a count of the number of each type of nucleotide duo in D_{train}. The data likelihood P(d|Model) is then calculated as the product of the individual probabilities for each nucleotide duo in d. Given a numbering of the S organisms composing the rows of a multiple alignment (s:1≤s≤S), we can refer to each organism's contribution to a column pair d as d^s. Thus, d^s represents the specific nucleotide duo contributed by organism s to column duo d. Treating each nucleotide duo as an independent observation, the probability P(d|Model) can be calculated as $\prod_{1\le s\le S} \mathrm{P}(d^s|\mathrm{Model})$.

In turn, P(d^s|Model) is generated from the renormalized distribution of observed nucleotide duos in all of the column duos of D_{train}. To obtain these model probabilities we first define a frequency distribution over the 16 possible nucleotide duos 0≤l≤15, $\hat{\phi}_l = \sum_{d\in D_{train}} \sum_{d^s\in d} \left[1 \text{ iff } d^s = l,\ 0 \text{ otherwise}\right]$. We can then define[8] P(d^s|Model) ≡ $\varphi_l = \hat{\phi}_l \Big/ \sum_l \hat{\phi}_l$. The distribution φ_l is, thus, the Maximum Likelihood estimate of P(d^s=l|D_{train}).

There are on the order of 1000 organisms in the multiple alignment used in this work. If we assume that the nucleotide duos are uniformly distributed in the multiple alignment, we might expect to see values for P(d|Model) of $\prod_{1\le s\le 1000} \mathrm{P}(d^s|\mathrm{Model}) \approx \prod_{1\le s\le 1000} (1/16) = (1/16)^{1000}$ or approximately 10^{-1200}. As numbers of such magnitude exceed the native floating point precision of most currently available digital computers, it is

[8] If the amount of data is small, or some transitions are found to have nearly 0 frequency, then a more sophisticated technique such as the Laplacian Estimator might be used to convert frequency to probability. This would take into account the finite amount of data used in the calculation. Such a correcting factor was not considered necessary in the current experiments.

convenient to adopt a formalism from information theory to represent such small numbers by their negative log likelihood (NLL). The NLL value of a probability p is $-\log_2(p)$. So long as we use the logarithmic base of 2, this NLL has an information theoretic interpretation as the mean number of bits required to encode an event of probability p. Typical NLL values for the Frequency Model might then be on the order of $-\log_2((1/16)^{1000})$ or about 4000 bits for a column duo. To further reduce this to a more intuitive level, the NLL value is normalized against the number of nucleotides found in valid[9] nucleotide duos in the column duo. For our "typical" column duo of 1000 nucleotide duos this leaves us with 4000 bits / (2×1000) nucleotides or about 2 bits/base[10]. The value of 2 bits/base might seem familiar. One of the simplest representational models for RNA sequences symbolizes each of the 4 types of nucleotides by placing them in correspondence with the 4 possible combinations of 2 bits.

When representing Frequency Model values of P(d|Model) as NLL values, we can rewrite our equation of $P(d|\text{Model}) = \prod_s P(d^s|\text{Model})$, as $\text{NLL}(P(d|\text{Model})) = -\log_2(\prod_s P(d^s|\text{Model})) = -\sum_s \log_2(P(d^s|\text{Model}))$. Alternatively, if there are n_l of each nucleotide duo ($0 \le l \le 15$) in column duo d, then P(d|Model) could also be written as $-\sum_l [n_l \cdot \log_2(\varphi_l)]$, which is far more computationally efficient. Now that we have

[9] A nucleotide duo is considered valid if both of its nucleotides are elements of {A ,C, G, U}. Nearly all column duos contain some invalid characters ("-" or "?"). Nucleotide duos containing one or more of these invalid characters are ignored by both the posterior probability calculations and the NLL normalization.

described how to train a Frequency Model from a set of column duos (D_{train}), and to obtain the posterior probability P(d|Model) for a novel column duo d, we may investigate applications for such probabilities.

2.2. 2 Discrimination

In order to perform discrimination for a model class (in this case, the Frequency Model) we need to tailor a set of model parameters for each classification category. Our present goal is the separation of paired multiple alignment column duos from nonpaired duos. We are given representative samples of each column duo population, namely Pair from the paired population (Pop_{Pair}) and Rand from the nonpaired population (Pop_{Rand}). Each of these samples is broken into a training set (D_{train}) and a validation set ($D_{test.}$). We then use the training data to configure two sets of model parameters. The first set of model parameters is calculated using D_{train}(Pair) and the second set is calculated using D_{train}(Rand). The models employing each parameter set will be referred to as $Model_{Pair}$ and $Model_{Rand}$ respectively. Given these models and a novel column duo d, which we wish to classify, we compute the data likelihoods P(d|$Model_{Pair}$) and P(d|$Model_{Rand}$) as described above. Classification then proceeds by comparing these probabilities and assigning d to the column duo population whose model generates the larger data likelihood (or lower NLL value)[11], that is $d \in$ Pair iff P(d|$Model_{Pair}$) > P(d|$Model_{Rand}$) and $d \in$ Rand otherwise.

---

[10] The term "base" is used interchangeable with the term "nucleotide" in this work.

[11] The larger probability corresponds to a smaller NLL value as $p_1 > p_2 \rightarrow \log_2(p_1) > \log_2(p_2) \rightarrow -\log_2(p_1) < -\log_2(p_2) \rightarrow NLL(p_1) < NLL(p_2)$.

This classification scheme is not in strict accordance with the Bayesian model, as it compares data likelihoods (P(*d*|Model)) rather than posterior model probabilities (P(Model|*d*)). This substitution of likelihoods for posterior probabilities is not general. However, in the present work, likelihood comparison provides a very good approximation of posterior probability comparison.

When we use a likelihood based classifier, we are effectively asking the, "Which data population, Pop_{Rand} or Pop_{Pair}, would be more likely to include *d* ?". However, this is not exactly the question we want to answer. We really want the answer to the question, "To which population is *d* more likely to belong?". This second question is analogous to classification based on the models posterior probability, P(Model|d). In order to answer the second question, we need some information regarding the relative sizes of Pop_{Rand} or Pop_{Pair}. This need is demonstrated by the following example.

Let us choose two columns from the multiple alignment at random and call them column duo *d*. Assume our models produce $P(d|Model_{Pair})=.99$ and $P(d|Model_{Rand}) = .003$. As these data likelihoods come from differing models, their probabilities need not sum to 1. Given no other information about the population from which *d* is drawn (Pop_d), we would probably argue that *d* is a paired duo. However, we have some information about Pop_d. Namely, we are told that *d* was selected at random from the columns of the multiple alignment.. As there are 2,688 columns in the multiple alignment, there are at most 1,344 paired column duos. This leaves approximately $2{,}688^2$-1,344 = 7,224,000 possible unpaired column duos. Given that the populations which generated Rand and Pair (Pop_{Rand} and Pop_{Pair}) are disjoint ($Pop_{Rand} \cap Pop_{Pair} = \varnothing$)

and complete ($\forall d,\ d \in \text{Pop}_{\text{Rand}} \cup \text{Pop}_{\text{Pair}}$), it seems *a priori* that *d* is overwhelmingly likely to have come from Pop_{Rand} with a probability of 7,224,000/(7,224,000+1,344) = 99.98%. The task left to us is how to combine this *a priori* information about Pop_d, with P(*d*|Model) to figure out population *d* is more likely to have come from (P(Model|*d*)). Bayes rule provides the solution to this problem in its statement of the relation:

$$P(\text{Model}|d) = P(d|\text{Model}) \cdot P(\text{Model}) / P(d).$$

We have been given $P(d|\text{Model}_{\text{Rand}})$ & $P(d|\text{Model}_{\text{Pair}})$. The probability P(Model) represents the a priori probability of *d* being drawn from Model, given only our information about Pop_d. The quantity P(*d*) represents the overall probability of observing *d*. Given that the two populations Pop_{Rand} and Pop_{Pair} are disjoint and complete, P(*d*) may be calculated as:

$$P(d) = P(d|\text{Model}_{\text{Rand}}) \cdot P(\text{Model}_{\text{Rand}}) + P(d|\text{Model}_{\text{Pair}}) \cdot P(\text{Model}_{\text{Pair}}).$$

For the present example, the posterior probabilities may now be calculated using Bayes Rule as follows.

$P(\text{Model}_{\text{Rand}})$	= 7,224,000/(7,224,000+1,344)	= .9998
$P(\text{Model}_{\text{Pair}})$	= 1,344/(7,224,000+1,344)	= .0002
$P(d\|\text{Model}_{\text{Rand}})$		= .003
$P(d\|\text{Model}_{\text{Pair}})$		=.99
P(*d*)	= .9998·.003 + .0002·.99	= .0031974
$P(\text{Model}_{\text{Rand}}\|d)$	= .9998·.003/.0031974	= 93.8%
$P(\text{Model}_{\text{Pair}}\|d)$	= .0002·.99/.0031974	= 6.2%

Because the two populations are disjoint and complete;

$$P(Model_{Rand}|d)+P(Model_{Pair}|d)$$

must sum to 1. The result that $P(Model_{Rand}|d) > P(Model_{Pair}|d)$ clearly indicates that *d* should be classified as having come from Pop_{Rand} rather than Pop_{Pair}. The overwhelming preponderance of random nucleotide duos in our Pop_d has clearly outweighed the likelihood's indication that *d* comes from Pop_{Pair} .

The above example, while informative, seems to counter our presumption that a likelihood comparison is acceptable for the present work. The example is presented to build an intuition for the type and magnitude of effects that might be invoked through the use of likelihood comparisons rather than posterior probability comparisons. In the present work, preliminary calculations were performed over similar sized data sets of 695 Rand column duos and 634 Pair column duos (*3.3 Preliminary Q Model Study*). In this preliminary work, over 99% of the column duos had data likelihoods that differed by more than a factor of two, and over 90% differed by more than a factor of 1,000. In the light of these overwhelming likelihoods, the prior probabilities on the order of 634/(634+695) ≈ 48% (Pair) and 695/(634+695) ≈ 52% (Rand) were deemed negligible.

When the data was again filtered, after the preliminary model calculations (*3.4 Secondary Data Preprocessing*), the size of the Pair data set was effectively halved. This resulted in a change in the model priors to $P(Model_{Rand}) = 695/(317+695) \approx 69\%$ and $P(Model_{Pair}) = 317/(317+695) \approx 31\%$. These prior probabilities were still considered negligible for the final experiments. This assumption was borne out by a

sample posterior probability calculation that showed the difference of less than 0.05% accuracy between simple discriminators based on posterior probability and likelihood (*6 Appendix B: Posterior Probability Classifier for IO*).

Nonlinearities in the preliminary Q Model results indicated that the simple probability comparison described above might not be an effective classifier. Thus, a neural network model was also used to determine column duo classifications. To preserve data integrity, this classifier was not trained on the validation data. This nonlinear classifier was found to reduce classification error in the IO Model from 11.2% for the simple classifier 11.2% to 9.3%. Details regarding the design, training and use of this discriminator are provided in section *3.5 Classifiers*.

2.2. 3 Motivation for Markov Trees

The Frequency Model is quite simple. To obtain this simplicity, a number of questionable assumptions are made. Potentially, the most crippling of these assumptions is the assumption that each of the *S* nucleotide duos that compose *d* are independent. For this independence assumption to be true, each organism that contributes to a column duo in a multiple alignment must be drawn randomly and independently from some stationary statistical distribution. Given the RNA sequence similarities between related families of organisms, this proposition is absurd.

The inaccurate assumption of statistical independence between nucleotides leads to two important flaws in this model. First, the assumption of independence can lead to unwarranted statistical biases due to statistically dependent clustering of the data.

Second, the denial of a dependent structure prevents the Frequency Model from specifically exploiting data dependencies for modeling purposes.

The multiple alignment is constructed from the sequences of numerous related organisms. These organisms tend to be found in genetically similar family groups. Within one group, there may be far less nucleotide sequence variance then between two differing families. The decision of how many closely related organisms from each family to include in a multiple alignment is not necessarily based on a desire to provide a statistically balanced sample over the multiple alignment. Organisms may be included in a multiple alignment due to their availability, phenotype, or their potential use to the sequencing party. This can provide a very biased sample of nucleotides from which to build a model. No model that is derived from a training set of data can be free of systematic sample set biases in the choice of training set. However, the Frequency Model's reliance on independence between nucleotide duos exacerbates its sensitivity to duo dependence by counting each contribution to the training set as equal, and completely ignoring the family bias problem.

Rather then being a problem, family sequence biases can be exploited to increase modeling accuracy. The Frequency Model is barred from any such modeling by its assumption of independence. The Frequency Model's probability calculations reduce to a summation over the number of nucleotide duos of each type. This prevents the exploitation of any information about the location of nucleotide duos within a column duo. If phylogenetic family groups are clustered together in the multiple alignment, then we expect to see ranges of similarly distributed sequences. However, the one

dimensional listing of organisms required by the multiple alignment is inadequate for the branching structure of family relationships. These family relationships are represented more accurately by the phylogenetic tree.

The following example demonstrates how a model that takes into account the phylogenetic relationships between organisms can serve to increase model accuracy over the Frequency Model. For this example, we limit our nucleotide duos to AU and GC only. The multiple alignment consists of four organisms numbered 1, 2, 3, and 4 four multiple alignment columns. The four columns are clustered into two column duos labeled Case 1 and Case 2. As the nucleotide duo frequency distribution is identical in both duos (50% AU and 50% GC), any discriminator based solely on the frequency

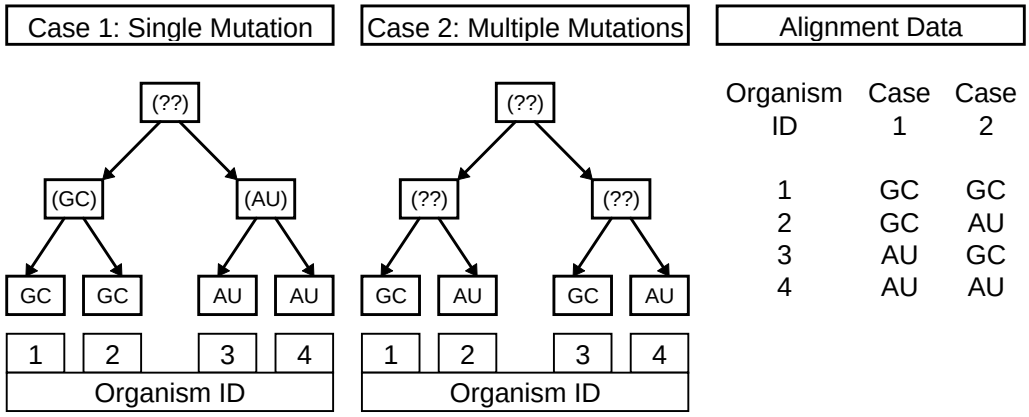

Figure 2-1: Exploiting Phylogenetic Distribution Bias

Examples of data that are irresolvable under models based solely on nucleotide duo frequency distributions. These examples, however, are quite distinguishable under a model that also takes into account the evolution of the column duo. Each leaf represents a single organism's contribution to a multiple alignment column duo. Parenthetical base duos are estimated from children, while leaf duos are directly observed data. A "?" represents an unknown base. Each case might represent a different column duo from a multiple alignment with Case 2 showing strong evidence of base pairing and Case 1 showing weak evidence for such pairing. Arrows represent the direction of evolution.

statistics would have to classify the two cases identically. However, a model taking into account the evolution of the nucleotides may come to a very different conclusion.

The above graph represents an imaginary phylogenetic tree with four organisms in it. The leaf nodes of each tree represent the nucleotide duos that compose a single multiple alignment column duo of length four (S=4). Each internal node represents the common ancestor of its child nodes and the arrows represent evolutionary decendency. The genetic makeup of the ancestors represented by the internal nodes is unknown and can only be inferred from that of their descendants.

The Case 1 data is likely to have been generated by a single mutation from the unknown root ancestor. While we can not be certain that only one mutation occurred, having two children with the same nucleotide duo is strong evidence that the parent shared that duo as well. In contrast, the tree shown in Case 2 can not be generated without at least two separate mutation events. As there is a tendency for a Watson-Crick (AU and CG) paired nucleotides to mutate to another Watson-Crick pair, the independent observation of two such mutations is stronger evidence of pairing than the observation of a single such event. Thus, we would expect that Case 2 would be more likely to be classified as paired then Case 1.

The Tree Model is developed to quantitatively exploit the evolutionary structure displayed in *Figure 2-1* by modeling the evolutionary relationships between organisms' RNA sequences. To this end we construct the Markov Tree, which serves as the central engine of the Tree Model. The Markov Tree represents the process of evolution as a

Markov process originating from the unknown common ancestral "root" of a phylogenetic tree and terminating with the known organisms in the "leaves" of the tree. We expect to see the Tree Model outperform the Frequency Model in two particular ways. First, we expect an increase in the ability of the Tree Model to represent the population from which the training set is drawn. This would be observed as a higher P(D_{test}|Model) for the Tree Model then for the Frequency Model. Second, we expect to see better differentiation between $\text{Model}_{\text{Rand}}$ and $\text{Model}_{\text{Pair}}$ for the Tree Model then for the Frequency Model. This will be observed as a higher accuracy in the discrimination between paired and nonpaired column duos.

2. 3 Tree Model Topology

This section is developed in five subsections. In *2.3.1 Phylogenetic Tree* we introduce and describe the phylogenetic tree. In *2.3.2 Markov Model* we review the properties of Markov Models that are relevant to the derivation of the Markov Tree. In *2.3.3 Markov Tree* we develop the Markov Tree by applying the inference methods of Markov processes to the topology of the phylogenetic tree. In *2.3.4 Notation Summary* we provide a concise notation summary for easy reference. Finally, in *2.3.5 Tree Model Sample Calculation*, we calculate an example of discrimination using the Markov Tree. This example includes a complete sample computation of a posterior probability for a column duo, P(*d*|Model).

2.3. 1 Phylogenetic Tree

As we will describe a mathematical model based on the evolutionary relationships embodied in a phylogentic tree, it is important to develop both an intuitive understanding of the tree (*Figure 2-2*) and a convenient notation with which to describe the model. A phylogenetic tree is a directed graph representing the evolution of all known organisms from a single progenitor organism. This graph conforms to the combinatoric definition of a tree in that it is completely connected, and contains no cycles. In particular, a

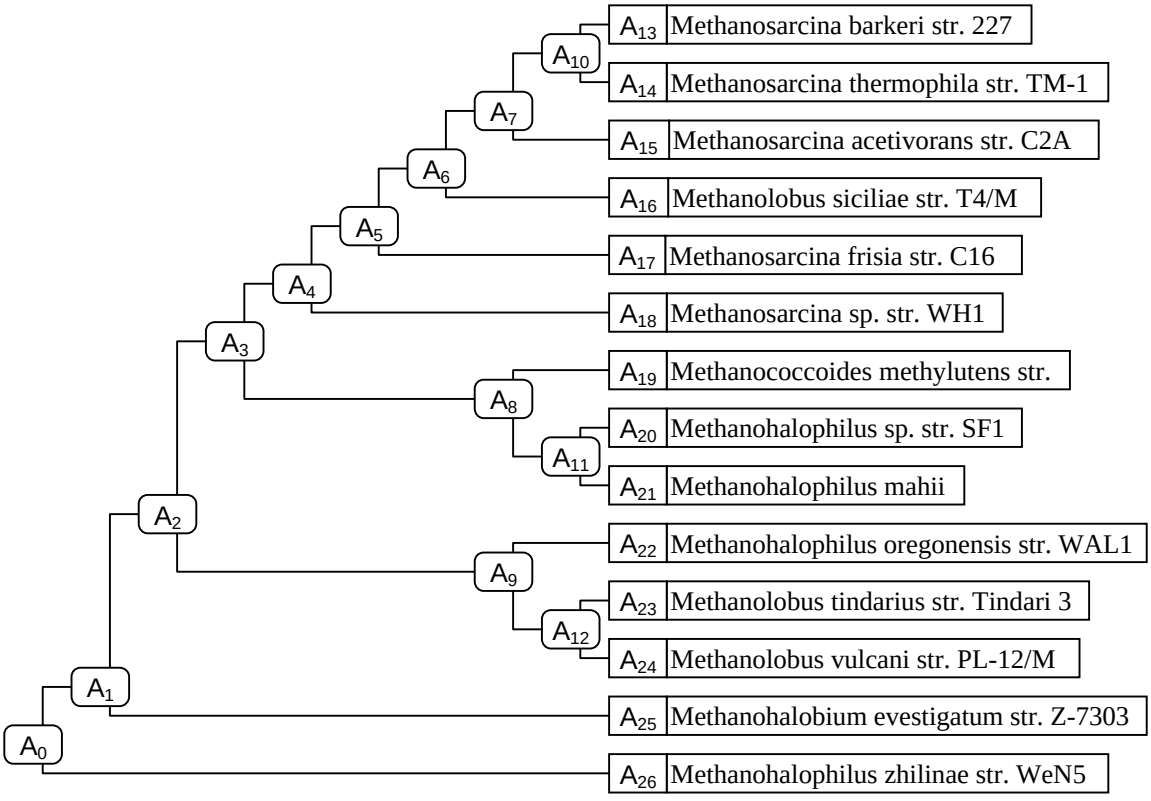

Figure 2-2: Nodal Notation for Phylogenetic Tree

This represents a sub-tree of a full phylogenetic tree [41]. This figure represents the same organisms as *Figure 1-3: Section of a Phylogenetic Tree*, however, in *Figure 2-2* nodes are labeled to demonstrate the notation. The root node A_0 represents the primeval ancestor from which all of the life represented in this tree descended.

phylogenetic tree is a binary tree where each node in the tree has either two descendants or none. Nodes with two descendants are referred to as "internal nodes", while nodes with no descendants are referred to as "leaf nodes".

In a phylogenetic tree, all of the known organisms are found at the leaves. Internal nodes represent "synthetic ancestors". While these synthetic ancestors might be taken as organisms that are believed to have existed but have not been observed, they more generally represent some degree of unexplained commonality between their children. Synthetic ancestors serve to group genetically similar organisms into proximal areas of the tree.

One objection to this scheme is that it does not allow a known organism to be a direct descendent of another known organism. There is strong biological evidence that the ancestor species of some currently living species still persist. This shortcoming is overcome when the concept of branch length is introduced later with the IOM Model *2.4.4 IOM Model.* At that point it will become clear that the phylogenetic tree described above can also represent such ancestral relationships.

Notationally, each node in the tree is given a unique label[12], A_i. A given the multiple alignment column duo fixes the nucleotides at the leaf nodes. The internal nodes represent random variables that could take on the value of any possible nucleotide

[12] As the phylogenetic tree is a binary tree, the number of total nodes and the number of branches are fixed by the number of leaves. If there are x leaves, then there are x-1 internal nodes, 2x-1 total nodes and 2x-2 branches. When nodes are labeled A_i, i will range from 0 to 2x-2, inclusively.

duo. The probability distribution over nucleotide duos for each internal node is determined by the data at the root and leaf nodes, as described in *2.3.3 Markov Tree.*

2.3. 2 Markov Model

The statistical inference used to develop the Tree Model derives from the calculation techniques used in Markov Models. As such, we digress into a brief discussion of Markov processes. We will subsequently employ existent formalisms and intuitions about Markov models in our Tree Model construction. A full treatment of Markov processes is not given here, see [33] for a tutorial. Instead we focus on the aspects of first-order Markov models that we will employ in the derivation of the Markov Tree. These characteristics include initial state assumptions, limited memory capacity, state transition probabilities and limited statistical independence between states.

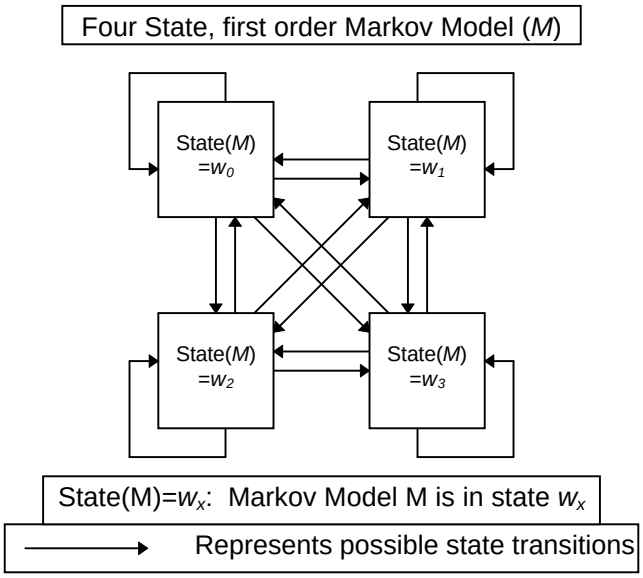

Figure 2-3: First Order Markov Model

Example of a first-order Markov model *M*. The model has 4 states w_0 to w_3 and is completely connected in that any state can transition to any other state with some (possibly 0) probability.

A first-order Markov model M (*Figure 2-3*) is generally defined as a mathematical 4-tuple $\{W, X, Y, Z\}$ where W is a set of states ($|W| = 4$ in this case), X is set of possible transitions between states ($X \subseteq W \times W$) between states, $Y \in W$ is an initial state and $Z \subseteq W$ is a set of terminal states. Notationally, M_t is a random variable over W indicating the state of M at time t. Thus, $P(M_t=w_l|X,Y,Z)$ represents the probability that the model M is found in state w_l at integral time step $t>0$. At $t=0$ the initial state may be defined deterministically as $M_0 = Y$, or probabilistically as $P(M_0=w_l) = P(Y=w_l)$. The transition matrix $X_{l,m}$, represents the probability per unit time that M in state w_l will be found in state w_m one time step later, or:

$$X_{l,m} \equiv P(M_{t+1}=w_l|M_{t+1}=w_m,Y,Z).$$

Given a probability distribution over states at some time t, $P(M_t=w_l)$, we can calculate the state probability distribution at time $t+1$ as:

$$P(M_{t+1}=w_l) =$$
$$\sum_m \left[P(M_{t+1} = w_l | M_t = w_m) \cdot P(M_t = w_m) \right] =$$
$$\sum_m \left[X_{l,m} \cdot P(M_t = w_m) \right].$$

This is referred to as the Markov induction property.

The above inductive step of the Markov model embodies several features that are critical to our later derivations. First, the initial state at $t=0$ must be defined in order to determine the state probabilities at a later time. In the first-order Markov model this initial state distribution is given as part of the model. Second, this model has only a limited memory capacity. The only contextual information passed from one time period to the next is the state of the system itself. Third, probability distribution over the

possible values for M_{t+1} ($P(M_{t+1}=w_m)$) is completely determined by the probability distribution over M_t ($P(M_t=w_l)$) and the model parameter $X_{l,m}$ ($P(M_{t+1}=w_m|M_t=w_l)$). It does not matter which route through the model states was taken to get to the state distribution $P(M_{t+1}=w_m)$. This property, while relatively obvious, is crucial to computational efficiency. Otherwise we might have to maintain a set of possible path histories that could grow exponentially with increasing t. Finally, the state transition matrix is a relatively compact ($|W|\times|W|$) structure that embodies all of the dynamic behavior of the system. If we were interested in the state distribution Δt time steps in the future, we could raise the transition matrix X to the power of Δt and apply it to the current state distribution to obtain the state distribution at Δt time steps in the future. Also, future state distributions in a Markov model have a limited form of statistical independence over time. That is, given a state probability distribution over M_t, $P(M_t=w_l)$, the distribution (over n),

$$P(M_{t-1}=w_n|M_t=w_l \wedge M_{t+1}=w_m) = P(M_{t-1}=w_n|M_t=w_l)$$

and

$$P(M_{t+1}=w_m|M_t=w_l \wedge M_{t-1}=w_n) = P(M_{t+1}=w_m|M_t=w_l)$$

While this characteristic may seem a trifling extension of the limited memory property of Markov models, its analogous implications for the Tree Model will be imperative for computational viability.

2.3. 3 Markov Tree Synthesis

We now have described the biological phylogenetic genetic tree (*2.3.1*), and the Markov process we intend to implement on that tree (*2.3.2*). Now we combine them to derive the Markov Tree that plays a critical role in our development of the Tree Model.

Our treatment of the probabilities derived from the Markov Tree will be identical to the treatment of the probabilities P(*d*|Model) from *2.2 Frequency Model.* However, now we use a more sophisticated model that takes into account phylogenetic dependencies between organisms. We will thus be replacing the likelihood P(*d*|Model) generated by the Frequency Model with P(*d*|**tree**∧Model). Separate versions of the mutation model, Model, will be trained as $\text{Model}_{\text{Rand}}$ and $\text{Model}_{\text{Pair}}$. These will be used to

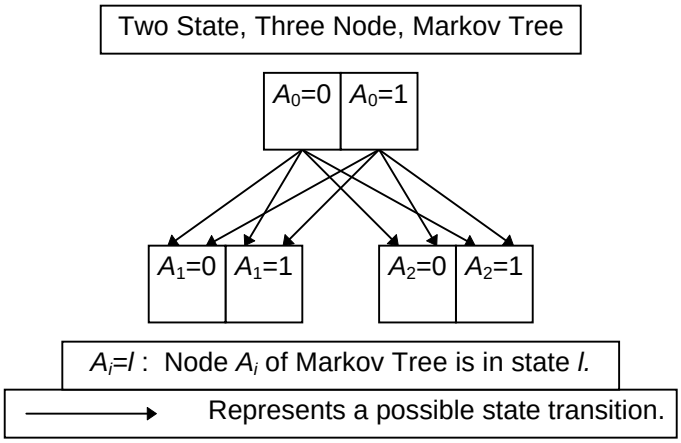

Figure 2-4: Markov Tree Model

Only a limited number of state transitions are allowable in the Markov Tree. The Markov Tree can have no cycles. States in the Markov Tree are organized into nodes where each node (A_i) represents a discrete random variable. In this figure, each random variable can take on the value 0 or 1. In a biological application, each possible value for the random variable might correspond to a nucleotide duo such as AU or GC. As all nodes in the tree represent data $P(A_i=0)+P(A_i=1) = 1$ for all A_i. Another unusual feature of the Markov Tree is that state transitions can only take place from the states of a parent node to the states of its child nodes.

produce $P(d|\text{Model}_{\text{Rand}})$ and $P(d|\text{Model}_{\text{Pair}})$ that will be compared to form a pairing discriminator.

For the Markov Tree derivation, it is convenient to draw several analogies between the Markov Tree and the Markov Model. It is also useful to limit the scope of the derivation to calculations on a single, given, column duo d of the multiple alignment. This assignment fixes the contribution of each organism (s) to the tree to a single nucleotide duo $d^s \in d$. These duos are found at the leaf nodes of the tree. Each tree node A_i corresponds to a discrete random variable over the 16 possible nucleotide duos. This is conveniently thought of as each A_i having 16 possible states, as we can then depict the probability distribution over the states l ($0 \le l \le 15$) of A_i as $P(A_i=l)$. Probabilities conditioned on $A_i=l$ can also be represented. One of the most important of these[13] is $P(d(A_i)|A_i=l)$ which represents the probability of all column duos contributed by organisms in the phylogenetic tree that are descended from A_i ($d(A_i)$), given that A_i is in state l.

Our goal is to compute $P(d)$, the likelihood of all of the nucleotide data at the leaves of the tree. If we knew $P(d|A_0=l)$ for each l, then we could easily calculate $P(d)$, since[14] $P(d) = \sum_l \left[P(d(A_0)|A_0 = l) \cdot P(A_0 = l) \right]$. As it is relatively easy to derive a value for $P(A_0=l)$; we will tackle that first, before the more complex calculation of

[13] Conditioning on the phylogenetic tree and mutation model parameters is not explicitly expressed in the clause $P(A_i=l|d)$. However, under the Tree Model, this probability does presuppose knowledge of both the tree and the mutation model. To avoid the cumbersome necessity of writing our $P(A_i=l|d \wedge \text{tree} \wedge \text{model})$, our remaining references to probabilities will presuppose conditioning on the tree and model. This does not affect our essential mathematics but does simplify our notation.

[14] As $A_i = A_0$ (the root node), $d(A_i)$ is all of the data descended from the root node that is all of the data, d.

$P(d(A_0)|A_0=l)$. We take as our *a priori* state probability distribution, the same nucleotide duo distribution generated by the Frequency Model. This distribution was produced from a renormalized count of the nucleotide duos in the training set D_{train} and represented by $P(A_0=l) = \varphi_l$. Our probability estimate $P(d)$ now becomes $\sum_l \left[P(d(A_0)|A_0 = l) \cdot \varphi_l\right]$. However, this derivation still requires knowledge of $P(d(A_i)|A_i=l)$, for the root node where i=0.

We next derive some notation that we will need to break $P(d(A_0)|A_0=l)$ down into an iterative calculation that will terminate in the leaves of the tree. The above chart (*Figure 2-5*) shows the relationship among three specific nodes (A_i, A_j and A_j), the root node A_0 and the column duo d where $d = d(\text{-}A_i) \wedge d(A_i) = d(\text{-}A_i) \wedge d(A_j) \wedge d(A_k)$. In this figure, the Markov Tree is partitioned into several distinct sections where A_i is the direct parent of A_j and A_k. The symbol $d(A_j)$ represents all of those nucleotide duos $d^s \in d$ that are contributed by organisms descended from node A_j, while $d(A_k)$ represents those duos descended from A_k . In addition $d(\text{-}A_i) = d(\text{-}A_i)' \wedge d(\text{-}A_i)$" represents the nucleotide duos from all of those branches of the phylogenetic tree that are "outside" of the sub-tree whose root is at A_i. If i=0, then A_i is the root node A_0 and $d(\text{-}A_i)$ is the null set, as all data is descended from the root node.

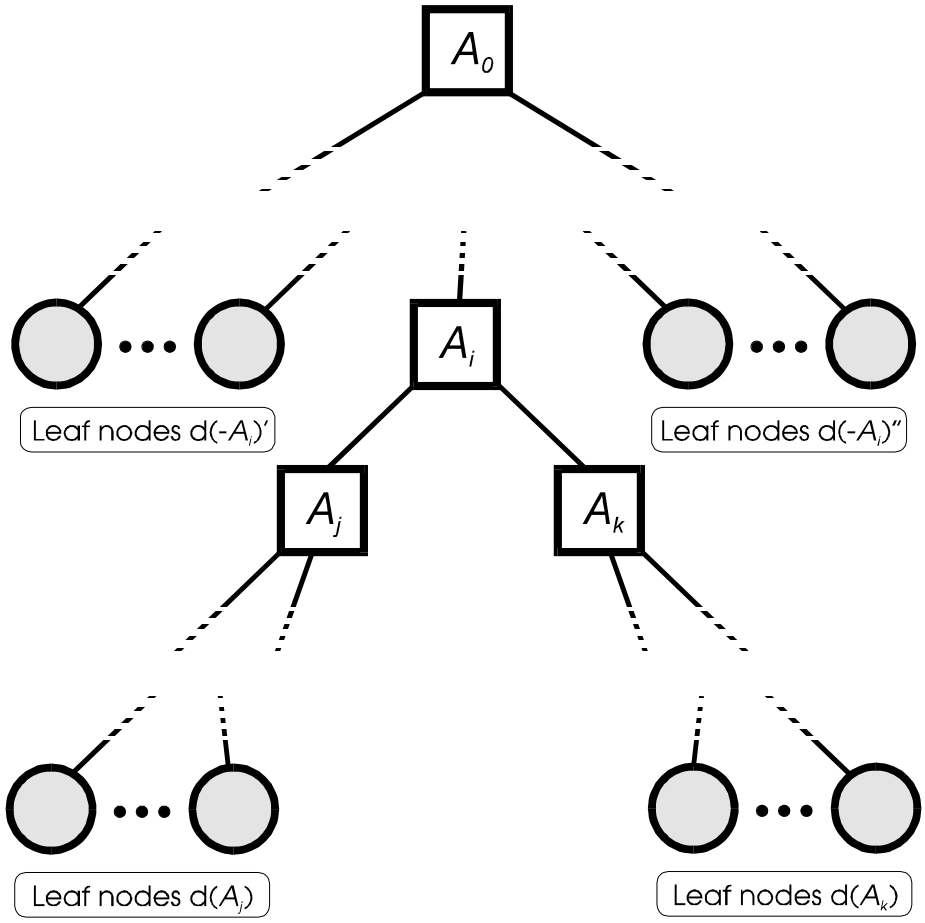

Figure 2-5: Phylogenetic Tree Node Structure

All nodes and data descend from the root node A_0. The complete column duo d consists of a set of nucleotide duos contained in leaf node sets: $d(-A_i)'$, $d(A_j)$, $d(A_k)$ and $d(-A_i)''$. Here $d(A_j)$ and $d(A_k)$ represent all of the leaf node data descended from A_i in the phylogenetic tree as $d(A_i)=d(A_j)\wedge d(A_k)$. Internal nodes A_j, and A_k are the sole children of A_i. The leaf nodes in $d(-A_i)'$ and $d(-A_i)''$ contain all of the nucleotide duos that are not directly descended from A_i to the left and right of A_i in the Tree. As $d(-A_i)'$ and $d(-A_i)''$ are never found independently in the following derivations, they are referred to collectively as $d(-A_i)$, where $d(-A_i)= d(-A_i)'\wedge d(-A_i)''$. In set theory notation $d(-A_i)=d/d(A_i)$.

Given the relationships depicted in *Figure 2-5*, we now derive the general expression for $P(d(A_i)|A_i=l)$. We replace $d(A_i)$ with $d(A_j)\wedge d(A_k)$, thus transforming,

$P(d(A_i)|A_i=l)$ into

$P(d(A_j)\wedge d(A_k)|A_i=l)$

We can use conditional decomposition to rewrite

$$P(d(A_j) \wedge d(A_k) | A_i = l) \text{ as}$$

$$P(d(A_j) | d(A_k) \wedge A_i = l) \cdot P(d(A_k) | A_i = l).$$

As any data in $d(A_j)$ must be descended from A_j, and similarly, data in $d(A_k)$ must come from A_k, we can rewrite

$$P(d(A_j) | d(A_k) \wedge A_i = l) \cdot P(d(A_k) | A_i = l) \text{ as}$$

$$\left[\sum_m P(d(A_j) \wedge A_j = m | d(A_k) \wedge A_i = l) \right] \cdot \left[\sum_n P(d(A_k) \wedge A_k = n | A_i = l) \right].$$

This sort of expansion is referred to as conjunctive inference. Through another application of conditional decomposition we can rewrite this as:

$$\left[\sum_m \left[P(d(A_j) | A_j = m \wedge d(A_k) \wedge A_i = l) \cdot P(A_j = m | d(A_k) \wedge A_i = l) \right] \right] \cdot$$
$$\left[\sum_n \left[P(d(A_k) | A_k = n \wedge A_i = l) \cdot P(A_k = n | A_i = l) \right] \right].$$

At this point, we can leverage the Markov model independence property. In section *2.3.2 Markov Model* we pointed out the seemingly trivial Markov Model property that, given a state probability distribution over M_t, $P(M_t=w_l)$, the distribution (over n)

$$P(M_{t-1}=w_n | M_t=w_l \wedge M_{t+1}=w_m) = P(M_{t-1}=w_n | M_t=w_l) \text{ and}$$

$$P(M_{t+1}=w_m | M_t=w_l \wedge M_{t-1}=w_n) = P(M_{t+1}=w_m | M_t=w_l).$$

Markov Trees have a similar property. Namely, that given $A_j=n$, any data $d(A_j)$ descended from a node A_j is independent of any state information outside the sub-tree descended from A_j. As we are looking at this derivation as a Markov process with an increasing t, A_j corresponds to M_t from the first-order Markov model. Similarly, $d(A_j)$

corresponds to the state information of $M_{t'}$, for $t'>t$. State probability distributions outside the sub-tree originating at A_j correspond to $M_{t''}$, for $t''<t$. This allows that

$$P(d(A_j)|A_j=m \wedge d(A_k) \wedge A_i=l) = P(d(A_j)|A_j=m)$$

as $d(A_k)$ and A_i are outside the sub-tree rooted at A_j. Thus, $d(A_k)$ and A_i can not affect $P(d(A_j))$ once the state of A_j is given. Similarly,

$$P(d(A_k)|A_k=n \wedge A_i=l)$$

is independent of $A_i=l$ and therefore is equal to

$$P(d(A_k)|A_k=n).$$

While, $P(A_j=m|d(A_k) \wedge A_i=l)$ is equal to $P(A_j=m|A_i=l)$. This leaves us with $P(d(A_i)|A_i=l) =$

$$\left[\sum_m \left[P(d(A_j)|A_j=m \wedge d(A_k) \wedge A_i=l) \cdot P(A_j=m|d(A_k) \wedge A_i=l)\right]\right] \cdot \left[\sum_n \left[P(d(A_k)|A_k=n \wedge A_i=l) \cdot P(A_k=n|A_i=l)\right]\right]$$

$$=$$

$$\left[\sum_m \left[P(d(A_j)|A_j=m) \cdot P(A_j=m|A_i=l)\right]\right] \cdot \left[\sum_n \left[P(d(A_k)|A_k=n) \cdot P(A_k=n|A_i=l)\right]\right].$$

We have now reduced our original probability $P(d(A_i)|A_i=l)$ to a calculation that relies solely on probabilities over A_i's children, $P(d(A_j)|A_j=m)$ & $P(d(A_k)|A_k=n)$, and our model's state transition probabilities $P(A_j=m|A_i=l)$ & $P(A_k=n|A_i=l)$. As the tree model uses a uniform state transition probability for all nodes in the tree, we can represent the probability $P(A_j=m|A_i=l)$ as a matrix indexed by l and m ($\rho_{l,m}$). The source of this distribution will be addressed in great detail later in this work in section *2.4 Mutation Models.* At the current level of abstraction, just accept that we have generated this

transition matrix from measurements made on $D_{train.}$. This further reduces our statement of $P(d(A_i)|A_i=l)$ from

$$\left[\sum_m \left[P(d(A_j)|A_j=m)\cdot P(A_j=m|A_i=l)\right]\right]\cdot\left[\sum_n \left[P(d(A_k)|A_k=n)\cdot P(A_k=n|A_i=l)\right]\right] \text{ to}$$

$$\left[\sum_m \left[P(d(A_j)|A_j=m)\cdot \rho_{l,m}\right]\right]\cdot\left[\sum_n \left[P(d(A_k)|A_k=n)\cdot \rho_{l,n}\right]\right].$$

Our derivation has only relied on the relative positions of A_i, A_j and A_k . As our derivation did not rely on any particular position for A_i within the tree, we may merely set i=0 and recursively calculate $P(d(A_0)|A_0=l)$. This recursive calculation will eventually require some $P(d(A_j)|A_j=m)$ and $P(d(A_k)|A_k=n)$ where one or both of A_j and A_k are leaf nodes. The leaf nodes represent the genetic contributions of known organisms, thus these nodes are in a completely determined state. The state distribution for some leaf node A_k , where A_k represents some organism numbered s in the phylogenetic tree, is given by

$P(d(A_k)|A_k=n)$ = 1 iff n corresponds to nucleotide duo d^s

and 0 otherwise.

This is illustrated by the following figure (*Figure 2-6*).

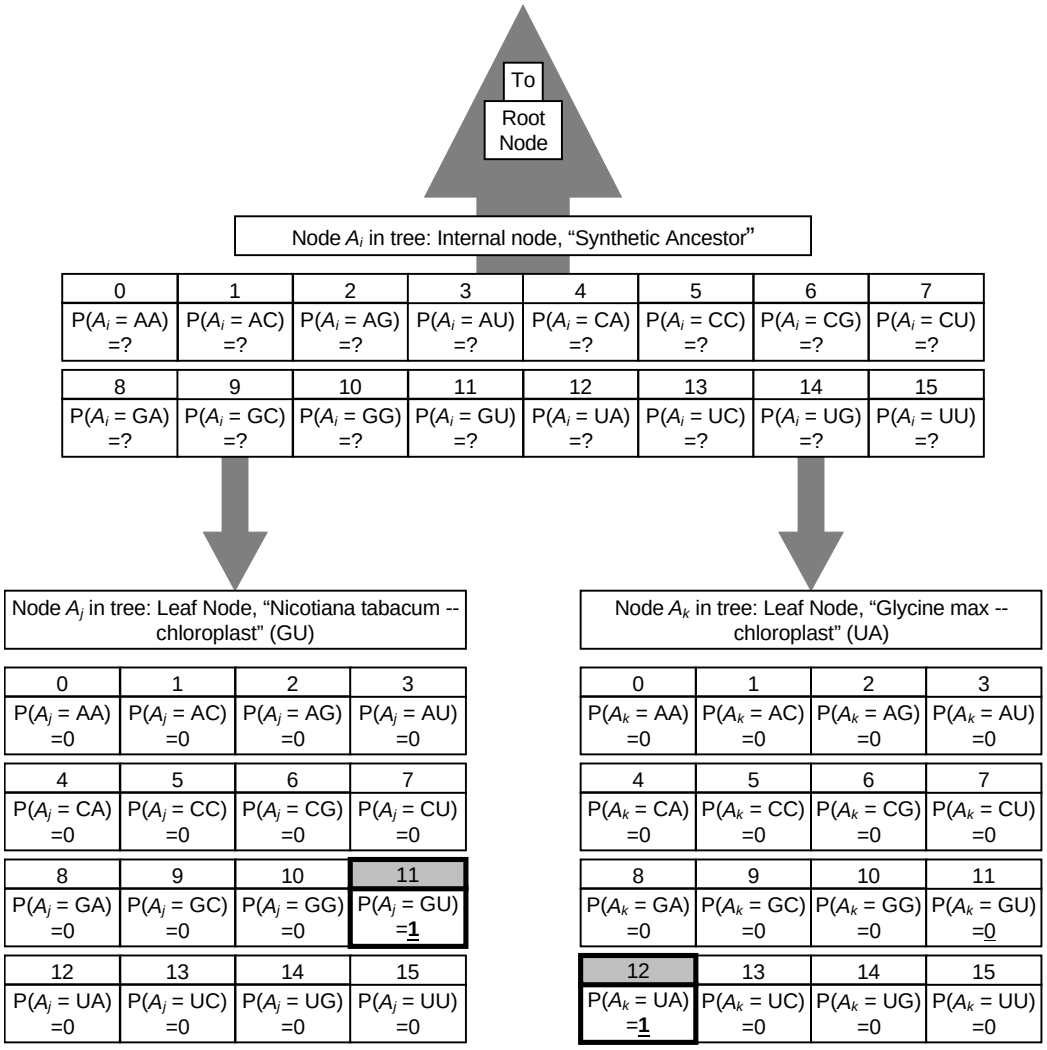

Figure 2-6: Phylogenetic Tree Leaves

Two sibling leaves of the phylogenetic tree and their parent node for column duo (89,168). The probability distribution over leaf states is defined by the nucleotides in the column duo. The distribution over states in the parent node is determined from the leaves by maximum likelihood inference through the mutation model which determines $\rho_{l,m}$.

As the probability $P(d(A_i)|A_i=l)$ plays a critical role in further derivations and experiments, we develop some special notation for it. We thus define $I_d(A_i=l) \equiv P(d(A_i)|A_i=l)$. Where d is clear from context, it may be omitted as $I(A_i=l)$. Because this probability calculation requires the knowledge of only those data inside the sub-tree

descended from A_i, the distribution is referred to as the *inside* distribution over i for A_i and d. A concise summary of the derivation of this inside probability follows:

$$\mathrm{P}(d) \equiv \sum_l \left[\mathrm{P}(d(A_0)|A_0 = l) \cdot \mathrm{P}(A_0 = l)\right] \quad \text{.....Definition.}$$

$$= \sum_l \left[\mathrm{P}(d(A_0)|A_0 = l) \cdot \varphi_l\right] \quad \text{.....Definition of } \varphi_l.$$

$$= \sum_l \left[I_d(A_0 = l) \cdot \varphi_l\right] \quad \text{.....Definition of } I_d(A_i=l).$$

Next, we develop a recursive definition of $I_d(A_i=l)$ that uses only fixed model parameters and nodes that are closer to the leaves then A_i. We then establish the base case where A_i is a leaf node.

$I_d(A_i=l) \equiv \mathrm{P}(d(A_i)|A_i=l)$ .....Definition of $I_d(A_i=l)$. If A_i is not a leaf node, see following recursive definition. If A_i is a leaf node, then $I_d(A_i=l) = 1$ if A_i's organism supplies nucleotide pair l to d and 0 otherwise.

$=\mathrm{P}(d(A_j) \wedge d(A_k)|A_i=l)$ .....Definition of $d(A_i)$, $d(A_j)$ & $d(A_k)$.

$=\mathrm{P}(d(A_j)|d(A_k) \wedge A_i=l) \cdot \mathrm{P}(d(A_i)|A_i=l)$. .....Conditional Decomposition.

$$= \left[\sum_m \mathrm{P}(d(A_j) \wedge A_j = m | d(A_k) \wedge A_i = l)\right] \cdot \left[\sum_n \mathrm{P}(d(A_k) \wedge A_k = n | A_i = l)\right]$$

.....Markov conjunctive inference.

$$= \left[\sum_m \left[\mathrm{P}(d(A_j)|A_j = m \wedge d(A_k) \wedge A_i = l) \cdot \mathrm{P}(A_j = m | d(A_k) \wedge A_i = l)\right]\right] \cdot \left[\sum_n \left[\mathrm{P}(d(A_k)|A_k = n \wedge A_i = l) \cdot \mathrm{P}(A_k = n | A_i = l)\right]\right]$$

.....Conditional Decomposition

$$\left[\sum_m \left[\mathrm{P}(d(A_j)|A_j=m)\cdot\mathrm{P}(A_j=m|A_i=l)\right]\right]\cdot\left[\sum_n \left[\mathrm{P}(d(A_k)|A_k=n)\cdot\mathrm{P}(A_k=n|A_i=l)\right]\right]$$

.....Markov independence property.

$$=\left[\sum_m \left[I_d(A_j=m)\cdot\rho_{l,m}\right]\right]\cdot\left[\sum_n \left[I_d(A_k=n)\cdot\rho_{l,n}\right]\right]$$ .....Definition[15] of I_d and ρ.

Equation 2-1: Summary Derivation of Inside Probability Distribution

2.3. 4 Notation Summary

Tree ..... Phylogenetic tree.

Pair & Rand ..... Sample set of multiple alignment column duos representing paired data or randomly selected unpaired data.

Pop_{Pair} & Pop_{Rand} ..... The Populations from which samples Pair and Rand are respectively drawn.

∧ ..... Logical conjunction ("and" operation).

A_i , A_j & A_k ..... Nodes in the Markov Tree generated from the phylogenetic tree. Where there is an ancestral relationship between the nodes, A_i is the parent of A_j and A_k.

l,m,n ..... Represent states of nodes A_i, A_j and A_k respectively. States are numbered 0 to 15 and correspond to the 16 possible nucleotide duos: AA, AC, AG, AU, CA, CC, CG, CU, GA, GC, GG, GU, UA, UC, UG & UU respectively.

d ..... A multiple alignment column duo. This is a nucleotide duo vector of length *S*, where *S* is the number of organisms in the multiple alignment. Each organism contributes one nucleotide duo to this vector.

d^s ..... Nucleotide duo *s* of column duo *d*.

$d(A_i)$ ..... Those nucleotide duos in column duo *d* that correspond to leaves of the Tree that are descended from node A_i.

$d(-A_i)$ ..... Those nucleotide duos in column duo *d* that correspond to leaves of the Tree that are *not* descended from node A_i , $d(-A_i) = d/d(A_i)$.

D ..... A set of multiple alignment column duos, also referred to as a data set.

D_{train} ..... A data set which is used to derive mutation model parameters.

D_{test} ..... A data set which is used for cross validation on a mutation model trained with a D_{train}. Each D_{train} has a corresponding D_{test} that is drawn randomly from the same population as D_{train} , but is necessarily disjoint from D_{train}.

$I_d(A_i=l)$ ..... $\mathrm{P}(d(A_i)|A_i=l)$. If A_i is not a leaf node, this is defined recursively. If A_i is a leaf node then $I_d(A_i=l)$ is 1 if A_i's organism contributes nucleotide pair *l*, to *d* and 0 otherwise

Model ..... A given mutation model, one of: Frequency, Q, IO or IOM (see *2.4 Mutation Models*).

---

[15] If one of A_i's children, say A_j is invalid, then there is no $d(A_j)$. In this case $\mathrm{P}(d(A_i)|A_i=l) = \mathrm{P}(d(A_k)|A_i=l) = \sum_n I_d(A_k=n)\cdot\rho_{l,n}$

$Model_{Rand}$..... Or $Model_{Pair}$. A mutation model trained on a data set which is known to be paired (Pair) or not known to be paired (Rand).

φ_l(Model) ..... The *a priori* probability of state *l* for any node in a Tree evaluated under Model. This is a vector of length 16 whose components sum to 1. Each component corresponds to one of the 16 possible nucleotide duos. Generally this is derived from a renormalized count of the number of each kind of nucleotide duo found in D_{train} for Model. Where Model is clear from context, this may be abbreviated as φ_l (see *2.2.1 Derivation of Frequency Model*).

$\rho_{l,m}$(Model)..... The *a priori* state transition probability from state *l* in a parent node to state *m* in a child node for Model. This is calculated in differing ways for differing mutation models. Where Model is clear from context, this may be abbreviated as $\rho_{l,m}$. For the IO mutation model this corresponds to the expectation value for $P(A_j=m|A_i=l\wedge\text{Tree}\wedge\text{Model}\wedge D_{train})$ over all nodes A_i, $A_j \in$ Tree, where A_i is the parent of A_j, see *2.4.3 IO Model.*

$\rho_{l,m}(r,\text{Model})$ ..... For the IOM mutation model, the state transition matrix ρ is also a function of the length of the branch connecting two directly related nodes A_i and A_j. The parameter r is used to indicate a bin number corresponding to a range of branch lengths, for which this ρ is applicable. See *2.4.4 IOM Model* for more details on this parameter. Where *r* or Model are clear from context, they may be omitted, and ρ will be referred to as $\rho_{l,m}$.

Table 2—1: Notation Summary

2.3. 5 Tree Model Sample Calculation

As the recursive definition of $P(d|\text{tree}\wedge\text{model})$ is rather complex, an example is presented here to show this recursive process in action. In order to focus on the process, the complexity of the model is reduced. The Phylogenetic tree has only four organisms, and thus four leaves and three internal nodes. This example reflects the tree shown in *Figure 2-7: Tree Model Example Genetic Data*. For simplicity, only two possible nucleotide duos are allowed at each node, AU or GC.

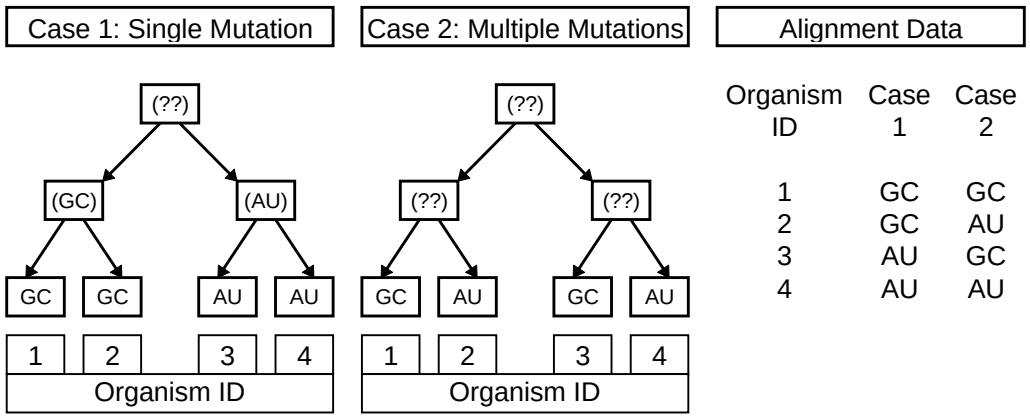

Figure 2-7: Tree Model Example Genetic Data

This figure represents the data used in this example. One phylogenetic tree with simulated nucleotide duo information for four organisms, for each of two multiple alignment column duos (Case 1 and Case 2). The purpose of this example will be to calculate $P(d(A_0)|\text{tree}\wedge\text{model})$ for the given model parameters.

The state transition matrix ρ and root node state distribution φ are approximated from actual 16S RNA data[16] (see *Table 2—2: Mutation Model Parameters for Example*). As described in the *2.2 Frequency Model* both $\text{Model}_{\text{Rand}}$ and $\text{Model}_{\text{Pair}}$ are presented and $P(d|\text{tree}\wedge\text{model})$ is calculated for each model on each of the two example column duos.

Paired Model		
P(AU→AU) = .954	P(AU→GC) = .046	P(AU) = .182
P(GC→AU) = .011	P(GC→GC) = .989	P(GC) = .818

Nonpaired (Random) Model		
P(AU→AU) = .975	P(AU→GC) = .025	P(AU) = .361
P(GC→AU) = .027	P(GC→GC) = .973	P(GC) = .639

Table 2—2: Mutation Model Parameters for Example

These numbers were derived from actual results obtained from the *2.4.3 IO Model.* The probability of no mutation occurring was maintained and the residual probability assigned to a mutation to the complimentary nucleotide duo. For the *a priori* state distribution, the relative proportions of AU and GC were maintained from the IO Model calculations, scaled up to total 100%.

[16] These transition probabilities are from the IO Model as calculated in section *3.9 IO* Model after normalization to account for the use of only 4 of 256 transition probabilities.

First we apply the multiple alignment nucleotide duos to the leaf nodes, then we calculate probabilities for the internal nodes of each model. This calculation requires the computation of $P(d(A_i)|A_i=n)$, for every state n of each node A_i in the tree,. This calculation begins at the leaves of the tree and progresses up to its root node. To save space, only one example of such a calculation (Case 1 data, Paired model) is given here. The others follow similarly.

$$P(d|\text{Model}) = \sum_{l=0,1} P(d(A_0)|A_0 = l \wedge \text{Model}) \cdot P(A_0 = l|\text{Model})$$

$$(.0408 \cdot .182) + (.0098 \cdot .818) = .021$$

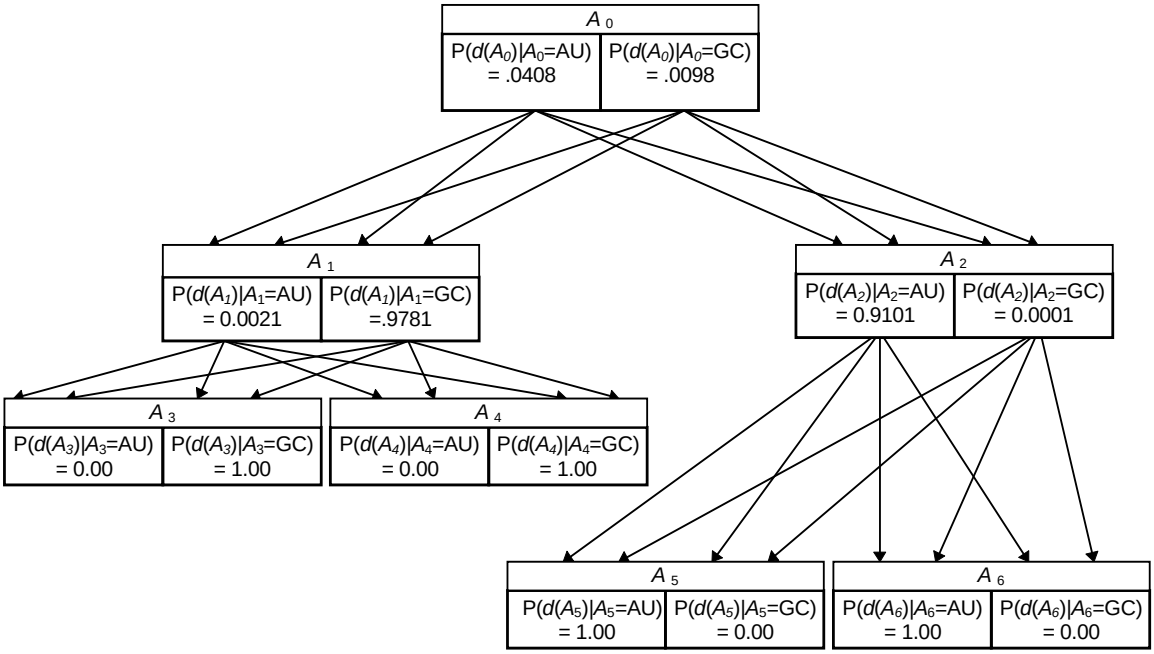

Figure 2-8: Calculation Tree for Example (Case 1, $Model_{Pair}$)

This tree shows the calculation process used to compute the posterior data probability $P(d|\text{tree}\wedge\text{model})$ for the column duo d described in *Figure 2-7: Tree Model Example Genetic Data.* The leaf nodes are initialized from the known nucleotide duo values. Other probabilities are derived from descendants according to the inference equation developed above in *Equation 2-1: Summary Derivation of Inside Probability Distribution.*

The calculations of P(*d*|Model) as described by the example in *Figure 2-8* leaves us with the following results (*Table 2—3*):

	P(*d*|Model) for Model Type			
Data *d*	$Tree_{Pair}$	$Tree_{Rand}$	$Freq_{Pair}$	$Freq_{Rand}$
Case 1	.02101	.02364	.05321	.02216
Case 2	.00073	.00066	.05321	.02216

	NLL(P(*d*|Model)) for Model Type, (bits/base)			
Data *d*	$Tree_{Pair}$	$Tree_{Rand}$	$Freq_{Pair}$	$Freq_{Rand}$
Case 1	0.697	0.675	0.529	0.687
Case 2	1.302	1.321	0.529	0.687

Table 2—3: Example Posterior Probability Result Summary

Summary of data likelihoods according to Markov Trees derived from Rand and Pair data. A sample calculation using the Frequency Model (*2.2 Frequency Model*) is also made using the same nucleotide duo frequency distribution as was used for the Markov Tree calculations.

The Frequency Model cannot distinguish between the column duos from Case 1 and Case 2 as it must assign them equal probabilities, based on their identical nucleotide distributions. On the other hand, the Tree Model does distinguish between the two cases assigning Case 2 a higher probability (lower NLL) according to $Tree_{Pair}$ than $Tree_{Rand}$. It can therefore be classified as a paired column duo. Alternatively, the single conserved mutation in Case 1 is not sufficient evidence to generate a more favorable probability from $Model_{Pair}$ than from $Model_{Rand}$, thus, Case 1 is classified as unpaired. The data from Case 1 may have come from a part of the RNA molecule that is evolutionarily stable, but suffered a random mutation at an early genetic ancestor.

The reader may find it unexpected that the Frequency Model provided data likelihoods that were about as high, or higher than those computed from the Markov Tree. This would tend to indicate that the Frequency Model is more accurately representing the sample data. The explanation for this lies in the fact that a mutation is a relatively rare event. From the statistics in *Table 2—2*, we would expect to see a mutation rate (per branch of the Markov Tree) of $0.182 \cdot 0.046 + 0.818 \cdot 0.011 = 1.7\%$ for paired column duos and $0.361 \cdot 0.025 + 0.639 \cdot 0.027 = 2.6\%$ for nonpaired column duos. Thus, our sample data is improbable in that it contains one or two mutations for only three branches. Given such improbable data, it is not surprising that the Frequency Model, which is insensitive to such mutation events, provided a higher data likelihood than the Markov Tree.

2. 4 Mutation Models

Up to this point, we have concentrated on developing the Tree Markov Model formalism that we use to derive $P(d|\text{tree}\wedge\text{model})$. We have assumed that the evolutionary model parameters $\varphi_l \equiv P(A_i=l|\text{Model})$ and $\rho \equiv P(A_j=m|A_i=l\wedge\text{Model})$ were somehow derived from the training set D_{train}. In this section, we formulate the derivation of these parameters and discuss the roles that they play in modeling the evolutionary process in a Markov Tree.

This section is broken up into four subsections. In *2.4.1 Rho and Phi (ρ and φ)* we review the specific form of the mutation models' parameters and discusses their meanings. In *2.4.2 Q Model*, we present the Q mutation model which derives ρ from a

biological interpretation of state transitions as a point-mutation process. In *2.4.3 IO Model,* we present the IO Model which calculates ρ through a purely statistical process of Expectation Maximization over a training set of column duos. In *2.4.4 IOM Model,* the final subsection, we incorporate phylogenetic tree branch lengths into the IO Model to create the most sophisticated mutation model in this work, the IOM Model.

2.4. 1 Rho and Phi (ρ and φ)

A local mutation model represents the process of nucleotide evolution between a parent organism and its children. This process is represented in two probability distributions $P(A_i=l|\text{tree}\wedge\text{Model})$ and $P(A_m=j|A_i=l\wedge\text{tree}\wedge\text{Model})$. For a given model, these distributions are represented by the 16-element vector φ_l and the 16×16 matrix $\rho_{l,m}$, respectively. As φ_l is computed in an identical fashion for each Model presented herein, the distinguishing characteristics of each model are represented completely in the calculation of $\rho_{l,m}$. As φ_l is the simpler distribution, it will be discussed first.

The probability distribution $\varphi_l \equiv P(A_i=l|\text{model})$ over $0\le l\le 15$, represents the *a priori* state distribution for all nodes in the Markov Tree. As described in the derivation of the Frequency Model, this distribution is computed through a maximum likelihood estimation over the training set of column duos D_{train}. The values in φ_l are computed by normalizing a simple count of the number of each type of nucleotide duo found in D_{train}. For each value of l, $\hat{\phi}_l$ is equal to the number of nucleotide duos of type l found in D_{train}, thus $\varphi_l = \hat{\phi}_l \Big/ \sum_{l'} \hat{\phi}_{l'}$. The interpretation of this distribution is very straightforward. Given

only that column duo d is selected at random from the population that generated D_{train}, our best Maximum Likelihood guess of the state distribution of a random node A_i in the Markov Tree operating on d would be that $P(A_i=l) = \varphi_l$. While it seems that such a distribution must have a profound impact on each step of a Markov Tree based calculation, we see that this is not the case. Nearly all of our calculations are conditioned on the assumption of a given column duo d and progress iteratively from this known leaf data. Thus, we are only required to rely on φ_l as a boundary condition at the root node. The assertion of $P(A_0=l) = \varphi_l$ for the Markov Tree's root and the nucleotide duos d^s at the Markov Tree's leaves, constitute a complete set of boundary conditions for the Markov Tree. These boundary conditions allow us to fix the state (nucleotide duo) probability distributions throughout the rest of the tree using the state transition matrix ρ for Maximum Likelihood inference on the architecture of the tree.

The state transition matrix ρ is significantly more complex than φ, in both calculation and interpretation. As with φ, ρ is extracted from calculation over the training set D_{train} where $\rho_{l,m}$ is an approximation to the expectation value of $P(A_j=m|A_i=l\wedge\text{tree}\wedge D)$, where D is the population from which D_{train} is drawn. This is the expectation value of the probability that a child node A_j will be found in state m, given that its direct parent node A_i was found in state l. Each node in a phylogenetic tree represents an organism either synthetic, for internal nodes, or observed, for leaf nodes. Thus, the biological interpretation of $\rho_{l,m}$ is the probability that an organism (A_j) which evolved directly from another organism (A_i) has a particular nucleotide duo (m) in a

particular column duo of the multiple alignment, given that its parent had some given nucleotide duo (l) in that same column. While mathematically $\rho_{l,m}$ is interpreted as a conditional probability distribution, it is also a reasonable definition of a point-mutation model. Both interpretations of $\rho_{l,m}$ are critical to the subsequent work and should be thoroughly understood before continuing.

According to the above definitions, $\rho_{l,l}$ can be interpreted as the probability per branch of the phylogenetic tree, that nucleotide duo of type l does *not* mutate, to some new nucleotide duo. We can thus represent the mutation rate per branch of the phylogenetic tree, for nucleotide type l, as $1-\rho_{l,l}$. If each branch of the phylogenetic tree is taken to represent a certain amount of chronological time, then this could represent a mean rate of mutation per unit of time, for nucleotide type l. We could similarly interpret $1-\sum_{l}\left[\varphi_{l}\cdot\rho_{l,l}\right]$ as the mean mutation rate per unit of time over all nucleotide duos.

While this biological interpretation of $\rho_{l,m}$ will be important to our derivation of $\rho_{l,m}$ from D_{train}, and thus our original search for P(d|tree$\wedge$Model), it may also be of significant interest to researchers in the field of evolutionary molecular biology. In particular, the methods used to derive $\rho_{l,m}$ for the IO and IOM Models will also provide an improved means for determining a plethora of other interesting evolutionary characteristics, including: columnar mutation rates, ancestral nucleotide distributions for

closely related organisms and mean dependence of nucleotide mutation rates on evolutionary time span (phylogenetic branch length)[17].

Returning to the problem of deriving P(d|tree∧Model), we are left with two similar interpretations for $\rho_{l,m}$ which we can leverage to derive its components. The Q, IO and IOM mutation models each derive $\rho_{l,m}$ in a different way. The Q Model is the most primitive and assumes an a priori mutation rate q which is used along with φ_l to approximate $\rho_{l,m}$. In this case, $\rho_{l,m}$ is derived according to the biological interpretation of $\rho_{l,m}$ as the measurement of evolutionary change per branch of the phylogenetic tree. The IO Model computes the components of $\rho_{l,m}$ through a more statistical interpretation of $\rho_{l,m}$ as the conditional probability distribution P($A_j=m|A_i=l\wedge$tree). To generate $\rho_{l,m}$ under this model, we begin with a Q Model approximation for $\rho_{l,m}$. We then employ an iterative process of Expectation Maximization to calculate the total number of state transitions actually observed in the Tree Model over some training set of column duos D_{train}. This count of state transitions between nodes is then normalized to become the new estimate for $\rho_{l,m}$. The reestimation calculation is then repeated using the new $\rho_{l,m}$ until no significant change is observed in $\rho_{l,m}$. Finally, the IOM Model uses nearly the same technique to evaluates $\rho_{l,m}$ as does the IO Model. However, unlike both the IO and the Q Models, IOM takes into account varying phylogentic tree branch length in its reestimation of $\rho_{l,m}$. Each branch of the phylogenetic tree has been assigned an evolutionary length by the program that generated the tree [41]. This branch length

[17] It is expected that the statistics gathered from this process will be superior to those gathered directly from measurements of a multiple alignment column duo in the same way that the Tree Model was able to resolve structure that the Frequency Model was not. The precise methods used to make such measurements will be discussed in *2.4.3 IO Model.*

represents a measure of genetic difference between a child organism and its parent. Under the interpretation of this length as a measure of time, the IOM Model groups branches of similar length. Each branch length group is then used to calculate a separate $\rho_{l,m}$. This allows for a crude variation of mutation rate with increasing evolutionary distance without having to state the form of the variation *a priori*. The IOM Model also addresses the problem of having sequence data from both parent organisms and their descendants in the phylogenetic tree at the same time. This problem is alleviated through the device of zero-length branches. For a detailed description of how this is accomplished, see *2.4.4 IOM Model.*

2.4. 2 Q Model

The Q Model leverages the biological interpretation of ρ to approximate its components $\rho_{l,m}$. Under this interpretation, a generic mutation probability per branch is given *a priori* as *q* [34][35]. If no mutation occurs between a child (A_j) and its parent (A_i), then we would expect the parent's state distribution to be the same as that of its child. This is represented as an identity transition matrix $P(A_i=l|A_j=m \wedge d \wedge \text{no mutation}) = \rho_{l,m}^{\text{no mutation}} = 1$ if $l=m$ and 0 otherwise. We model the case that a "mutation"[18] does occur as a state change to one drawn randomly from the models *a priori* state distribution. This is represented by the state transition matrix $\rho_{l,m}^{\text{mutation}} = P(A_j=m|\text{model}) = \varphi_m$. As we are interested in forming a transition matrix, embodying both the possibility of mutation, and the possibility of conservation (non-mutation) we take our

[18] . The term "mutation" is used loosely in the context of the Q Model transition function. The result state of a "mutation" modeled by the random selection of a new state according to φ_m, could be any state. As it is possible to randomly select the original state, a state change is not guaranteed under a Q Model

composite ρ to be a linear combination of the two transition matrices blended according to q as $\rho_{l,m} = (q)\cdot\rho_{l,m}^{\text{mutation}} + (1\text{-}q)\cdot\rho_{l,m}^{\text{no mutation}}$.

$$\rho_{l,m} \equiv \begin{cases} q\cdot\varphi_m & \text{if } l \neq m \\ q\cdot\varphi_m + (1-q) & \text{if } l = m \end{cases}$$

Equation 2-2: Q Model Mutation Probabilities

The mutation probability parameter q represents the probability per branch that a point-mutation will occur. If no mutation occurs, a child's state distribution is the same as its parents. If a "mutation" does occur the child's state distribution is set to equal the *a priori* state distribution φ. This does allow $l = m$ (no change) as a possible result from a mutation event. As it is not known *a priori* whether or not a mutation event occurs, a linear superposition of these possibilities is used at each state transition.

The above probability distribution (*Equation 2-2*) was proposed by Felsenstein [34] for use in phylogenetic tree construction. While this Q Model is relatively crude and requires an empirical determination of the optimal value for q, it does serves as a plausible preliminary estimate for ρ. Tree Model calculations derived using this estimate for ρ serve as a basis against which to measure the performance of the more detailed IO and IOM Models. One serious argument against this model is that there is no *a priori* reason to believe that the result of a mutation event can be accurately drawn at random from the stationary distribution φ_l. This would indicate that $\forall l,l',m,l\neq m,l'\neq m\ \rho_{l,m} = \rho_{l',m}$, which seems intuitively unlikely.

For the purpose of comparison with IO and IOM Models, the Q Model has 16 degrees of freedom. This is because φ has 16 independent parameters which are normalized to unity, reducing the number of degrees of freedom in φ by 1. The

---

"mutation".

mutation rate parameter q adds one additional degree of freedom. Thus, the number of degrees of freedom is $|q|+|\varphi| = 1+(16-1) = 16$.

2.4. 3 IO Model

Since the IO Model is appreciably more complex than the Q Model, its derivation is broken into four separate subsections. In *2.4.3.1 IO Model Overview* we provide a general overview of the reestimation process used to calculate ρ from D_{train}. In *2.4.3.2 Frequency Reestimation* we derive the specific cumulative frequency function that is renormalized to form ρ. In *2.4.3.3 Outside Probability* we derive a new state probability distribution called the "outside" distribution that is critical for the calculation of the cumulative frequency function. In *2.4.3.4 Summary* we combine the derivations of the previous subchapters into a compact representation for the reestimation process.

2.4.3.1 IO Model Overview

The IO Model is so named for to its similarity to the *Inside-Outside* method for the training of Stochastic Context Free Grammars [36] [37]. This model directly estimates the parameters of $\rho_{l,m}$ from a given Markov Tree and a training set of column duos referred to as D_{train}. This model is significantly more complex then the Q Model as each element of the 16×16 state transition matrix $\rho_{l,m}$ is reestimated independently. As each row l of the matrix $\rho_{l,m}$ is required to be normalized, one degree of freedom is removed for each of the 16 rows in the 256-element matrix. This yields a total number of degrees of freedom for the IO Model of $|\varphi| + |\rho| = (16-1) + (256-16) = 255$, which is much larger than the 16 degrees of freedom of the Q Model.

The parameters φ and ρ of the IO Model are initialized from the corresponding parameters of the Q Model. A value for q is given *a priori*, φ is extracted from D_{train} as for the Frequency Model and these are combined to construct the initial estimate for $\rho_{l,m}$. We now present each element $d \in D_{train}$ at the leaves of the tree and use dynamic programming to fill in the conditional state probability distributions $P(A_i=l|d)$ at each node. Once this calculation is complete, we aggregate the number of times that we observe a state transition from each state l of a parent node A_i to each state m of a child node A_j.

As the Markov Tree is probabilistic, state transitions are not observed as discrete events. Rather, they are observed as probabilities that a particular state was occupied $P(A_i=l|d)$, multiplied by the probability that a particular transition was made from a state l of A_i to some state m of its child A_j, $P(A_j=m|A_i=l \wedge d)$. Thus the number of transitions observed between each pair of states (l,m), from parent node A_i to its child A_j, is represented as $P(A_j=m|A_i=l \wedge d) \cdot P(A_i=l|d)$. Further, we know that $P(A_j=m|A_i=l \wedge d) \cdot P(A_i=l|d)$ is equal to $P(A_j=m \wedge A_i=l|d)$ by conditional decomposition. Such transitions are referred to as fractional transitions as the sum of all fractional transitions to states of a given child A_j from its parent A_i must total to 1. These fractional transitions are then aggregated over all child-parent combinations in the Markov Tree. This aggregation is equivalent to the summation $\sum_{\substack{i,j \\ A_i \text{ parent of } A_j}} P(A_i = l \wedge A_j = m|d) = \hat{f}_{l,m}(d)$. The matrix $\hat{f}_{l,m}(d)$ then contains the relative frequencies of the state transition $l \rightarrow m$ in the Markov Tree generated by d.

These transitions are then be accumulated over all $d \in D_{train}$. to form $\hat{f}_{l,m} = \sum_{d \in D_{train}} \hat{f}_{l,m}(d)$.

As we are summing over all $d \in D_{train}$, and D_{train} is assumed to be drawn at random from the population of column duos that we are attempting to model, no additional weighting is required to reflect P(d|Model). This is because d is expected to appear in D_{train} approximately as often as it appears in the population from which D_{train} is drawn. Thus, the frequency statistics generated from measurements on d will automatically be weighted by the number of times that d appears in D_{train}.

Once we have $\hat{f}_{l,m}$, we can normalize it to obtain our next estimate for $\rho_{l,m}$ as $\hat{f}_{l,m} \Big/ \sum_{m'} \hat{f}_{l,m'}$. This iterative reestimation process for $\rho_{l,m}$ represents the heart of the Expectation Maximization method [38]. This method is guaranteed to produce a model which locally maximizes P(D_{train}|model) over the components of $\rho_{l,m}$.

2.4.3.2 Frequency Reestimation

The reestimation procedure described above relies on the computation of the probability distribution P($A_i=l \wedge A_j=m|d$). In this section we attempt to formulate this probability distribution in terms of Model parameters, and the boundary conditions of the phylogenetic tree at its leaves (d). We will discover that we can not do this directly and will, instead rely on the calculation of two recursively defined probability distributions. Both of the recursive calculations needed to derive the distributions will eventually terminate with either a model parameter (φ), or a boundary condition at a tree leaf (d).

$P(A_i=l \wedge A_j=m|d)$ ..... Initial quantity.

$= P(A_i=l \wedge A_j=m|d) \cdot P(d)/P(d)$ ..... Multiplicative Identity.

$= P(A_i=l \wedge A_j=m \wedge d)/P(d)$ ..... Bayes' Rule.

Note: $P(d)$ is simply defined as $\sum_l \left[I_d(A_i = l) \cdot \varphi_l \right]$, as in *2.3 Tree Model Topology*.

$= P(A_i=l \wedge A_j=m \wedge d(-A_i) \wedge d(A_i))/P(d)$ ..... Definition of d (*Figure 2-5*).

$= P(A_j=m \wedge d(A_i)|A_i=l \wedge d(-A_i)) \cdot P(A_i=l \wedge d(-A_i))/P(d)$ ..... Conditional Decomposition.

$= P(A_j=m \wedge d(A_i)|A_i=l) \cdot P(A_i=l \wedge d(-A_i))/P(d)$ ..... Markov Independence on $A_{i..}$

$= P(d(A_i)|A_j=m \wedge A_i=l) \cdot P(A_j=m|A_i=l) \cdot P(A_i=l \wedge d(-A_i))/P(d)$ .. Conditional Decomposition.

$= P(d(A_i)|A_j=m \wedge A_i=l) \cdot \rho_{l,m} \cdot P(A_i=l \wedge d(-A_i))/P(d)$ ..... Definition of $\rho_{l,m.}$

$= P(d(A_j) \wedge d(A_k)|A_j=m \wedge A_i=l) \cdot \rho_{l,m} \cdot P(A_i=l \wedge d(-A_i))/P(d)$ ..... $d(A_i) = d(A_j) \wedge d(A_k)$.

$= P(d(A_j)|A_j=m \wedge A_i=l) \cdot P(d(A_k)|A_j=m \wedge A_i=l) \cdot \rho_{l,m} \cdot P(A_i=l \wedge d(-A_i))/P(d)$

..... Independence of $d(A_j)$ and $d(A_k)$ given $A_i=l$.

$= P(d(A_j)|A_j=m) \cdot P(d(A_k)|A_j=m \wedge A_i=l) \cdot \rho_{l,m} \cdot P(A_i=l \wedge d(-A_i))/P(d)$

..... Independence of $d(A_j)$ and $A_i=l$ given $A_j=m$.

$= I_d(A_j=m) \cdot P(d(A_k)|A_j=m \wedge A_i=l) \cdot \rho_{l,m} \cdot P(A_i=l \wedge d(-A_i))/P(d)$ .. Definition of $I_d(A_j=m)$.

$= P(d(A_k)|A_j=m \wedge A_i=l) \cdot I_d(A_j=m) \cdot \rho_{l,m} \cdot P(A_i=l \wedge d(-A_i))/P(d)$. Rearrange terms.

$= P(d(A_k)|A_i=l) \cdot I_d(A_j=m) \cdot \rho_{l,m} \cdot P(A_i=l \wedge d(-A_i))/P(d)$..... Independence of $A_j=m$ and $d(A_k)$ given $A_i=l$.

$= \sum_n P(A_k = n \wedge d(A_k)|A_i = l) \cdot I_d(A_j=m) \cdot \rho_{l,m} \cdot P(A_i=l \wedge d(-A_i))/P(d)$

..... Conjunctive Inference.

$= \sum_n \left[P(d(A_k)|A_k = n \wedge A_i = l) \cdot P(A_k = n|A_i = l) \right] \cdot I_d(A_j=m) \cdot \rho_{l,m} \cdot P(A_i=l \wedge d(-A_i))/P(d)$

..... Conditional Decomposition.

$= \sum_n \left[P(d(A_k)|A_k = n \wedge A_i = l) \cdot \rho_{l,n} \right] \cdot I_d(A_j=m) \cdot \rho_{l,m} \cdot P(A_i=l \wedge d(-A_i))/P(d)$

..... Definition of $\rho_{l,n}$.

$$= \sum_n \left[\mathrm{P}(d(A_k) \mid A_k = n) \cdot \rho_{l,n} \right] \cdot I_d(A_j{=}m) \cdot \rho_{l,m} \cdot \mathrm{P}(A_i{=}l \wedge d(\text{-}A_i))/\mathrm{P}(d)$$

..... Independence between A_i and $d(A_k)$ given $A_k{=}n$.

$$= \sum_n \left[I_d(A_k = n) \cdot \rho_{l,n} \right] \cdot I_d(A_j{=}m) \cdot \rho_{l,m} \cdot \mathrm{P}(A_i{=}l \wedge d(\text{-}A_i))/\mathrm{P}(d)$$. Definition of $I_d(A_k{=}n)$.

Equation 2-3: IO Model Transition Frequency Reestimation Derivation (Part I)

By the end of *Equation 2-3* we have reduced our state transition frequency measurement to summations over model parameters (ρ), previously derived values (P(d) and $I_d(A_j{=}m)$), and P($A_i{=}l \wedge d(\text{-}A_i)$). This last term is troubling as it is not readily reducible to model parameters and boundary values. If we can calculate this probability distribution, our frequency reestimation will be complete. Fortunately, there is such a calculation.

Just as the inside distribution was defined for a given node state $A_i{=}l$ as a recursive calculation on the nodes of the sub-tree rooted at A_i, we can define a corresponding recursive calculation, which we will call the *outside* distribution. This distribution is over the states of a given node, such as A_i, however, it involves only those nodes *outside* of the sub-tree roots at A_i. Clearly it is just such a calculation that is needed to derive P($A_i{=}l \wedge d(\text{-}A_i)$) from the boundary conditions outside of A_i, namely $d(\text{-}A_i)$.

2.4.3.3 Outside Probability

The outside distribution $O_d(A_i=l) \equiv P(A_i=l \wedge d(-A_i))$ will take into account all of the leaf node data that fall outside of $d(A_i)$. As we will see, the calculation of $O_d(A_i=l)$ will proceed recursively beginning at the root node and working towards the leaves. This is exactly the opposite direction that the inside calculation took. Because the inside probability distribution relies solely on model parameters and known boundary conditions, it will be assumed that this distribution has already been calculated at every node. This is necessary, as we will be using the inside distribution in our formulation of the outside distribution. In addition, it will conserve our notation if we define our outside distribution on $O_d(A_j=m)$ rather than $O_d(A_i=l)$. This is merely a notational convenience and has no underlying mathematical significance. Once this distribution is calculated, we will plug it into the missing step in the frequency estimation and we will be able to completely reestimate the IO Model ρ parameters.

We begin with the unresolved reduction from our previous transition frequency calculation, namely[19]:

$O_d(A_j=m) \equiv P(A_j=m \wedge d(-A_i) \wedge d(A_k))$.

$$= \sum_l P(A_j=m \wedge A_i=l \wedge d(-A_i) \wedge d(A_k)) \quad \text{.....Conjunctive Inference.}$$

$$= \sum_l P(A_j=m \wedge d(A_k) | A_i=l \wedge d(-A_i)) \cdot P(A_i=l \wedge d(-A_i))$$

$$\text{.....Conditional Decomposition.}$$

$$= \sum_l P(A_j=m \wedge d(A_k) | A_i=l \wedge d(-A_i)) \cdot O(A_i=l) \quad \text{.....Definition of Outside.}$$

[19] Please note that we have redefined our node notation here. In the transition frequency reestimation, we were left with $P(A_i=l \wedge d(-A_i))$ unresolved. As we have relabeled A_i as A_j for notational convenience, $d(-A_i)$ must be relabeled as $d(-A_i) \wedge d(A_k)$. This can be made more clear by a glance at *Figure 2-4: Markov Tree Model.* If we relabel the node A_i as A_j, then the leaf nodes that were previously covered by $d(-A_i)$ will now include the new $d(-A_i)$ as well as the data descended from the new A_k, that is $d(A_k)$.

$$= \sum_l \mathrm{P}(A_j=m \mid A_i=l \wedge d(-A_i) \wedge d(A_k)) \cdot \mathrm{P}(d(A_k) \mid A_i=l \wedge d(-A_i)) \cdot O(A_i=l)$$

.....Conditional Decomposition.

$$= \sum_l \mathrm{P}(A_j=m \mid A_i=l) \cdot O(A_i=l) \cdot \mathrm{P}(d(A_k) \mid A_i=l \wedge d(-A_i))$$

.....Independence of $d(-A_i)$, $d(A_k)$ and A_j given $A_i=l$.

$$= \sum_l \rho_{l,m} \cdot O(A_i=l) \cdot \mathrm{P}(d(A_k) \mid A_i=l \wedge d(-A_i))$$ .....Definition of $\rho_{l,m}$.

$$= \sum_l \left[\rho_{l,m} \cdot O(A_i=l) \cdot \sum_n \mathrm{P}(d(A_k) \wedge A_k=n \mid A_i=l \wedge d(-A_i)) \right]$$

.....Conjunctive Inference.

$$= \sum_l \left[\rho_{l,m} \cdot O(A_i=l) \cdot \sum_n \left[\mathrm{P}(d(A_k) \mid A_k=n \wedge A_i=l \wedge d(-A_i)) \cdot \mathrm{P}(A_k=n \mid A_i=l \wedge d(-A_i)) \right] \right]$$

.....Conditional Decomposition.

$$= \sum_l \left[\rho_{l,m} \cdot O(A_i=l) \cdot \sum_n \left[\mathrm{P}(d(A_k) \mid A_k=n \wedge A_i=l \wedge d(-A_i)) \cdot \mathrm{P}(A_k=n \mid A_i=l) \right] \right]$$

.....Indep. of $d(-A_i)$ and A_k given $A_i=l$.

$$= \sum_l \left[\rho_{l,m} \cdot O(A_i=l) \cdot \sum_n \left[\mathrm{P}(d(A_k) \mid A_k=n) \cdot \mathrm{P}(A_k=n \mid A_i=l) \right] \right]$$

.....Independence of $d(A_k)$ given $A_k=n$.

$$= \sum_l \left[\rho_{l,m} \cdot O(A_i=l) \cdot \sum_n \left[I(A_k=n) \cdot \rho_{l,n} \right] \right]$$ .....Definition of $\rho_{l,n}$[20].

Equation 2-4: IO Model Outside Probability Distribution Derivation

The final form of the derivation of *Equation 2-4* gives us a recursive formula for the calculation of the outside probabilities in terms of model parameters, inside probabilities and previously calculated outside probabilities. However, we still have not established the recursive terminating condition at the root node. This anchor step is:

[20] If A_j has no valid sibling A_k then there is no $d_3 = d(A_k)$. In this case $O_d(A_j=m) \equiv \mathrm{P}(A_j=m \wedge d(-A_i) \wedge d(A_k))$ becomes $\mathrm{P}(A_j=m \wedge d(-A_i)) = \sum_l (\rho_{l,m} \cdot O_d(A_i=l))$.

$O(A_0{=}m) \equiv \mathrm{P}(d(-A_0) \wedge d(A_k) \wedge A_0{=}m)$

$= \mathrm{P}(A_0{=}m)$ ..... As A_0 is the root node, it has no sibling node A_k and $d(A_k)$ is null. Similarly, as A_0 has no parent node, so $d(\text{-}A_0)$ is null.

$= \varphi_m$ ..... Definition of φ_m.

2.4.3.4 Summary

In section *2.4.3.2 Frequency Reestimation* we established our state transition probability estimation as $\rho_{l,m} = \hat{f}_{l,m} \Big/ \sum_{m'} \hat{f}_{l,m'}$, where $\hat{f}_{l,m} = \sum_{d \in D_{train}} \hat{f}_{l,m}(d)$. We have further established that $\hat{f}_{l,m}(d) =$

$$\sum_{\substack{i,j \\ A_i \text{ parent of } A_j}} \mathrm{P}(A_i = l \wedge A_j = m | d) =$$

$$\sum_n \left[I_d(A_k = n) \cdot \rho_{l,n} \right] \cdot I_d(A_j{=}m) \cdot \rho_{l,m} \cdot \mathrm{P}(A_i{=}l \wedge d(\text{-}A_i)) / \mathrm{P}(d).$$

With the addition of our newly derived outside probability distribution and an expansion of the P(d) term, $\hat{f}_{l,m}(d)$ may be conclusively rewritten as:

$$\sum_n \left[I_d(A_k = n) \cdot \rho_{l,n} \right] \cdot I_d(A_j{=}m) \cdot \rho_{l,m} \cdot O_d(A_i{=}l) \Big/ \sum_l \left[I_d(A_i = l) \cdot \varphi_l \right]$$

Equation 2-5: IO Model Transition Frequency Reestimation Derivation (Part II)

2.4. 4 IOM Model

While the IO Model expands significantly on the adaptability of the Q Model, it still leaves open the problem of broadly differing branch lengths in the phylogenetic tree. The IO Model treats all branch lengths in the phylogenetic tree as having equal length. These branch lengths represent the amount of genetic diversity between a child organism

and its parent. If mutation is taken to be a stochastic process, then it is reasonable to assume that a larger branch length represents a greater amount of chronological time. The branch lengths found in our phylogenetic tree span five orders of magnitude. It is reasonable to expect that providing some model variance to represent this range would lead to an increase in modeling accuracy. In addition, there remains the unresolved problem of observed organisms that are descended from other observed organisms. As the organization of the phylogenetic tree forces all observed organism data to be at the leaves of the tree, it seems that no observed organism may be represented as the descendent of another. Clearly there is biological evidence to contradict this structure. The IOM Model (*Inside-Outside-Multiple*) is designed to address both the branch length variance issue and the decendency issue. This is accomplished through the modeling of differing phylogenetic branch lengths (r) with differing state transition matrices $\rho_{l,m}(r)$.

Our concept of evolutionary time assumes that there is some underlying point-mutation process occurring continuously with time. Let us define a matrix M, similar to our $\rho_{l,m}$ matrix, which represents the probability per unit of evolutionary time Δt that a given nucleotide duo type l will mutate into a nucleotide duo type m. If the mutation process corresponds to our model, we would expect that observed mutation rate could be modeled over any time period T as $M^{(T/\Delta t)}$. This M would thus embody both long and short time period behavior for such randomly selected mutations. This kind of process corresponds exactly to the first-order Markov process described in section *2.3.2 Markov Model.*

Given such a mutation process, as well as a measure of evolutionary time for each branch length on our tree, we would expect that we could reestimate the matrix M as well as the branch lengths of each branch in the Markov Tree. A problem arises, however, in the transition frequency reestimation process of the IO Model. In the reestimation equations derived above (section *2.4.3.2*), we assume that all observed transition counts in the tree occur over the same period of time. This is embodied in the equation

$$\sum_{\substack{i,j \\ A_i \text{ parent of } A_j}} \mathrm{P}(A_i = l \wedge A_j = m|d) \;=\; \hat{f}_{l,m}(d)\,,$$

where all node descendant (A_i parent of A_j) are considered equally related. In a model that takes into account relative branch lengths (evolutionary time), the state transitions observed between a given A_i and A_j are drawn from a mutation process over a potentially unique amount of evolutionary time $T_{i,j}$. Thus, each unique combination nodes *i,j* could yield an estimate for the generic mutation rate *M* over a different time scale. When we are done aggregating the fractional state transitions in the Markov Tree we will thus have a series of estimates for *M* over differing scales of time in the form of $M^{(T1/\Delta t)}$, $M^{(T2/\Delta t)}$, $M^{(T3/\Delta t)}$… where each *T1*, *T2*, *T3*… represent the evolutionary time between a unique pair of nodes (i_1, j_1), (i_2, j_2), (i_3, j_3)… While such a computation could easily be performed, it is unclear how the resulting estimates could be combined into a single estimate for *M*. Many techniques are known for exponentiating and taking the roots of such square matrices. These could be combined, for example, to take the geometric mean of the observed matrices. However, there is no method known to the author for combining these matrices into a single estimate of *M* that preserves the behavior of $M^{(T/\Delta t)}$ over both

long and short time periods. This task is made particularly complex by the presence of uncertainty in the estimation data.

To circumvent this issue, a nonparametric method was chosen that did not rely on the explicit exponentiation of *M*. This method allows variation of $\rho_{i,j}$ with differing branch lengths by grouping the branches into bins based on their branch lengths. Bin boundaries are collateral, non-overlapping and constructed so as to provide approximately the same number of branches in each bin. The set of all bins is defined as *R*, while each bin *r* is enumerated such that $0 \leq r \leq (|R|-1)$. All of the branches within a given group *r*, are then treated as having the same branch length. Within each branch length range, the transition probability reestimation proceeds exactly as with the IO Model. The single difference is that only branches within *r* contribute to the reestimation of $\rho_{i,j}(r)$. While this process is crude, it allows for a variation of the mutation frequency matrix $\rho_{i,j}(r)$ with greatly differing evolutionary spans[21]. In some ways, this non-parametric method is potentially preferable to one that requires a particular form of time variation for $\rho_{i,j}(r)$. While the non-parametric method employed here allows for only a crude time variation of $\rho_{i,j}(r)$, it does allow the form of the variation to be completely driven by the training data. This form can then be examined to gain further understanding regarding the time variance of $\rho_{i,j}$. A more restrictive parameterization of $\rho_{i,j}(r)$ runs the risks of misfitting the data or mismodeling the evolutionary process. The

[21] While not explored in this work, the comparisons of the mutation matrices $\rho_{i,j}(r)$ for differing *r* may provide interesting insights into the way molecular RNA evolution occurs on differing time scales. In particular, a comparison of the eigenvectors and eigenvalues of these matrices could provide evidence for or against the hypothesis that molecular evolution is a stateless process, and thus representable over any time scale by a mutation rate matrix *M* raised to some evolutionary time component *T* (M^T).

current model for $\rho_{i,j}(r)$ might provide the information needed to build an accurate parametric model for $\rho_{i,j}(r)$. Furthermore, the present model for $\rho_{i,j}(r)$ allows us to elegantly resolve the problem of concurrently observed ancestors and descendants in the phylogenetic tree.

The use of differing branch lengths to represent differing degrees of evolutionary time allows us to represent such ancestral relationships among observed organisms through the device of 0 length branches. The setting of a branch length to 0 indicates that there is no difference between an ancestor organism and its descendent, as no evolutionary time had passed. As shown below in *Figure 2-9*, any ancestral relationship can be represented in our phylogenetic tree, while limiting observed organisms to the leaf nodes. In the IOM Model, a special bin (r=0) is set aside to represent 0 length branches. This matrix $\rho_{i,j}(0)$ is set to the identity matrix *a priori* to prevent any transitions from taking place as $\rho_{i,j}(0) = 1$ iff $i=j$, and 0 otherwise. This process is described graphically in the following figure.

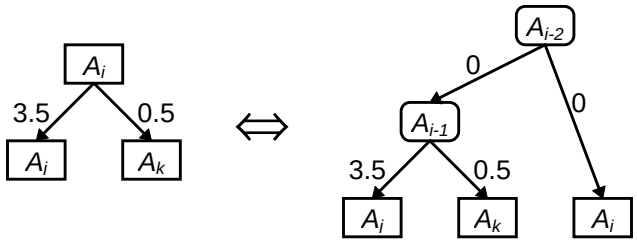

Figure 2-9: Use of Zero Length Branches in Phylogenetic Tree

The above phylogenetic trees are computationally identical. Rectangular boxes represent observed organisms, while boxes with rounded corners represent internal "synthetic ancestors". The device of zero-length branches can be used to move observed ancestors (A_i) into leaf nodes. When the branch length is 0, then no evolutionary time is considered to have passed. Nodes connected by branches of length 0 must have the same state distribution. Observed organisms can not be connected by branches of length 0 unless their genetic makeups are identical.

We expect that the increased complexity of the IOM Model over the IO Model will yield additional modeling accuracy. However, this increased complexity does not come without additional cost. The IO Model provided $|\varphi|+|\rho| = (16-1) + (256-16) = 255$ degrees of freedom. For an IOM Model with $|R|$ bins, we would have $|\varphi|+(|R|-1)\cdot|\rho|$ degrees of freedom. The magnitude $|R|$ is decremented by one because $\rho_{l,m}(0)$ is forced to unity, and thus provides no additional freedom. For the experiments in this work, $|R| = 7$. This the IOM Model has nearly 1,455 degrees of freedom, nearly 6 times as many as does the IO Model. With so many degrees of freedom the specter of over-fitting arises and we must ask whether our model is really capturing salient characteristics of the D_{train}'s population. Perhaps we are merely encoding the exact information of D_{train} in the parameters of the Model. To maintain vigilance against this possibility, the available data sets are broken into disjoint training and validation sets. The difference between each Model's performance on training data is diligently compared to its performance on the validation data. If the performance on the two data sets begins to diverge, we expect that we are over-fitting the data. However, at this point our subject matter has left the proper realm of our theoretical development for Chapter *2: Theory*. We are now ready to move into the experimental domain of Chapter *3: Experiments.* Model validation, parameter selection and other such important issues will be addressed therein.

3 Experiments

This chapter is broken into ten sections which follow the experimental development of the Tree Model. In *3.1 Data Sources,* we discuss the source and format of the multiple alignment, phylogenetic tree, paired and random column duos used in these experiments. After obtaining the data, it had to be filtered as described in *3.2 Preliminary Data Preprocessing* to remove unusable sections. This filtered data was then tentatively evaluated under the Q Model described in *3.3 Preliminary Q Model Study* to obtain an approximately optimal value for *q*. The results of the initial Q Model investigation indicated a need for cross validation and some further filtering of the data, as described in *3.4 Secondary Data Preprocessing.* In addition, the initial Q Model results exhibited certain nonlinearities which were addressed through the development of a more sophisticated classification scheme set forth in *3.5 Classifiers.* As the results presented here are relatively complex, *3.6 Results Format* provides a brief overview of the graphical, and statistical format which will be used to present the final results. Finally, the results of each Model are reviewed and discussed briefly in the last four sections: *3.7 Frequency Model, 3.8 Q Model, 3.9 IO Model* and *3.10 IOM Model.*

3. 1 Data Sources

All of the data used in the following experiments was obtained through the Ribosomal Data Project (RDP) of the University of Chicago, Urbana-Champaign [39]. In particular, we used data from prokaryotic Small Subunit (SSU) RNA also known as 16S RNA. This family of ribosomal RNA was selected because it has nearly as many

known sequences as the shorter tRNA, yet is approximately 25 times as long, providing a greater challenge for structural modeling.

Three essential data files were retrieved from the RDP: a multiple alignment (SSU_Prok.gb) [40], the phylogenetic tree (SSU_Prok.newick) [41] and a list of column duos which are known to be paired (pairs) [43]. These files were from revision 3.0 of the RDP database. All of the experimental results derived herein stem solely from these three data files. The multiple alignment data file fixed the SSU primary sequence data into 2688 columns and contained alignments for 1381 organisms. This alignment is, at the time of this writing, a comprehensive listing of sequenced 16S RNA. Alignment has been performed by hand through contributions of numerous research biologists over years of work. The phylogenetic tree data file contained 1376 organisms and was generated by the fastDNAml program, version 1.0.6 [21], which is also available from the RDP. The phylogenetic tree and multiple alignment had 1375 organisms in common, the rest of the data was disregarded. The paired column duo file contained a list of 2-tuples of column identification numbers for those multiple alignment columns that are believed to chemically interact. These column pairs included column duos that are believed to compose: helixes (secondary structure), end-caps for helixes and individual pairs (tertiary structure). The paired column duo data did not include any duos related solely by ternary (3 nucleotide) interactions.

3. 2 Preliminary Data Preprocessing

Several aspects of the initial data received from the RDP made it difficult to use. To insure a uniform data set for experimentation, several preprocessing steps proved necessary.

First, the organism names in the phylogenetic tree data file had to be coordinated with the names in the multiple alignment. Since the naming conventions were similar, but not exactly the same, and there was no common keying field, name space coordination was accomplished by hand through the addition of a tag line to each entry in the multiple alignment data file [42]. This line contained the phylogenetic tree's organism name for each corresponding multiple alignment sequence.

Next, a list of random column duos was generated to sample the nonpaired (Rand) population. All paired column duos (Pair) [43] found in the nonpaired sample were removed[22]. Since there were more symbols in the multiple alignment than the nucleotide designators (A, C, G, U), both Pair and Rand data were filtered to insure a certain amount of valid data in each column duo. Additional characters included symbols representing gaps (-), omissions (.) and uncertain sequencing data (Y, R, N).

In all Models, only "valid" nucleotide duos contributed to the probability calculations. A nucleotide duo was considered valid if each of its constituent bases was one of {A, C, G, U}. Both Rand and Pair were filtered to insure that each column duo

[22] Only duos from the nonpaired set which were found in the paired set were removed. If a nonpaired duo had one column in common with a paired duo, it was not removed from the nonpaired data set. A nonpaired column duo could have both of its columns present in paired duos, so long as those columns were not themselves paired. This filtering was chosen to represent experimental conditions where the Tree

contained at least 75% valid nucleotide duos. This process reduced Rand and Pair from 3500 and 944 column duos to 695 and 634 column duos respectively. In probability calculations, non-valid duos in a column duo were treated as nonexistent[23]. All resultant NLL scores for column duos were then normalized by the number of nucleotides in valid nucleotide duos for each column duo. This normalization maintained a consistent interpretation of NLL as the mean number of bits of information per valid base in the data set.

3. 3 Preliminary Q Model Study

After the data had been filtered, the preliminary studies using the Q Model were performed in order to determine a working value for the mutation frequency parameter q. NLL values for both Rand and Pair data were calculated under using q values of: 0.99, 0.9, 0.5, 0.1, 0.01, 0.001, 0.0001 and 0.00001. Since no cross validation was being implemented at this phase of the work, all column duos were included in the calculations used to generate the following results. In this preliminary work, a Dirichlet mixture was used to determine the values for φ, which was combined with q to construct ρ. The Dirichlet values were drawn from earlier work by Brown et. al. [44]. The actual values used are available in *Table 3—1* and *Table 3—4*.

Model would be exhaustively scanning a multiple alignment for possible paired column duos.

[23] For leaf nodes of a Markov Tree, a node is considered valid if its corresponding nucleotide duo is valid. For internal nodes, the node is valid if either of its children are valid. Please refer to footnote *15* on page *47* for calculation of the inside probability distribution when one child is invalid. Please refer to footnote *20* on page *65* for calculation of outside probability when a sibling node is invalid. State transition frequency estimations omit from summation any fractional state transition to an invalid child.

Left Base	Right Base			
	A	C	G	U
A	0.160	0.135	0.193	1.591
C	0.177	0.135	3.404	0.163
G	0.219	1.719	0.247	0.533
U	2.616	0.152	0.784	0.249

Table 3—1: Nucleotide Pair Relative Frequency

This distribution is renormalized to compute the Dependent model φ for the preliminary Q Model study.

Base Name	Relative Freq.
A	0.26
C	0.21
G	0.18
U	0.20

Table 3—4: Nucleotide Relative Frequency

This is renormalized to compute the Independent or null model φ for the preliminary Q Model study.

As there was no D_{train} available at this point in the experimentation, $\text{Model}_{\text{Rand}}$ was constructed by calculating φ using the relative individual nucleotide frequency from

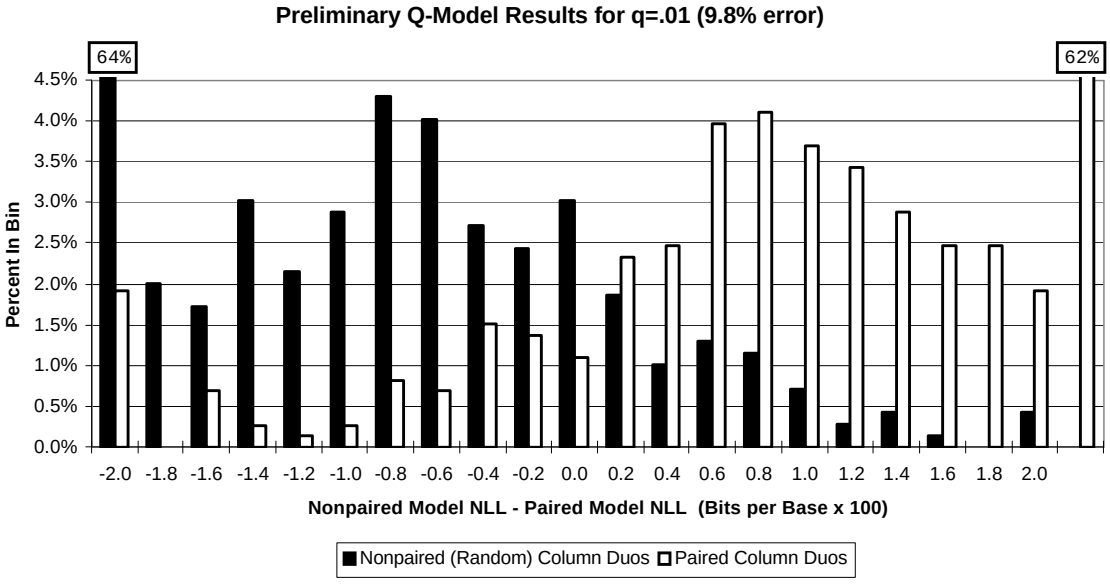

Figure 3-1: Preliminary Q Model Study - Result Sample for q=0.01

The Negative Log Likelihood (NLL) scores reported here represent the difference in mean values or, $\text{NLL}(P(d|\text{Model}_{\text{Rand}}))$ - $\text{NLL}(P(d|\text{Model}_{\text{Pair}}))$ for a given column duo *d*. Note that most of the column duos are at the far edges of the graphs where the bars go off the end of top of the y-axis by more than an order of magnitude. Classification error is calculated as the total overlap between these the two distributions divided by 2 (see *Equation 3-1: Preliminary Q Study Error* Calculation).

Table 3—4 under the assumption of nucleotide independence[24]. $Model_{Pair}$ was constructed by taking φ to be the nucleotide duo distribution for paired duos found in *Table 3—1*. For each of the two models, $-\log_2(P(d|tree \wedge Model))$ was calculated for each column duo (d) in both the Pair and Rand data sets. For each d, the NLL value under $Model_{Pair}$ was subtracted from the NLL value under $Model_{Rand}$. This yielded a net score that was expected to be less than zero for nonpaired column duos, and greater than zero for paired column duos[25]. Each net NLL score was then divided by the number of valid bases in valid nucleotide duos for the column duo to determine the mean NLL score in units of bits of per base.

The above graph (*Figure 3-1*) represents the NLL scores for q=0.01, as applied to $Model_{Rand}$ and $Model_{Pair}$. Similar NLL score distributions were calculated for each of several values of the model parameter q. To determine the error rate for each distribution, the percentage of overlap between the distributions was calculated as follows.

N	Number of Bins of NLL scores.
Un(n)	Percentage of unpaired data that is in bin n.
Pr(n)	Percentage of paired data that is in bin n.

$$\text{Error Rate} = \frac{1}{2} \cdot \sum_{1 \le n \le N} \text{Min}(\text{Un}(n), \text{Pr}(n))$$

Equation 3-1: Preliminary Q Study Error Calculation

[24] The nucleotide independence assumption is that for a nucleotide duo xy, $P(xy) = P(x) \cdot P(y)$.

[25] If $P(d|Model(Pair)) < P(d|Model(Rand))$, then $NLL(P(d|Model(Pair))) > NLL(P(d|Model(Rand)))$ and thus $NLL(P(d|Model(Rand))) - NLL(P(d|Model(Pair))) < 0$. This is basically a posterior probability classifier.

These error rates were calculated for each *q* value (0.00001, 0.0001, 0.001, 0.01, 0.1, 0.5, 0.9 and 0.99). The results of these error calculations are as follows (*Figure* 3-2):

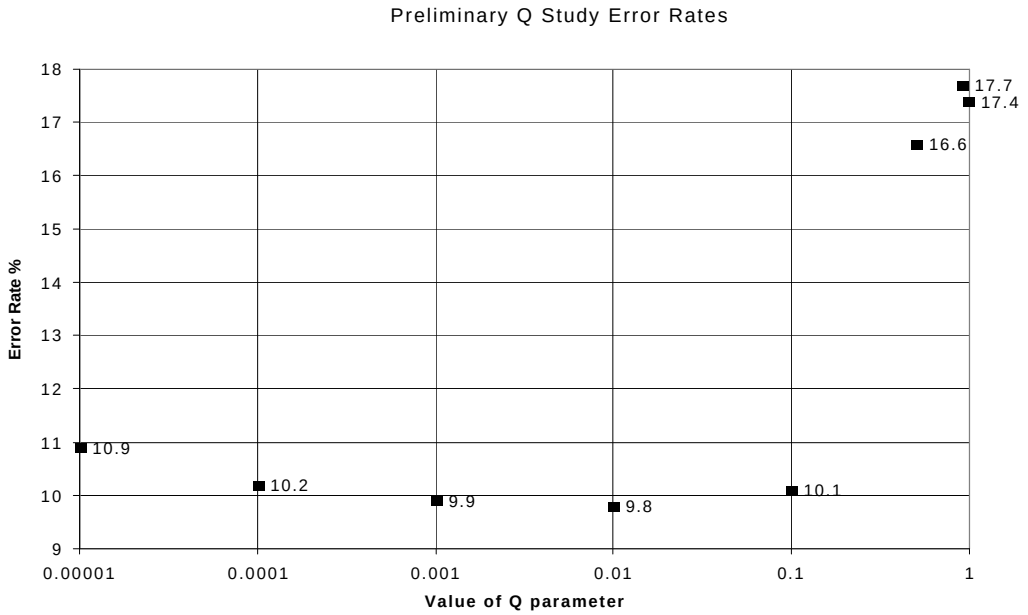

Figure 3-2: Preliminary Q Model Error Rates

According to the above, the value of $q = 0.01$ was found to provide the lowest error rate. Consequently, this value was used as an initial estimate of the value of *q* for the following Q Model studies.

3. 4 Secondary Data Preprocessing

After the preliminary Q Model studies were completed, both the Rand and Pair data sets were filtered again; this time to insure uniqueness and remove column duos that

were inverses of one another[26]. This additional filtering did not effect Rand, which remained at 695 column duos. However, by eliminating inverses in Pair, that data was effectively halved, leaving 317 column duos. These remaining duos were oriented consistently, with the column having the lower column ID number to the left. This is considered biologically plausible due to the inherent direction in an RNA nucleotide sequence induced by asymmetry in the molecule's phosphate backbone.

To address concerns of over-fitting in the more complex IO and IOM Models, it was decided that cross validation should be implemented. To this end, Rand and Pair were each divided randomly into 4 sets of approximately equal size (Rand: 173, 173, 173, 176; Pair: 79, 79, 79, 80). Cross validation was implemented by training on three of the four sets, and then validating on the fourth. Each possible combination of three training sets and one validation set will be referred to as a partition. Partitions are numbered Rand1-Rand4 and Pair1-Pair4. A partition's number corresponds to the number of that partition's validation set.

The training and testing of the nonlinear classifier (see *3.5 Classifiers*) required training examples, as well as test examples from both Rand and Pair. This provided sixteen groups of train/test data, one group for each combination of one Rand and one Pair partition. The same three sets from each partition that was were to derive a Model were also used to train the classifier. The fourth set from each partition served as the validation (test) set for both the model, and the classifier. This preserved validation

[26] A column duo is stored as an ordered 2-tuple of column identification numbers x,y. If y,x is also found in the data set, it was removed. All tuples were then arranged with the lower column identification number

integrity while allowing each column duo to serve as validation data for some training set.

3. 5 Classifiers

Once a Model had been constructed from a training set, it was used to produce likelihoods for each $d \in D_{test}$, P(d|Model∧Tree). Up to this point, d was classified based on the Model that provided the higher likelihood. If P(d|Model$_{Rand}$) > P(d|Model$_{Pair}$) then d was classified as nonpaired, otherwise it was classified as paired[27]. This classification scheme will be referred to as the "simple discriminator". During the preliminary Q

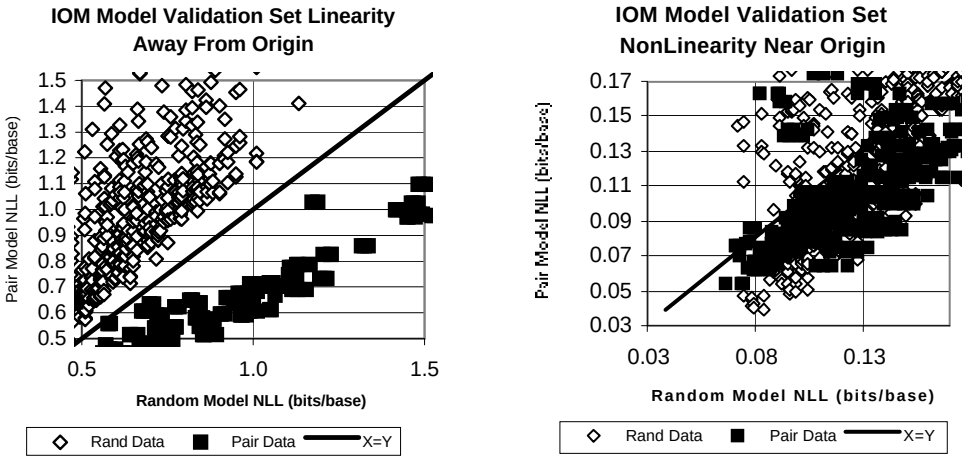

Figure 3-3: Typical Nonlinearity Near Origin for NLL Values

While a direct comparison of posterior probabilities is suitable for tuples with relatively small posterior probabilities (large NLLs), it serves as a poor classifier for higher probabilities (small NLLs). These graphs represent the probability generated for each given column duo by each model . The X=Y line represents the boundary for the simple classifier. These plots contain about 50% of the total data from the IOM Model calculations. For a complete chart of these results which includes the data in this chart, please see *Figure 3-11: IOM Model Results Graphical Summary.*

first, i.e. x,y where x<y. Column duos of the form x,x were removed as well.

[27] Please see sections *2.2.2 Discrimination* and *6 Appendix B: Posterior Probability Classifier for IO* for a justification of the use of likelihoods (P(d|Model)), rather than posterior model probabilities (P(Model|d)).

Model study, it was found that there were certain nonlinearities in the distribution of the data points that might foil the simple classifier (see *Figure 3-3*). In particular, NLL values from both data sets tended to be unexpectedly low, according to $Model_{Pair}$, when the data was highly probable according to both Models. This did not come as a complete surprise as the data sets contained a number of highly conserved column duos that are assigned high likelihoods under both $Model_{Pair}$ and $Model_{Rand}$. These dada are nearly indistinguishable under the Tree Model. The problems arising from this sort of data are discussed at considerable length in *4 Discussion and Conclusion*

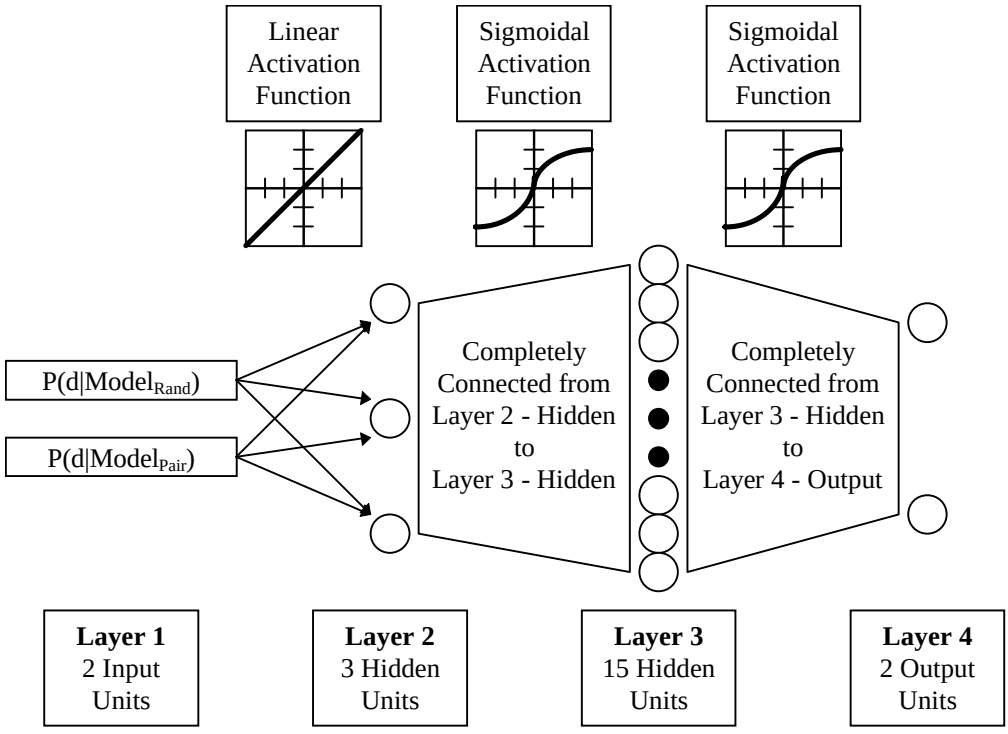

Figure 3-4: Neural Net Discriminator

A four layer, feed forward, neural net is used to aid in discrimination. Raw NLL values are fed into the input units. The linear activation function in Layer 2 serves to rescale the NLL scores. Layers 3 and 4 provide the actual discrimination computation. Learning is accomplished through a classical back propagation technique. The Output layer training values are tuples of either (0,1) or (1,0) which indicate paired or nonpaired data respectively. Novel inputs produce a tuple of real numbers that represents the strength of the net's belief that the input data are from a paired or nonpaired column duo respectively.

To overcome this difficulty, a second, nonlinear discriminator was implemented as an artificial neural network (ANN or "Neural Net Discriminator", see *Figure 3-4*). This class of discriminator is capable of finding locally optimal classification strategies given even a strongly nonlinear training set. In addition, the classification strategy found by the neural network was derived from a training set of exemplars and required virtually no manual intervention.

The primary difficulty in implementing the neural network discriminator was the selection of training and test data. Since this type of network required examples from both Rand and Pair, careful preparation of training and test set partitions was required to prevent any contamination of the validation data from the training set. The network was trained and tested 16 times, corresponding to each of the 16 possible combinations of Pair & Rand data partitions (see *3.4 Secondary Data Preprocessing*). Network training was performed on the same elements used to train Model, and tested on Model's validation column duos.

For each individual train/test cycle, each element of the ANN training set consisted of two NLL values for *d* according to $\text{Model}_{\text{Pair}}$ and $\text{Model}_{\text{Rand}}$ and an additional 2-tuple containing the correct classification of *d*. This 2-tuple consisted of (1,0) if the *d* was nonpaired column duo and (0,1) if *d* was paired. After training was complete, the classifier would accept two NLL values generated by $\text{Model}_{\text{Pair}}$ and $\text{Model}_{\text{Rand}}$ for a novel column duo *d*. The ANN then produced a real valued output 2-tuple (X,Y) where X was the network's belief that *d* was not paired and Y was the network's belief that *d* was paired. Due to the network architecture and choice of

training techniques, X and Y fell within the constraints that $0 \leq X,Y \leq 1$ and $X + Y = 1$. Final classification of *d* was done by comparing X and Y and predicting that *d* belonged the data class corresponding to the output with the larger magnitude.

The Neural Network Discriminator performed as expected, providing superior discrimination performance on all of the training data (up to 26% fewer errors). The ANN also significantly improved discrimination performance on validation data for the IO and IOM Models (up to 17% fewer errors). However, the ANN actually decreased the validation set classification accuracy for the Q and Frequency Models (13% & 5% more errors). This degradation in validation set discrimination is attributed to the ANN's over fitting of the relatively simple Q and Frequency Model training sets. Please see the following table for a more complete description of the ANN's performance.

Model Type	Mean Training Error %			Mean Validation Error %		
	Simple	ANN	Δ	Simple	ANN	Δ
Frequency	13.74	11.88	1.86	14.38	15.09	-0.71
Q	14.87	13.02	1.85	14.77	16.70	-1.93
IO	8.65	6.37	2.28	11.27	9.34	1.93
IOM	9.42	7.21	2.21	12.38	10.50	1.88

Table 3—2: Discrimination Error for Simple Classifier vs. ANN Classifier

All numbers listed are percentages. Numbers listed under "Simple" and "ANN" columns are the misclassification rates of each discriminator, for each evolutionary model. The "Improve" column is the percentage reduction in the misclassification rates from the Simple classifier to the ANN classifier. In every case, the ANN classifier decreases discrimination error against the Simple classifier for the training set. However, the validation data classification error rate increases for the Q & Frequency Models, presumably due to the over learning on the part of the ANN model.

As an in depth treatment of Artificial Neural Networks (ANN's) is not within the scope of this work, interested parties are referred to [45] and [46] for a general treatment of ANN's. The network configuration and computation was programmed in PlaNet [47].

3. 6 Results Format

The results for each Model class (Frequency, Q, IO & IOM) are summarized in its own section (*3.7, 3.8, 3.9* & *3.10*). In each of these sections, there is a numerical summary of the results for the corresponding model, as well as a graphical summary. The numerical summary presents the performance of the model in terms of mean posterior probability and classification accuracy, for both the training data and the test data of Rand & Pair. The graphical summary presents a coarse visual representation of each models encoding and classification performance.

The mean data likelihood for each set is represented in units of bits per base (NLL). The lower the NLL value, the greater the mean probability of the data given the model, and thus the better the fit of the model to the data. In general, the NLL values derived from training set data measure how well the model fits the training data. The NLL values derived from for validation data measure how well the model generalized to the training sample's generating population. A substantial increase in mean NLL values between the training set and the testing set is often an indication that the model is over-learning. Over-learning occurs when a model begins to represent statistical fluctuations in the training sample that are not representative of the population from which the sample was drawn.

In the results section for each model, there is also a numerical summary of the classification accuracy for each of the two classifiers described in *3.5 Classifiers.* This summary displays the number of nucleotide duos from each of the Rand and Pair data sets, and how they were classified. Data sets are presented in columns ("Actual") with a differing evaluation model on each row ("Predicted"). This a count in the Pair column and Rand row represents a paired column duo which was classified as nonpaired by the discriminator

Training data is displayed separately from validation data. A substantial drop in classification accuracy from the training set to the validation set is a likely indicator of over-learning by the classifier. This is relevant only for the neural net discriminator as the simple discriminator is not trained. It is realistic to expect the optimal validation classification accuracy to be lower than the neural net training accuracy and higher than the greater of the simple classifier accuracy, and the neural net validation accuracy.

It may seem odd that the total number of column duos represented in the data columns of the classification summaries are greater than the number of column duos in the data set. This is due to the cross validation data partitioning described in *3.4 Secondary Data Preprocessing.* According to this data partitioning, each data type (Rand/Pair) is partitioned four different ways. In each of the four partitionings, all of column duos are separated into two sets with 75% of the data for training and a disjoint 25% for validation. As each of Pair and Rand are configured into four such partitionings, there are sixteen possible combinations of one Rand partition and one Pair partition. A classifier is trained and tested separately for each of these sixteen

combinations. The results are then accumulated for Rand and Pair across all sixteen partition combinations. Separation between training set statistics and validation set statistics is maintained across this accumulation. For the Rand data set of 695 unique column duos, the number of counts in the Actual columns of the classification summary will be 16·(695·25%) = 2,780 validation duos and 16·(695·75%) = 8,340 training duos. Clearly, each validation set column duo is used in discrimination four times, once with each Pair partition. Each training duo is used in discrimination twelve times, three times with each Rand partition. Similarly the number of counts in the Actual columns for the 317 column duo Pair data set are: 16·(317·25%) = 1,268 for the validation set and 16·(317·75%) = 3,804 for the training set.

In addition to the numerical results, the results summary section for each model class contains a chart that summarizes model performance for the validation results for that model. This graph is relatively complex and requires some explanation. As outlined above, each validation set column duo *d* generates four validation probability tuples, for example if $d \in D(Pair)$ then we would have the four tuples $(P(d|Model_{Pair}), P(d|Model(RandX))$ for $(1 \leq X \leq 4)$. Each of these four tuples would be plotted as a separate point with the NLL generated by $Model_{Rand}$ model as the X-axis coordinate and the NLL generated $Model_{Pair}$ as the Y-axis coordinate. As we expect to see data from the paired data set (*d*) generate $P(d|Model_{Pair}) > P(d|Model_{Rand})$, we would expect this data to cluster below the X=Y line[28]. Similarly, we would expect to see data drawn from Rand

[28] We are plotting NLL values rather then probabilities. For a given probability p, $NLL(p) = -\log_2(p)$. Thus a higher probability indicates a lower NLL value. Thus a datum with a high likelihood under a given model will be found near the origin of the axis corresponding to that model.

cluster above the X=Y line. In general, this was found to be the case. In addition, the data was found to cluster strongly near the origin on both axes. To compensate for this, the graphs are plotted in log-log format to provide for a more uniform graphical distribution[29]. To allow for direct comparisons, all of these charts use the same scaling and numerical X and Y-axis range (0.01 to 10). In the case of the Frequency Model results, however, all of the data was found to cluster in a relatively small region, and thus a linear-linear detail of the area of interest is included.

In all of these charts, there is a secondary Y axis providing some additional information about the accuracy of the simple classifier. The separation between the Rand and Pair data sets seems to decrease closer to the origin. This seems to indicate that the classification accuracy could be increased by ignoring certain data elements that had high probabilities according to both $Model_{Pair}$ and $Model_{Rand}$. This would correspond to a three way classification scheme that would classify each column duo *d* into one of three classes: Paired, Rand and Unknown. The thick gray line labeled Accuracy on *Figure 3-5:* Sample Graphical Summary, indicates the classification accuracy of the Simple Classifier. Each point on this line represents the cumulative classification accuracy for all data to right of that point (duos with larger $Model_{Rand}$ NLL values). This line is used to show how model classification accuracy improves as data with lower Rand Model NLL values are excluded from the classification validation set.

[29] Due to a 4000 data point limitation in the graphing software (Microsoft's Excel), 25% of the data tuples were removed at random to keep the number of plotted points below the maximum allowed. As these points were selected at random, their removal is not expected to affect the shape of the distributions, though there is a chance that individual, potentially interesting, outliers might be absent.

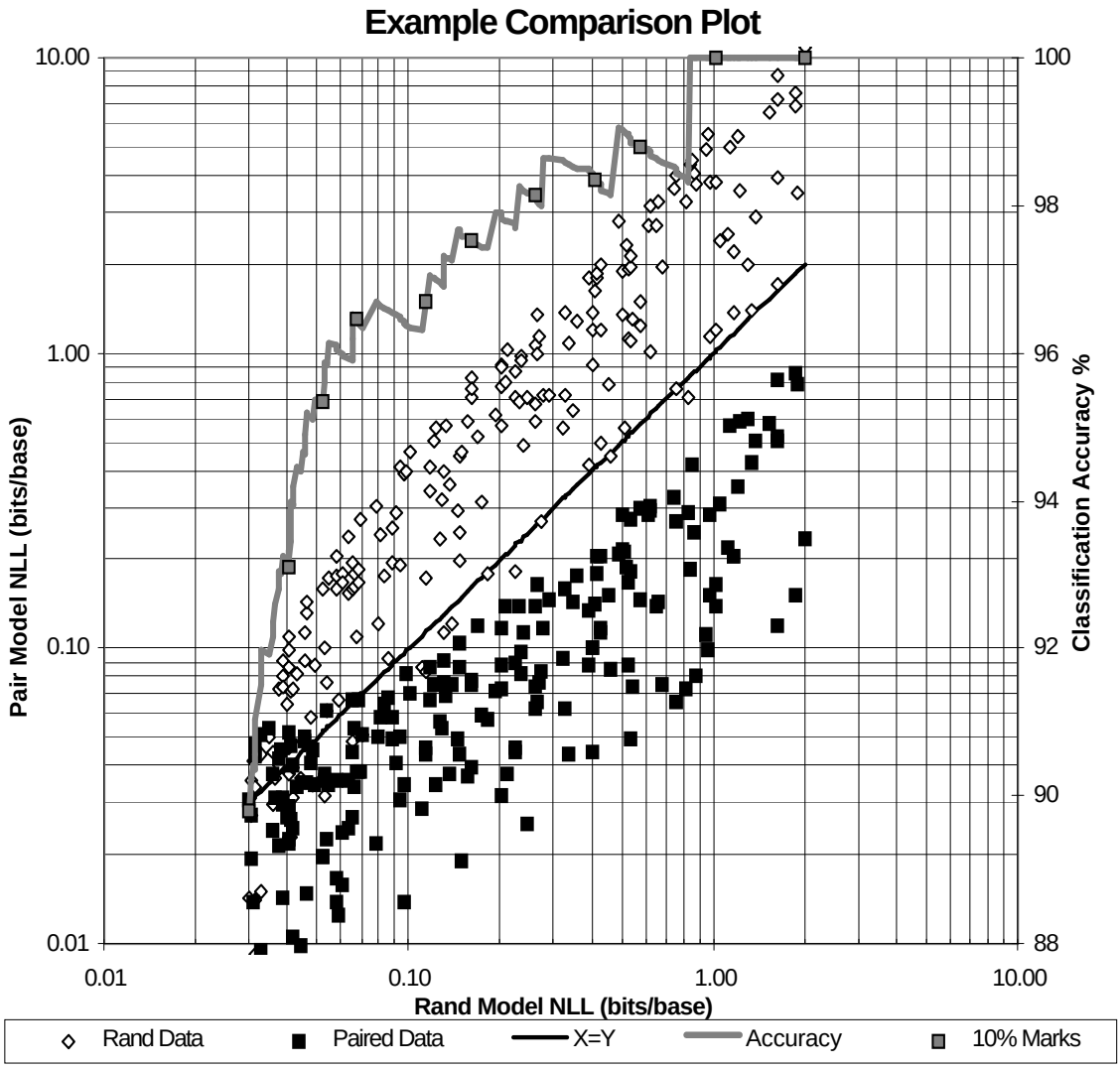

Figure 3-5: Sample Graphical Summary

The data for this plot (400 duos) was generated synthetically to highlight the chart chart's characteristics. For actual Tree Model data, a tuple of per base NLL values are computed for each column duo according to $Model_{Pair}$ and $Model_{Rand}$. A simple classifier assigns the data to the model with the higher probability (lower NLL value). The area above the X=Y line is the region of Rand Data classification, and the region below the X=Y line is the region of Paired Data classification. The synthetic data above does reflect lower classification accuracy near the origin. Each point on the gray line represents the classification accuracy (right hand Y-axis) of all data points to its right, thus showing how the simple discriminator's accuracy increases as points near the origin are excluded as ambiguous. The gray squares are dividers that mark every 10% of the data points excluded (every 40 data points).

This use of Random Model NLL scores may seem arbitrary, however, there is a theoretical motivation for it. The Random Model NLL value for a column duo *d* is a

reasonable, though crude, indication of the number of mutations that *d* has undergone during its evolution. Because the Tree Model bases its calculations predominantly on the mutation process, column duos that are highly conserved and thus experience few evolutionary changes, may be far more difficult to classify then those which mutate frequently. By this reasoning, such slowly mutating column duos might be excluded as classification candidates.

As the $Model_{Rand}$ is trained from a set of non-paired columns drawn randomly from the multiple alignment, it is a reasonable approximation of the mean evolutionary process for a given mutation model class (Frequency, Q, IO or IOM). By measuring the state transition rates from the IO Model, the mean mutation rate per branch of the phylogenetic tree is calculated to be approximately[30] 0.03. Unlike the *q* parameter in the Q Model, this is the *a priori* probability that over a given phylogenetic tree branch a given parental nucleotide pair will change to a *different* nucleotide pair in a given child. Any evolutionary model with similar mutation rates will thus penalize the probability of column duos that have high mutation rates with high NLL values. The magnitude of this penalty will be approximately $-\log_2(0.03) = 1.58$ bits per mutation. Assuming that the distribution of individual nucleic acids is relatively uniform, each base that does not mutate would contribute approximately $-\log_2(1.0-0.03) = 0.044$ bits per non-mutation branch. As the Phylogentic Tree is a binary tree[31] there are approximately twice as many

[30] For brevity, the calculation of this value is not shown. It is derived from the state transition counts for training set 1 of the Rand data. The nucleotide conservation rate was 96.51%, and thus the mutation rate was 3.49% that is approximated as 3% for this rough calculation.

[31] Each node is either a leaf node or has exactly 2 children. Thus, if there are N leaves there are N-1 internal nodes, for a total of 2N-1 nodes and 2N-2 branches (transitions).

branches as there are leaves, and each valid leaf contains two bases. Thus, to the extent that the mean NLL per base over a column duo is greater than

$$0.044 \text{ bits/branch} \times (\#\text{ of branches} / \#\text{ of bases}) \approx 0.044 \text{ bits/base},$$

the NLL of the column duo derives predominately from the mutations encountered, rather than the static nucleotide composition of the columns[32]. As the Rand Model NLL values are uniformly above 0.03 bits/base, and the vast majority are above 0.044 bits/base, they are a reasonable estimator of the number of mutations encountered in a column duo's evolution. These NLL values are thereby anticorrelated with the strength of the conservation in the column duo. The interpretation of $\text{Model}_{\text{Rand}}$ NLL value as a measure of genetic stability, is invoked to construct the accuracy line, which successively excludes the most conserved remaining data (as measured by random model NLL on the X-axis of the graph) as the line progresses to the right. The increase in discrimination accuracy with increasing exclusion of conserved column pairs may be a more informative measure of the resolving power of the model, then the model's overall accuracy. To facilitate the interpretation of this accuracy line, square markers are placed for each 10% of the data that is excluded. Thus, if there are 100 data points, solid squares will be found on the cumulative accuracy line at the first X-axis data position (0% excluded), the 11'th (10% excluded), the 21'st (20% excluded) and so forth until the last at the 91'st position (90% excluded).

[32] Plus a small factor to determine the initial configuration of the system. This decreases quickly as 1/N where N is the number of evolutionarily unchanged base duos. As this approximation is for columns that are highly conserved, and thus relatively stable over evolutionary periods, this factor is neglected.

3. 7 Frequency Model

The Frequency Model is the simplest model class. It is used as a "null" model against which to compare other models. The Frequency Model computes posterior probabilities for a given column duo d based on the assumption that all nucleotide duos $d \in D$ are independent observations drawn from the distribution $\varphi_l(D_{train})$, $P(d|\text{Model}) = \prod_{d^s \in d} \varphi_{d^s}$, or alternatively $\text{NLL}(P(d|\text{Model})) = -\sum_{d^s \in d} \log_2(\varphi_{d^s})$. As in all models discussed in this work, φ_l is the nucleotide frequency distribution found in the training set for a given data class (Rand or Pair). If $d \in D_{train}$ then the nucleotide duos $d^s \in d$ were included in the calculation of φ. If $d \in D_{test}$, or from the data class opposite that of d, then d is not used in the calculation of φ. The independent model mentioned below differs from the dependent model in the calculation of φ. In the independent model, the probability distribution over nucleotide duos is computed from the marginal probability distribution over individual nucleotides, assuming that the marginal nucleotide distributions are statistically independent. Thus, for a given nucleotide duo xy which corresponds to state l, $\varphi_l = P(xy) = P(x) \cdot P(y)$. An independent model is constructed for the Frequency Model only as a verification of the hypothesis that randomly selected column duos will produce a nucleotide duo distribution that is similar to the independent joint distribution of individual nucleotides. This hypothesis is borne out by the following tables that show that NLL scores generated by the independent and dependent $\text{Model}_{\text{Rand}}$ are quite similar.

Frequency Model (dependent)				
NLL (bits/base)	Validation Set		Training Set	
	Rand Data	Pair Data	Rand Data	Pair Data
Rand Model	1.996	1.906	1.978	1.906
PairModel	3.007	1.390	3.007	1.244

Frequency Model (independent)				
NLL (bits/base)	Validation Set		Training Set	
	Rand Data	Pair Data	Rand Data	Pair Data
Rand Model	1.990	1.960	1.981	1.960
PairModel	2.059	2.037	2.060	1.887

Table 3—3: Frequency Model NLL Summary

One common null model used in NLL calculations for RNA is to merely assume that all nucleotides are drawn independently from a uniform distribution. This would yield a per base NLL value of $-\log_2(1/4) = 2$ bits/base. It is interesting to note that the Pair Model statistics are an extremely poor representation of the Rand data class, requiring 3 bits/base for the data. This is significantly worse than the simpler 2 bits/base null model. This distribution is so bad that the independent probability distribution generated by the Pair Model is a better measure of the Rand data class (2.1 bits/base) than the dependent model (3.0 bits/base). The 2 bits/base model is upheld by the statistics generated by the Rand Model indicating a nearly uniform NLL score of 1.9-2.0 bits per base over all data sets. In the case of a model being applied to its own training data, the NLL scores in *Table 3—3: Frequency Model NLL Summary* may be interpreted as the entropy, or self information content, of that training set. For the pairwise model, this is 1.3 bits/base, indicating a strong, but not perfect trend towards pairing in nucleotide duos in paired column duos.

Frequency Model Classification Accuracy - Simple Discriminator				
	Actual (train)		Actual (validate)	
Predicted	Rand	Pair	Rand	Pair
Rand	6912	241	2291	89
Pair	1428	3563	489	1179
Accuracy	86.26%		85.72%	

Frequency Model Classification Accuracy - Neural Net Discriminator				
	Actual (train)		Actual (validate)	
Predicted	Rand	Pair	Rand	Pair
Rand	7382	485	2459	290
Pair	958	3319	321	978
Accuracy	88.12%		84.91%	

Table 3—4: Frequency Model Classification Summary

The classification accuracy table (*Table 3—4*) shows some unusual features as well. This table shows that the classification accuracy for the nonlinear classifier (Neural Net Discriminator) is actually worse than the simple classifier for the validation set. While no definitive explanation for this was sought, it seems likely that this was due to over fitting of the training data by the network. As shown in *Figure 3-6: Frequency Model Results Graphical Summary (detail)*, the clustering of data is fairly straightforward. The Neural Network complexity needed to fit the more complex IO & IOM Model results, probably goes to fit noisy fluctuations in the training set. This enhances training set discrimination at a cost in validation set accuracy. The over-fitting hypothesis is supported by the neural network classification accuracies that are higher than the simple classifier for the training sets, but slightly lower for the validation sets.

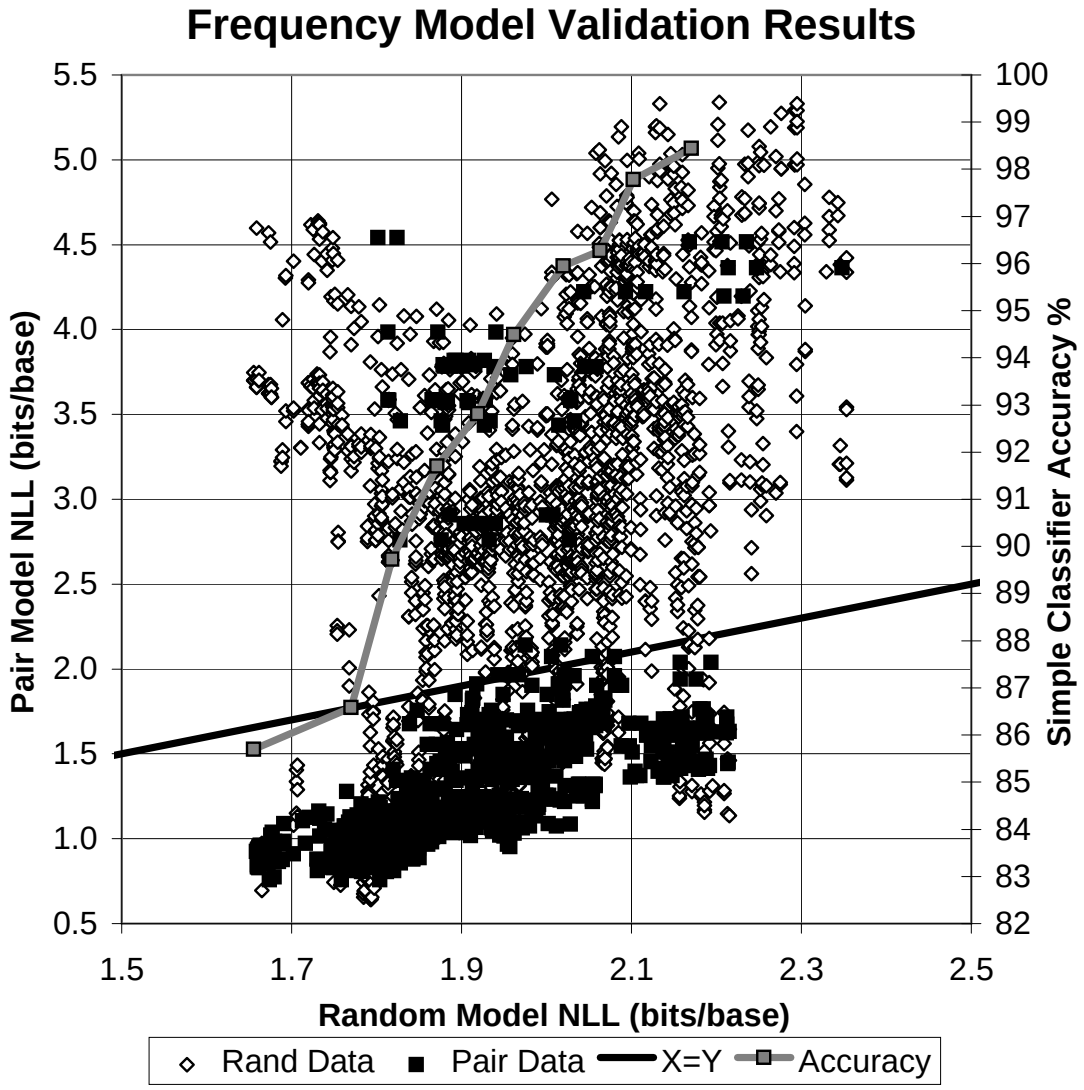

Figure 3-6: Frequency Model Results Graphical Summary (detail)

See also *7 Appendix C: Data Separation Charts* , page 140 for seperation chart of this data.

The above graph (*Figure 3-6*) indicates that the Random represents all of the data in 1.6 to 2.4 bits/base. Much of the discrimination capability comes from the paired model statistics that spread column duo encodings from 0.5 to 5.5 bits/base. The results from this model also differ from those of the Q, IO and IOM Models in that they are

clustered in a relative small region. For comparison purposes, the following graph is drawn to the same scale as the graphs for other models (*Figure 3-7*). Clearly, only a tiny fraction of the total scale is used to represent all of the available data.

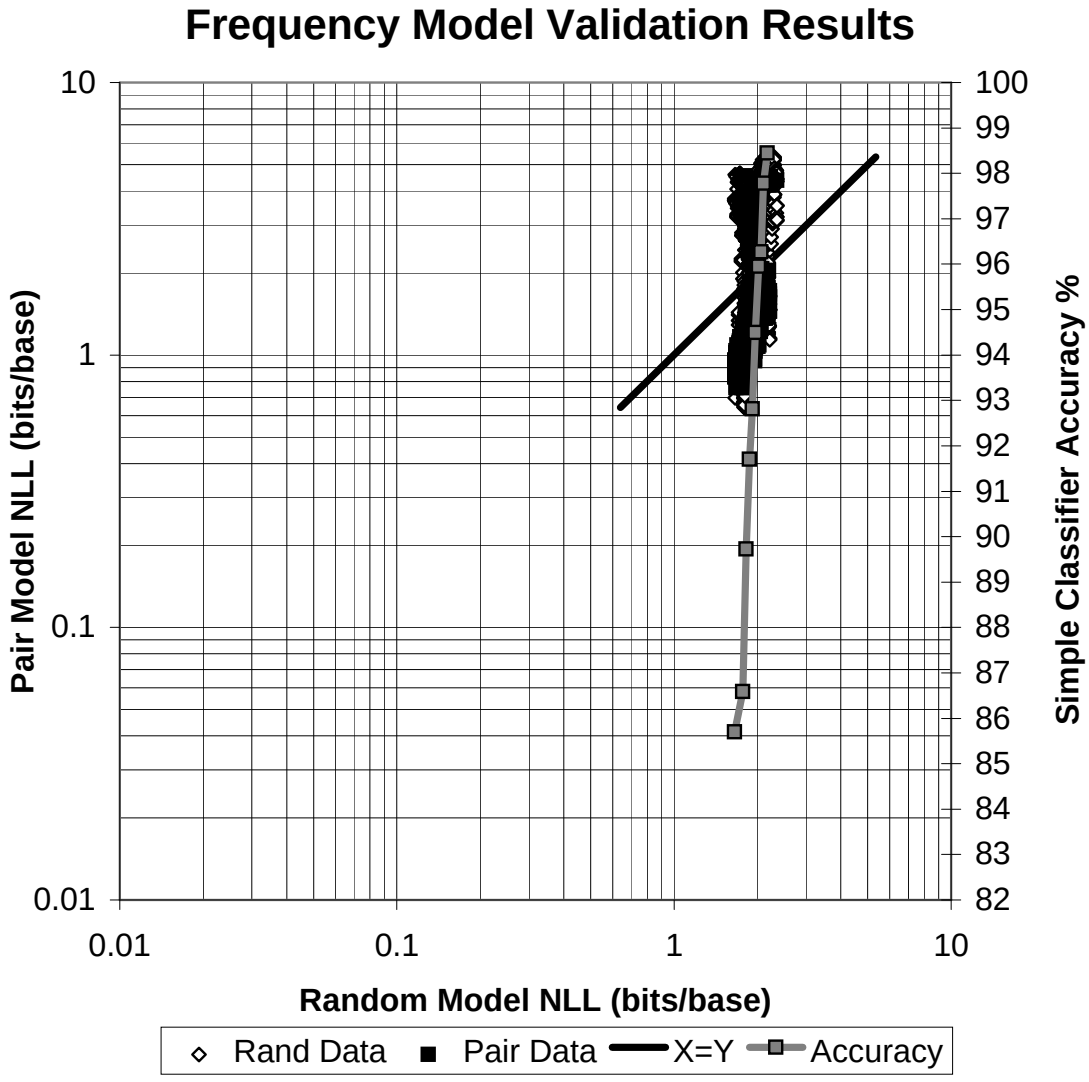

Figure 3-7: Frequency Model Results Graphical Summary

3. 8 Q Model

The results from *Figure 3-2: Preliminary Q Model Error Rates* indicate that the optimal value for the *q* parameter is around 0.01. To the order of magnitude calculations that were performed, this is consistent with the mutation rate parameter of 0.03 obtained from the IO Model in *3.5 Classifiers*. An additional exploratory value of *q*=0.0001 was attempted and found to have a marginally better cross validation discrimination accuracy. However it had far higher NLL values, and the difference in classification accuracy was deemed sufficiently small to be insignificant. Thus, the performance of the Q Model at *q*=0.01 is considered to be more indicative of its peak performance. For a summary of results for *q*=0.0001, please see *5 Appendix A*. Due to resource constraints, it was not feasible to explore other values of *q* for this work.

Q Model (dependent)				
NLL	Training Set		Validation Set	
(bits/base)	Rand Data	Pair Data	Rand Data	Pair Data
Rand Model	0.389	0.329	0.389	0.329
PairModel	0.462	0.306	0.462	0.320

Table 3—5: Q Model NLL Summary for q=0.01

The difference in NLL values between the Frequency Model (*Table 3—3*) and the Q Model (*Table 3—5*) are striking. While the Frequency Model provided validation NLLs on the order 2.0 bits/base for Rand column duos under $Model_{Rand}$ and 1.4 bits/base for Pair column duos under $Model_{Pair}$, the Q Model produces 0.39 and 0.32 bits/base respectively. This tremendous savings indicates that even for such a crude local

evolution model as the Q Model, the phylogenetic tree holds a tremendous amount of information.

Q Model Classification Accuracy - Simple Discriminator				
	Actual (train)		Actual (validate)	
Predicted	Rand	Pair	Rand	Pair
Rand	8172	1638	2729	547
Pair	168	2166	51	721
Accuracy	85.13%		85.23%	

Q Model Classification Accuracy - Neural Net Discriminator				
	Actual (train)		Actual (validate)	
Predicted	Rand	Pair	Rand	Pair
Rand	7888	1129	2620	517
Pair	452	2675	160	752
Accuracy	86.98%		83.28%	

Table 3—6: Q Model Classification Summary for q=0.01

The validation set NLLs for a given data set indicate the degree to which a model fits the data population. However, it is the difference between the NLL values generated by $Model_{Rand}$ and $Model_{Pair}$ on the same data that should be indicative of the discrimination power of the Model. For the Q Model, we see a much better fit of the model to the data than we saw in the Frequency Model. However, we see very little differentiation in NLL values produced by $Model_{Rand}$ and $Model_{Pair}$, on the same data. This leaves us with a discrimination capability that is similar to that of the Frequency Model, about 85% accuracy (*Table 3—6*).

As the following graph indicates (*Figure 3-8*), all data points are clustered sharply around the X=Y line, and are somewhat difficult to distinguish. It is thus not surprising under such a simple distribution that the Neural Network's 17% validation set error rate is so much higher than the 13% training set error rate. This 17% validation error rate is even higher than the Simple Discriminator error rate of 15%. This is strong

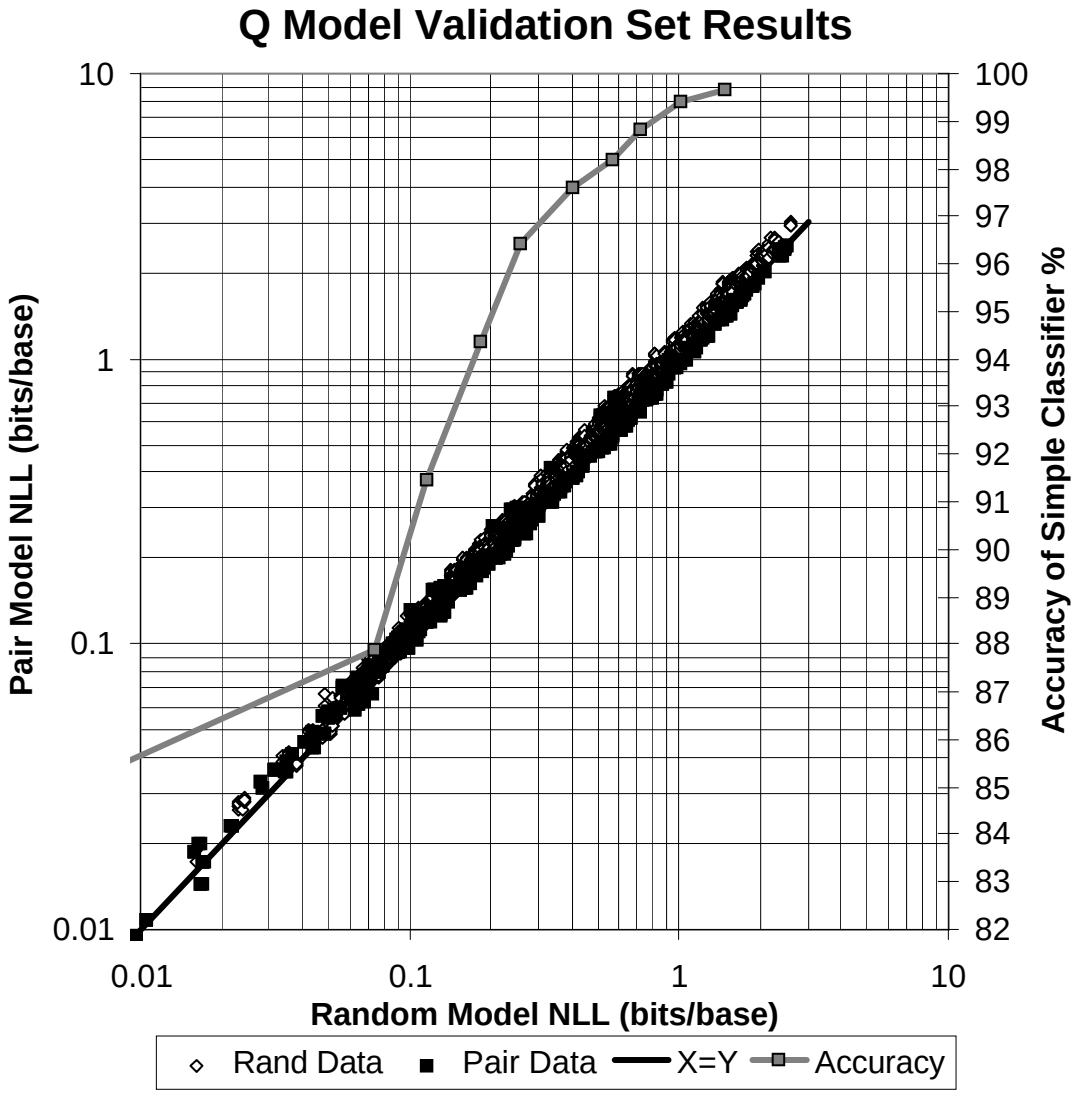

Figure 3-8: Q Model Results Graphical Summary for q=0.01

See also *7 Appendix C: Data Separation Charts* , page 141 for seperation chart of this data.

evidence of over learning in the training of the Neural Network. It might reasonably be assumed that the validation accuracy of an optimal classifier would lie somewhere in the range between the accuracy of the Neural Network on the training data and its accuracy on the validation data.

We can see, from the above chart both data sets are clustered tightly around the X=Y line. This indicates poor distinction between Rand and Pair under the Q Model. What is more difficult to see is that this indeterminacy is most severe near the origin. As points lying near the origin are excluded, the classification accuracy rises from its base rate of 85%, to 89% with 10% excluded, 95% with 30% excluded to 98% with 50% excluded. It is reasonable to ask why these data points are so difficult to classify. Data points near the origin are those with relatively high probabilities according to both models (Rand and Paired). In general, these are the data points with few mutations, as transition to a differing nucleotide pair is a relatively rare event (see *3.5 Classifiers & 3.6 Results Format*). As the Tree Model derives the bulk of its efficacy from making approximations of the evolutionary history of a column duo, those duos with little mutation in their evolution have few distinguishing characteristics. For these column duos, only the composition of the column duo remains as a distinguishing factor. At this point the Q Model is expected to perform differentiation with accuracy similar to that of the Frequency Model. For example, a perfectly uniform column duo provides little information to indicate whether it is composed of two randomly selected columns that are independently conserved, or whether it is a highly conserved paired column duo.

3. 9 IO Model

The IO Model was initialized from the state transition matrix ρ generated by the Q Model with *q* set to 0.0001. This model was then trained through repeated reestimation of the probability transition matrix ρ (*2.4.3 IO Model*) for approximately 10 iterations at which point the change in training set NLL value was less than 0.01%. The resultant values for ρ were then saved and applied to the validation data as described above.

IO Model (dependent)				
NLL (bits/base)	Validation Set		Training Set	
	Rand Data	Pair Data	Rand Data	Pair Data
Rand Model	0.316	0.364	0.313	0.365
PairModel	0.496	0.283	0.495	0.260

Table 3—7: IO Model NLL Summary

The difference between the Q Model (*Table 3—5*) and IO Model (*Table 3—7*) NLL values for each data set on its own validation data is relatively small. The Rand data NLL values dropped from 0.389 to 0.316 bits/base while those of the Paired Data dropped from 0.320 to 0.283 bits/base. This shows nowhere near the dramatic shift that was observed between the Frequency Model (treeless) and the Q Model (uses tree). What is more significant is the change in the separation between mean $Model_{Pair}$ & $Model_{Rand}$ NLLs on the same data. For the IO Model validation set, the difference between the $Model_{Rand}$ NLL and $Model_{Pair}$ NLL was 0.180 bits/base for Rand data and 0.081 bits/base for Pair data. This is a tremendous improvement from the Q Model's

differences of 0.073 and 0.009 bits/base. These differences represent the separation of the centroids of Rand and Pair data clusters, on the graphical summaries. For tightly clustered data, this separation is expected to indicate the resolving power of the model.

Another effect we did not see in either of the simpler Frequency or Q Models is the beginning of over fitting in the NLL values. For $Model_{Rand}$ and $Model_{Pair}$ results we see a degradation in NLL values between the Training Set and the Validation set of 0.003 and 0.023 bits/base respectively.

IO Model Classification Accuracy - Simple Discriminator				
	Actual (train)		Actual (validate)	
Predicted	Rand	Pair	Rand	Pair
Rand	7443	153	2462	134
Pair	897	3651	318	1134
Accuracy	91.35%		88.83%	

IO Model Classification Accuracy - Neural Net Discriminator				
	Actual (train)		Actual (validate)	
Predicted	Rand	Pair	Rand	Pair
Rand	7970	404	2626	224
Pair	370	3400	154	1044
Accuracy	93.63%		90.66%	

Table 3—8: IO Model Classification Summary

As might be expected from the marked increase in data cluster centroid separation, we do see a strong increase in accuracy over the Q Model. Validation set error drops from approximately[33] 15% to 9%. For the first time we also see possible

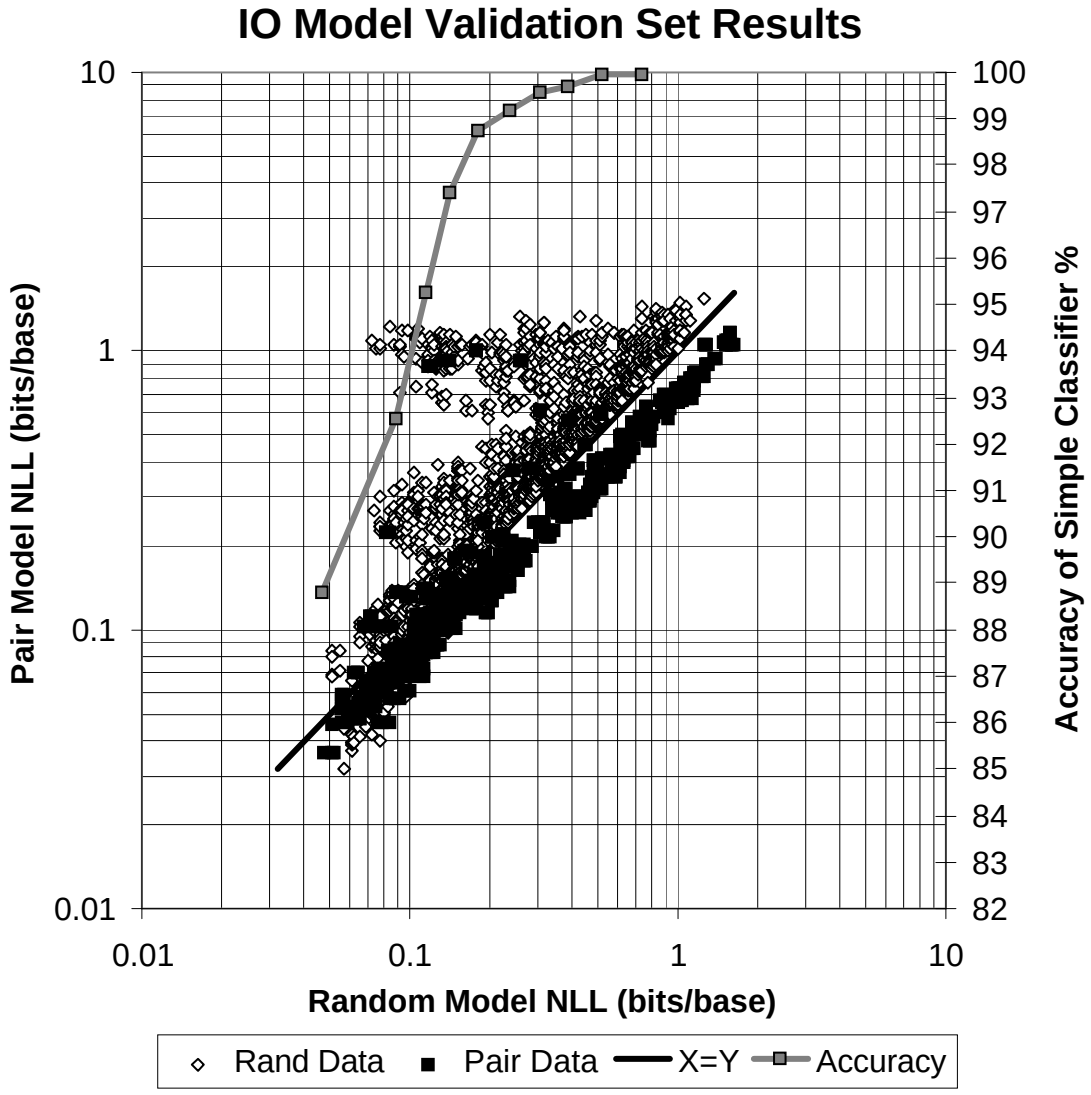

Figure 3-9: IO Model Results Graphical Summary

See also *7 Appendix C: Data Separation Charts* , page 142 for seperation chart of this data.

[33] As the validation accuracy of the Neural Network classifier was lower than the accuracy of the Simple classifier, the accuracy of the Simple classifier was thought to be a more accurate representation of overall resolving power. Thus the simple classifier's statistics are used for comparison.

results of over learning displayed in the Simple Discriminator, whose error rate rises from 9% to 11% between training set and validation set. In the Q and Frequency Models no such change was evident. This is clearly not a result of over-learning in classifier training, as the simple classifier is not trained. Rather, it is more indication of the beginning of some over-learning of the IO Model.

As with the Q Model, the IO Model's classification accuracy rises dramatically as degenerate data near the origin is excluded (*Figure 3-9*). Simple Discriminator classification accuracy rates rise from an 89% baseline, to 92% with 10% exclusion, 97% with 30% exclusion and over 99% with 50% exclusion. It is important to reiterate that the above graph is rendered in log-log scale, thus a spatial difference of 0.003 bits/base near the lower left edge of the data would appear as large as a difference of 1.0 bits/base near the upper right edge of the data. Despite this monumental change in scaling, data points clustered near the origin appear far closer to the X=Y line then those far from the origin. This indicates a tremendous increase in resolving power as the number of mutations in a column duo increase. In particular, the capacity of the Pair Model to reject elements of the Rand Data set has increased, as is indicated by the broad fan of Rand Model data points in the upper left region of the graph. As there is no similar spread of points to the lower right of the X=Y line, it may be conjectured that most of the discrimination is coming from the Paired Model, as the Rand Model fits both Pair and Rand data equally well.

3.10 IOM Model

For the IOM experiments the number of transition matrices was arbitrarily chosen to be 7. This allowed one identity matrix for zero-length branches, one matrix to model outliers with near-zero branch lengths, and five matrices for the bell shaped distribution containing the remaining branch lengths (*Figure 3-10*). Each matrix accounted for 236, 20, 499, 499, 499, 499, 496 branches respectively.

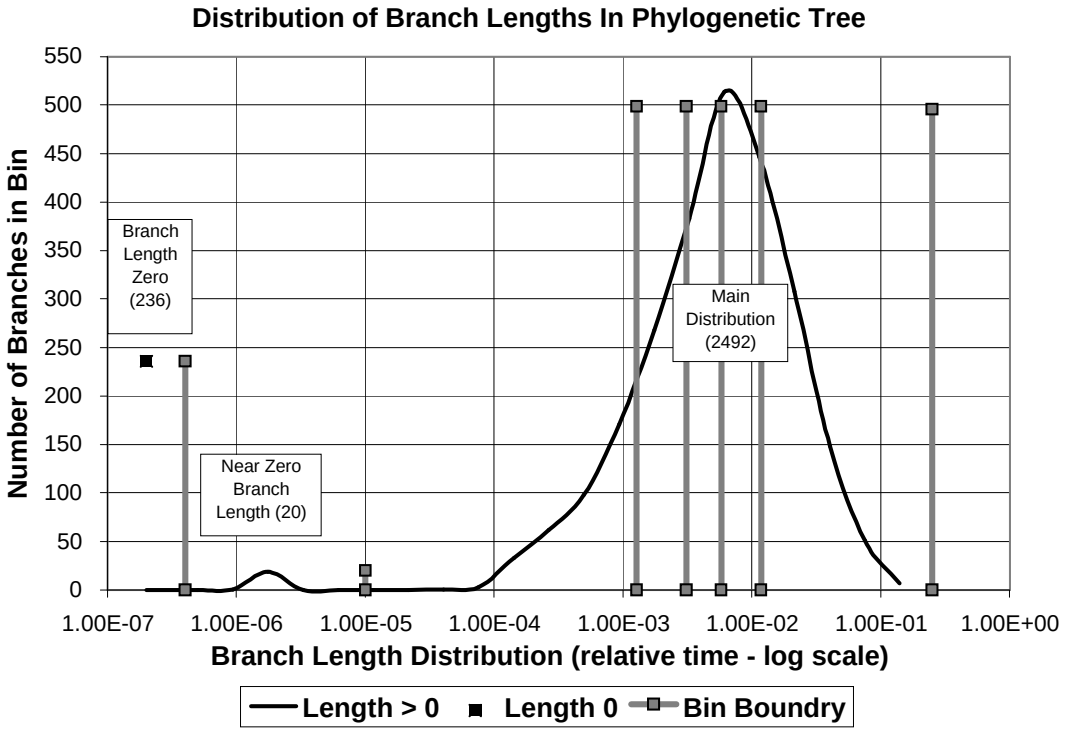

Figure 3-10: Phylogenetic Tree Branch Length Distribution

The dark curve represents the branch length frequency distribution, by length. As the range of branch lengths is broad, a logarithmic scaling is used on the X-axis. This means that bins containing the same number of branches may not appear to have the same area under the frequency density curve. The height of the gray bars may be read on the Y-axis as the number of branches in the bin whose ceiling is the X-axis location of the gray bar. As branches of length zero may not be represented on a logarithmic plot, the length zero bin is represented as a point near 1e-7 on the X-axis.

In the above figure, the X-axis is in a logarithmic scale, thus the area under the density curve may not be equal for bins that contain equal numbers of branches. The height of the gray bin markers is a correct representation of the number of branches in a bin, and the location of a bin marker on the X-axis is the location of the upper boundary of the bin.

IOM Model (dependent)				
NLL (bits/base)	Validation Set		Training Set	
	Rand Data	Pair Data	Rand Data	Pair Data
Rand Model	0.305	0.363	0.302	0.363
PairModel	0.541	0.281	0.541	0.252

Table 3—9: IOM Model NLL Summary

The change in NLL results between the IO Model (*Table 3—7*) the IOM Model (*Table 3—9*) was disappointing. Despite the greater complexity of IOM, and its potentially greater evolutionary accuracy, only modest improvements in NLL values were observed. Validation NLLs for Rand and Pair data sets dropped from 0.313 and 0.260 bits/base to 0.302 and 0.252 bits/base respectively. The separation between mean NLL values between $Model_{Pair}$ and $Model_{Rand}$, over the same data, did increase from the IO modem to the IOM Model. The difference in the mean NLL values for Rand data increased from 0.182 bits/base to 0.239 bits/base, and for Pair data the difference increased from 0.105 to 0.111 bits/base. While this seems to indicate an increase in the resolving power, the following accuracy results show that this is not the case.

A negligible increase in NLL values between training and validation data served to lower fears of over fitting. Despite an approximately 6-fold increase in the number of

model parameters, the difference between Training and Validation NLL values remained constant at 0.003 bits/base between IO and IOM on Rand data. The NLL separation for Pair increased slightly from 0.023 to 0.029 bits/base.

IOM Model Classification Accuracy - Simple Discriminator				
	Actual (train)		Actual (validate)	
Predicted	Rand	Pair	Rand	Pair
Rand	7405	209	2439	160
Pair	935	3595	341	1108
Accuracy	90.58%		87.62%	

IOM Model Classification Accuracy - Neural Net Discriminator				
	Actual (train)		Actual (validate)	
Predicted	Rand	Pair	Rand	Pair
Rand	7917	452	2609	254
Pair	423	3352	171	1014
Accuracy	92.79%		89.50%	

Table 3—10: IOM Model Classification Summary.

While an increasing separation in mean NLLs between $Model_{Pair}$ and $Model_{Rand}$, might be expected to indicate an increase in resolving power, we actually observe a uniform decrease in classification accuracy (*Table 3—10*). Validation set accuracy for the Simple Discriminator dropped from 89% to 88% between IO and IOM, while the Neural Net Discriminator accuracy dropped from 91% to 90%. Though this drop may not seem significant, it is still disheartening given that an increase had been expected.

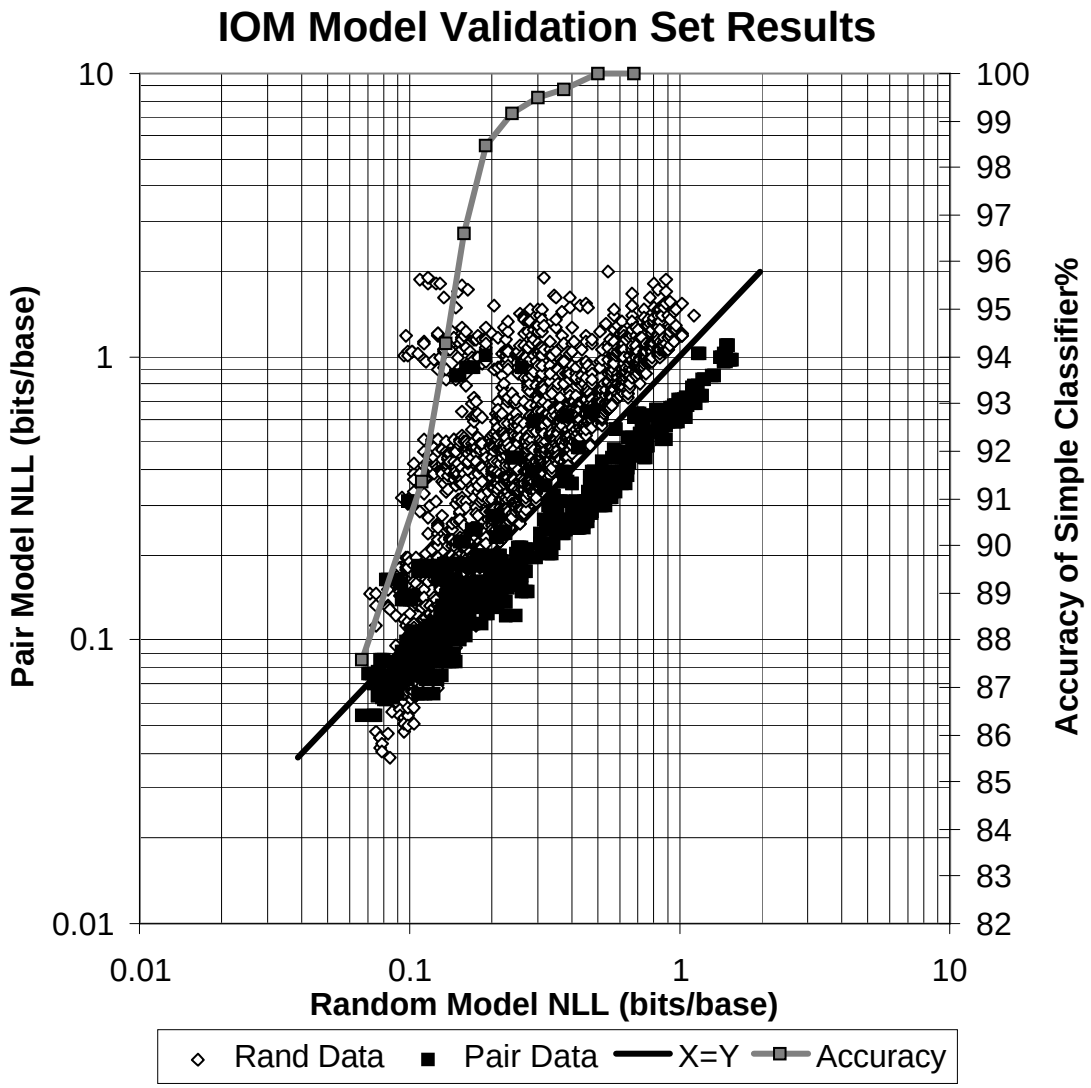

Figure 3-11: IOM Model Results Graphical Summary

See also *7 Appendix C: Data Separation Charts* , page 143 for seperation chart of this data.

A comparison between the scatter plots for the IO (*Figure 3-9*) and IOM Model (*Figure 3-11*) serves to resolve some of the questions regarding the lack of increase in resolving power between the IO and IOM Models.

The fan of Rand points in the upper left corner of the plot is more uniformly distributed in the IOM chart then it is in the IO chart. In the IO chart, more of these points clustered closer to the X=Y axis. This accounts for a greater variance in Rand Model scores. We find that correctly classified points were being driven further into their classification zone (away from the X=Y axis), while the resolution of ambiguous points was not being increased. As most of our errors came from points near the origin, which have low NLL values from both models, the increasing certainty about relatively well classified points served to separate data cluster centroids, without increasing their classification error rate. This is supported by a comparison of the accuracy lines on plots of the IO and IOM data. For small X-axis values, the accuracy line is about 1% lower on the IOM chart then it is on the IO chart. However as more ambiguous points are removed, the difference in classification accuracy quickly diminishes until, at the 50% exclusion marker, the lines meet. At the far right of the chart, IOM accuracy is slightly higher than IO accuracy, with IOM reaching 100% accuracy after 80% of the data has been excluded while IO does not reach this accuracy even at 90% exclusion.

4 Discussion and Conclusion

This final chapter is broken into four sections. In *4.1 Discussion*, we summarize and compare the most important quantitative results from the experimental section. Results from the IO and IOM Models are given the most attention. In *4.2 Algorithm Speed and Size* we provide an informal derivation of the asymptotic resource use of the Tree Model, as well as the actual running time needed to compute the experiments in this work. In *4.3 Author's Note and Conclusion* we review the results of this work qualitatively and informally in the context of modeling processes in general. Finally, in *4.4 Future Directions* we propose some follow-on and closely related research and propose new applications for the Tree Model.

4. 1 Discussion

This subsection provides a comparative summary of our experimental work. First, the issue of discrimination accuracy is addressed focusing mainly on the IO and IOM Models. The major sources of ambiguity in the data are addressed. Next, the NLL values generated by the Models are discussed. As these NLL values are derived directly from data likelihoods, the NLLs represent how well a given model fits a training set (training data NLL) or the population from which the training data is drawn (validation data NLL). This section closes with a reconciliation of the IOM Model's superior data modeling with its less impressive discrimination capability.

It is critical to note that nearly all of the Tree Model classification error comes from the regions where both $\text{Model}_{\text{Pair}}$ and $\text{Model}_{\text{Rand}}$ accept the data with high probability (low NLLs). This effect becomes progressively more evident as the sophistication of the mutation models increases. Overall, the RNA structure is highly conserved across RNA evolution: the IO Model produces a mean phylogenetic tree mutation rate of 0.03 mutation/branch. This mutation rate is consistent with the 0.01 mutation/branch order of magnitude calculation for q. Thus, mutations are relatively rare and much of the NLL value comes from these mutations during evolution. To this extent, the NLL values for the models trained on randomly selected data may be used as a rough approximation of the mutation frequency for a given column duo.

Clearly, the Tree Models discriminate more poorly when the data is relatively uniform, with few mutations and low NLL values. This makes a certain amount of intuitive sense, as the observation of a highly conserved column duo such as AU (a classic Watson-Crick base pair) gives little information as to whether the duo is highly conserved because it is paired, or highly conserved because its constituent columns are conserved independently. While such a conserved column duo will generate a marginally lower NLL under a paired mutation model than under a random one, the difference is small and thus the discrimination ability poor. However, the knowledge of this phenomenon could be used to place confidence bounds on discrimination predictions. Such bounds would be parameterized by the NLL values of $\text{Model}_{\text{Rand}}$, as well as the difference in NLL values between models.

The increase in prediction accuracy with decreasing pair conservation, as approximated by $Model_{Rand}$ NLL, clearly demonstrates the superior resolving power of the more complex mutation models. As each column duo was used exactly once as validation data, there were a total of 1012 column duos used for validation, 317 of these were paired and 695 were unpaired. When 45% of the most conserved data was removed (557 duos remaining), both the IO and IOM Models were able to correctly identify the remaining data with greater than 99% accuracy. The Q Model did not reach this accuracy until 75% of the most conserved data had been removed (253 remaining duos), and the Frequency Model did not reach a 99% accuracy level even after 90% of the most conserved data had been removed (101 remaining duos). This is clear evidence of the accuracy gains that may be realized when phylogenetic bias in the mutation statistics is considered through the use of the Tree Model. These values compare favorably with the 70%-80% accuracies obtained by previous methods of secondary structure detection [32][48].

The use of a neural network as a nonlinear classifier provided some additional accuracy over a simple NLL comparison classifier for the IO and IOM Models. However, error rates for the Frequency and Q Models actually rose when this technique was applied from 14% to 15% and 14% to 17% respectively. As the complexity of the mutation model grew, so did the effectiveness of this nonlinear classifier. The IO Model's error rate dropped from 11.27% to 9.34%, and the IOM Model's error rate dropped from 12.38% to 10.50% accounting for 17% and 15% of the total residual error, respectively (see *Table 4—1*).

Model Type	Training Data Error %		Validation Data Error %	
	Simple	ANN	Simple	ANN
Frequency	13.74	11.88	14.38	15.09
Q	14.87	13.02	14.77	16.70
IO	8.65	6.37	11.27	9.34
IOM	9.42	7.21	12.38	10.50

Table 4—1: Tree Model Classification Error Summary

All numbers listed are percentages. Numbers listed under "Simple" and "ANN" columns are the misclassification rates of each discriminator, for each evolutionary model. This table is excerpted from *Table 3—2: Discrimination Error for Simple Classifier vs. ANN Classifier.*

While there were some concerns of over-fitting the Neural Network training data, the 3% increase in error between the training set and the validation set is also reflected in the simple NLL-based discriminator. This tended to indicate that the mild over-fitting observed was occurring in the modeling of the Markov Tree, rather than in the training

Model Class	Rand Data			Pair Data		
	Training Set (bits/base)	Validation (bits/base)	Δ	Training Set (bits/base)	Validation (bits/base)	Δ
Frequency	1.978	1.996	0.018	1.244	1.390	0.146
Q	0.398	0.398	0.000	0.306	0.320	0.014
IO	0.313	0.316	0.003	0.260	0.283	0.023
IOM	0.302	0.305	0.003	0.252	0.281	0.029

Table 4—2: NLL Overfitting Summary by Model Class

Table shows the mean NLL value (bits/base) for each model on its own data set: $Model_{Rand}$ on Rand Data and $Model_{Pair}$ on Pair Data. The greater the difference in NLL scores between the training set and the validation set, the larger the expected over-fitting. Beyond the Frequency Model, the more complex the model, the greater the number of degrees of freedom and the greater the potential for over fitting. The Q Model has 16 degrees of freedom in q and φ. The IO Model has 255 degrees of freedom in φ and ρ. The IOM Model has 1,455 degrees of freedom in φ and its 6 configurable ρ distributions. The column labeled Δ represents the arithmetic difference between the training NLL value and the validation NLL value. Substantial increases in Δ may represent over-fitting by the Tree Model.

of the classifier. This hypothesis was also supported by the slight increase in the separation between training set accuracy and validation set accuracy for the simple classifier. This separation rose from 2.6% in IO, with 255 degrees of freedom to 3.0% in IOM, with 1,455 degrees of freedom.

In general, IOM showed some improvement in NLL values and no improvement in classification accuracy over IO (see *Table 4—2* and *Table 4—1* respectively). The overall error rate of IOM was marginally higher than that of IO, 10.5% as compared to 9.3%. However, the error rate for IOM did decline faster than that of IO, as more and more of the most conserved column duos were excluded from the test set. The IOM Model actually had a lower error rate than the IO Model after 50% of the most conserved data was removed. In addition, the IOM Model had lower validation set NLL values and greater mean NLL differentiation between $Model_{Pair}$ and $Model_{Rand}$.

Model Class	Rand (bits/base)	Pair (bits/base)
Frequency	1.996	1.390
Q	0.389	0.320
IO	0.316	0.283
IOM	0.305	0.281

Table 4—3: NLL Summary by Model Class

This table summarizes the Validation set results for each model. It is summarized from *Table 4—4* for clarity. The values represent mean bits/base of the validation set for each model. These numbers summarize how well each model represents the data population from which the training sample is derived. Lower values represent a better mean fit, and a higher $P(D_{test}|Model)$.

The mean Rand validation set NLL values were 0.305 bits/base under IOM_{Rand} and 0.541 bits/base under IOM_{Pair} as compared with 0.316 bits/base for IO_{Rand} and 0.496

bits/base for IO_{Pair}. As Rand was given a 9% higher NLL by IOM_{Pair} than by IO_{Pair}, we may infer that the IOM_{Pair} is better able to reject Rand data than IO_{Pair}.

Model Class	Rand Data			Pair Data		
	Rand Model (bits/base)	Pair Model (bits/base)	Δ	Rand Model (bits/base)	Pair Model (bits/base)	Δ
Frequency	1.996	3.007	1.011	1.906	1.390	0.516
Q	0.389	0.462	0.073	0.329	0.320	0.009
IO	0.316	0.496	0.180	0.364	0.283	0.081
IOM	0.305	0.541	0.236	0.363	0.281	0.082

Table 4—4: Summary of Separation of Mean Data Set NLL Values

This table shows the difference in mean NLL score (bits/base) between different models on the same validation data. For the tightly clustered data sets in Q, IO and IOM the larger the Δ, the better the expected discrimination for that data set.

The paired validation data was given a 4% lower NLL by IOM_{Pair} than by IO_{Pair} . This is evidence indicating that IOM is modeling the population from which Pair is drawn better than IO does (see *Table 4—4*). However, this ability to more accurately model the Rand and Pair populations did not translate into higher classification accuracy. This is consistent with the idea that IOM is more sensitive than IO over more volatile column duos, while IOM performs no better than IO on more consistent column duos with few mutations in their evolutionary history.

4. 2 Algorithm Speed and Size

The calculation of the inside probability

$$I_d(A_i = l) = \left[\sum_m \left[I_d(A_j = m) \cdot \rho_{l,m}\right]\right] \cdot \left[\sum_n \left[I_d(A_k = n) \cdot \rho_{l,n}\right]\right]$$

(*Equation 2-1: Summary Derivation of Inside Probability Distribution*)

requires an accumulation of probability from each state (l) of a tree node (A_i) to each state (m & n) of its descendant's nodes (A_j & A_k). If there are N organisms in the phylogenetic tree and S states corresponding to the possible nucleotide duos, then this calculation requires O(NS^2) time for each column duo. If we wish to check every possible column duo in a multiple alignment of M columns, then our running time will be bounded in time by O(NS^2M^2) and space by O(NS^2+M^2). Usually S will remain fixed at 16 for the 16 possible nucleotide duos in RNA. However, if other symbols such as the gap symbol are added to the alphabet, this term may grow.

The calculation of the outside probability

$$O_d(A_j = m) = \sum_l \left[\rho_{l,m} \cdot O_d(A_i = l) \cdot \sum_n \left[I_d(A_k = n) \cdot \rho_{l,n}\right]\right]$$

(*Equation 2-4: IO Model Outside Probability Distribution Derivation*)

seems to require O(S^2) calculations for each state S of each node A_i of the tree, due to its nested calculation over the inside probabilities of A_j's sibling, $I_d(A_k=n)\cdot\rho_{l,n}$,. This would yield a complexity of O(NS^3) for each column duo. However, the nested section of this calculation is also computed during the inside calculation and may be stored for later use. This leaves us with O(NS^2). As the inside calculation, which is a prerequisite, is also O(NS^2) we have a total bound on the time to calculate the outside distribution of O(NS^2) per column duo.

As with the outside distribution calculation, the frequency reestimation formula

$$\hat{f}_{l,m}(d) = \sum_{l}\left[\rho_{l,m}\cdot O_d(A_i = l)\cdot\sum_{n}\left[I_d(A_k = n)\cdot\rho_{l,n}\right]\right]$$

(*Equation 2-5: IO Model Transition Frequency Reestimation Derivation (Part II)*) appears to require $O(S^3)$ operations between each parent and child node ($A_i \rightarrow A_j$ and $A_i \rightarrow A_k$). However, at the heart of this equation is the same nested loop over $I_d(A_k=n)\cdot\rho_{l,n}$ that was found in the outside derivation. This was stored for us during the inside calculation, leaving us with an $O(NS^2)$ calculation for each column duo in the training set.

This leaves us with an NLL evaluation algorithm for column duo *d* that is $O(NS^2)$ in both space and time per column duo. The training algorithm also requires $O(NS^2)$ in space and time for a column duo. However, as this algorithm is iterative, it is not absolutely clear how *N* and *S* effect the number of iterations through algorithm convergence. For the experiments in chapter *3,* the algorithm never required more than a small number of iterations (7-10) over the training set to converge to a change of less than 0.01% in the training set NLL values per iteration.

The Tree Model experiments (*3 Experiments*) were implemented in the Gnu Project's gcc (C++) version 2.5.5 on a Digital Equipment Corporation, DECstation 3000/400 APX (with a RISC microprocessor running at over 150 MHz). The operating system was DEC's OSF/1 Version 3.0. The standard level of compiler optimization was used (gcc -O). The resultant program required approximately 1.9 seconds to produce an NLL value for a column duo of approximately 1000 valid nucleotide duos, under both of the IO & IOM models. The program took approximately 5.5 seconds to generate *f*(*d*) for

a single training set column duo, when no other appreciable activity was occurring on the computer. To generate two sets of transition matrices (Rand and Pair) over a training set size of 521+238 = 759 column duos using 10 iterations of expectation maximization, required approximately 11.6 hours. The generation of the 174+79 = 253 validation set NLL values required an additional 8 minutes. This process was repeated four times, once for each train/test partition. The program required approximately 11 megabytes of memory in which to run, and no excessive virtual memory swapping was observed during program execution.

While the NLL value generation for the validation set was negligible in the present work, an exhaustive search for secondary structure using this technique might prove cumbersome. Given 2,688 columns is 16S RNA, there would be $2{,}688^2$ = 7,225,344 column duos to examine. On the hardware employed here, this would require approximately 159 days of uninterrupted CPU time. This running time is not completely unreasonable, given that the present level of hardware is available as a workstation. Nonetheless, the NLL generation for each column duo may be performed concurrently. Thus, thirty similar workstations should be able to complete such an exhaustive calculation in 1/30 of the time required by one workstation, or approximately 5.5 days.

Our software was not tuned especially well, and software refinement should be able to increase performance substantially. No effort was made to unroll loops, or replace arrays with pointers. In addition, the broad range of probabilities encountered during calculation required a special implementation of floating point numbers. As probabilities on the order of 2^{-1000} are quite commonly encountered during frequency

reestimation, the standard C Language double precision floating point representation was insufficient. To circumvent this problem, a C++ class called *Prob* was obtained from the computational biology group at the University of California at Santa Cruz [49]. This class maintained a logarithmic representation of a number, thus reducing the number's precision, but enhancing its range. This logarithmic representation made multiplication of probabilities a very cheap operation, but addition expensive. While this class aided phenomenally in ease of coding and portability, it might be faster to maintain the probabilities in a NLL form explicitly, without the overhead of an abstract class.

4. 3 Author's Note and Conclusion

Well, here we are, nearly at the end of this work. Before this work continues into *4.4 Future Directions* spending several pages talking about what didn't get done, a little time is spent in this section ruminating over what has been done. If the reader is put off by long sentences, first person narrative, wild conjecture or self congratulatory prose, it is suggested that the reader skip this section and go on to the critiques presented in *4.4 Future Directions.* This section provides an insight into some of the personal motivation for, and achievements in, the completion of this work.

I began this document by introducing a relatively new paradigm into the field of RNA modeling, a field where more traditional techniques rely on physical modeling. Physical modeling requires a very large number of simplifying assumptions in order to model any system larger than a single atom (and a simple one at that). The choice of these simplifying assumptions must be made *a priori* through subjective decisions based

on the particular expertise of the model's builder. From a scientist's perspective, this process is a little like playing the lottery. If a researcher makes the correct assumptions, then the researcher is rewarded with positive results that can bring fame, glory and continued funding. If a researcher guesses incorrectly, they can lead an entire branch of the physical sciences down a blind alley for years. Especially if they are an authority in their field. In addition, extensions can only be made to such physical models by people deeply schooled in a diversity of fields including: numerical computation, modeling theory and the specific physical science involved (which in this case is molecular biology). In the place of physical modeling, I advocate a new paradigm. This paradigm replaces the exciting, technically arcane, labor intensive, fundamentally exploratory and eminently fundable, procedures of physical modeling with a boring, simple, automatic and reliable process of statistical modeling. I certainly hope that this statistical process is fundable as well.

The statistical paradigm directly addresses issues of generalization, over-fitting and data support for model complexity. These issues are implicit in, and ignored by, most physical modeling techniques. The techniques of physical modeling were well suited to situations where a large amount of *a priori* information had to be employed, without the benefit of much data, to model a process of great complexity. However, in the field of molecular genetics, the plethora of sequence information has obviated the physical technique to some extent. Direct probabilistic modeling techniques are now feasible. In a sense, the tables have turned. Formerly the observational statistics of a system were implicit in a model derived from physical properties. Now the physical

properties of a system are implicit in the observational statistics used to derive the model. An exciting reversal of roles!

As the language of machine learning is the very language of statistical inference, it is only reasonable to use machine learning techniques whenever they are supported by a sufficient amount of physical data. There may always be a margin on the frontiers of science where sparse observational data, in the face of complex systems, renders physical modeling mandatory. However, the application of automated data collection techniques to fundamental research is continually narrowing that margin. If the statistical assertions implicit in physical models are made explicit, then physical properties could easily be included as *a priori* information in a statistical model. Sparse amounts of data could then be combined with these statistically represented physical laws, to further shrink the realm over which physical modeling holds primacy.

While the nucleotide base pair detector developed herein is of relatively low complexity, it serves marvelously as an example of this new paradigm. Very little knowledge of chemistry is required to understand the concepts behind the model. While the equations of statistical inference were moderately complex to derive, they require only a cursory grasp of probability theory to understand. The modeling process arises directly from the data, requiring no specialized knowledge, tweaking or tuning[34]. Only three parameters were chosen in an *ad hoc* manner: the number of cross validation partitions (4), the percentage of valid nucleotides required in a valid column duo (75%),

[34] While this is strictly true, the experiments performed herein utilized a 16S multiple alignment that had been carefully constructed by both hand tuning and automated techniques [40]. While the Tree Model

and the number of branch length bins in the IOM model (7). All other relationships in the Tree Model were inferred from the data sets. With somewhat greater resources, all of the arbitrary parameters could have been selected statistically. Finally, potential problems in over-fitting are addressed automatically through cross validation. While the comparison of training values with validation values was treated informally here, there are statistical methods for determining and limiting its effects.

However, all of this wonderful automation comes to naught if the process performs worse than other currently available processes. So, how good is the statistical model provided here? The answer is very good indeed! The secondary structure detector developed here is found to have 90%-99% accuracy in detecting secondary structure. Techniques based on energy minimization and manual phylogenetic analysis have shown accuracies of 70%-80%, and require a substantial amount of experienced manual arrangement. Not only that, but staggering probabilistic gains in model accuracy are hidden in the use of NLL values. The mean Tree Model validation set NLL for a nucleotide was found to be about 0.3 bits, for both paired (0.28 bits) and nonpaired (0.31 bits) nucleotides. This translates into a data likelihood of approximately $2^{-.3} = 81\%$. This is compared to the frequency model that generated mean NLL scores around 1.4 bits for paired nucleotides or 1.9 bits for unpaired nucleotides, or 38% and 27% respectively. If we look at an entire column in a multiple alignment (about 2000 nucleotides) we find that the improvement in data likelihood is $.81^{2000}/.38^{2000} \approx 10^{657}$ for paired nucleotides or $.81^{2000}/.27^{2000} \approx 10^{954}$ for unpaired nucleotides. This is 500 to 1000 *orders of magnitude*

---

could easily be applied a less refined cruder alignment, the accuracy levels may not be as high.

of data probability that was unaccounted for by the Frequency Model[35]. To get such statistics for the entire multiple alignment, we can make the crude assumption that the column duos are all independent. A model assuming some form of dependence should do better, but even under this relatively weak independence assumption we may raise our astronomical ratio to the power of approximately 1000 (column duos per alignment) and arrive at nearly a million orders of magnitude likelihood increase over the entire multiple alignment! This means that the observed multiple alignment data is approximately $10^{1,000,000}$ times more likely under the Tree Model than it was under the Frequency Model, without substantially overfitting the training set. Moreover, this Frequency Model was about as good at ferreting out nucleotide pairing (about 85% accuracy) as current physical modeling techniques (70-80% accuracy) [15][32][50].

These results are generated by a first generation modeling technique. A Bayesian analyst might scoff at the Tree Model's restricted use of prior information. Numerous prior data such as columnar mutation rates and nucleotide dependency between adjacent columns, are not explicitly exploited by the Tree Model. To this I say… Excellent! By all means, go ahead and develop more accurate priors and evolve the statistical paradigm! This model is designed merely as a road sign pointing in an alternative

[35] I do acknowledge a certain amount of hyperbole in this number. The Frequency Model is relatively weak from an information-theoretic point of view as it does not use mutual information from neighboring nucleotide duos within a given column duo, much less any of the more complex frequency weighting schemes [51][52]. The Tree Model does use this information. Recent, and very preliminary, results based on the self-information of each column duo has yeilded some striking results. Though this self-information model attained approximately the same discrimination accuracy as the IO Model, its mean NLL values were much higher. The NLL values were 0.50 bits per base for paired (dependent) nucleotide data and .58 bits per base for unpaired (independent) data. This would reduce the per-column-duo likelihood advantage of the Tree Model to factors of 10^{114} & 10^{164} respectively.

direction to current processes, a relatively accessible first step down a long inferential path.

This is by no means to say that the Tree Model is a toy. While this pair detector has significant uses in the area of RNA base pair modeling, its potential for other sorts of measurement is tremendous. Remember that during the process of deriving posterior probabilities for column duos, we calculate a nucleotide duo state distribution for every ancestor in the phylogenetic tree. This is a statistical depiction of the entire evolutionary process for that column duo, from the primordial cell to each and every observed organism that contributes to the multiple alignment. Merely using this process to compute the posterior probability of the column duos is like swatting a fly with a telephone pole.

Through the use of the probability distributions in this tree, one need no longer be limited in genetic measurements to a single value for statistics, nor a single arguably "correct" ancestral nucleotide configuration. Measurements of genetic quantities such as mutation rates, genetic composition and phylogenetic branching distances, can now be calculated as expectation values over all possible evolutionary developments. Furthermore, instead of calculating a single quantity, one could calculate probability distributions over a range of possible values. The replacement of a single arguable statistic with a distributions over possible values is as fundamental to the analysis of uncertain data as is the concept of the Gaussian distribution.

While the grandiose statements made in this section might seem out of character with the rest of this work… well… they are. The research leading to this work was designed to be conservative and self critical so as to present a lower bound on the reasonable expectations for its type of approach. However, behind every difficult and drawn out research effort, there must be at least a flicker of excitement guiding the work through its darker periods. While much of the motivating enthusiasm can become smothered by the rigorous constraints of academia, I felt that this work would not be complete without a taste of the excitement that drove it to completion. You will now be returned to your regularly scheduled academic skepticism.

4. 4 Future Directions

This section provides some potentially interesting areas for future research and is broken into four sections. In , we look at some additional tests that could be run using the current Tree Model to verify its accuracy and generality. In , we look at some new experiments which might be run with the Tree Model. In , we look at ways that the Tree Model might be theoretically extended. Finally, in , we propose another application for the Tree Model. In general, each idea for further work is kept to its own paragraph and no attempt is made to connect the ideas.

4.4. 1 Tree Model Verification

It would be interesting to apply the models generated with 16S data to 23S RNA. A strong positive result with 23S RNA would provide convincing evidence that the evolutionary model developed here has general validity. More detailed studies of the

tRNA family might as well be able to discern novel tertiary pairing structure, as the tRNA structure is known well enough to provide a training set of tertiary column duos. One preliminary test of this method would be to use the nucleotide duo mutual information content of a column duo as a $\text{Model}_{\text{Pair}}$, and the individual nucleotide mutual information content as a corresponding $\text{Model}_{\text{Rand}}$.

During the work leading to *3.3 Preliminary Q Model Study* several of the nucleotide duos that were known to be paired distinguished themselves by scoring very poorly under Q_{Pair} values of $q > 0.1$. These columns were found to be composed predominantly of non-Watson-Crick base pairs (some of these duos were found to be helix end caps). These were originally used to construct a third model called $\text{Model}_{\text{Exotic}}$. However due to the small number of these samples (approximately 10) as well time constraints, the investigation of $\text{Model}_{\text{Exotic}}$ had to be terminated and these column duos were returned to the Pair data set. It might be of interest to perform a similar experiment by constructing an IO or IOM model solely to represent those column duos from Pair that scored appreciably better under $\text{Model}_{\text{Rand}}$ than they did under $\text{Model}_{\text{Pair}}$. Such an experiment could provide transition matrices that would be particularly useful in ferreting out the most difficult to detect paired nucleotide duos, and thereby reducing false negatives under $\text{Model}_{\text{Pair}}$.

The Frequency Model is a particularly weak null model as it incorporates neither mutual information between nucleotide duos in a column duo, nor any explicit weighting for phylogenetic similarity between organisms [53]. Techniques for addressing these concerns have been explored in the field of protein structure detection [51][52]. These

methods involve weights for column duos based on their degree of nucleotide variability, as well as weighting to help account for the similarity between phylogenetically related organisms (they do not use a full phylogenetic tree). Such techniques could be applied to RNA and used as a stronger non-phylogenetic-tree based opponent for the Tree Model.

It is unclear how much of the effectiveness of the Tree Model stems from our use of a well developed multiple alignment. To test the Tree Model with less refined data, a more primitive multiple alignment could be developed using a completely automated alignment technique such as those found in [18] and [29]. This alignment could then be used to test the Tree Model's performance under more adverse circumstances.

4.4. 2 Experimental Tree Model Extension

The filtering for valid pairs significantly reduced the useful range of testing and training data, especially for the randomly generated data. As it is exactly this data that is to be scanned automatically for pairing structure, some way should be found to expand the field of valid pairs. By far, the most commonly rejected symbol was the gap symbol (-). While a biological interpretation of this symbol is relatively complex, it should not be excessively difficult to incorporate this symbol into the known alphabet. However, while this extension may be technically simple, the inclusion of gap information in the pairing decision process might make the resultant model's decisions too dependent on structural information introduced during the alignment process. In particular, a gap symbol in a multiple alignment does not necessarily correspond to a physical object. Thus, it would be unclear exactly what the inclusion of this symbol would be modeling.

Despite these trepidations, the inclusion of gap information in the terminal alphabet would appreciably expand the range of acceptable data, and would allow the exploitation of evolutionary nucleotide insertion and deletion information. One might reasonably expect that the evolutionary insertion of a gap into a paired column is far less likely to be genetically stable then the introduction of such a state into an unstable column. Thus the evolutionary production of many gaps would be evidence against pairing. Since the major source of error stems from a lack of rejection on the part of the models, this should provide a marked increase in performance.

While it is clear that the Tree Model performs better on column pairs with larger numbers of mutations in their evolutionary histories, the exact relationship between the numbers of mutations and discrimination accuracy is not clear. Further research in this area could provide estimated probability bounds on column duo classifications. These bounds could be included in a context-sensitive model that would take into account the classifications of adjoining column duos in its determination of pairing status of the given column duo. Such a method could provide the templates used by a Stochastic Context Free Grammar [29] for complete secondary structure determination.

4.4. 3 Theoretical Tree Model Extension

While it does not seem that the overfitting is an issue for the IO or IOM models, it might become so if these methods are applies to smaller data sets, such as 23S. In this case, a variant of the Q Model might serve to bridge the complexity gap between the Q Model and the IO Model. This variation would involve the use of 16 q_l parameters rather

than a single value for q. Each of these would correspond to a particular source state in the transition matrix $\rho_{l,m}$. This would allow greater flexibility in the transition matrix which could then be calculated as $\rho_{l,m} = q_l \cdot \varphi_m$ for $l = m$, and $q_l \cdot \varphi_m + (1-q_l)$ otherwise [53]. The values for q_l could then be calculated through Expectation Maximization.

Currently, the branch length is incorporated into the modeling in a relatively coarse nonparametric manner. If a provably correct parametric method could be developed, a common mutation transition function could be developed which concentrated all of the available sequence data. There are numerous problems in mixing counts from differing branch lengths, however, and even theoretically this problem is not trivial. If solved, though, this might produce a marked improvement in resolution. In addition, a parametric time-mutation model would allow the reestimation of the branch lengths in the tree. This could be of tremendous use to biologists, who currently use more biased statistics derived directly from multiple alignments, as well as individual expert knowledge, to hand tune their phylogenetic trees. As more data becomes available, and the trees grow larger, this hand manipulation will become more and more cumbersome and techniques for automated generation and refinement of the phylogenetic trees will become increasingly necessary.

In the current work, both the IO and the IOM Models heavily leverage the mean model parameters φ and ρ to determine P(d|model). However, the use of these mean statistics can lower the resolution of the model. It is quite possible that improved resolution could be obtained by executing the expectation maximization training

algorithm on each column duo presented to a model. This would produce a ρ specifically tailored to that column duo, *d*. Such a method once more raises the specter of over-fitting the model to each specific column pair. It may be possible to avoid this, however, by using only a fraction of the reestimated ρ, while maintaining a certain fraction of the initial ρ as a prior distribution. This is similar to the Laplacian probability estimator that initializes the number of observed counts in a count-based system to 1, before any data has been observed. The precise fraction of the prior information to use could be obtained from a Dirichlet prior method as described in [54] and [55] or simple taken *a priori* like the *q* parameter in the Q Model.

Another variant of the Tree Model would eschew the use of randomly generated column duos for generation of the unpaired model, $\text{Model}_{\text{Rand}}$. Instead, a reference model would be generated by the explicit assumption of statistical independence. Thus, the Rand data would be replaced with an evolutionary history of individual columns, combined using the independence criterion for nucleotide duos *xy*, $P(xy)=P(x)\cdot P(y)$. This would help the nonpaired model to reject any kind of statistical dependency, rather than rejecting only those forms not found in the Rand sample.

The current Tree Model relies heavily on the correctness of the phylogenetic tree to quantify genetic relationships. For a given multiple alignment, the phylogenetic tree is generally known only approximately, and in some cases differing regions of an alignment may have differing trees (as in cases of genetic crossover) [56]. For this reason it might be a good idea to replace the single phylogenetic tree with a phylogenetic matrix

representing the genetic similarity between each pair of organisms in the multiple alignment. Phylogenetic trees may represent a single optimal path through a genetic mutual information matrix. However, using the single best estimate for a solution may be far less robust then estimating an expectation value over all possible solutions. Dynamic programming techniques might be employed to efficiently estimate secondary structure over all phylogenetic trees, weighted by each one's likelihood.

The Tree Model performs discrimination by training two models on differing classes of data and then comparing the performance of each model on novel data. It seems likely from the degenerate situation where both models find the same data probable, that both of these models are encoding some of the same evolutionary dynamics. As a result, it might be wiser to look for a Tree Model based technique that attempts to model only the differences between the two data sets. Such a model might help resolve the column duo degeneracy problem.

4.4. 4 Additional Areas of Interest

Finally, Tree Model is not limited to measurements of RNA. Its evolutionary methodology should be easily extensible to the modeling of protein evolution. As a sensitive detector of structure and phylogeny, this technique might assist in protein identification, classification and possibly *in vitro* design.

5 Appendix A: Q Model Results for q=0.0001

The Q Model was run with a *q* parameter value of 0.0001 as well as the previously reported value of 0.01. While the value of 0.01 which is reported in *3.8 Q Model* was found to produce significantly NLL values, this value for q = 0.0001 produced marginally better discrimination accuracy. As such, it might be of some interest and is made available here.

Q Model (dependent)				
NLL (bits/base)	Validation Set		Training Set	
	Rand Data	Pair Data	Rand Data	Pair Data
Rand Model	0.615	0.518	0.615	0.518
PairModel	0.690	0.512	0.690	0.509

Q Model (independent)				
NLL (bits/base)	Validation Set		Training Set	
	Rand Data	Pair Data	Rand Data	Pair Data
Rand Model	0.569	0.757	0.569	0.758
PairModel	0.574	0.764	0.574	0.763

Table 5—1: Q Model NLL Summary for q=0.0001

Q Model Classification Accuracy - Simple Discriminator				
	Actual (train)		Actual (validate)	
Predicted	Rand	Pair	Rand	Pair
Rand	8191	1868	2731	613
Pair	149	1936	49	655
Accuracy	83.39%		83.65%	

Q Model Classification Accuracy - Neural Net Discriminator				
	Actual (train)		Actual (validate)	
Predicted	Rand	Pair	Rand	Pair
Rand	7772	1224	2590	440
Pair	568	2580	190	828
Accuracy	85.24%		84.44%	

Table 5—2: Q Model Classification Summary for q=0.0001

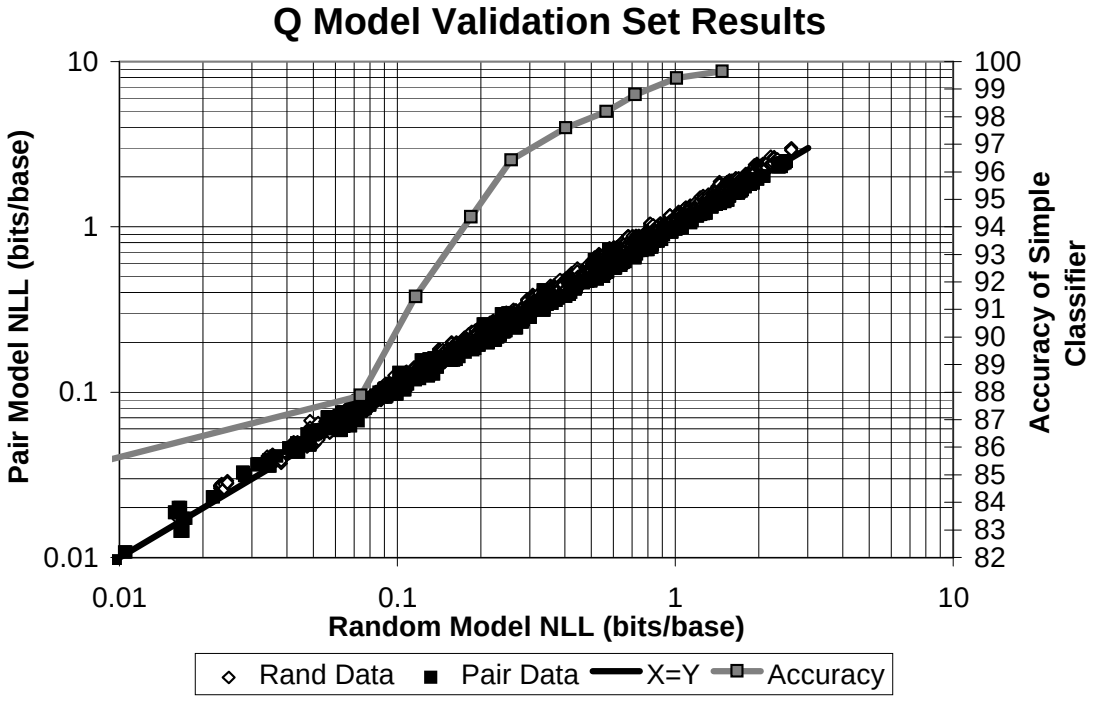

Figure 5-1: Q Model Results Graphical Summary for q=0.0001

6 Appendix B: Posterior Probability Classifier for IO

In the present work, data likelihoods (P(*d*|Model)) rather than posterior probabilities (P(Model|*d*)) are used for the classification of column duos (*d*). This technique is not generally valid according to Bayesian analysis. However, in the case that the model prior probabilities are nearly equal ($P(Model_{Rand}) \approx P(Model_{Pair})$), a direct comparison of data likelihoods should yield similar results to a direct comparison of posterior probabilities. While the prior model probabilities are close to one another in the current work (see *2.2.2 Discrimination*), the precise impact of this substitution was not clear.

To provide a greater insight into the the use of likelihoods instead of posterior probabilities, the posterior probabilities were calculated for the IO validation data as outlined in *2.2.2*. Each column duo (*d*) in the IO validation set was then reclassified according the *simple discriminator* presented in *2.2.2* using posterior model probabilities rather than likelihoods. These posterior probabilities were calculated from the likelihoods using Bayes' Rule as P(Model|d) = P(d|Model)·P(Model)/P(d) where:

|Rand| =	Number of column duos in the validation set for Rand =	695
|Pair| =	Number of column duos in the validation set for Pair =	317

$P(Model_{Rand})$ = |Rand|/(|Rand| + |Pair|) = 695/(695+317) = 68.7%

$P(Model_{Pair})$ = |Pair|/(|Pair|+|Rand|) = 317/(695+317) = 31.3%

$P(d) = P(d|Model_{Rand})\cdot P(Model_{Rand})+P(d|Model_{Pair})\cdot P(Model_{Pair})$.

A summary of these results follows in *Table 6-1*.

IO Model Classification Accuracy: Simple Discriminator, Validation Data				
	Likelihood		Posterior Probability	
Predicted	Rand	Pair	Rand	Pair
Rand	2462	134	2467	138
Pair	318	1134	313	1130
Accuracy	88.83%		88.86%	

Table 6-1: Comparison of Likelihood vs. Posterior Probability Classification

A discriminator formed from the direct comparison of posterior probabilities performs less than 0.05% better than the same discriminator using actual posterior probabilities P(d|Model).

We find that a simple discriminator, based on Bayesian posterior model probabilities increases classification accuracy by less than 0.05% over a similar comparison based on data likelihoods. This lends additional credibility to our approximation of posterior probabilities as proportional to data likelihoods, for column duo classification purposes on the present data set.

As a separate investigation, we can estimate the classification error that the Tree Model would be expected to produce if were it were applied to all possible column duos in the multiple alignment. To answer this question, we assume values for |Pair| and |Rand| consistent with an exhaustive search through all column duos of the multiple alignment for paired column duos. We classify each column duo on our data sets according to the new model priors, and then separate the misclassification rates by data

class (Rand & Pair). Finally, we rescale these error rates to reflect the relative percentages of unpaired and paired column duos as a fraction of all possible column duos in the multiple alignment. This extrapolation relies heavily on the assumption that the training data in Rand and Pair accurately represent thier generating population (Pop_{Rand} and Pop_{Pair}). To comply with this condition, we model only those column duos in the multiple alignment that would meet our 75% valid nucleotide duo criterion. This limits us to 19.9% of the all nonpaired column duos and 67.2% of all paired column duos.

|Pair| = # of paired column duos in 16S alignment = 944·0.672 = 634

|Rand| = # of unpaired column duos in alignment =
(# of columns)^2-|Pair| = (2,688^2-944)·0.199
= 1,437,656

$P(Model_{Pair})$ = |Pair|/(|Pair|+|Rand|) = 634/(634+1,437,656) = 0.044%

$P(Model_{Rand})$ = 1.0-$P(Model_{Pair})$ = 99.956%

The result of this proceedure, along with our estimate of the model's error rates are found below in *Table 6-2*.

IO Model Classification Accuracy: Extrapolation to Entire Multiple Alignment			
Predicted Data Set	Actual Data Set		Model Accuracy
	Rand	Pair	
Rand	1,306,819	88	>99.99%
Pair	130,837	546	0.04%
Accuracy By Data Set	90.90%	86.12%	90.90%

Table 6-2: Posterior Probability Extrapolation

This table contains the results of a posterior probability extrapolation of the Tree Model from Rand and Pair to the set of all column duos in the multiple alignment. The cumulative accuracy is in the lower right cell of the table.

This extrapolation clearly brings to light a problem with secondary structure prediction based on column duo classification. Even given a classification accuracy of 99%, the vastly greater number of unpaired column duos will overwhelm the smaller number of column pair. In the previous example, we correctly identified 546 of the paired duos (86%) and 90% of the nonpaired duos. However, this left approximately 130,000 nonpaired duos classified as paired, obscuring the 546 which actually were paired. If a column pairing detector incorrectly accepts even 0.1% of the random column duos as paired (about 1,300 in this example), there will still be substantially fewer corretly identified paired duos than incorrectly classified nonpaired duos!

The effects of this enormous bias may be reduced by tightening the restrictions on duos that are accepted as paired. Currently, the model with the higher posterior probability from a column duo, is assigned to that duo. We can make this assignment more stringent by requiring a higher posterior probability ($P(Model_{Pair}|d)$) before classifying a column as paired. As we raise this threshold towards 100%, more paired duos will be incorrectly be classified as unpaired, but duos accepted as paired will have a greater certainty of actually being paired. See *Figure 6-1* for a graphical depiction of this transition.

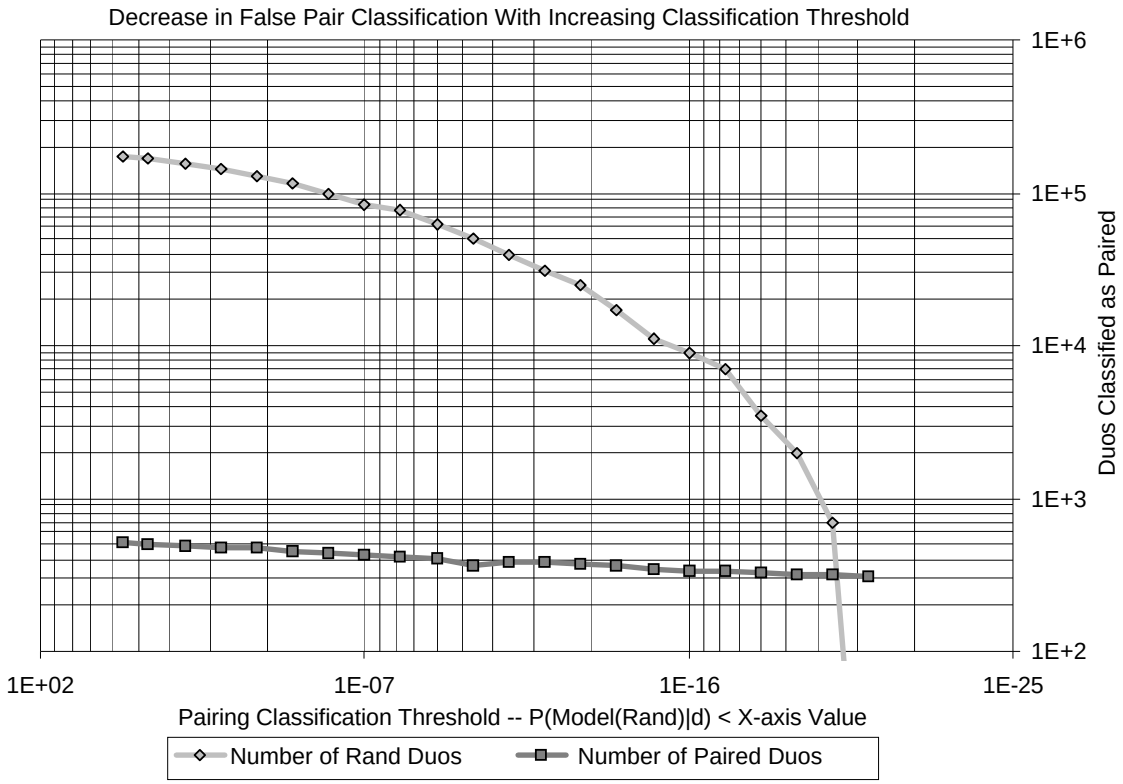

Figure 6-1: Duos Classified as Paired vs. Classification Threshold

The above chart shows how the number of nonpaired column duos classified as paired drops off quickly as the classification threshold increases, while the number of paired duos drops off more slowly. Points are drawn at x axis values of .5, .1, 0.01 and every each factor of 10 thereafter. The final point is at $x=10^{-21}$. At this point all of the 311 column duos classified as paired are actually paired. For the purposes of logarithmic representation, the number of nonpaired column duos at $X=10^{-21}$ is represented as 1 rather than 0.

At the far left hand side of the chart, the classification threshold is 50%. At this point 516 of the 634 paired column duos (81%) are classified as paired, but so are approximately 175,000 of the 1.8 million unpaired duos. When the required probability threshold is raised to $1\text{-}10^{-21}$ (very close to 1), *all* 311 of the column duos that are classified as paired are actually paired, though approximately 51% of the paired duos are incorrectly classified as unpaired.

While no unpaired duos were classified as paired at as classification threshold of $1\text{-}10^{-21}$, it should be noted that the results for thresholds near 100% may be statistically unreliable. This is because we are extrapolating performance on a very large set of unpaired column duos (1,437,656) from a relatively small discrimination validation set[36] of 2780 (paired, nonpaired) example 2-tuples. This discrimination set is, in turn, derived from an even smaller set of 695 unpaired column duos. The result is that each original discrimination set element counts for $1{,}437{,}656/2780 \approx 517$ elements in the extrapolated data. The result of this data leveraging is that the three rightmost points in the *Rand Duos* data set in *Figure 6-1,* which represent approximately 3500, 2000 and 500 misclassified unpaired column duos, are generated by only 7, 4, and 1 discrimination set elements. These in turn could be generated from as few as 2, 1 and 1 unpaired column duos, respectively. While the threshold for the correct classification of these last unpaired column duos seems visually consistent with the previous data in the chart, it is hardly statistically robust.

[36] Each element of the discrimination validation set is a 2-tuple of column duo probabilities, namely $P(\text{Model}_{\text{Rand}}|d)$ and $P(\text{Mode}_{\text{Pair}}|d)$. As we are using four fold cross validation, there are four paired models against which to classify each Rand column duo. As we have 695 unpaired column duos, each of which is used exactly once for validation, we have $4 \cdot 695 = 2780$ unpaired elements of the discrimination validation set.

7 Appendix C: Data Separation Charts

This section contains graphs that were deemed useful, but were generated too late for formal inclusion into the main body of the thesis. These graphs are generated from the same data that was used to generate the NLL graphical results summary figures for each model in *3 Experiments.* These summaries are found in:

Figure 3-6: Frequency Model Results Graphical Summary (detail) on page 94
Figure 3-8: Q Model Results Graphical Summary for q=0.01 on page 98,
Figure 3-9: IO Model Results Graphical Summary on page 102 and
Figure 3-11: IOM Model Results Graphical Summary on page 107.

As the data in these summaries was found to cluster strongly around the X=Y line on each chart, it was frequently difficult to pick out salient data characteristics. To further distinguish the clusters, the following charts were produced. The data points on each chart in this section correspond to the data points found on the graphical summaries mentioned above. For each Model, except Frequency, this section contains two charts, an overview and a detail. The overviews are all scaled similarly to provide a common reference for comparison. The details are scaled separately to represent the area of greatest interest. The X-axis on of each chart represents the Rand model NLL score for each column duo in a validation set, just like the graphical summaries. All X-axes are identical in range and scale to one another, as well as to the graphical summaries. However, the Y-axes of the following charts represents the directed distance from a given graphical summary data point, to its X=Y line. Specifically, this Y-axis value is

$$(\mathrm{NLL}(\mathrm{P}(d|\mathrm{Model}_{\mathrm{Pair}}))-\mathrm{NLL}(\mathrm{P}(d|\mathrm{Model}_{\mathrm{Rand}}))) / \sqrt{2}$$

and is scaled linearly in units of bits per base. Thus, all of the data points above the X-axis are classified as unpaired, and all of the data below the X-axis are classified as paired by the simple classifier. For brevity, the chart legends refer to correctly classified data are as *good* and incorrectly classified data as *bad.* Thus, the all data in the half-plane above the X-axis would be Good Rand or Bad Pair, having been classified as unpaired. Similarly, all data below the X-axis is Bad Rand or Good Pair, having been classified as paired. The Frequency Model's data presents a special case. As it was tightly and uniformly clustered, it is presented in a single graph with a linear X-axis. No detail of this graph was deemed necessary.

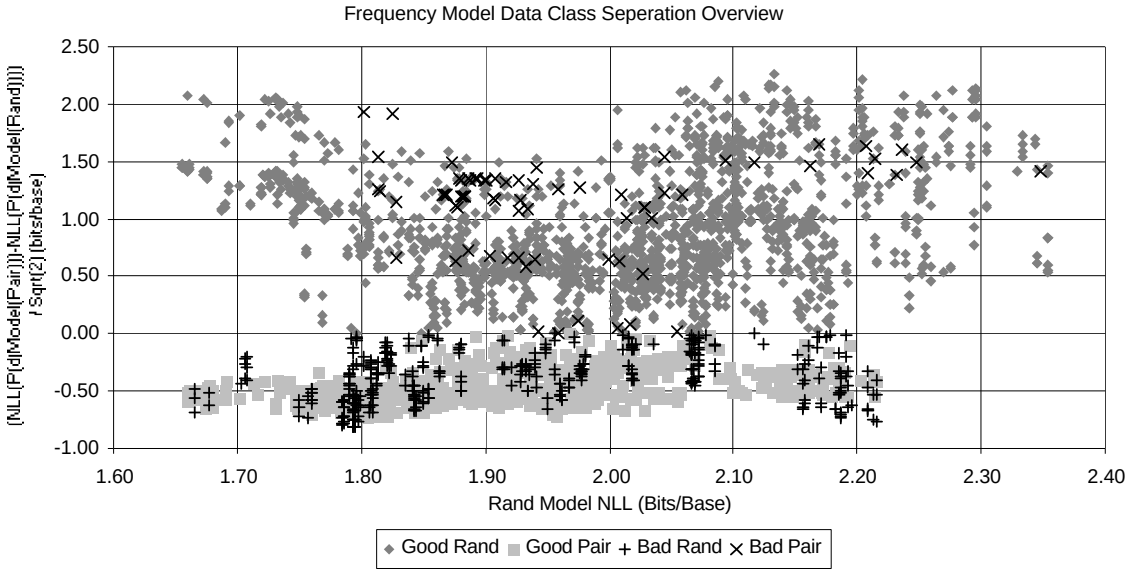

Figure 7-1: Frequency Model Likelihood Separation Chart

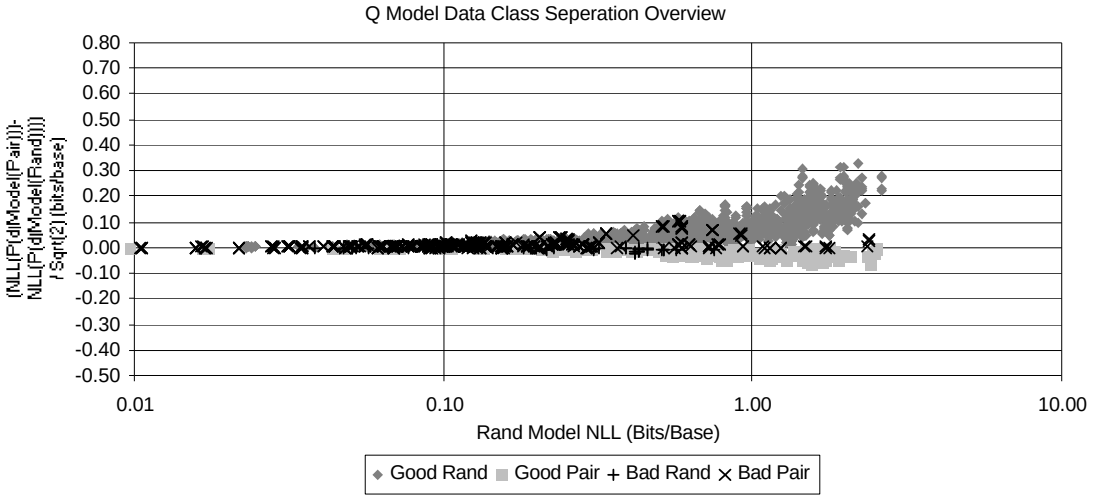

Figure 7-2: Q Model Likelihood Separation Chart -- overview

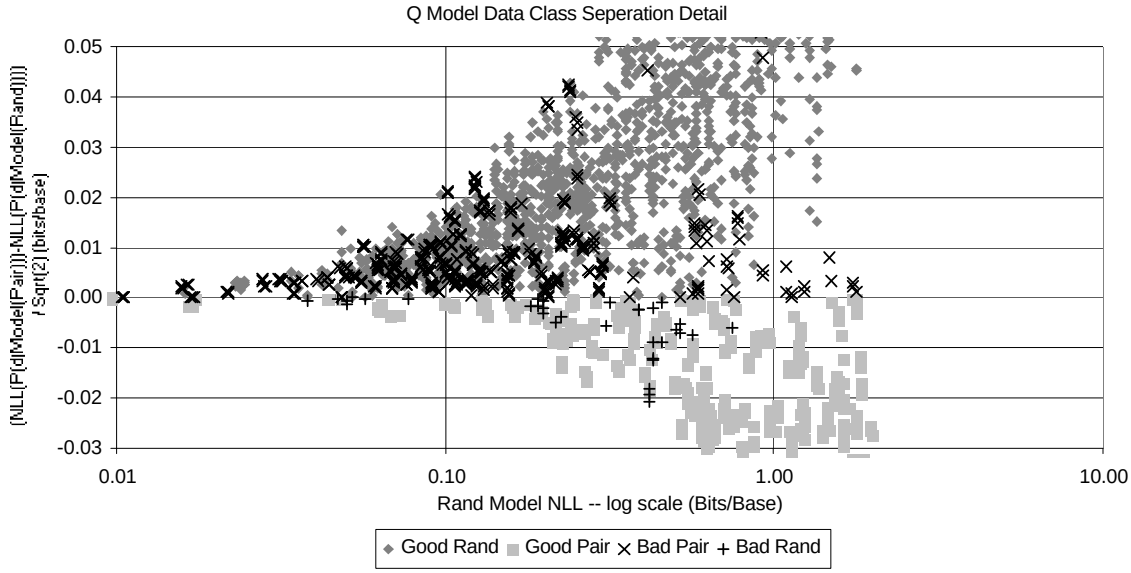

Figure 7-3: Q Model Likelihood Separation Chart -- detail

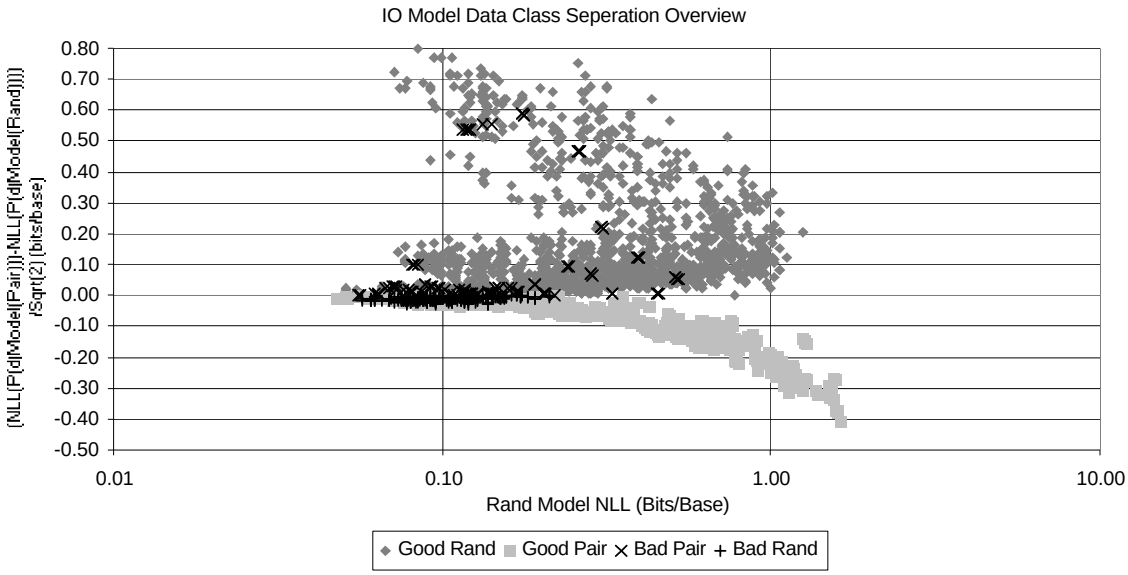

Figure 7-4: IO Model Likelihood Separation Chart -- overview

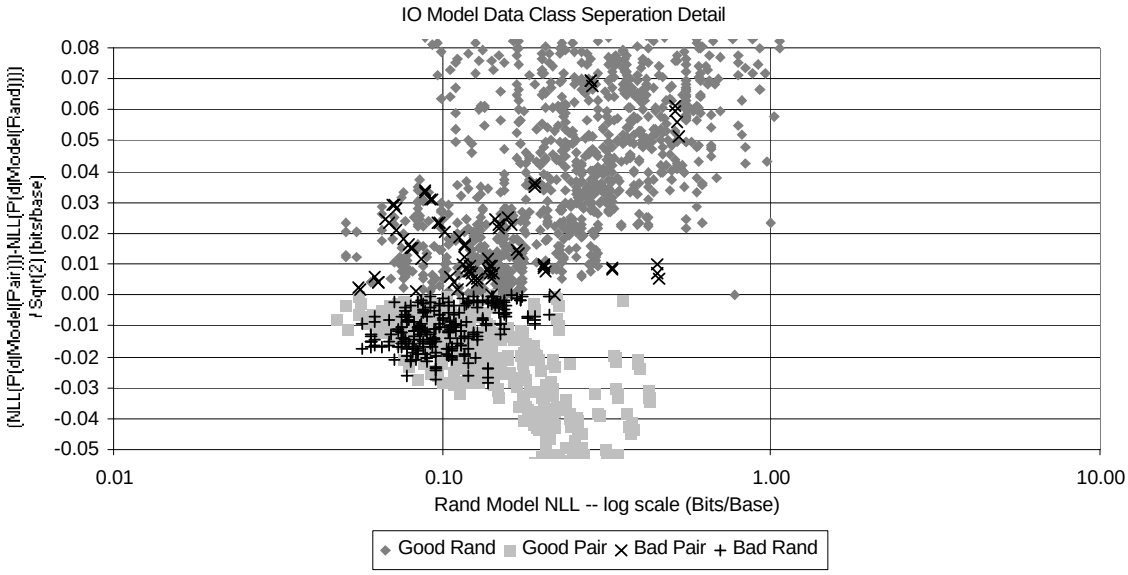

Figure 7-5: IO Model Likelihood Separation Chart -- detail

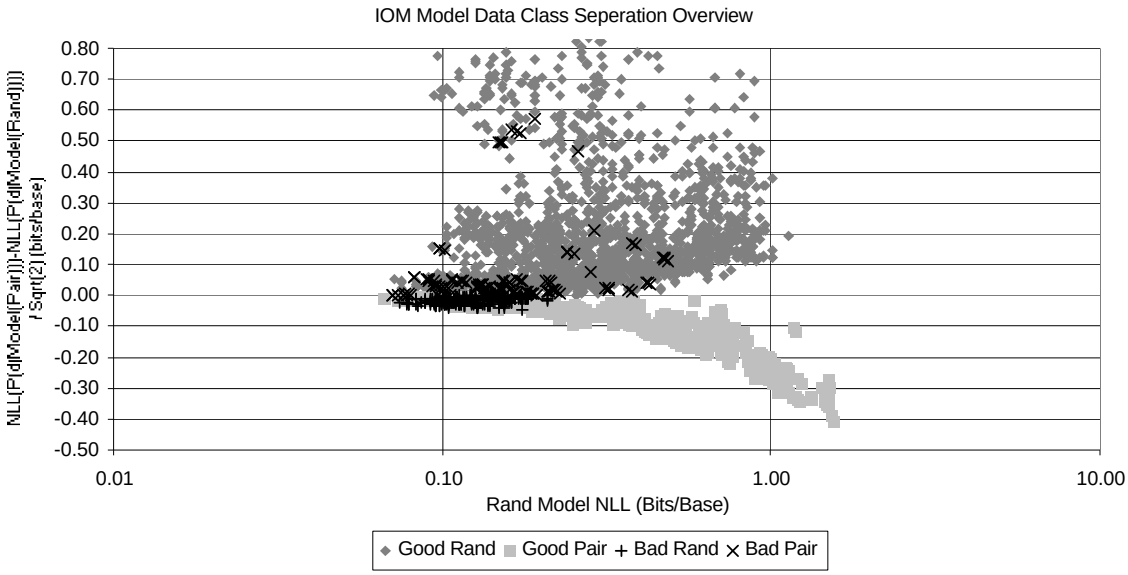

Figure 7-6: IOM Model Likelihood Separation Chart -- overview

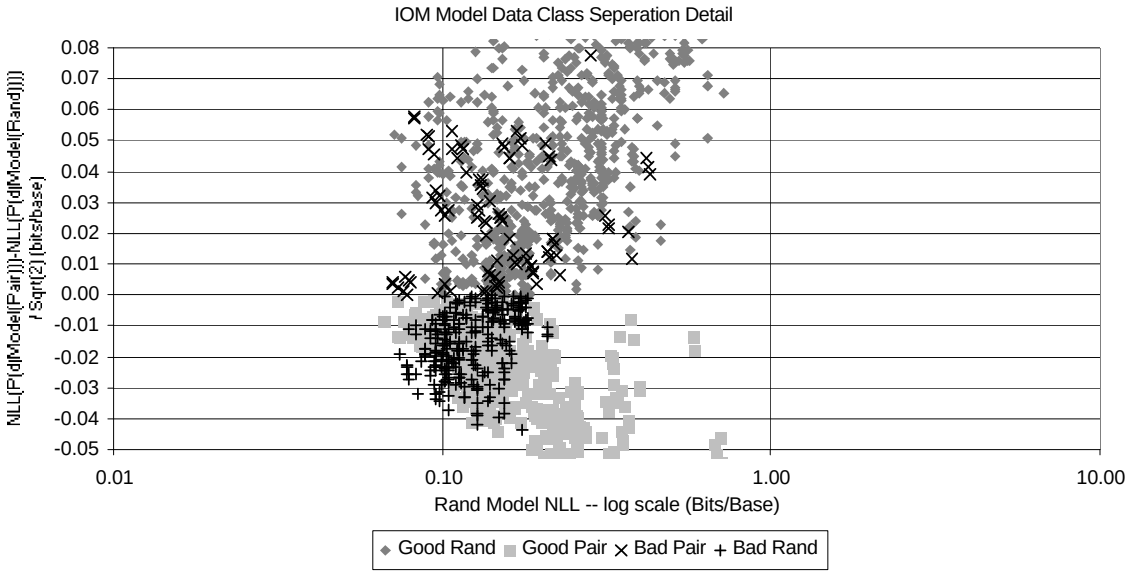

Figure 7-7: IOM Model Likelihood Separation Chart -- detail

8 Annotated References

1 U.S. Department of Health and Human Services, Public Health Report 1992. *Public Health Reports*, November & December 1992, 107:740-741. Federal funding increases for "Grand Challenges" in computational biology.

2 J. A. Board, Junior. Grand challenges in biomedical computing. *Critical Reviews in Biomedical Engineering*, 1992, 20(1-2):1-24. Review of state of the art in biomedical computing in the context of "Grand Challenges" of computational biology.

3 Robert B. Denman. Using RNAFOLD to Predict the Activity of Small Catalytic RNAs. *Biocomputing*, 1993, 15(6):1090-1095. References to Higher order structure (tertiary and secondary) being critical in RNAs functioning.

4 L. Chan, M. Zuker and A. B. Jacobson. A computer method for finding common base paired helices in aligned sequences: an application to the analysis of random sequences. *Nucleic Acids Research*, 1990, 19(2):353-358. References to use of statistical dependence to infer chemical pairing. Function of RNA is dependent on its structure.

5 Giorgio Benedetti and Stefano Morosetti. Three-Dimensional Folding of Tetrahymena Thermophila rRNA IVS Sequence: A Proposal. *Journal of Bimolecular Structure and Dynamics*, 1991, 8(5):1045-1055. Denaturing an RNA molecule effects its behavior. Knowledge of 3D structure is imperative for modeling of behavior. Use of X-ray crystallography and NMR to calculate 3D structure, once crystallized.

6 Y. T. van den Hoogen, A. A. van Beuzekom, E. de Vroom, G. A. van der Marel, J. H. van Boom, and C. Altona. Bulge-out structures in the single-stranded trimer AUA and in the duplex (CUGGCGCGG)·(CCGC-CCAG): A model building NMR study. *Nucleic Acids Research*, 1988, 16:5013-5030. MRI measurement of RNA fragment.

7 E. Westhof, P. Dumas and D. Moras. Crystallographic refinement of Yeast aspartic acid transfer RNA. *Journal of Molecular Biology*, 1985, 184:119-145. X-ray based measurement of entire tRNA molecule.

8 C. Pelling and T. D. Allen. Scanning electron microscopy of polytene chromosomes (I). *Chromosome Res*, November 1993, 1(4):221-37. Electron microscopy techniques to resolve individual nucleotides.

9 V. Mandiyan; S. Tumminia; J. S. Wall and M. Boublik. Visualization of ion-dependent conformational changes in Escherichia Coli 23 S rRNA by scanning transmission electron microscopy. *Archives of Biochemistry and Biophysics*, February 1, 1990, 276(2):299-304. Technique does not require denaturing of sample.

10 R. R. Gutell, A. Power, G. Z. Hertz, E. J. Putz and G. D. Stormo. Identifying constraints on the higher order structure of RNA: continued development and application of comparative sequence analysis methods. *Nucleic Acids Research*, 1992, 20(21):5785-5795. Correspondence of structure with function in RNA. Seminal work on Frequency Model. Use of nucleotide column duo mutual information contents without phylogenetic tree information.

11 B. Lesyng; J. A. McCammon, Molecular modeling methods: Basic techniques and challenging problems. *Pharmacology and Therapeutics*, November 1993, 60(2):149-67. Overview of techniques from quantum modeling to molecular "degrees of freedom" modeling.

12 J. Brickmann, The Darmstadt workshop on molecular modeling: past and future. *Journal of Molecular Graphics*, June 1994, 12(2):82-3. Proceedings from modern conference in molecular modeling including most recent techniques and challenges.

13 Michael Zuker and David Sankoff. RNA secondary structures and their prediction. *Bulletin of Mathematical Biology*, 1984, 46:591-621. Energy minimization techniques for RNA folding.

14 Ann B. Jocobson and Michael Zuker. Structural Analysis by Energy Dot Plot of a Large mRNA. *Journal of Molecular Biology*, 1993, 233:261-269. Details of an energy minimization (EM) folding. Subjective factors in EM. Other shortcomings of EM. Use of suboptimal foldings (energy degeneracies). Limits on the accuracy of the energy potential function. Preference for probabilistic modeling, but uses thermodynamics to obtain probabilities and declares such a technique currently infeasible.

15 Michael Zuker, Hohn A. Jaeger and Douglas H. Turner. A comparison of optimal and suboptimal RNA secondary structures predicted by free energy minimization with structures determined by phylogenetic comparison. *Nucleic Acids Research*, 1991, 19(10):2707-2714. Specific accuracies for energy minimization techniques. References to definition of free energy.. Nuclear systems are not at equilibrium so there is no reason that the minimum energy should be the correct folding. Explores suboptimal foldings as a method of circumventing these problems. Uncertainties in energy parameters of folding model (10%). Method finds 80% of the secondary structure helices.

16 Kyungsook Han and Hong-Jin Kim. Prediction of common folding structure of homologous RNAs. *Nucleic Acids Research*, 1993, 21(5):1251-1257. Computes secondary structure through recursive estimation over a small (10-35) set of sequences. Uses heuristic, rather then probabilistic modeling. History of secondary and 3D structure for tRNA. Notes on deficiencies of energy minimization techniques and manual phylogentic techniques. This work is the closest to my own (see footnote *2* on page *5* of *1.1 General Motivation*).

17 M. S. Waterman. Sequence Alignments. *Mathematical Methods for DNA Sequences*. CRC Press, 1989. Example of use of Dynamic Programming the extract optimal statistics from distributions.

18 L. Grate, M. Herbster, R. Hughey, I. S. Mian, H. Noller and D. Haussler. RNA Modeling Using Gibbs Sampling and Stochastic Context Free Grammars. *Proceedings, 2nd International Conference on Intelligent Systems for Molecular Biology*, February 1994, 235:1501-1531.

19 J. D. Watson and F. H. C. Crick. Genetical Implications of the Structure of Deoxyribonucleic Acid. *Nature*, May 30, 1953 171b:964-967. Follow on work to [31]. Greater detail regarding nucleotide-nucleotide structure and nucleotide-sugar-phosphate structure.

20 Yves Van der Peer, Jean-Marc Neefs, Peter De Dijk and Rupert De Wachter. Reconstructing Evolution from Eukaryotic Small-Ribosomal-Subunit RNA Sequences: Calibration of the Molecular Clock. *Journal of Molecular Evolution*, 1993, 37:221-227. Universality of SSU RNA among cellular organisms. Areas of high variability due to evolution. Is an example of potential biases in statistics gathered directly from multiple alignments, due to phylogentic similarities.

21 Gary J. Olsen, Hideo Matsuda, Ray Hagstrom and Ross Overbeek. fastDNAml: a tool for construction of phylogenetic trees of DNA sequences using maximum likelihood. *Computer Applications in the Biosciences*, 1994, 10(1):41-48. Describes the phylogenetic tree construction technique used to generate the phylogenetic tree that was employed in this work.

22 Ahrun Malhotra, Robert K. Z. Tan and Stephan C Harvey. Modeling Large RNAs as Ribonucleoprotien Particles using Molecular Mechanics Techniques. *Biophysical Journal*, 1994, 66:1777-1795. Deriving 3D structure from secondary structure information and other constraints. 3D structure for RNA is lagging behind that of other macromolecular systems. General availability of measured 3D structures for RNA (x-raw crystallography & MRI), specific examples. Low resolution data available. Citations for 3 other 3D modeling techniques.

23 D. G. Higgins and P. M. Sharpp. Fast and sensitive multiple sequence alignments on a microcomputer. *Computer Applications in the Biosciences*, 1989, 5:151-153. Automated multiple alignment generation.

24 Jeffery L. Thorne, Hirohisna Kishino and Joseph Felsenstein. An Evolutionary Model for Maximum Likelihood Alignment of DNA Sequences. *Journal of Molecular Evolution*, 1991, 33:114-124. maximum likelihood technique for creating DNA multiple alignments. Uses a Markovian process, but no phylogenetic information. Concentrates on aligning pairs of sequences.

25 C. E. Lawrence, S. F. Altschul, M. S. Boguski, J. S. Liu, A. F.Neuwald, J. C. Wootton. Detecting subtle sequence signals: a Gibbs sampling strategy for multiple alignment. *Science*, 8-October 1993, 262(5131):208-14. Automated multiple alignment for proteins.

26 S. R. Holbrook, C. Cheong, I. Tinoco Jr,., and S. H. Kim. Crystal structure of an RNA double helix incorporating a track of non-Watson-Crick base pairs. *Nature*, 1991, 353:579-581.

27 Elisabeth R. M. Tillier and Richard A. Collins. Neighbor Joining and Maximum Likelihood with RNA Sequences: Addressing the Interdependence of Sites. *Molecular Biology and Evolution*, 1995, 12(1):7-15. Double Stranded method for phylogenetic tree generation assumes a dependent relationship between nucleotides in nucleotide evolution. Comparison with maximum likelihood and neighbor-joining methods.

28 Joseph Felsenstein. A likelihood approach to character weighting and what it tells us about parsimony and compatibility. *Biological Journal of the Linnean Society*, 1981, 16:183-186. A comparison of methods for building phylogenetic trees, including the likelihood methods.

29 Yasubumi Sakakibara, Michael Brown, Rebecca C. Underwood, I. Saira Mian and David Haussler. Stochastic context-free grammars for tRNA modeling. *Nucleic Acids Research*, 25-November 1994, 22(23):5112-20.

30 M. Zuker. Prediction of RNA secondary structure by energy minimization. *Methods in Molecular Biology*, 1994, 25:267-94.

31 J. D. Watson and F. H. C. Crick. Molecular Structure of Nucleic Acids: A structure for Deoxyribose Nucleic Acid. *Nature*, April 25, 1953, 171a:737-738. Seminal work introducing helical structure for DNA. Also indicates a belief that helical RNA is unlikely.

32 Jan Pieter Abrahams, Mirjam van der Berg, Eke van Batenburg and Cornelis Pleij. Prediction of RNA secondary structure, including pseudoknotting, by computer simulation. *Nucleic Acids Research*, 1990, 18(10):3035-3044. Gives relative accuracies for base pair discrimination for several energy minimization strategies as 70% - 79%.

33 L. R. Rabner. A tutorial on hidden Markov models and selected applications in speech recognition. *Proceedings of the IEEE*, 1989, 77(2):257-286. General introduction to hidden Markov models.

34 Joseph Felsenstein. Evolutionary Trees from DNA Sequences: A Maximum Likelihood Approach. *Journal of Molecular Evolution*, 1981, 17:368-376. Presentation of Q mutation model for use in Maximum Likelihood inference for the derivation of phylogenetic trees.

35 David Haussler. Personal communication. September-1994. The Q Model state transition matrix [34] was also developed independently by David Haussler for specific application to the problem of RNA secondary structure determination using phylogenetic trees. David Haussler is presently reachable at the Computer and Information Sciences Department of the University of California at Santa Cruz as haussler@cse.ucsc.edu.

36 K. Lari and S. J. Young. The estimation of stochastic context-free grammars using the inside-outside algorithm. *Computer Speech and Language*, 1990, 4:35-56. Presentation of the inside-outside algorithm.

37 J. K. Baker. Trainable grammars for speech recognition. *Speech Communication Papers for the 97th Meeting of the Acoustical Society of America*, 1979, p 547-550. Presentation of the forward-backward algorithm on which the inside-outside algorithm is based.

38 A. P. Dempster, N. M. Laird and D. B. Rubin. Maximum Likelihood from Incomplete data via the EM algorithm. *Journal of the Royal Statistical Society*, 1977, 39(1):1-38.

39 Ribosomal Data Project, University of Illinois in Urbana-Champaign. Revision 3.0 of the database. Retrieved from rdp.life.uiuc.edu in /pub/RDP. This source also requests citation of Niels Larsen, Gary J. Olsen, Bonnie L. Maidak, Michael J. McCaughey, Ross Overbeek, Thomas J. Macke, Terry L. Marsh and Carl R. Woese, "The Ribosomal Database Project", *Nucleic Acids Research*,, 1993, Volume 21 Supplement, pp. 3021-3023.

40 Multiple Alignment data is drawn from [39] /rdp/SSU_rRNA/SSU_Prok.gb.

41 Phylogenetic Tree data is drawn from [39] /rdp/SSU_rRNA/tree/SSU_Prok.newick.

42 Rodrigo Garces. Personal comminucation and data. 1994. Rodrigo Garces is currently available at the University of California at Santa Cruz Department of Computer and Information Sciences as garces@cse.ucsc.edu. Manual synchronization of multiple alignment data [40] with phylogenetic tree data [41].

43 T. J. Macke. AE2. Ribosomal Data Project, University of Illinois in Urbana-Champaign. 15, February 1993. The list of base paired column duos was drawn from a data library accompanying the AE2 sequence editor. This data is available through anonymous ftp from rdp.life.uiuc.edu in /pub/rdp/programs/Editor_AE2/ae2.tar.Z. The particular data file used is found at ae2/lib/paircon.16 in the archive file. The author of this program, T. J. Macke, is reachable at macke@scripps.edu. Includes helical pairs, endcaps and other tertiary interactions but no 3-nucleotide interactions. For more information on the RDP see [39].

44 Michael Brown, Richard Hughey, Anders Krough, I. Saira Mian, Kimmen Sjölander and David Haussler. Deriving Dirichlet Priors for RNA and Proteins. University of California at Santa Cruz, Department of Computer and Information Sciences. 1994. Unpublished.

45 David E. Rumelhart, James L. McClelland and the PDP Research Group. *Parallel Distributed Processing: Volume 1, Foundations.* The MIT Press. Cambridge, Mass. 1986.

46 John Hertz, Anders Krough, and Richard G. Palmer. *Introduction to the Theory of Neural Computation.* Addison-Wesley Publishing Company. Redwood City, California. 1991.

47 Yoshiro Miyata. *A User's Guide to PlaNet Version 5.6: A tool for Constructing, Running and Looking into a PDP Network.* Computer Science Department, University of Colorado, Boulder. December 20, 1990. Author should be reachable as miyata@boulder.colorado.edu.

48 J. A. Jaeger, D. H. Turner and M. Zuker. Predicting optimal and suboptimal secondary structure for RNA. *Methods in Enzymology*, 1990, 183:281-306. Energy minimization techniques for RNA folding.

49 Richard Hughey, Anders Krough and John Panzer. *Prob* class for C++. 1994. Information regarding this class is available at URL: http://www.cse.ucsc.edu/research/compbio/doc/ultimate.html#SEC66 or directly from [56].

50 S. C. Chan, A. K. C. Wong and D. K. Y. Chiu. A survey of multiple sequence comparison methods. *Bulletin of Mathematical Biology*, 1992, 54:563-698. Comparison of totally automated methods for folding multiple alignments.

51 J. D. Thompson; D. G. Higgins and T. J. Gibson. Improved sensitivity of profile searches through the use of sequence weights and gap excision. *Computer Applications in the Biosciences*, February 1994, 10(1):19-29.

52 R. Luthy; I. Xenarios and P. Bucher. Improving the sensitivity of the sequence profile method. *Protein Science*, January 1994, 3(1):139-46.

53 Kevin Karplus. Personal communication. March 1995. Kevin Karplus is presently reachable at the University of California at Santa Cruz in the Computer Engineering Department as karplus@cse.ucsc.edu.

54 Michael Brown, Richard Hughey, Anders Krough, I. Saira Mian, Kimmen Sjölander and David Haussler. Using Dirichlet mixture priors to derive hidden Markov models for protein families. *Proceedings, First International Conference on Intelligent Systems for Molecular Biology*, 1993, 47-55. Application of Dirichlet priors in cases of little information.

55 Anders Krogh, Michael Brown, I. S. Mian, Kimmen Sjölander and David Haussler. Hidden Markov models in computational biology. Applications to protein modeling. *Journal of Molecular Biology*, 4-February 1994, 235(5):1501-31. Use of regularizers (pseudo counts) to prevent over-fitting in hidden Markov models.

56 Richard Hughey. Personal communication. March 1995. Richard Hughey is reachable at the University of California at Santa Cruz, Computer Engineering Department as rph@cse.ucsc.edu.