Modeling Non-homologous Protein Sequences
Using a Variation on Traditional $k$th Order Markov Models

A Thesis Presented

by
Daphne A. Kaxiras

to
The Department of Applied Mathematics

in partial fulfillment of
the honors requirements
for the degree of Bachelor of Arts

March 26, 2021
Harvard University
Cambridge, Massachusetts
Abstract

Protein homology search, or the search for evolutionarily related proteins, has a number of useful applications in biology. Relating novel sequences to existing large databases of annotated proteins allows scientists to make predictions about the structure and function of these novel sequences, and also helps further understanding of protein evolution and phylogenies. Current sequence similarity search programs that detect homology, such as BLAST and HMMER, are powerful and continue to improve in their ability to detect more and more distantly related proteins. Despite these improvements, such search programs still face challenges when it comes to proteins featuring periodic patterns of compositional bias. Common protein patterns that appear in many evolutionarily unrelated proteins can lead to inaccurate estimations of the significance of similarity between sequences, and thus to inaccurate predictions of homology. We hypothesize that the log-odds scores at the basis of sequence similarity search can be improved by a more accurate model of non-homology, also known as the null model. We propose a variation on the traditional Markov model which aims to detect the existence of biased patterns in protein sequences. A conventional \( k^{\text{th}} \) order Markov model assumes that the identity of each amino acid in a sequence depends on the \( k \) amino acids that immediately precede it. Our model allows these dependencies to have gaps in them; i.e. for a 3\(^{\text{rd}} \) order Markov model the dependencies could be on the 1\(^{\text{st}} \), 4\(^{\text{th}} \), and 7\(^{\text{th}} \) residues preceding each amino acid, instead of the traditional 1\(^{\text{st}} \), 2\(^{\text{nd}} \), and 3\(^{\text{rd}} \). Current null models are based on 0\(^{\text{th}} \) order Markov models, causing them to inaccurately estimate the probabilities of some proteins with patterns of compositional bias. We test models with different dependencies in search of one that best models real protein sequences. We find that Markov models of orders greater than 3 show significant improvement as compared with the 0\(^{\text{th}} \) order null model on protein regions with patterns of compositional bias. We identify certain models that best capture specific types of compositional bias, including the heptad repeats of coiled-coils. These results support the use of a higher order Markov model in the calculation of log-odds scores for sequence similarity search.
Acknowledgements

This thesis would not have been possible without the support and guidance of many people over the past 2.5 years. First and foremost I would like to thank my thesis advisor Professor Sean Eddy, for giving me this project sophomore year, as well as the support and encouragement needed to turn it into a senior thesis. Thank you for giving me a project I could make my own, for helping me learn the tools to carry it out, and for your invaluable feedback every step of the way. Your enthusiasm for your work, your attention to detail, and your kindness made my research experience incredibly fun and rewarding. I would also like to thank you for one of the best classes I have taken at Harvard, MCB112, and for being an exemplary role model, both as a scientist, and as an advisor.

I would like to express a huge thank you to my thesis supervisor Nicholas Carter, without whose countless hours of help I would not have made it very far. Thank you for working with me until I had gotten the math right, for always helping me debug my programs, for walking me through code over Zoom, and for your patient and detailed explanations. Thank you for your feedback on my writing, and for always making the time to meet with me when I needed it. Your dedication to finding and understanding the right answer to a question has made me a better critical thinker and an overall better scientist.

Many further thanks to all of the members of the Eddy lab, for being incredibly inclusive, kind, and fun. Your enjoyment of computational biology and your thoughtful conversations have had a large impact on how I envision my future in science. Special thank you to Tom Jones, for sending me detailed guides on sed/awk which I still use to this day, and to Elena Rivas, for introducing me to DAGs and for your early constructive feedback on my math.

I would also like to thank my academic advisor, Margo Levine, who has helped me through many difficult points in my undergraduate career, and without whose advice I would not have had as fruitful a time at Harvard as I did. Thank you Margo, for being the best teacher, role model, and advisor I could have asked for.

A heartfelt thank you to my roommates, Katie Mabbott, Skylah Reis, and Otter the dog, for putting up with me throughout this long process, and for helping me make the best of Zoom senior year. I could not have asked for a better group to spend my senior spring with.

Finally, and most importantly, I would like to thank my parents, Eleni and Efthimios Kaxiras, for their unconditional support, love, and encouragement. You have inspired me with your passion for and dedication to science my whole life, and it is because of you that I have been able to make my undergraduate career, and this thesis, a reality. Thank you for everything, and a special thank you for your thoughtful feedback on early versions of this thesis.
# Contents

1 Introduction 3  
1.1 Protein Homology Search 3  
1.2 The Log-Odds Score 4  
1.3 Challenges in Homology Search 6  
1.4 Examples of Pattern-Based Compositional Bias 7  
1.5 Current Standards for Detecting Bias 9  
1.6 Experimental Question and Hypothesis 10  

2 Approach 12  
2.1 Overview of Discontinuous Markov Model Approach 12  
2.2 Calculating Exact Sequence Probabilities 13  
2.3 Approximating Sequence Probabilities 14  
2.4 Basis for the Current Null Model 16  
2.5 The Proposed New Null Model 17  
2.6 Calculating Conditional Probabilities 19  

3 Materials and Methods 22  
3.1 Definition of Model Accuracy and the $Q$ Score 22  
3.2 A Note on Notation 24  
3.3 Choice of Indices to Test 25  
3.4 Choice of Training Data Set 25  
3.5 Choice of Test Data Sets 27  
3.6 Assessing Model Performance 29  

4 Results 30  
4.1 Amino Acid Patterns Hold Information 30  
4.2 Jackknife Resampling Standard Deviation 32  
4.3 Model Parameter $k$ has Largest Effect on $Q$ Score 33  
4.4 Models with $k$ Greater Than 4 Can Detect Biased Composition 34  

5 Conclusion 38  
5.1 Implications of Results 38  
5.2 Further Work 38  

6 References 40
Supplemental Materials

For the purposes of reproducibility, we have provided the manually curated test sets needed to reproduce the results of this thesis at the following link: http://eddylab.org/publications/Kaxiras21/Kaxiras21-supplement.tar.gz.
1 Introduction

1.1 Protein Homology Search

Sequence similarity searching is a commonly used and powerful strategy for transferring information from well studied proteins to newly determined ones. Protein sequence similarity search tools such as BLAST (Altschul et al., 1990) and HMMER (http://hmmer.org) search for statistically significant excess similarity between protein domains, or more similarity than could be explained by chance alone. Such an excess of similarity between sequences is widely used as a way to infer evolutionary relation to a common ancestor, known as “homology”. Making such connections between proteins gives scientists a better understanding of sequence evolution, and can be used as a way to infer structural and functional information about newly determined proteins, or at least, to construct useful hypotheses (Pearson, 2013). With recent advancements in computational methods, homology search through sequence similarity has become increasingly more powerful, allowing for the discovery of more and more distant homologous connections. Current research has focused, among other goals, on increasing the
1.2 The Log-Odds Score

Most widely used programs for finding evolutionarily related proteins compare the amino acid sequences of whole proteins or protein domains to produce statistical estimates for the significance of similarity between those sequences, using a variety of methods. BLAST, HMMER, and other search methods (Remmert et al., 2012; Pearson, 1990; Pearson and Lipman, 1988; Altschul et al., 1997), either explicitly or implicitly, conduct a hypothesis test to determine sequence similarity significance. They compare the probability that two sequences share a certain amount of similarity under the hypothesis that they are homologs, to the probability that they share this amount of similarity due only to chance. As an example of high sequence similarity, Figure 1.1 shows hemoglobin proteins from different organisms, highlighting conserved amino acid positions.

**Figure 1.1:** Sequence alignment of the first 60 to 62 amino acids in homologous hemoglobin subunit beta (HBB) proteins in bovine, chimpanzee, human, cat, gorilla, and rabbit. Conserved positions are highlighted in blue.

One common way to conduct a hypothesis test in bioinformatics is by calculating a log-odds likelihood score (Altschul et al., 2010; Eddy, 2009), which compares the probability of seeing such similarity under the two hypotheses in the form of a fraction, whose numerator
is the probability given the homologous hypothesis, and denominator is the probability given the non-homologous (null) hypothesis. It is important to note that the term “log-odds” as used in bioinformatics differs from traditional statistical log-odds, which is normally expressed as $\log \frac{p}{1-p}$ for some probability $p$. For the purposes of homology search, rather than comparing the probability of some event and its complement, we would like to compare the probability of an event given some model of interest relative to the event’s probability under a null model.

The hypothesis test for sequence similarity search aims to answer the question “is the target sequence $x$ homologous to $y$, the query sequence (or alignment of sequences)?” using a comparison of two hypotheses. More specifically, the two hypotheses are:

1. hypothesis $H_y$: the target sequence $x$ is homologous to $y$, the query sequence (or alignment of sequences)
2. hypothesis $R$ (the null hypothesis): the target sequence $x$ is a non-homologous “random” sequence

The log-odds likelihood score is expressed as follows:

$$S(x|y) = \log_2 \frac{P(x|H_y)}{P(x|R)}$$

Significant research effort in computational biology has been invested toward improving the accuracy of the homologous hypothesis model (Eddy, 2008), while relatively little work has been directed toward creating a more accurate probabilistic model for the null hypothesis (the “null model”). This thesis explores one approach for increasing the accuracy of the null model by using Markov models to identify common repetitive patterns in amino acid sequences.
1.3 Challenges in Homology Search

In trying to increase the sensitivity of homology search tools, thereby reducing the number of missed homologs ("type II" errors), there exists the risk of increasing the number of false-positive results ("type I" errors). False-positives arise when two sequences share a certain amount of similarity but are not evolutionarily related, and sequence similarity search tools fail to recognize that the patterns in these sequences are likely to have evolved independently. As protein database sizes increase with growing sequencing capabilities and computing power, the consequences of false-positives become more pronounced. With small searches, results can be inspected manually, and homology assumptions can be checked through other methods. However, searches with millions of results, which are becoming more common due to larger databases, make it impractical, and sometimes even impossible, to check for false-positives manually.

Most protein sequences share as much similarity with globally unrelated sequences as they do with synthetically generated random sequences (Lavelle and Pearson, 2009). Because of this, when two sequences share more similarity than would be expected under a model of random, or unrelated, proteins, they are inferred to be homologous (Pearson, 2013). There do exist, however, certain protein families that share more similarity than would be expected by chance despite being evolutionarily unrelated. These sequences feature easy-to-evolve patterns of amino acids that, for some structural or functional reason, came to exist in multiple different protein families, independently. Proteins that feature regions of common protein motifs are said to have compositional bias, because their amino acid composition differs from what would be expected if such patterns were not prevalent in many different proteins. Subsequences of biased
composition are also called low-complexity regions. Convergent evolution, or the evolution of similar structures through different pathways, is one way in which non-homologous similar protein sequences can come to exist.

Excessively similar, non-homologous proteins interfere with the way sequence similarity search programs, such as BLAST and HMMER, determine significance. In the process of calculating a log-odds score to test the two hypotheses—one being that the sequences are homologous, and the other (the null) being that they are unrelated—methods with inaccurate null models can overestimate the significance of observed similarity. Null models that cannot recognize certain biased composition regions as common throughout all organisms will assign lower probability to the event that two sequences share such a region by chance. This lower null model probability leads to an inaccurately high score for the significance of the observed similarity. It is therefore necessary that sequence similarity search programs use a null model that can identify common patterns to avoid overestimating the associated log-odds score.

1.4 Examples of Pattern-Based Compositional Bias

One particularly salient example of convergent evolution in protein domains is the coiled-coil motif (Crick, 1953), which consists of alpha-helices coiled around each other to form a cohesive structure. The simple yet strong structure of coiled-coil regions makes them integral parts of a wide range of proteins. Long fibrous coiled-coils as seen in tropomyosin, a component of actin filaments in most animal muscles (Phillips et al., 1986), are composed of heptad repeats and can be hundreds of amino acids long. Shorter coiled-coil domains such as leucine zippers, which are found in many gene regulatory proteins (Landschulz et al., 1988), are also composed of heptad repeats, but span on the order of tens of amino acid residues. Because of the structural
advantages provided by coiled-coil regions, similar heptad repeats have evolved convergently in many different proteins, without the need for a common ancestor. These similarities can lead to the types of false-positive errors discussed above. When long stretches of two proteins feature coiled-coils domains, sequence similarity search programs have a hard time distinguishing between this chance similarity and similarity that should imply homology. The coiled-coil motif is prevalent in many non-homologous proteins, which breaks the assumption that allows sequence similarity to imply homology.

Periodic patterns of compositional bias, such as coiled-coils, were identified in the 2013 paper Challenges in homology search: HMMER and convergent evolution of coiled-coil regions (Mistry et al., 2013) as more likely to lead to inaccurate significance estimates in HMMER. The paper tested HMMER’s ability to correctly detect homology between select proteins and manually curated groups of homologous proteins, called families. It identified problematic protein motifs by looking for protein regions that matched two or more families not yet annotated as related in Pfam (Mistry et al., 2020), a large database of curated protein families. The paper established that HMMER struggled disproportionately when estimating similarity significance involving proteins with patterns of compositional bias. The problem of false-positives due to compositional bias exists in other homology search algorithms as well, and remains a source of error to this day (Jin et al., 2020). With a better model of common protein structures, similar low-complexity regions in two proteins would not be enough to signal significant sequence similarity, and to imply homology. While coiled-coils are a particularly well known example of sequences that can trip up current search algorithms, there are many forms of compositional bias in proteins which can cause inaccuracies in statistical estimation of similarity significance, including the RGG/RG motif (Thandapani et al., 2013), and transmembrane regions (Stevens
and Arkin, 2000). Figure 1.2 illustrates the highly biased nature that proteins with RGG repeats can exhibit. The RGG motif is not specific only to one protein family, and can be found in non-homologous proteins due to convergent evolution. From the prevalence of RGG repeats in the protein in Figure 1.2, we can see that two unrelated sequences that both independently feature the RGG motif might share a high amount of similarity by chance alone. The RGG motif and the coiled-coil motif are two particularly important examples of convergent evolution that can lead to inaccuracies in homology search.

![Figure 1.2: The full sequence of the protein HACA ribonucleoprotein complex subunit, from the gene GAR1 in the small-eared galago (a lemur-like primate endemic to Africa). This protein is necessary for ribosome biogenesis and telomere maintenance, and contains two arginine/glycine enriched regions, otherwise known as RGG regions. Repeats of RGG are highlighted in the above sequence to illustrate the degree of compositional bias of proteins containing the RGG motif.](image)

### 1.5 Current Standards for Detecting Bias

To decrease the number of false-positive hits, different sequence similarity search algorithms use a variety of methods to detect, and account for, compositional bias (Shin and Kim, 2004; Li and Kahveci, 2006; Lee et al., 2008; Pearson et al., 2016). Many of these methods fall into two categories: low-complexity sequence masking or biased composition score correction. Masking, as done by the program SEG (Wootton and Federhen, 1996) in BLAST, is used to filter out low-complexity regions in searches involving amino acid sequences. By identifying and ignoring low-complexity regions, BLAST can instead focus on regions that in theory feature less
compositional bias, and therefore produce more accurate search results. Although masking can be an effective way of limiting the effects of biased composition on homology search, it does not directly address the problem of inaccurate models of non-homology, but rather tries to make up for it.

Bias score correction involves adjusting the estimate of significance based on an assessment of the amount of bias in the sequences involved in the search. HMMER is one example of a program that uses bias score correction to lower the scores (which are estimates of similarity significance) of biased sequences. It uses two methods for bias score correction, both of which model protein sequences as having regions of biased composition and regions of unbiased composition. Models for the biased regions, which are generated on the spot and are search specific, are 0th order models, meaning that they cannot recognize local correlations that exist in protein sequences. The Challenges in homology search paper (Mistry et al., 2013) argued that HMMER’s 0th order bias score correction methods made it vulnerable to false-positive search results, because many common protein patterns—such as the coiled-coil—exhibit higher order local correlations. This suggests a possible way to improve the null model: use a higher order Markov model.

1.6 Experimental Question and Hypothesis

The scientific question this paper aims to address is the following: How can we reduce the number of false-positive results in protein similarity searches? We hypothesize that by creating a null model that more accurately captures the probability of observing certain patterns simply due to chance, we can increase the threshold of similarity needed for significance in sequences that feature compositional bias. This leads to the more specific question: How can we
best improve the null model? Given past work on the prevalence of coiled-coils in sequences that lead to many false-positive hits, we hypothesize that we can create a model to account for the presence of coiled-coils and other examples of compositional bias using variations on traditional Markov models.
2 Approach

2.1 Overview of Discontinuous Markov Model Approach

Looking at the log-odds likelihood score, shown below, we revisit the probability in the denominator: the probability that the two sequences being compared are non-homologous, and any similarity between them arose by chance and independently.

$$S(x|y) = \log_2 \frac{P(x|H_y)}{P(x|R)}$$ (1)

Building on the Markov model that is explicitly used in HMMER3, we first discuss an ideal scenario in which we are able to calculate exact probabilities of sequences. We then discuss approximating this ideal in order to be able to apply the resulting model in real calculations, and the assumptions made for this approximation. Finally, we discuss the new proposed null model, which uses a *discontinuous* $k$th order Markov model—for $k > 0$—to capture pattern compositional bias in proteins. Because capturing long patterns (of 7 or more residues) with traditional $k$th order Markov models would require a large $k$ and therefore a very large number
of model parameters, we decided to explore a non-traditional way of allowing small $k$ to span larger regions instead. We name the resulting Markov models \textit{discontinuous} because we aim to take into account a select subset of the residues that come before each amino acid when calculating its probability contribution—not just the residues immediately before it.

### 2.2 Calculating Exact Sequence Probabilities

We wish to calculate the probability of observing a certain protein sequence by chance. Let $X$ be a random vector of unknown length, that represents a sequence made up of amino acids. In order to evaluate the probability of observing a certain sequence, we can evaluate the probability that the random vector $X$ takes on a certain value, $x$, where $x$ represents a specific protein sequence of length $L$, which has elements $(x_1, x_2, ..., x_L)$. Evaluating the probability that we observe specific sequence $x$ is represented by $P(X = x)$. Examples of $x$ include a sequence of length $L = 3$ consisting of the amino acids R, G and G in this order, represented as (RGG). The order of the elements in $x$ is important; (RGG) $\neq$ (GGR). For next few sections we will be using the vector ABCDE as a simple fake protein, for the purposes of clarifying some of the equations with concrete examples.

For simplicity of notation, for the rest of this discussion we will abbreviate $P(X = x)$ using $P(x)$, and $P(X_n = x_n)$ using $P(x_n)$. Thus $x_n$ becomes the abbreviation for the event $X_n = x_n$. Using the chain rule of probability, we can obtain an exact formula for the probability of observing a given protein sequence:

$$P(x) = \prod_{n=1}^{L} P(x_n|x_1, ..., x_{n-1})$$  \hspace{1cm} (2)

Going back to the ABCDE fake protein example, equation (2) gives:

\[ P(ABCDE) = P(E|ABCD) P(D|ABC) P(C|AB) P(B|A) P(A) \]

While exact, this equation requires that we know the value of each conditional probability involved. The best way to estimate conditional probabilities for such Markov models is to calculate the frequency of appearance of all patterns in protein data. In order to use the probability \( P(x_L|x_1, \ldots, x_{L-1}) \) as part of our exact model, we would need to know the value of that probability for all possible values of \( x_1, \ldots, x_L \). For the protein alphabet, which consists of 20 amino acids, this leads to a total of \( 20^L \) different conditional probability values. The average protein sequence is hundreds of amino acid residues long, with many proteins stretching well into the thousands; a model with \( 20^{300} \) conditional probabilities would not be applicable in a real setting. It is therefore necessary to find an approximation to this exact calculation for practical application.

### 2.3 Approximating Sequence Probabilities

Protein sequence composition and structure is affected by intramolecular interactions between both proximal and distant amino acid residues. On average, however, closer residues have a more significant impact on each other than distant ones. Because for computational feasibility we needed to approximate the conditional probabilities of observing different amino acids based on the rest of the residues in the sequence, we decided to make the simplifying independence assumption that sequences exhibit an extension of the Markov property. Under this assumption, the probability of any individual residue depends only on a select number of the residues that came before it, rather than of all of them. This allows us to make the following approximation:
2.3 Approximating Sequence Probabilities

\[ P(x_n | x_1, ..., x_{n-1}) \approx P(x_n | x_{n-k}, ..., x_{n-1}) \]  (3)

Where \( k \) is some number, of our choosing, between 0 and \( L - 1 \). When \( k = 0 \), there is no index that falls in the range \( x_n, ..., x_{n-1} \), and thus the conditional probability in equation (3) becomes a marginal probability, i.e. \( P(x_n) \). The approximation in equation (3) is equivalent to saying that the identity of each amino acid in the sequence depends at most on \( k \) other amino acids. In this case, we will condition only on amino acids that come before our residue of interest, as opposed to considering amino acids before and after it. Through this formulation we can model protein sequences as \( k^{th} \) order Markov models.

There arises the problem of negative indexes into the sequence, when \( n - k \leq 0 \). To resolve this, we replace the conditional probabilities that involve \( n - k \leq 0 \) with the joint probability of all \( x_n \) for which \( n - k \leq 0 \). For \( k = 2 \) in our previous example, this approximation would yield:

\[ P(ABCDE) \approx P(E|CD) \ P(D|BC) \ P(C|AB) \ P(AB) \]

In general terms, this is:

\[ P(x) \approx P(x_1, ..., x_k) \prod_{n=k+1}^{L} P(x_n | x_{n-k}, ..., x_{n-1}) \]  (4)

For large \( k \), however, this joint probability of the first \( k \)-mer in the sequence can pose computational difficulties. Therefore we make yet another simplifying independence assumption for this last joint probability, and say that each residue is independent from all others. This allows us to replace the joint probability with the product of the marginal probabilities. For our example this becomes:

\[ P(ABCDE) \approx P(E|CD) \ P(D|BC) \ P(C|AB) \ P(A) \ P(B) \]
And again in general terms:

$$P(x) \approx \prod_{n=1}^{k} P(x_n) \prod_{n=k+1}^{L} P(x_n|x_{n-k}, \ldots, x_{n-1})$$

(5)

This additional independence assumption makes computing probability approximations for protein sequences more efficient, because it is easier to store 20 marginal probabilities, which can be used for all $k$, rather than store $20^k$ joint probabilities, which change based on $k$.

### 2.4 Basis for the Current Null Model

The most basic way to calculate the probability $P(X = x)$ using the approximation above, is to set $k = 0$. This assumes that the individual residues in the sequence are independent (as set by $k = 0$), and identically distributed. Identical distribution is key because it allows us to assume that the distributions for random variables $X_1, X_2, \ldots, X_L$ are all approximately the same. In other words, when generating the conditional probabilities we do not discriminate based on where in the sequence a particular residue falls. Following our example, this leads to the following approximation:

$$P(ABCDE) \approx P(E)P(D)P(C)P(B)P(A)$$

Going back to the homology search problem, the above model, when trained on a certain dataset $D$, is one possibility for the null model $R$. Changing the training dataset or the value of $k$ would define a new null model $R$. The training set is the source of the values for the different conditional and marginal probabilities in the models as described above.

A 0th order Markov model cannot, however, capture amino acid patterns, and thus will not fit all real protein sequences well. An example of why this might be a problem can be illustrated with RGG repeats in proteins. The $k = 0$ null model would assign the sequences
RGGRGG and GGRGRG the same probability. Given what we know about protein structure, however, we know that seeing RGGRGG is much more common, and should thus be assigned a higher probability by the null model.

### 2.5 The Proposed New Null Model

To allow the new model to capture information about amino acid patterns, we introduce an additional independence assumption. We first define $b$ as the farthest residue on which the conditional probability of $x_n$ depends. We use $k$ for the order of the Markov model. Next we hypothesize that $x_n$ may not heavily depend on all $b$ residues that come before it, but rather a subset of them. As an example, for $b = 4$ and $k = 2$ this could be:

\[
P(x_n|x_1, ..., x_{n-1}) \approx P(x_n|x_{n-4}, x_{n-3}, x_{n-2}, x_{n-1}) \approx P(x_n|x_{n-4}, x_{n-2})
\]

Where we decide that the residues $x_{n-1}$ and $x_{n-3}$ are not as important as $x_{n-2}$ and $x_{n-4}$ in calculating conditional probabilities for $x_n$. We call the model that arises from this additional approximation a discontinuous Markov model, because we don’t condition on continuous sets of prior residues. The dependencies of our discontinuous Markov models can be represented visually using a directed acyclic graph:

![Directed Acyclic Graph](image)

Where each arrow indicates a dependency of the node (representing amino acid residues) at the head of the arrow, on the node at the origin of the arrow. The beginning 3 nodes have no
dependencies, because there is no node \( n - 4 \) for them, and as defined above, we have decided to approximate those probabilities using each residue’s marginal probability.

Next we introduce notation to generalize this approach. To do this, we introduce a binary vector \( \mathbf{m} \), of length \( b \), with elements \( m_i \) where \( i \) ranges from 1 to \( b \). Each element \( m_i \) of \( \mathbf{m} \) takes on the value 1 if the residue \( x_{n-i} \) is to be included in the conditional probability of \( x_n \), and 0 otherwise. The total number of ones in vector \( \mathbf{m} \) is equal to \( k \), the order of the Markov model.

In the above example for \( b = 4 \), where we excluded \( x_{n-1} \) and \( x_{n-3} \) in the condition, the \( \mathbf{m} \) vector would be \( \mathbf{m} = [0, 1, 0, 1] \); \( m_1 = 0 \), which tells us that we do not condition on \( x_{n-1} \), while \( m_2 = 1 \), which tells us that we do condition on \( x_{n-2} \). In our ABCDE example, the approximation for the probability given \( b = 2 \) and \( \mathbf{m} = [0, 1] \) would turn into:

\[
P(ABCDE) \approx P(E|C)P(D|B)P(C|A)P(B)P(A)
\]

Using the vector \( \mathbf{m} \), we can further approximate equation (5) by dropping any terms \( x_{n-i} \) from the condition that involve an \( i \) for which \( m_i = 0 \). We now have a new null model \( R \) which depends on (1) the length of farthest look-back \( b \), (2) a subset of \( k \) prior residues to look at for conditional probabilities (indicated by \( \mathbf{m} \)), and (3) the training dataset \( D \). Using this notation, we can specify any null model using \( R(b, k, \mathbf{m}, D) \). We define \( A_n(\mathbf{m}, x_n) \) to be the set of look-back events defined by \( \mathbf{m} \) and position \( n \) of sequence \( x \).

Given that each conditional probability is less than (or rarely equal to) 1, multiplying hundreds of them will result in a number so small that modern computers will not be able to accurately store it. As an example, assume all conditional probabilities are equal to the naive estimation 0.05, based on the fact that there are 20 amino acids. The probability of a sequence with 400 residues would then be \( 0.05^{400} \approx 3.8 \times 10^{-521} \), a number which is outside the range of
2.6 Calculating Conditional Probabilities

double precision floating-point numbers, and would get rounded down to 0 on most computers.
To avoid such errors due to arithmetic underflow, sequence probabilities are often computed in log space. Using this, we turn the products in equation (5) into a sum:

$$\log_2 P(x|R(b, k, m, D)) \approx \sum_{n=1}^{k} \log_2 P(x_n) + \sum_{n=b+1}^{L} \log_2 P(x_n|A_n(m, x_n))$$

(6)

Where $A_n$ is the set of look-back events defined by $m$ and $x_n$, for each residue in the sequence, indexed by $n$. The vector $m$ is sufficient for defining $A_n$ because given any $m$ we know the corresponding $b$ and $k$ without further information; we include them in the model for the purpose of clarity. For notational convenience, we introduce an abbreviation for the i.i.d. $0^{th}$ model: let $R_0(D)$ be equal to $R_0(0, 0, [\ ], D)$, sometimes abbreviated just $R_0$ when the training set is implied or irrelevant. The core of this problem is in finding the best value $m$ that gives us the most accurate null model $R(b, k, m, D)$.

2.6 Calculating Conditional Probabilities

In order to implement the above Markov model, we create a transition matrix that gives us the conditional probability of observing each residue, depending on which discontinuous set of residues preceded it. The transition matrix can be displayed in a 2D table whose $(i, j)$ entry is the probability $P(X_n = x_i|X_{n-j} = x_{n-j})$ where $j$ is a set of numbers defined by $b$ and $m$.

Let us look at the concrete example of a transition matrix for the model $R(b = 3, k = 3, m = [1, 1, 1], D = \text{UniProtKB/Swiss-Prot})$ in Table 2.1 (Poux et al., 2017). From this table we can see:

$$P(X_n = A|X_{n-3} = A, X_{n-2} = A, X_{n-1} = C) = 0.0973$$

In order to calculate these probabilities from a training set of protein sequences, we calculate the number of times we see the residue of interest given the prior condition, and divide
by the total number of times we see the prior condition appear in the training set. Additionally, we use additive smoothing to avoid potential divisions by zero, in the case when some patterns do not exist in the training dataset. We do this by adding 1 to the numerator and 20 to the denominator; this way if there are no instances of a given pattern in the training dataset, we assume we have no information on the upcoming residue, and assign it a naive $\frac{1}{20}$ probability.

<table>
<thead>
<tr>
<th>Residue</th>
<th>AAA</th>
<th>AAC</th>
<th>AAD</th>
<th>...</th>
<th>YYV</th>
<th>YYW</th>
<th>YYY</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>0.2047</td>
<td>0.0973</td>
<td>0.1172</td>
<td>...</td>
<td>0.0676</td>
<td>0.0742</td>
<td>0.0843</td>
</tr>
<tr>
<td>C</td>
<td>0.0122</td>
<td>0.0173</td>
<td>0.0105</td>
<td>...</td>
<td>0.0168</td>
<td>0.0153</td>
<td>0.0197</td>
</tr>
<tr>
<td>D</td>
<td>0.0362</td>
<td>0.0418</td>
<td>0.0603</td>
<td>...</td>
<td>0.0828</td>
<td>0.0804</td>
<td>0.0680</td>
</tr>
<tr>
<td>E</td>
<td>0.0594</td>
<td>0.1058</td>
<td>0.0643</td>
<td>...</td>
<td>0.0520</td>
<td>0.0539</td>
<td>0.0409</td>
</tr>
<tr>
<td>...</td>
<td>...</td>
<td>...</td>
<td>...</td>
<td></td>
<td>...</td>
<td>...</td>
<td>...</td>
</tr>
<tr>
<td>V</td>
<td>0.0756</td>
<td>0.0564</td>
<td>0.0696</td>
<td>...</td>
<td>0.0599</td>
<td>0.0474</td>
<td>0.0473</td>
</tr>
<tr>
<td>W</td>
<td>0.0076</td>
<td>0.0107</td>
<td>0.0113</td>
<td>...</td>
<td>0.0073</td>
<td>0.0157</td>
<td>0.0108</td>
</tr>
<tr>
<td>Y</td>
<td>0.0182</td>
<td>0.0239</td>
<td>0.0316</td>
<td>...</td>
<td>0.0407</td>
<td>0.0421</td>
<td>0.0569</td>
</tr>
</tbody>
</table>

Table 2.1: Conditional probabilities for a model $R$ with $m=[1,1,1]$ trained on the UniProt protein database UniProtKB/Swiss-Prot. Each row represents the conditional probabilities of observing a certain amino acid in a protein, and each column represents the prior dependencies (e.g. $P(A|AAC)=0.0973$).

As an example, let us fix $b = 3$, $k = 2$, and $m = [1, 0, 1]$. Thus for each residue we are looking to calculate the probability $P(X_n = x_n | X_{n-3} = x_{n-3}, X_1 = x_{n-1})$. In order to calculate the probability $P(X_n = M | X_{n-3} = W, X_{n-1} = D)$, for example, we count the number of times we observe the sequence “WxDx”, and the number of times we observe the sequence “WxDM”, where “x” means any amino acid. The probability is then calculated by dividing $C(WxDM)$ by $C(WxDx)$ where the function $C$ counts the number of times a sub-sequence appears in the
2.6 Calculating Conditional Probabilities

We use the training set, and by applying additive smoothing,

\[ P = \frac{C(WxDM) + 1}{C(WxDM) + 20} \quad (7) \]

We do this for every combination of amino acids, out of the 20 possible, to fill out the 20 by 20^k transition matrix.
3 Materials and Methods

Given the pattern-based nature of protein sequences, we have hypothesized that a discontinuous Markov model will be able to more accurately model real non-homologous protein sequences than the i.i.d. $R_0$. Below we define model accuracy, how we tested our hypothesis based on this definition, and what we looked for when ranking the benefits of each model.

3.1 Definition of Model Accuracy and the $Q$ Score

The null model is meant to measure the probability that we would observe the sequence we are interested in if we pulled it randomly out of a bag of all real protein sequences. We therefore are looking to create a model that fits all real protein sequences well, and fake or shuffled sequences poorly. Such a model would assign sequences with more common patterns a higher probability, as opposed to sequences with no common patterns but the same residue frequency. This is because common patterns appear more often in many real protein sequences, and are therefore more likely to be observed. The more a model can distinguish between real protein sequences and shuffled sequences with the same residue frequency, the more accurate
it is. For the purposes of improving the log-odds score, we want to specifically find an index $m$ that best models real sequences with patterns of biased composition. We therefore look for a model that not only assigns high probabilities to the average protein sequence, but assigns sufficiently higher probabilities to sequences with patterns of biased composition.

We introduce score $Q(T, R)$ which we use as a measure of a model’s performance on a certain test dataset. $Q$ is the average improvement of a new model $R$ as compared to the i.i.d. model $R_0$ (trained on the same dataset) per residue in a test set. It is a function of this test dataset $T$ and a model $R(b, k, m, D)$. It is calculated as follows:

$$Q(T, R) = \frac{\sum_{j=1}^{N} \log_2(P(x_j|R(b, k, m, D)))) - \sum_{j=1}^{N} \log_2(P(x_j|R_0(D))))}{M}$$  \hspace{1cm} (8)

Where $N$ is the total number of sequences in the test set $T$; $x_j$ is the $j^{th}$ sequence in $T$, for $j$ ranging from 1 to $N$; and $M$ is the total number of amino acid residues in $T$. Intuitively, $Q$ is the per residue change in the log$_2$ probability of a sequence if we were to use a new null model $R$ instead of the i.i.d. $R_0$. It has a direct impact on the log-odds score used to detect homology, which can be rewritten from equation (1) as the following:

$$S(x|y) = \log_2 P(x|H_y) - \log_2 P(x|R)$$

All else being equal, the log-odds score $S(x|y)$ would change the following amount, if we were to change null models from $R_0$ to a new $R$:

$$\Delta S(x|y) = \log_2 P(x|R_0(D))) - \log_2 P(x|R(b, k, m, D)) = -Q(x, R) * L$$  \hspace{1cm} (9)

Where $L$ is the length of sequence $x$ in amino acid residues. A positive $Q$ indicates that the new model $R$ fits the sequence $x$ better than the old model $R_0$, and vice versa. We use $Q$ as our measure of model accuracy going forward, and look for new models with positive $Q$ on real
protein sequences, and negative $Q$ on fake sequences.

Using $Q$ as our metric of model accuracy, we tested different discontinuous Markov models based on a selection of indices $\mathbf{m}$ on a variety of test sets with pattern based biased composition. Figure 3.1 describes the experimental pipeline for testing different models, and the below sections describe the choices made for each part of the pipeline.

![Figure 3.1: Experimental pipeline for using metric $Q$ to test for the most accurate new null model.](image)

### 3.2 A Note on Notation

For notational convenience, we introduced $\mathbf{m}$ as a binary vector, which fit well into the mathematical notation of a Markov model. For computational purposes however, we needed to turn $\mathbf{m}$ into a binary number $m_b$ by flipping it and concatenating all of its members: $\mathbf{m}=[0,1,0,1]$ turns into $m_b=1010$, and $\mathbf{m}=[1,0,0,1,1]$ turns into $m_b=11001$. We further simplify this by con-
verters $m_b$ to base 10, and distinguish between vector $m$, binary $m_b$ and decimal $m_d$ accordingly. Additionally, we introduce another equivalent, which is more intuitively readable: $m_h$ is a vector of the prior residues taken into account by the model. For the example $m=[0,1,0,1]$, $m_h$ is equal to [2,4] because the 2nd and 4th elements of $m$ are set to 1. We use $m_h$ when discussing the comparative performance of each model in the results section. As a final example, all of the following are equivalent: $m=[0,1,1,1]$, $m_b=1110$, $m_d=14$, and $m_h=[2,3,4]$.

### 3.3 Choice of Indices to Test

Each new null model is defined by a file of conditional probabilities, each of which has file size $8 \times 20^{k+1}$ bytes; 8 bytes for each of the $20^{k+1}$ probabilities, represented as double-precision floating point numbers. To keep within reasonable computer disk usage, we decided to limit our file sizes to a maximum of 100 GB each. With this constraint, our maximum $k$ value was set to 6; $8 \times 20^7 = 10,240,000,000$ or 10.24 GB. The next step up would be $k = 7$, or a file size of 204.8 GB. Given our motivating compositional bias motif, the coiled-coil, and its heptad repeats, we decided to allow the maximum $b$ to be equal to 7, so as to capture the $i-7$ position using our model. We tested all indices that were within our maximum constraints of $k = 6$ and $b = 7$, which corresponds to all $m_d$ between 0 and 126 (equal to $m_h = [2, 3, 4, 5, 6, 7]$), inclusive.

### 3.4 Choice of Training Data Set

Each null model is defined by probabilities calculated from a certain training dataset $D$. When deciding which training dataset to use in our search for the best index $m$, we took into consideration two factors. The first was that the dataset needed to be large and comprehensive; it needed to include as many protein sequences as possible, in order for the model to get an
accurate sense of common protein patterns in all areas of life. One standard for comprehensive protein datasets is the Universal Protein Resource (UniProt) (Consortium, 2020), which curates protein sequence databases with different levels of annotation. The second factor was that we wanted to make sure the dataset was not enriched with redundant (identical or nearly identical) sequences, but still had complete coverage of protein sequence space. One of UniProt’s databases, the UniProt Reference Clusters (UniRef) (Suzek et al., 2015), provided the coverage of sequence space that we were looking for, while eliminating redundant sequences. Specifically, we chose to use the UniRef50 database, which is curated such that no two sequences in it share more than 50% identity.

To verify that UniRef50 evenly covers protein sequence space, we used jackknife resampling (Miller, 1974) to estimate the variance of scores $Q$ due to noise. We randomly shuffled the order the sequences in UniRef50 (but not the amino acids in each sequence), to create 10 training and test set pairs. Figure 3.2 shows the relationship between the test set and training set of each split. Using the ten test and training set pairs allowed us to run the same calculations ten times on different data.
3.5 Choice of Test Data Sets

For each training set \( D_i \) (where \( i \) ranges from 1 to 10) generated from UniRef50, and for each index \( m \) discussed in section 3.3, we trained a model \( R(b, k, m, D_i) \). This gave us a total of 1270 trained models to test for best performance, discussed in section 3.6.

3.5 Choice of Test Data Sets

We calculated \( Q \) scores for each of the 1270 models \( R \) on the following four types of test datasets. Manually curated test sets and instructions for reproducing all other datasets used in this thesis are provided at this link: Kaxiras21-supplement.tar.gz.

**Control Sets:** The control test sets are the ones generated from the jackknife resampling described above. For each of the ten splits, the test set is a representative random sample of 10% of UniRef50 that we then used to calculate the estimated \( Q \) for an average protein sequence. In other words, if we were to use a new model \( R \) instead of the i.i.d. \( R_0 \) to calculate
log-odds scores, then the $Q$ score on these control sets would represent the amount by which we expect log-odds scores to change on average.

**Shuffled Control Sets:** To check if our new null models $R$ could distinguish between real protein sequences and shuffled ones (which the i.i.d. $R_0$ cannot do), we created shuffled versions of each of the ten control test sets. To do this, we took each sequence in each of the test sets and shuffled its amino acid residues. This preserves the number of each amino acid in the sequence, but randomizes all patterns, so that the only difference between the shuffled control test set and the control test set is the removal of real protein patterns.

**Long Coiled-Coils Set:** Given the particularly problematic nature of sequences with coiled-coils in homology search, we manually curated a test set of 172 myosin and tropomyosin proteins to assess each model. We chose myosin and tropomyosin proteins due to the length of their coiled-coil regions—on the order of hundreds of amino acids long.

**RGG Set:** The other common pattern of compositional bias we tested was the RGG motif, discussed in section 1.4. For this motif, we manually curated a test set of 100 proteins enriched with RGG repeats (Thandapani et al., 2013).
3.6 Assessing Model Performance

To find the best replacement for the i.i.d. null model, we search for a new model that has a marginally positive $Q$ score for average sequences, and an even higher positive $Q$ score for sequences with patterns of compositional bias. To do this, we calculate $Q$ scores for the four datasets described above, and for each model, compare the performance on each dataset. The relative comparisons are detailed in the following section.

<table>
<thead>
<tr>
<th>Test Set</th>
<th>Number of Sequences</th>
<th>Number of Residues</th>
<th>Average Seq Length</th>
<th>Smallest Sequence</th>
<th>Longest Sequence</th>
</tr>
</thead>
<tbody>
<tr>
<td>Control Set #1</td>
<td>4,941,398</td>
<td>1,399,925,568</td>
<td>283.3</td>
<td>11</td>
<td>36,805</td>
</tr>
<tr>
<td>Shuffled Set #1</td>
<td>4,941,398</td>
<td>1,399,925,568</td>
<td>283.3</td>
<td>11</td>
<td>36,805</td>
</tr>
<tr>
<td>Coiled-Coil Set</td>
<td>172</td>
<td>174,089</td>
<td>1,012.1</td>
<td>146</td>
<td>2,245</td>
</tr>
<tr>
<td>RGG Set</td>
<td>100</td>
<td>34,352</td>
<td>343.5</td>
<td>194</td>
<td>714</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Other Sets</th>
<th>Number of Sequences</th>
<th>Number of Residues</th>
<th>Average Seq Length</th>
<th>Smallest Sequence</th>
<th>Longest Sequence</th>
</tr>
</thead>
<tbody>
<tr>
<td>UniRef50</td>
<td>49,410,134</td>
<td>13,992,611,441</td>
<td>283.2</td>
<td>11</td>
<td>45,354</td>
</tr>
<tr>
<td>Training Set #1</td>
<td>44,468,736</td>
<td>12,592,685,873</td>
<td>283.2</td>
<td>11</td>
<td>45,354</td>
</tr>
</tbody>
</table>

Table 3.1: Statistics for datasets used to produce the results in this thesis. The first of the ten of each training, control, and shuffled control sets are shown as an example. The remaining nine are approximately equal in number of sequences and average sequence length. Statistics on the full UniRef50 are given for reference.
4 Results

4.1 Amino Acid Patterns Hold Information

The foundation for the discontinuous Markov models is our hypothesis that some amino acid patterns are more common than their i.i.d. probability would lead us to believe. We hypothesize that by creating a model sensitive to these patterns, we can create an overall more accurate model of non-homologous proteins. Given the highly specific and amino acid-dependent nature of protein structure and function, we hypothesize that a model which assumes amino acids in a sequence are independent and identically distributed is not an accurate model of real proteins. A simple test of this hypothesis is to compare the joint probability of k-mers to the probability assigned by an i.i.d. model. For the 3-mer example ATG, the two probabilities to compare would be the i.i.d. probability $P(A)P(T)P(G)$ and the joint probability $P(ATG)$. We expect for some patterns that the joint probability will be much larger than the probability expected from an i.i.d. model, because those patterns are disproportionately frequently expressed in proteins. Figure 4.1 shows the results of this comparison for patterns of 2, 3, 4, and 5 residues.
4.1 Amino Acid Patterns Hold Information

RESULTS

Joint probabilities were calculated by counting number of occurrences of each k-mer pattern and dividing by the total number of patterns of length $k$. Using the counting function from section 2.6, the joint probability and i.i.d. probability respectively would be:

Figure 4.1: (A) Probability of observing two-residue patterns (2-mers such as AA or SL), as calculated by the i.i.d. model R0 on the x-axis vs. the observed probability of those patterns (calculated by taking number of occurrences of each pattern and dividing it by the total number of tuples in the dataset) on the y-axis. (B) The same calculation for 3-mer patterns, (C) 4-mer patterns, and (D) 5-mer patterns. All calculations done using the UniRef50 dataset. Outlier points labeled with their corresponding amino acid pattern.
Figure 4.1 validates our hypothesis that the observed joint probabilities of some amino acid patterns deviate greatly from their expected i.i.d. probability. Specifically, we see that repeats of the same amino acid display the greatest such deviation. Additionally, we see that as $k$ increases, the maximum relative difference between joint probability and i.i.d. probability increases as well, indicating that higher order Markov models (larger $k$) could assign more accurate probabilities to protein sequences than an i.i.d. model. In Figure 4.1B we point out the 3-mer RGG, discussed in section 1.4 as a motivating example of compositional bias. We see that the joint probability of observing RGG is only marginally higher than the i.i.d. probability, indicating that other, less studied, amino acid patterns may contribute even more bias to protein sequences.

4.2 Jackknife Resampling Standard Deviation

We found that the standard deviation of the average $\log_2$ probability per residue of the ten control test sets was negligible compared to the mean. Table 4.1 shows the indices $m_d$ with the lowest and highest standard deviations. Because of this low variance, which we also observed in the raw probabilities of models created by the jackknifed training sets, we decided to use the mean of the ten $\log_2$ probabilities for calculating our $Q$ scores. We thus turned the 1270 $\log_2$ probabilities per test set into 127 average $Q$ scores, one per model.
4.3 Model Parameter \( k \) has Largest Effect on \( Q \) Score

We found that the factor with the most influence on the score \( Q \) for the control test sets was the order of the Markov mode \( k \). While there was some differentiation between models with the same \( k \) and different index \( m \), the majority of the difference in \( Q \) score remained dependent on the order of the Markov model, especially for higher \( k \). Figure 4.2 illustrates this relationship. Figure 4.2 also illustrates one of the most important findings of this paper: that there is indeed information in protein patterns that can be captured by a simple Markov model. As \( k \) increases, we see that the models assign higher probabilities per residue to the average real sequence, while assigning a lower and lower probability to the same sequences but shuffled. This difference in \( Q \) scores between real sequences and shuffled sequences indicates that the new null models are able to distinguish between real and fake protein sequences. It also negates the possibility that the new null models are assigning higher probabilities to all, or to a random portion of, protein sequences, making the models useless for detecting compositional bias.

**Table 4.1:** Log\(_2\) probability per residue of each control test set, calculated using the model trained on the corresponding training set. Indices with the smallest standard deviation between log\(_2\) probabilities (\( m=2 \)) and largest standard deviation (\( m=111 \)) shown. For reference, a naive model that assigned every amino acid a probability of 0.05 would give all test sets a log\(_2\) probability of -4.322 per residue.
4.4 Models with \( k \) Greater Than 4 Can Detect Biased Composition

To be functionally useful in solving the problem of inaccurate estimates of sequence similarity significance, our new null models need to be able to distinguish between protein sequences with patterns of compositional bias and those without. We expect new models that are overall more accurate at estimating the probability of observing a random sequence to perform better, as compared to the i.i.d. model, on biased sequences than unbiased ones. This is because we hypothesized that the i.i.d. null model was more inaccurate for sequences with patterns of bias; therefore a model that now fits all real protein sequences about equally well should show

![Variance of Model Scores on Test Sets Based on \( k \)](image)

**Figure 4.2**: Swarm plot of average \( Q \) scores from each Markov model based on the order of the model \( k \). Each blue point represents one model’s \( Q \) score on the control test sets, and each red point represents one model’s \( Q \) score on the shuffled control test sets. The \( y=0 \) line indicates the score at which the new null model assigns equal probabilities to the sequences in the test set as the i.i.d. model \( R_e \). Positive scores indicate higher probabilities assigned by the new null model as compared with the i.i.d. model. Negative scores indicate that the new null model assigns a lower probability than the i.i.d model to the event that the test set contains real protein sequences.
more improvement on pattern-based biased sequences. Figure 4.3 shows the results of a comparison between on average unbiased sequences (the control test set), and highly biased ones (the coiled-coil set and the RGG set).

![Q Scores on Long Coiled-Coils and RGG vs Test Sets](image)

**Figure 4.3:** Increase in log2 probability per residue (Q) of sequences with patterns of biased composition vs. increase in log2 probability per residue of random samples of sequences from UniRef50. Each data point represents a different new null model R, some of which are labeled with their corresponding m, for reference. Boxed are all models with k=4, seen enlarged in Figure 4.4.

We see that as \( k \) increases, i.e. as the models incorporate more past information, \( Q \) scores on the biased test sets greatly surpass \( Q \) scores on the control test sets. We additionally see that the main factor affecting \( Q \) scores of the biased test sets is \( k \), just as it is on the con-
4.4 Models with k Greater Than 4 Can Detect Biased Composition

The effect of $k$ increases with higher $k$, as we see all models with $k = 6$ vastly outperform models with $k = 5$ on both the coiled-coil and RGG test sets. The black $x = y$ line indicates scores at which the new models would be assigning the same probability to the biased sequences as the control test sequences, on average. A positive deviation from this line indicates that the new null models are showing greater improvement, as compared to the i.i.d. model, on biased sequences than on unbiased ones. As discussed in section 4.2, the error bars for Figure 4.3 were too small to be visible, and were therefore omitted.

To get a deeper look at the effect of discontinuous vs. continuous dependencies on different Markov models, we inspect models with $k = 4$ and all possible combinations of prior residue dependencies. The results are shown in Figure 4.4 for both the RGG test set and the coiled-coil test set. Of particular interest is the highest point on the coiled-coil graph (Figure 4.4B), which represents the model with prior dependencies $m_h = [1, 3, 4, 7]$. Given our knowledge of coiled-coil heptad repeats (Cohen and Parry, 1986) with the most structurally important, and therefore most conserved, positions at $i - 3$, $i - 4$, and $i - 7$, we see that a model that takes these positions into account is able to detect coiled-coil regions. Furthermore, we see that the models that perform the best on the coiled-coil test set overwhelmingly take into account positions $i - 4$ and $i - 7$, while positions $i - 2$, $i - 5$, and $i - 6$ do not seem to contribute much. Similarly, position $i - 3$ seems to have a disproportionately large effect on a model’s performance, as compared with other positions. Even though the error bars on each point are quite low, and the spread of models models with different $m$ is significant, it is still too small to be functionally useful.
Figure 4.4: Relative Q scores for (A) the RGG test set vs. the control test sets, and (B) the coiled-coil test set vs. the control test set. Graphs zoomed in to encompass only models with \( k = 4 \), to show difference in Q score due only to varying which 4 prior residues the model takes into account. Points labeled with their \( m_i \) indices.
5 Conclusion

5.1 Implications of Results

The new models we tested showed significant improvement over the i.i.d. null model, however models of the same $k$ and different $m$ did not differ as significantly as we expected. This implies that any impact a new null model would have on the log-odds score would be mostly determined by the order of the Markov model, $k$, and that a discontinuous model (such as $m_h = [1, 4, 7]$) is not significantly better than one with continuous prior dependencies (such as $m_h = [1, 2, 3]$). Despite this result, continuous Markov models do significantly outperform the i.i.d. null model, and thus implementing a new null model $R$ for use in real homology search programs such as HMMER could help lower the number of false-positive results caused by proteins subject to convergent evolution and patterns of compositional bias.

5.2 Further Work

While the discontinuous models we tested showed potential for increasing the accuracy of the non-homology model, there are still several areas to explore for further improvement.
Given a limited amount of time, we only tested 126 different indices \( m \), which were constrained by a maximum \( k = 6 \) and \( b = 7 \). Increasing the maximum \( k \) and maximum \( b \) would allow us to search a wider area of models, potentially uncovering models that significantly outperform others of the same \( k \). Of particular interest are the models with \( 3 \leq k \leq 5 \), which have corresponding conditional probability files of sizes between 64KB and 25.6MB. Due to their file sizes, such models are more practically useful than models with more parameters.

The other area of potential improvement is in the training dataset. We hypothesized that a training set that evenly covered protein space would be the best to train a model of non-homology that fit all real proteins approximately equally. However, different domains of life have differing frequencies of amino acids in their proteins (Bogatyreva et al., 2006). This information could be harnessed to make an even more accurate new null model for the different domains of life. We could train several different models, perhaps one each for archaea, bacteria, and eukarya, as well as one general one, for when the taxonomic classification of a certain search is unknown. Although the current new null models with \( k \geq 4 \) show significant improvement as compared to the i.i.d. null model in detecting biased composition, we have yet to exhaust all possible further improvements.
6 References


42