Sequence Conservation Metrics:
Implementation and Comparative Analysis

Master's Thesis
Bioinformatics Masters
Degree Programme,
Institute of Medical Technology,
University of Tampere, Finland.
Ayodeji, E. Olatubosun.
August 2009.

ACKNOWLEDGEMENTS

I owe great gratitude to my supervisors, Martti Tolvanen and Pentti Riikonen, for the time devoted to putting me through in the course of undertaking this project. Special thanks to Martti for reading and re-reading the manuscript. My acknowledgements also go to my group leader, Prof. Mauno Vihinen, for his corrections.

I also want to acknowledge the financial and moral contributions of my parents, Mr. and Mrs. Olatubosun, to the success of this program. My thanks also go to Senator Bode Olajumoke (Ph.D., LL.M., BL.) for his assistance. To my siblings and all my friends, I say thank you all. Finally, all thanks to God for sparing my life till this day and giving me the wisdom to undertake this task.

August, 2009.

Ayodeji E. Olatubosun.
Bioinformatics Programme,
Institute of Medical Technology,
University of Tampere, Finland.

MASTER'S THESIS

Place:	UNIVERSITY OF TAMPERE Bioinformatics Masters Degree Programme, Faculty of Medicine, Institute of Medical Technology, Tampere, Finland.
Author:	Ayodeji, E. Olatubosun.
Title:	Sequence Conservation Metrics: Implementation and Comparative Analysis.
Pages:	61 pp + appendices 18 pp
Supervisors:	Martti Tolvanen, Pentti Riikonen.
Reviewers:	Prof. Mauno Vihinen, Martti Tolvanen.
Time:	August, 2009.

Abstract

Background and Aims: Multiple Sequence Alignments (MSAs) help identify regions of similarity and dissimilarity between its constituent sequences, which is essential in understanding structural, functional and evolutionary relationships of DNA or protein. Different authors from different fields have made efforts to develop new quantitative methods or apply metrics used in other fields in scoring conservation at different positions in MSAs, using different programming languages. This work is an effort to implement many of these methods on a single platform familiar to most bioinformaticians, which will allow direct comparison of the methods, and make them more accessible as a tool to the research community.

Methods: Seven conservation methods namely: Entropy, Variance, Sum of Pairs, Relative Entropy, Jensen-Shannon Divergence method, and the deviation from maximum frequency method and its log form were implemented in Object Oriented Perl. Two weighting schemes and two background frequency schemes were also implemented, leading to 27 variants of the methods. These were used in computing conservation scores for 502 alignments. Correlation analysis and Receiver Operating Characteristic analysis was conducted on the conservation data.

Results: The correlation result agrees closely with that of a previous study. The independent counts scheme gave the best performance as compared to other weighting options. Relative Entropy, Variance and the Jensen-Shannon Divergence were the best three methods as measured by their performance on the catalytic site dataset.

Conclusion: The conservation metrics were successfully implemented and comparative evaluation carried out. A correct specification of the true positive and true negative classes is however needed in order to make generalised statements about the comparative performances of the methods tested.

CONTENTS

ABBREVIATIONS

CSA	Catalytic Site Atlas
Diff	Conservation calculated using the deviation from maximum frequency method
Entropy	Conservation calculated using Entropy method
HDiff	Conservation calculated using the deviation from maximum frequency method and position-based sequence weights scheme
HEntropy	Conservation calculated using Entropy method and position-based sequence weights scheme
HJSD_Local	Conservation calculated using Jensen-Shannon Divergence method and background residue frequencies from the alignment and position-based sequence weights scheme
HJSD_Swiss	Conservation calculated using Jensen-Shannon Divergence method and background residue frequencies from SwissProt and position-based sequence weights scheme
HLogDf	Conservation calculated using the Log of the deviation from maximum frequency method and position-based sequence weights scheme
HRE_Local	Conservation calculated using relative entropy method and background residue frequencies from the alignment and position-based sequence weights scheme
HRE_Swiss	Conservation calculated using relative entropy method and background residue frequencies from SwissProt and position-based sequence weights scheme
HVariance	Conservation calculated using the variance method and position-based sequence weights scheme
icDiff	Conservation calculated using the deviation from maximum frequency method and the independent counts scheme
icEntropy	Conservation calculated using Entropy method and the independent counts scheme
icJSD_Local	Conservation calculated using Jensen-Shannon Divergence method and background residue frequencies from the alignment and the independent counts scheme
icJSD_Swiss	Conservation calculated using Jensen-Shannon Divergence method and background residue frequencies from SwissProt and the independent counts scheme
icLogDf	Conservation calculated using the Log of the deviation from maximum frequency method and the independent counts scheme

icRE_Local	Conservation calculated using relative entropy method and background residue frequencies from the alignment and the independent counts scheme
icRE_Swiss	Conservation calculated using relative entropy method and background residue frequencies from SwissProt and position-based sequence weights scheme
icVariance	Conservation calculated using the variance method and the independent counts scheme
JSD_Local	Conservation calculated using Jensen-Shannon Divergence method and background residue frequencies from the alignment
JSD_Swiss	Conservation calculated using Jensen-Shannon Divergence method and background residue frequencies from SwissProt
LogDf	Conservation calculated using the Log of the deviation from maximum frequency method
MSA	Multiple Sequence Alignment
RE_Local	Conservation calculated using relative entropy method and background residue frequencies from the alignment
RE_Swiss	Conservation calculated using relative entropy method and background residue frequencies from SwissProt
ROC	Receiver Operating Characteristics
Variance	Conservation calculated using the variance method

1. Introduction

Positional conservation scoring is a method of quantifying the variability in columns of multiple sequence alignments, based on the notion that sequences in the alignment are homologues. The more invariable a column is, the higher the conservation level of that column. Conservation scores have been computed considering phylogenetic relationship among the sequences in the alignment (Armon *et al.*, 2001; Lichtarge *et al.*, 1996; Mayrose *et al.*, 2004; Pupko *et al.*, 2002) and also without any such consideration (Liu and Guo, 2008; Mirny and Shakhnovich, 1999; Pei and Grishin, 2001). Valdar (2002) gives a comprehensive review focused mainly on methods in the latter category. These methods are easier to implement and make no assumption about the phylogenetic relationship between the sequences in the alignment under consideration.

Over time, different efforts have been made by different authors from diverse fields to develop new quantitative methods or apply metrics used in other fields of science in scoring the conservation at different positions of an alignment. Unfortunately, many of these methods have been implemented on different platforms, in different languages, and most often for specific purposes, often making only their publications widely accessible. Additionally, some new methods have been implemented between the last popular review of residue conservation scoring methods (Valdar, 2002) and now. A new effort is thus needed to review these methods, implement them, compare them, thereby providing a singular reference point and unbiased assessment of the performance of these different methods. This thesis work was thus embarked upon to achieve the following.

1.1 Aim and Objectives

The general aim of this research work is to develop various positional conservation indices in the Perl language, and thus make them available to the entire research community.

Objectives include:

a. Develop the program modules using object oriented Perl and BioPerl. Using object oriented development techniques will enhance scalability of the

program. Additionally, the use of Perl and BioPerl will enhance accessibility and usability in the bioinformatics domain.

b. Measure the relative efficiency and/or correspondence of some of the different methods, using the following procedures:
   i. Measuring conservation indices for different columns of alignments retrieved from earlier publication(s) in the same research domain. This will enhance comparability of results.
   ii. Measurement of correlation between the different methods.
   iii. Carrying out ROC analysis to determine the performance of the methods on catalytic site dataset.

1.2 Significance of the study

This research work aims to implement many of these residue conservation scoring metrics in a language which is easily accessible and usable for bioinformatics and life science specialists – Perl. The comparative analysis of the methods will provide an up-to-date and independent assessment of their performance. Having the methods implemented on a single platform with the same language will also enhance comparability.

2. Review of Literature

2.1 Multiple Sequence Alignment

Protein, DNA and RNA sequence information are represented as sequences over a finite alphabet; {ACDEFGHIKLMNPQRSTVWY} for protein, {ACGT} for DNA and {ACGU} for RNA. A global pairwise alignment of two or more sequences arranges the sequences into a rectangular array with the goal that amino acid residues (or nucleic acid bases) in a given column are homologous (i.e. derived from a single position in an ancestral sequence), superposable (in a rigid local structural alignment) or play a common functional role (Edgar and Batzoglou, 2006). Gaps are introduced in a sequence position if it does not match the residues in the other proteins.

Several techniques have evolved to exploit the conservation encoded in sequence alignments. Some methods assume that homologous sequences share conserved motifs, which are presumably crucial to the structure or function of the protein. The structures of these motifs are used to build discriminators for particular protein families. An unknown query sequence may be searched against a library of such descriptors (pattern database) to determine whether or not it contains any of the predefined characteristics. The structure or function of the query sequence could be deduced based on those of the matching sequence. Searches of pattern databases thus theoretically offer a fast track to the inference of biological function. Examples of such resources include PRINTS (Attwood, 2002) and BLOCKS (Henikoff *et al.,* 1999; 2000). Domain profiles (or HMM) based upon such formalism (Sigrist *et al.,* 2002, Mulder *et al.,* 2003; Finn *et al., 2008*) are important tools in protein characterization. All the aforementioned methods are based on multiple sequence alignments (MSAs).

The rate of generation of new protein primary structures far outpaces that at which their secondary structures could be experimentally determined. It is this gap that protein comparative modeling hopes to bridge. According to Marti-Renom *et al.*(2000) the steps involved in comparative modeling include finding suitable template protein(s) related to the target; aligning target and template(s) sequences; identifying structurally conserved regions; predicting structurally variable regions, including insertions and missing N and C termini; modeling side chains; and refining

and evaluating the resulting model. Indeed, without properly aligned homologous sequence(s) input into the process, the quality of the output models will be less than optimal. MSAs are thus critical to the success of comparative modeling of protein structure.

Molecular phylogenetics attempts to determine the rates and patterns of change occurring in DNA and proteins and to reconstruct the evolutionary history of genes and organisms. The process of phylogenetic analysis requires a valid MSA as input. The validity of subsequent phylogenetic analyses is dependent, to a great extent, on the validity of the input multiple sequence alignment (Morrison and Ellis, 1997). Succinctly put, all analyses of phylogenetic relationships derived from sequence data are fundamentally based upon the input MSA (Phillips *et al.,* 2000). This underscores the importance of multiple sequence alignments in phylogenetic analyses.

Additionally, MSAs are useful in sequence annotation and gene finding, and in many other biological analyses. They are thus very invaluable tools in biological analyses.

The problem of pairwise alignment has quite simple algorithms using dynamic programming (Needleman and Wunsch, 1970; Smith and Waterman, 1981), but directly extending these to aligning multiple sequences is computationally infeasible for more than a few sequences (Lipman *et al.,* 1989; Wang and Jiang, 1994). As a result, many algorithms based on heuristics have been designed to tackle this problem. Progressive methods (Feng and Doolittle, 1987) assemble a multiple alignment by making a series of pairwise alignments of sequences or pre-aligned groups, the order of which is guided by a dendrogram, so that similar sequences are aligned before divergent ones. These methods do not guarantee an optimal solution, and errors made in earlier steps cannot be corrected in later stages of the alignment process. Classical methods such as ClustalW (Thompson *et al.*, 1994) use general residue substitution models to implement a fast algorithm and produce rather reasonable results, but they display limited success in aligning divergent sequences accurately. KAlign (Lassmann and Sonnhammer, 2005) enhanced the progressive alignment approach by employing an approximate string-matching algorithm to calculate sequence distances and by incorporating local matches into the otherwise global alignment. The algorithm was well suited in terms of speed and accuracy to deal with large-scale comparative

genomics. This algorithm has recently been improved for faster speed and better memory utilization (Lassmann *et al.,* 2009).

Iterative refinement and consistency scoring are two methods commonly used to minimize errors in progressive alignment steps. Iterative refinement involves repeatedly dividing the aligned sequences into sub-alignments and realigning the sub-alignments. MAFFT (Katoh *et al.*, 2002) and MUSCLE (Edgar, 2004) are contemporary programs that mainly rely on iterative refinement with scoring based on general amino acid substitution models to enhance alignment quality.

The consistency-based scoring function for two sequences depends on their pairwise alignments, as well as on how they can be aligned with regard to other sequences. This ensures that the alignment scoring function for two sequences contains information of their alignments to other sequences as well. This approach was first employed by the program T-Coffee (Notredame *et al.*, 2000). For progressive alignments without refinement steps, consistency scoring is superior to scoring based on general amino acid substitution models.

Other approaches have also been employed. PCMA (Pei *et al.*, 2003) measures consistency of sequence profiles instead of that of individual sequences. ProbCons (Do *et al.*, 2005) employs a probabilistic treatment of consistency through pairwise hidden Markov models (HMMs) (Pei and Grishin, 2006) while in MUMMALS (Pei and Grishin, 2006), more complex HMMs are designed that implicitly capture information of unalignable regions, secondary structure and solvent accessibility in probabilistic consistency scoring. HMMs are advantageous in that they give an estimation of the probability of any two residues being aligned. This technique has yielded additional improvement to consistency-based progressive alignments. The consistency scheme can effectively incorporate different sources of constraints such as local alignments, global alignments and structure-based alignments when available.

Aligning sequences with similarity in the “twilight zone” (below 25-20% identity) is quite problematic. Methods that rely solely on sequence information perform poorly in this problem area. Many methods have thus sought to address this problem by exploring additional information apart from the input sequences. Such information include sequence homologs, predicted secondary structures and 3-dimensional

structures. DbClustal (Thompson *et al.*, 2000) combines local alignments found in database searching with global ClustalW alignments. Mafft-homologs (in MAFFT package (Katoh *et al.*, 2005) aligns target sequences together with database homologs found from searches in order to obtain more accurate alignments for the targets. PROMALS (Pei and Grishin, 2007) identifies database homologs using PSI-BLAST (Altschul *et al*., 1997) searches. This information is applied in building sequence profiles and predicting secondary structures, employing profile-to-profile comparisons enhanced with secondary structural information in the alignment processes. 3DCoffee (O'Sullivan *et al.*, 2004) uses SAP (Taylor, 1999) structure-based alignments and FUGUE (Shi *et al.*, 2001) sequence-to-structure alignments to improve alignment quality. Recently, the Expresso server (Armougom *et al*., 2006) extended the 3DCoffee method by automatically identifying highly similar 3D structural templates for target sequences and using structural alignments for consistency-based alignments.

The problem of estimating quality of MSAs has been studied intensively (see Elofsson, 2002; Ahola *et al.,* 2006, 2008). The measures quantifying alignment quality can be divided into those relying on conservation (Pei and Grishin, 2001; Carrillo and Lipman, 1988; Thompson *et al.*, 2001; Beiko *et al.*, 2005; Tomovic and Oakeley, 2007), consistency (Mevissen and Vingron,1996; Lassmann and Sonnhammer, 2005; Landan and Graur, 2007, 2008) or structural similarity (Armougom *et al.*, 2006; O'Sullivan *et al.*, 2003). Conservation is thus also important as a measure of alignment quality.

2.2 Conservation

A position in a multiple sequence alignment is said to be conserved if there is a residue at that position that is represented in a significantly higher amount, as compared to other residues in the same column, or if substitutions occur in such column such that some physicochemical properties are conserved. According to Valdar (2002), the patterns of amino acid variability in an MSA column gives insight into the evolutionary pressure, mutation, recombination, and genetic drift that often spans millions of years. Highly conserved positions are deemed to have been under functional and/or structural constraint, and as such amino acid substitutions or deletions at these positions are deleterious and such homologues are promptly eliminated. Deductively, the identification of highly conserved positions in an

alignment is therefore of very great importance because it helps in the identification of the structural and functional parts of the constituent proteins.

Valdar (2002) gives some desired properties of a good conservation scoring function. It should be a function that maps a set of arguments in the input space (which includes the aligned column and possibly other information) to a number in the output space. It is desired that the output space be continuous and bounded. The score should take account of the amino acid frequency in the alignment column, recognize conservative replacement and the fact that some substitutions incur more chemical and physical change (such as change in size) than others. The score ought to penalize positions with numerous gaps. Additionally, a good conservation score should normalize against sequence redundancy. An additional criterion is that an invariant position should obtain a maximum score regardless of the particular amino acid type (Fischer *et al.*, 2008).

2.2.1 Sequence Weighting

Numerous highly identical sequences in a multiple sequence alignment could lead to misleading conservation score results. Weighting strategies have thus been devised that assign lower weights to highly identical sequences and higher weights to more divergent ones. Various weighting schemes are described or referenced in Pei and Grishin (2001) and Valdar (2002), but the current discussion will be limited to the ones utilized in this study.

The easiest but naïve approach to weighting is to have no weight at all, that is, to use the actual fractional frequency of residues in a column directly in conservation score calculation. The position-based sequence weights scheme, a more widely used formulation by Henikoff and Henikoff (1994), seeks to maximize the spread of data in aligned columns. Sequences are first weighted at individual positions then these positional weights are combined to derive the overall sequence weights. The weight of the i^{th} sequence at position x is computed as:

$$w_x^i = \frac{1}{k_x n_x^i}$$

where k_x is the number of amino acid types present in position x and n_x^i is the frequency of the i[th] sequence's amino acid at position x. This is then averaged over all positions so that the ultimate weight of sequence *i* becomes

$$w^i = \frac{1}{L}\sum_{x=1}^{L} w_x^i$$

where L is the length of the alignment.

Pei and Grishin (2001) introduced the notion of estimated independent counts. The estimated independent count scheme was designed to correct for correlation between aligned sequences. They stated that the number of independent observations (counts) of amino acid *a* at alignment column *x* is equal to the effective number of sequences that contain amino acid *a* at that column. The independent count for amino acid *a* at position x is given as:

$$N_x^a = \ln\left(1 - F_x^a \big/ 20\right) \Big/ \ln 0.95$$

F_x^a was chosen as the average number of different amino acids per position, considering only those sequences containing amino acid *a* at position *x*.

2.3 Measures of Protein Sequence Conservation

Many computational methods have thus been devised to quantify protein sequence positional conservation in multiple sequence alignments. These methods are highlighted subsequently.

2.3.1 Shannon Entropy

The Shannon information theoretic entropy (Shannon, 1948) is one of the popular measures employed in studying conservation at alignment positions. Its original usage was in information theory (Durbin *et al.*, 1998) but it was proposed for use in conservation analysis by Shenkin *et al.* (1991).

Suppose there are *n* possible events with probabilities of occurrence $p_1, p_2, \ldots, p_n$, then the Shannon entropy is a measure of the amount of choice involved in the selection of the events. It could also be viewed as how uncertain we are of the outcome. It is given by the formula:

$$-\sum_{i=1}^{n} p_i \log p_i$$

Some Properties of Shannon Entropy (H)

H = 0 if and only if one of the p_i's is one and the rest are zeros. This means that when we are certain of the outcome H vanishes. This means that the more certain the outcome is, the lower the value of H. For a given n, H is maximum and equal to log*n* when all the p_i's are equal (1/n). This scenario is the most uncertain situation, and thus H is again correct by assigning the maximum entropy to the most uncertain scenario.

Some residues are easily substituted with others based on similarity in physicochemical properties (eg. Lysine (K) is much more easily substituted with Arginine (R) than with Isoleucine (I), thus a column consisting of K and R should have a better conservation score than a column having K and I, in the same proportion as the latter). The main shortcoming of conservation scores based on the entropy score is the fact that they fail to account for physicochemistry, and will thus assign the same conservation score to the two columns described in the above scenario (Valdar, 2002).

In an attempt to remedy the insensitiveness of entropy based conservation scores to the physicochemical properties of amino acid residues in an alignment column, amino acids were divided into 6 classes based on physicochemical properties (Mirny and Shakhnovich, 1999). The classes and their constituent amino acids are aliphatic {AVLIMC}, aromatic {FWYH}, polar {STNQ}, positive {KR}, negative {DE}, and special {GP}. The resultant entropy derived conservation score was then

$$s(l) = \sum_{i=1}^{6} p_i(l) \log p_i(l)$$

Although this scoring scheme basically recognizes the similarity between amino acids grouped into the same class, it fails to recognize the relative distribution of amino acids in the same class. As an example, under this scheme, a column having residues from only a single class will have a conservation score of zero (maximum conservation). The scheme therefore leads to a reduction in the sensitivity of the entropy score to residue diversity, while it conveys some degree of physicochemical sensitivity.

2.3.2 Conservation Measure Based on Joint Entropy

Mihalek *et al.* (2007) used the idea of joint entropy between two variables and additionally incorporated the information about the mutational preferences of amino acids to quantify conservation in MSA columns. The joint entropy between two distributions X and Y is given as:

$$H(X,Y) = -\sum_{x}\sum_{y} P(x,y)\ln P(x,y)$$

where $P(x,y) = probability(X = x \ \ and \ \ Y = y)$

They then applied this to a single distribution of amino acid at a column, X, repeatedly, and also incorporated the information about mutational preferences thus:

$$H_{BB}(X,X) = -\sum_{x_1}\sum_{x_2<x_1} P(x_1,x_2)\ln\frac{P(x_1,x_2)}{Q(x_1,x_2)}$$

$$P(x_1,x_2) = \frac{N(x_1,x_2)}{L(L-1)/2}$$

$Q(x_1,x_2)$ is said to play the role of the "background" mutational propensity. This is incorporated in a Kullback-Leibler-like approach (see 2.3.3 below). $Q(x_1,x_2)$ is estimated from the original set of matrices used in constructing BLOSUM using some transformations including a Monte Carlo procedure.

Unfortunately, even in the authors' own words, this method still lacks the ability to detect the pressure to conserve a particular physicochemical characteristic. The added complexity is not justified considering the fact that it has nothing to offer that makes it better than the computationally simpler entropy scheme.

2.3.3 Conservation Measure Based on Relative Entropy

Given two probability distributions P(x) and Q(x), there are two relative entropies (Thomas and Joy, 1991) (also referred to as Kullback-Leibler divergence) associated with these viz:

$$D(P \parallel Q) = \sum_{x\in X} P(x)\log\frac{P(x)}{Q(x)} \quad \text{and} \quad D(Q \parallel P) = \sum_{x\in X} Q(x)\log\frac{Q(x)}{P(x)}$$

It is assumed that these two distributions have the same range, i.e. $P(y_i) > 0$ if and only if $Q(y_i) > 0$, for all i.

The relative entropy between two distributions measures the dissimilarity between them, but it is not symmetric (i.e. $D(P||Q) \neq D(Q||P)$, and thus not a true distance measure, though often used in this sense. $D(P||Q)$ is a measure of the inefficiency of assuming that the distribution is Q when the true distribution is P (Thomas and Joy, 1991). It is always non negative, zero if and only if P = Q, but its upper limit is not bounded.

Some attempts have been made at applying the notion of relative entropy in measuring conservation. Williamson (1995) divided the amino acids into 9 sets according to the limited conservative amino acid substitution scheme viz: {Met, Leu, Val, Ile}, {His, Arg, Lys}, {Ser, Thr}, {Ala, Gly}, {Asp, Glu}, {Gln, Asn}, {Phe, Trp, Tyr}, {Pro} and {Cys}. They then applied the relative entropy approach as given in the following formula:

$$I(S) = \sum_{i=1}^{9} f(G_i, S) \ln\left[f(G_i, S) / g(G_i)\right]$$

Where $f(G_i,S)$ is the frequency of group G_i in alignment position S and $g(G_i)$ is the fractional frequency of group G_i in the whole alignment. This grouping resembles closely the one taken by Mirny and Shakhnovich (1999) and is subject to the same shortcomings (see 2.3.1).

Wang and Samudrala (2006) used the relative entropy in the context of measuring conservation. Assuming p_i is the frequency of residue i in an alignment column and q_i is the frequency of the same residue i in an appropriately large collection of sequences (e.g. in SwissProt, or even in the alignment itself), then their application of relative entropy is:

$$D(P \,\|\, Q) = \sum_{i=1}^{20} p_i \log_2 \frac{p_i}{q_i}$$

This function measures conservation in a column as the dissimilarity between the amino acid distribution in this column and the amino acid distribution in the background frequency. In other words, it measures the inefficiency of assuming the distribution of amino acids in the column comes from the same distribution as the one in the background. Highly conserved columns will have a distribution highly different from the background distribution, while columns with low conservation will have a

distribution quite similar to the background distribution. The higher the conservation of an alignment column, the higher it will be scored by this formula. This method is thus intuitively correct.

The main shortcoming of this method of measuring conservation is that it is not bounded at the upper end. The discriminating power of the method is also very much dependent on the distribution used as the background.

2.3.4 Conservation Measure based on Jensen-Shannon Divergence

Jensen-Shannon divergence is a symmetrized and smoothed version of the Kullback-Leibler divergence (Topsøe 2003). The general form of the Jensen-Shannon divergence could be given as

$$\sum_v \alpha_v D(P_v \parallel Q)$$

where P_v and Q are distributions, α_v is a scalar and $D(P_v \parallel Q)$ is a relative entropy term, meaning the distance between P_v and Q.

According to Topsøe (2003), the JSD formula could be viewed in terms of a source which generates a string $x_1x_2\ldots x_n$ of "letters", each letter selected independently of previous letters and according to a specific distribution among the P_vs and in such a way that the probability that P_v is used for the selection is α_v. A Q that minimizes the JSD expression is thus a distribution that could be used to code for the output of the source with minimal errors. Such a Q has been shown to be $\overline{P} = \sum_v \alpha_v P_v$ (Topsøe, 2003).

The specific Jensen-Shannon divergence between two distributions P and Q could be given as:

$$JSD(P,Q) = \lambda D(P \parallel M) + (1-\lambda) D(Q \parallel M)$$
$$M = \lambda P + (1-\lambda) Q$$

with M being a weighted average of distributions P, and Q. λ, which is the weighting factor, is often taken to be 0.5,

Given an alignment column with amino acid residues having probability distribution $P = p_1, p_2, \ldots, p_{20}$, over the 20-letter amino acid alphabet, and given that $Q = q_1, q_2, \ldots, q_{20}$, is the distribution of the amino acids in a suitable background protein sequence

collection, then JSD(P,Q) is the most likely common source distribution of both distributions P and Q, with λ as a prior weight (Yona and Levitt, 2002). If distributions P and Q are very similar, then JSD(P,Q) gives a value quite near to zero, with JSD(P,Q) being zero for identical P and Q. Very dissimilar P and Q gives values near to 1. JSD(P,Q) is bounded in the range [0,1].

Capra and Singh (2007) used the two-distribution version of the Jensen-Shannon divergence to characterize the level of conservation at different positions in an alignment. The two distributions used are the column amino acid distribution and the background distribution. Their background distribution was derived from the overall amino acid distribution in the BLOSUM62 alignments. They alluded to a slight improvement in the performance of the system, if alignment-specific backgrounds were used, but they did not implement this due to the added complexity envisaged.

The two-distribution Jensen-Shannon divergence basically computes the minimum weighted distance between two distributions to a third distribution. Using this method boils down to the same idea used in the relative entropy approach, but with the added advantage that the resulting conservation scores are bounded, so that a maximally conserved column gets a score of 1, with the score going down to zero as the columnar conservation becomes more degenerate. Since the conservation is measured as a distance from a background frequency, it still suffers from the same problem of dependency on the background frequency used.

2.3.5 Conservation Measure based on Variation

In their work on evolutionarily conserved pathways of energetic connectivity in protein families, Lockless and Ranganathan (1999) viewed conservation at a given column in a multiple sequence alignment (MSA) as the overall deviance of amino acid frequencies at that column from their mean values in the MSA. They based this on the argument that evolution of a protein fold is the result of large-scale random mutagenesis, with selection constraints imposed by function, thus the lack of evolutionary constraint at one position should cause the distribution of observed amino acids at that position in the MSA to approach their mean abundance in all proteins, and deviances from the mean values should quantitatively represent conservation.

Given that the frequency of amino acid *i* in a column *k* of MSA is p_i^k, and that the mean frequency of the amino acid *i* in the whole alignment is $\overline{p}_i$, then conservation in that column could be scored as follows, under this formulation:

$$\sqrt{\sum_{i=1}^{20}\left(p_i^k - \overline{p}_i\right)^2}$$

The more similar the distribution of amino acids in a column is to the mean distribution of amino acids in the alignment, the smaller the score given by this formulation, and the score becomes zero if the two distributions are the same. The maximum conservation value is reached by an MSA column occupied by an invariant amino acid whose frequency in the alignment is minimal. According to Pei and Grishin (2001), the advantage of this method is its use of overall amino acid frequencies, which differ for different protein families. The method may thus be better suited to capture variations that could have been insignificant if a general background frequency were used.

On the negative side, conservation values computed using this method do not have an upper bound, and they are relative to a background frequency derived from the alignment. This makes it impossible to compare directly conservation values derived from different alignments using this formulation.

2.3.6 Sum of Pairs Conservation Score

The sum-of-pairs score describes conservation by calculating the sum of all possible pairwise similarities between residues in an aligned column. Karlin and Brocchieri (1996) used the sum-of-pairs scoring scheme in their study of the structure and function of *RecA* genes. Their score was given as:

$$\sum_{i=1}^{N}\sum_{j>i}^{N} M\left(s_i(x), s_j(x)\right) \times \frac{2}{N(N-1)},$$

where $s_i(x)$ is the amino acid at column *x* in the i^{th} sequence and $M\left(s_i(x), s_j(x)\right)$ is the similarity between amino acids $s_i(x)$ and $s_j(x)$.

In Pei and Grishin (2001), it was expressed in a more computationally tractable format as:

$$\sum_{a=1}^{20}\sum_{b=1}^{20} f_a(x) f_b(x) M(a,b),$$

M(a,b) is the similarity between amino acids a and b, and $f_a(x)$ is the frequency of amino acid a in column x. The similarity matrix M was normalized according to the formula

$$M(a,b) = \frac{m(a,b)}{\sqrt{m(a,a)m(b,b)}},$$

where *m(a,b)* is the similarity score given. *M(a,a)* thus equals 1 and $-1 < M(a,b) \leq 1$ for all *a*, *b*. This conservation score is bounded in the interval [-1,1].

The main criticism against conservation scoring using the sum-of-pairs approach is that the formulation does not make sense in that the conservation at an aligned column could not have come about as a result of pairwise similarities (Valdar, 2002).

In a formulation similar to the sum-of-pairs scoring, but with the assumption that all substitutions must have been from the dominant amino acid (amino acid with the highest frequency) at the position to all others, Liu *et al.* (2006) developed a conservation scoring method formulated thus:

$$\sum_{i=1}^{20} f_i(x) \log M(i, D'),$$

where $f_i(x)$ is the fractional frequency of residue *i* in column x and $M(i, D')$ is the normalized BLOSUM62 matrix substitution score for a substitution from the dominant amino acid at that column, D', to amino acid *i*. Normalization was carried to ensure that *M(a,a)* = 10, and $2 \leq M(a,b) \leq 10$ for any amino acid a and b. To mitigate against the complication of $M(i, D')$ being zero, in which case it would be a complication for the log function, Liu and Guo (2008) reformulated the conservation function as follows:

$$\sum_{i=1}^{20} f_i(x) M(i, D').$$

It is easily seen that these formulations of Liu *et al*. (2006) and Liu and Guo (2008) are basically a stripped-down version of the sum-of-pairs scoring scheme, especially

when we consider the sum-of-pairs formulation that uses the fractional frequencies of residues at the column: $\sum_{a=1}^{20}\sum_{b=1}^{20} f_a(x) f_b(x) M(a,b)$.

2.4 Object Oriented Programming

Object-oriented programming (OOP) is a programming paradigm that uses "objects" and their interactions to design applications and computer programs. Object orientation is mature and well proven, being applied in such areas as software, databases and networks. The paradigm aids in building very complex systems, easily allowing for integrated team effort. Some advantages of objected oriented programming include (Haltermann, 2008; Lafore, 1998):

Data Encapsulation: Object oriented concepts encourage data encapsulation, which means objects are tightly bound to the data on which they work. Data and processes are kept together in small, easily managed packages; data is never separated from algorithms. We thus end up with less complex and easily maintainable code, which is also less sensitive to change in requirements.

Code Reusability: In object oriented programming, the system design is based on concepts that exist in the system being modelled, and the interrelationship between the concepts. It is consequently easy to modify these in the future based on newer developments. Systems derived using object oriented concepts are thus highly scalable and easily maintainable.

Easier Understanding and Better Communication: Objects are quite much easier to understand for novice and experts alike, and even for the users too. This is because they are derived from real life concepts, which are modelled in such objects, before they are realized in the form of computer concepts and programming languages. With the advent of object orientation came design specification languages such as the Universal Modelling Language (UML), which aids in general system design and also provides a unified language for communicating design decisions. Such a language is also easy for users to understand, being highly graphical and intuitive.

2.4.1 Concepts in Object Oriented Programming

Some important concepts in object oriented programming, as described in Conway (2000) and O'Docherty (2005) are summarized below.

Objects

An object is a self-contained entity that consists of both data and procedures to manipulate the data. Object attributes are the variables in the objects, and the actual data those variables hold are referred to as attribute values. Objects also have methods, which are the public interface through which other routines can perform actions on the data within the object.

A class is a formal specification of the attributes of a particular kind of object and the methods that may be called to access those attributes. A class thus provides the template from which objects are made.

Encapsulation

Encapsulation means the inclusion of all the resources needed for an object to function within the object itself - basically, the methods and the data. Object attributes are accessed only through specific methods provided by the object, and not directly by any external routine. Encapsulation enables the internal representation of the class to be changed, without adversely affecting any other part of the whole system. This helps to maintain simplicity and locality in the system design, by providing coordinated access to and operation on data, thus ensuring easier maintainability of the program in the future.

Association and Aggregation

All objects in a particular application are connected to each other, strongly or loosely, directly or indirectly. Objects may be connected to one another in two principal ways: association and aggregation. Association is a weak form of connection, in which the objects connected together are not completely dependent on each other, i.e. one can exist without the other. In a uni-directional association, two classes are related, but only one class knows that the relationship exists. Aggregation, on the other hand, is much stronger. It means putting objects together to make more complex ones. Giving

a simplified example, a gene is made up of one or more introns and exons. A gene is thus basically an aggregate of introns and exons.

Inheritance

Inheritance is the ability to derive a new class (subclass) from an older class (super class, base class) by extending it i.e. adding extra features. Advantages of inheritance include:

- Support for richer, more powerful modelling. This benefits the development team as well as programmers reusing the code.
- It affords code reuse and also leads to shorter codes.
- It is very natural; the notion of inheritance is seen in real life as well as in abstract concepts.

2.4.2 UML Class Diagram

Class diagrams are used to describe the types of objects in a system and their relationships. They model class structure and contents using design elements such as classes, packages and objects. Class diagrams also act as very good documentation, giving a holistic view of the entire application depicted in the diagram. Conway (2000) discusses the use of UML in object oriented modeling extensively.

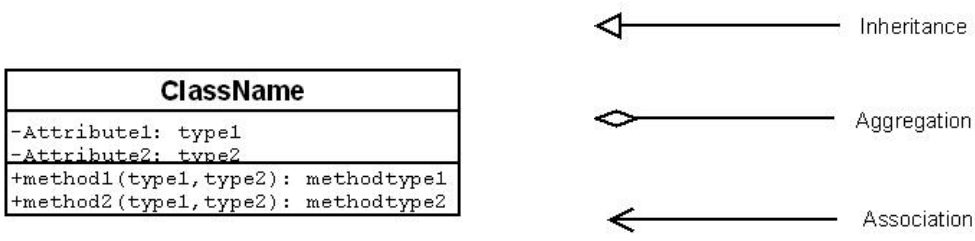

Figure 2.1 UML Notations of Some Class Diagram Elements
(a) Representation of a class, showing the class name, attributes and methods.
(b) An inheritance relationship between two classes is depicted by an arrow with a triangular head pointing from the subclass to the base class. An aggregation is depicted by a diamond-headed arrow pointed to the aggregate class. A uni-directional association is drawn as a solid line with an open arrowhead pointing to the known class.

The depiction of a class is a rectangular box with three compartments. The topmost compartment contains the class name, the middle contains the attributes specification and the last compartment contains the methods specification. A minus before an attribute or method name signifies that the attribute is private, while a plus indicate that it is public. A colon is used to separate the attribute or method from its type specification. Additionally, for a method, arguments are enclosed in brackets after the method name (only argument types indicated in the diagram).

2.5 Receiver Operating Characteristics Curves

A classification model (or classifier) is a mapping from instances to predicted classes. Conservation measurement schemes, as an example, produce continuous output, making it necessary to specify a threshold above (or below) a scored column gets classified as conserved (or not conserved). Receiver operating characteristic (ROC) graphs are used for visualizing, organizing and selecting classifiers like these based on their performances at varied threshold points.

Given a classifier and an instance, there are four possible outcomes: a positive instance classified as positive is counted as a true positive, a negative instance classified as positive is counted as a false positive, a negative instance classified as negative is counted as a true negative, and as a false negative if it were a positive instance classified as negative. Given a classifier and a set of instances, metrics that could be and which are used in making the ROC analysis include (Fawcett, 2006):

$$true\ \ positive\ rate = \frac{Positives\ correctly\ classified}{Total\ \ positives}$$

$$false\ \ positive\ rate = \frac{Negatives\ incorrectly\ classified}{Total\ \ negatives}$$

$$specificity = \frac{True\ negatives}{False\ \ positives + True\ negatives}$$
$$= 1 - false\ \ positive\ rate$$

The true positive rate is also referred to as the hit rate, sensitivity, as well as recall, while the false positive rate is also referred to as the false alarm rate. Other relevant metrics include:

$$precision = \frac{true\ \ positives}{true\ \ positives + false\ \ positives}$$

$$accuracy = \frac{true\ \ positives + true\ negatives}{positives\ +\ negatives}$$

$$F - measure = \frac{2}{1/precision + 1/recall}$$

The ROC graph is a plot of sensitivity (true positive rate) against specificity. It depicts relative tradeoffs between benefits (true positives) and costs (false positives) (Fawcett, 2006). An ROC curve begins at point (0, 0), corresponding to the strictest decision threshold whereby all columns are classified as not conserved and ends at (1, 1), corresponding to the most lenient decision threshold whereby columns are labeled as conserved. Each point on the curve corresponds to a specific pair of sensitivity and specificity and the area under each curve gives an overview of the overall performance of the classification method that generated the values used in the plot. When comparing ROC-curves of different tests, good curves lie closer to the top left corner and the worst case is a diagonal line (Lena, 2001). The diagonal line basically represents the strategy of a classifier that randomly guesses a class (Berman *et al.*, 2003).

The area under the ROC curve is equivalent to the probability that the classifier will rank a randomly chosen positive instance higher than a randomly chosen negative instance (Berman *et al*., 2003). Classifiers are thus often tested on the basis of the area under their ROC curve, with larger areas indicative of better discriminative power of the associated classifier. Often, the area under the ROC curve is too global a summary measure (Obuchowski, 2005). A good instance of this is in conservation scoring, where the region of interest is in the area with low specificity and high sensitivity values. In such cases, the partial area under the ROC curve, choosing an appropriate cutoff specificity, or sensitivity, is an appropriate discrimination measure for the classifier (Obuchowski, 2005). Some recent studies in conservation scoring have used this approach (Capra and Singh, 2007; Fischer *et al*., 2008).

ROC curves are able to provide a richer measure of classification performance than scalar measures such as accuracy, error rate or error cost. In some cases, the conclusion of which classifier has the superior performance can change with a shifted

distribution (Berman *et al*., 2003). Fortunately, ROC curves remain unperturbed by changes in class skews and error costs, in contrast to other appealing substitutes, such as precision-recall curves.

3. Materials and Methods

3.1 Data Set

The data set used in this study is a subset of that used by Capra and Singh (2007) which they derived from the Catalytic Site Atlas (http://www.ebi.ac.uk/thornton-srv/databases/CSA/) (Porter *et al.*, 2004). The Catalytic Site Atlas (CSA) provides catalytic residue annotation for enzymes in the Protein Data Bank. It consists of a set of non-homologous enzymes of known structure with well-defined active site and plausible catalytic mechanism. Enzymes are annotated if the X-ray crystal structure or NMR model exists, and if sufficient information concerning active site, overall reaction catalysed and catalytic mechanism can be obtained from primary literature. Residues fulfilling the following criteria were defined as catalytic (Bartlett *et al*, 2002; Porter *et al*., 2004):

- direct involvement in the catalytic mechanism, e.g. as a nucleophile;
- alteration of the pKa of a residue or water molecule directly involved in the catalytic mechanism;
- stabilization of a transition state or intermediate, thereby lowering the activation energy for a reaction;
- activation of the substrate in some way, e.g. by polarizing a bond to be broken.

PSI-Blast searches were then made, using each element of this non-homologous set and a cut off < 0.0005, against a composite database of non-redundant database from NCBI and protein sequences from structures found in the PDB. An alignment of the sequences found then facilitated the identification of homologues, which were further scrutinized to ensure the conservation of residues that are supposed to be catalytic.

Capra and Singh (2007) created the data for their work using known catalytic residues obtained from the Catalytic Site Atlas. For each literature based entry in the CSA as of June 8, 2006, they obtained the 3D structure of the protein chain from the Protein Data Bank (PDB) (http://www.pdb.org) (Berman *et al*., 2003). The structures' sequences were then clustered at 95% sequence identity and redundant structures were removed. Sequence alignments for each remaining structure were then obtained from HSSP (http://swift.cmbi.kun.nl/gv/hssp/) (Dodge *et al*, 1998). These alignments were filtered to improve alignment quality by removing sequences with more than

95% sequence similarity to the original CSA sequence or whose length was more than two standard deviations away from it. Any alignment with fewer than five sequences was removed.

The alignment data for this study is a subset of the one by Capra and Singh (2007), but with the additional constraint that the least number of sequences in each alignment must be 20. This is in line with the findings of Pei and Grishin (2001), who concluded that not less than approximately 20 representative sequences are required in estimating the conservation patterns in a protein family. Additionally, during processing, conservation scores were not calculated for positions having gaps in 50% or more positions of the sequences in the alignment.

The Relative Entropy and Jensen-Shannon divergence methods calculated conservation as a distance between the frequency in an MSA column and a given background frequency. In this study, two background frequencies were utilized: overall amino acid distribution from the particular alignment under consideration and also the amino acid distribution from all the sequences in SwissProt (http://www.uniprot.org/) (The Universal Protein Resource (UniProt), 2008).

To calculate the background frequency based on SwissProt, the entire sequences in SwissProt database (The Universal Protein Resource (UniProt), 2008) were downloaded on 2nd February, 2009. This consisted of 408,099 sequences and 147,085,246 amino acid residues. The frequency for each amino acid was then computed using a Perl script that counted the different amino acids in each of the sequences and divided the counts by the total number of amino acids in the sequences (see Appendix 7.2). Selenocysteine (U) and Pyrrolysine (O) are quite rare in nature and were discovered quite recently, compared to other amino acids (Atkins and Gesteland, 2002; Krzycki, 2005; Cone *et al.*, 1976; Zinoni *et al.*, 1986). Certain protein sequencing techniques do not distinguish among certain residue pairs. In such cases, B is used to represent a residue that could either be asparagine or aspartic acid, and Z for one that could either be glutamine or glutamic acid. These residue representations (U, O, B and Z) have very low frequency among available protein sequences (in the order of 10^{-6}, as estimated from all protein residues in Uniprot) due to their rare occurrence in nature. They were consequently excluded from further

analysis. In the event of the occurrence of any of these residues in an alignment column, it will be replaced by the gap character and the user will be notified.

3.2 Implementation Details

The application which I have written for sequence conservation metrics is composed of eight (8) modules viz:

conservation::Conservation
conservation::Statistics
conservation::AlnWrapper
conservation::Weights
conservation::SubstitutionMatrix
conservation::Iterators::AlnIterator
conservation::Iterators::SeqIterator
conservation::Iterators::ColumnIterator

Apart from these, there are 3 additional files, blosum62.txt, blosum_liu.txt and backgroundFrequencies.dat. blosum62.txt is used to store the Blosum62 substitution matrix, blosum_liu.txt is used to store the version of the Blosum matrix used by Liu *et al*. (2006) and Liu and Guo (2008) while the backgroundFrequencies.dat is used to store the background frequencies calculated from SwissProt.

Fig. 3.1 shows the class diagram of the modules that constitute the conservation indices system. The conservation::Iterators::SeqIterator and conservation::Iterators::ColumnIterator modules provide iteration facilities over the sequences and columns in the alignment respectively. These both inherit from the conservation::Iterators::AlnIterator module which iterates sequentially over a set of integers, given the minimum and maximum points. The conservation::Weights module handles the calculation of the position-based sequence weights (Henikoff and Henikoff, 1994) as well as the calculation of the independent counts (Pei and Grishin, 2001) for each amino acid in each position. It also inherits from the conservation::Iterators::AlnIterator module. Any other weight calculation scheme to be implemented later would be added to the conservation::Weights module.

The conservation::AlnWrapper module, on the other hand, takes as input the alignment object or an alignment file, retrieving the alignment using bioperl. It

provides the ability to iterate over the sequences and columns of an alignment by returning conservation::Iterators::SeqIterators and conservation::Iterators::ColumnIterators as needed respectively. It thus aggregates both classes and provides navigation capabilities over the alignment.

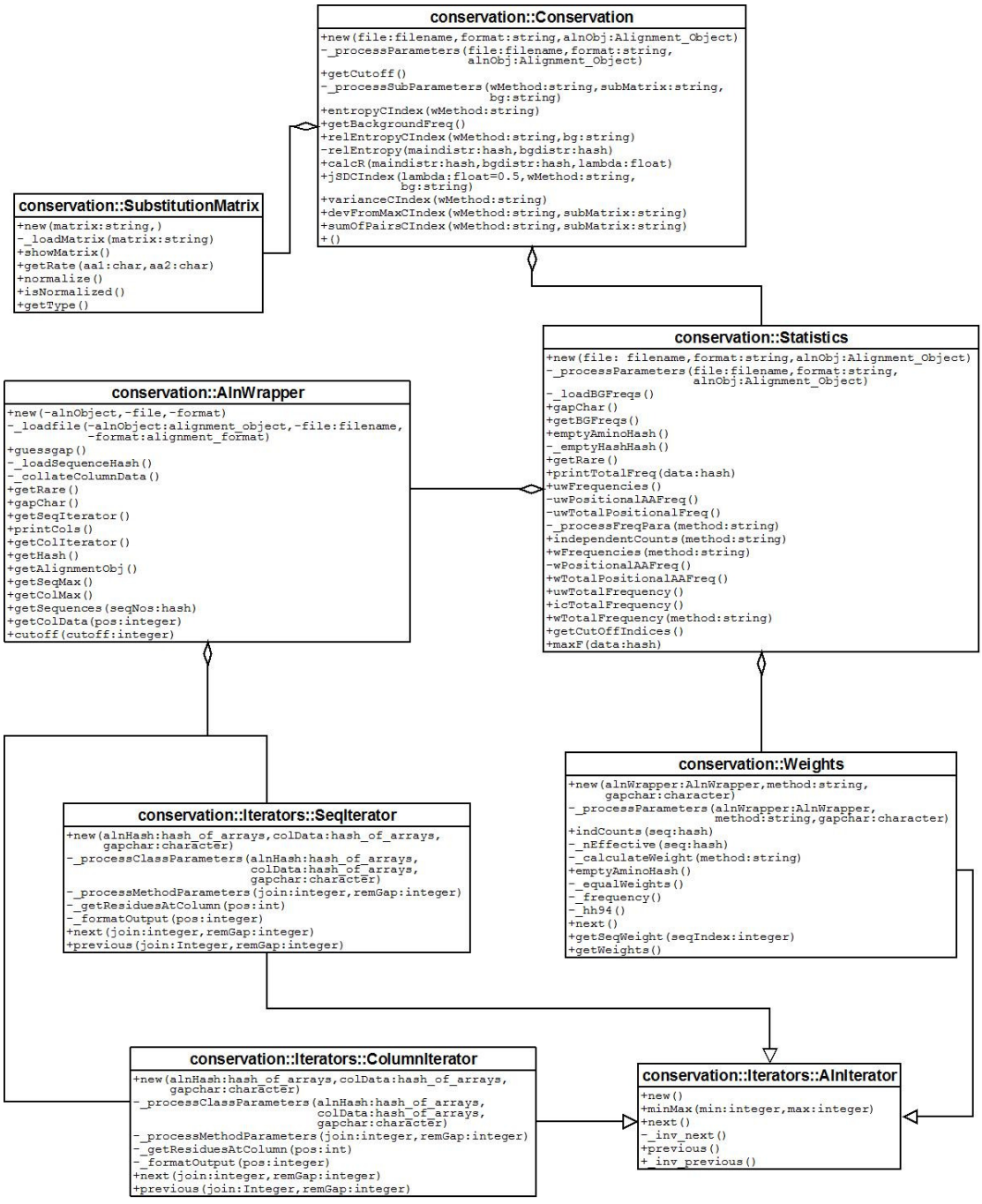

Figure 3.1: The Class Diagram for the Application Modules

The conservation::Statistics module aggregates both the conservation::AlnWrapper and conservation::Weights class, using the former to retrieve sections of the alignment and the latter to calculate appropriate weights. It then uses this information to calculate various statistics needed for calculating various conservation indices. In calculating some statistics, it needs and thus loads the background frequencies from the backgroundFrequencies.dat file. The conservation::SubstitutionMatrix module loads from disc the two substitution matrices, blosum62 and blosum_liu (Liu *et al.*, 2006; Liu and Guo, 2008). It also provides some scaling operation for the blosum62 matrix, scaling the leading diagonals to 1 (Pei and Grishin, 2001).

The conservation::Conservation module aggregates the conservation::Statistics and the conservation::SubstitutionMatrix class, deriving appropriate statistics from the former and substitution rates from the latter, and using these to compute the different conservation indices.

3.3 Module Testing

Testing the modules was done iteratively during the development process. Specifically, as each module was being developed, test scripts were also written to utilise the methods and ensure that the returned results conform to the expected. Additionally, since the modules were connected together in a hierarchical structure, the correct output in a module higher up in the hierarchy implies implicitly the correct functioning of those ones lower down.

For those modules that needed an alignment file to work, the model alignment used in Valdar (2002) was used (Fig. 3.2). This alignment is quite small, consisting of 10 sequences and 11 columns, but it is constructed in such a way as to highlight some of the most prominent conditions that arise in protein sequence alignments. The choice of the alignment was based upon the preceding fact and also because it affords the manual computation of the various conservation values and their associated statistics, due to the small alignment size. It was thus possible to compare values generated from the modules with those manually calculated (Appendices, section 7.1).

The system has additionally been used on over 13,000 protein sequence alignments retrieved from the Conserved Domain Database (Marchler-Bauer *et al.*, 2009), thus

testing its ability to work on a variety of alignments. Additionally, it was tested on 502 alignments retrieved from the work of Capra and Singh (2007).

```
seq1            DDDDDDIPDLL 11
seq2            DDDDDDIPVLL 11
seq3            DDDDDDIPYLL 11
seq4            DDDDDDIPALL 11
seq5            DDDDDDLWT-- 9
seq6            DDEDEELWK-- 9
seq7            DDEDEELWP-- 9
seq8            DDEDEELWC-- 9
seq9            DDEDEFVSR-- 9
seq10           DEEFFFVSH-- 9
                *::    :
```

Figure 3.2: Model Alignment (Valdar, 2002)

3.4 Usage

This conservation::Conservation module is the only one to be interacted with by the user desiring to calculate conservation indices. It is therefore desirable to provide more explanation about the mode of usage of this module.

The following Perl script provides commented code examples about calculating different conservation measures for an alignment.

```
#take the module into use
use conservation::Conservation;

#if the alignment needs to be opened for preprocessing, take Bio::AlignIO into use
use Bio::AlignIO;

#open alignment file using Bio::AlignIO. #This example assumes there is an
#alignment  file  named test.aln in current directory.
my $in = new Bio::AlignIO(-file => "test.aln", -format =>
"ClustalW");

#get the alignment object
my $aln = $in->next_aln;

#preprocess the alignment object

#after preprocessing, create a new conservation::Conservation object using the
#alignment object
my $cons = conservation::Conservation->new(-alnObj =>
$aln);
```

```
#on the other hand, if no preprocessing is needed, the object could be created directly
#from the alignment file
my $cons = conservation::Conservation->new(-file =>
"test.aln", -format => "ClustalW");


#calculate unweighted entropy conservation for each column of the alignment
my $uwEn = $cons->entropyCIndex;

#calculate entropy conservation using Position-based sequence weights (Henikoff and
#Henikoff, 1994)
my $wEn_hh94 =  $cons->entropyCIndex(-wMethod => 'hh94');

#calculate entropy conservation using independent count scheme (Pei and Grishin,
#2001)
my $icEn = $cons->entropyCIndex(-wMethod => 'indcount');

#calculate relative entropy conservation using SwissProt background
my $uREn_swiss =  $cons->relEntropyCIndex(-bg =>
"swiss");

#calculate relative entropy conservation using using SwissProt background and
#Position-based sequence weights
my $wREn_swiss_hh94 = $cons->relEntropyCIndex(-bg =>
"swiss", -wMethod => 'hh94');

#calculate relative entropy conservation using background statistics derived from the
#alignment and position-based sequence weights
my $wREn_local_hh94 = $cons->relEntropyCIndex(-bg =>
"local", -wMethod => 'hh94');

#calculate JSD-based conservation using background statistics derived from the
#alignment and independent count scheme
my $icJSD_swiss = $cons->jSDCIndex(-bg => "local", -
wMethod => 'indcount');

#calculate conservation based on the variance scheme (Pei and Grishin, 2001) using
#Position-based sequence weights
my $wVar_hh94 = $cons->varianceCIndex(-wMethod =>
'hh94');

#calculate sum-of-pairs conservation using normalized blosum62 (Pei and Grishin,
#2001) and Position-based sequence weights
my $wSP = $cons->sumOfPairsCIndex(-subMatrix =>
"blosum62", -normalize => 1, -wMethod => 'hh94');

#calculate conservation based on divergence from amino acid with highest frequency,
#using the independent count scheme this calculates both the log and non-log forms
$results = $cons->devFromMaxCIndex(-subMatrix =>
"blosum_liu", -wMethod => 'indcount');
```

```
#retrieve the non-log form
my $icDN = $results->{'nonLogForm'};

#retrieve the log form
my $icDL = $results->{'logForm'};

#each of the results are in references to hashes, so to access the conservation value for
#each column then the contents of the hash needs to be retrieved get the alignment
#length
my $aln_len = $aln->length -1;

#for each column
for my $col (0..$aln_len){
  #do something with each column conservation

  #for example, print each of the conservation values of each column on a separate
  #line.
  #in this case the values are those of conservation measured using
  #relative entropy with SwissProt background and weighted using position-based
  #sequence weights
   print $wREn_swiss_hh94->{$col},"\n";

   if($uwEn->{$col} == -100){
  #if the conservation value is -100, it means that the
  #percentage gap in the column is more than the cutoff
  #Something can then be done here to handle this condition
   }
}
```

The documentation of the available public interfaces from all the modules is provided in the appendices (section 7.2).

3.5 Analyses

Conservation values were calculated for the 502 alignments using seven methods viz: Entropy method, Variance method, Sum of Pairs method, all from Pei and Grishin (2001) , and Relative Entropy method (Wang and Samudrala, 2006), Jensen-Shannon Divergence method (Capra and Singh, 2007), and the deviation from maximum frequency method (Liu and Guo, 2008) and its log form (Liu *et al*., 2006). Variants of the methods which utilized no weights, position-based sequence weights (Henikoff and Henikoff, 1994) as well as the independent count scheme of (Pei and Grishin, 2001) were employed in the calculation of the conservation scores. Additionally, gaps

were factored in by multiplying the conservation score for each method and column by the percentage of residues (1 – fraction of gaps) in that column.

For the case of the Entropy method, the conservation index was first subtracted from 1, then multiplied by the percentage of residues, and the resulting score subtracted from 1 again, to obtain the gap-weighted entropy score. This was carried out because entropy conservation scores reduce with increasing conservation, while the other measures increase with increasing conservation. In the case of weighted scheme, the percentage of gaps in the columns was calculated using the sequence weights, while for the unweighted scheme as well as the independent counts scheme, there was no weighting employed in the calculation of the percentage gaps. All positions with more than 50% gaps were eliminated from further analysis.

Both the relative entropy scheme as well as the Jensen-Shannon divergence scheme measure conservation relative to a background frequency. Conservation scores were thus calculated with both methods, using two different background frequencies: amino acid frequency from SwissProt as well as overall frequency from the alignment itself. In all, conservation scores were calculated using 27 methods, which are variants of the seven methods with the weighting scheme as well as the background frequencies factored in.

3.5.1 Correlation Analysis

Conservation indices for each of the 502 alignments were normalized to zero mean and unity variance. Pearson product-moment correlation was then calculated for each possible method pair. The convention for naming the methods is as follows. The seven methods are named Entropy, REntropy (relative entropy), JSD (Jensen-Shannon divergence), Variance, SOP (sum of pairs), Diff (divergence from amino acid with maximum frequency) and LogDf (log of divergence from amino acid with maximum frequency). Each method is preceded by H (for sequences weighted using the position-based sequence weight scheme), ic (for independent count scheme) or nothing at all, for unweighted conservation scoring. In the case of schemes using background frequencies, such schemes are suffixed with either S or L, for schemes utilizing SwissProt or background frequency derived from the alignment itself, respectively.

3.5.2 ROC Analysis

For the sake of the ROC analysis, true positives and true negatives were defined for each column of each alignment, in the same order in which the normalized conservation values of the alignments exist on file. Columns classified to be catalytic are designated as the true positives, and these are given the label 1. All other columns are classified as true negatives, and are labeled -1. The annotations data are part of the data used by Capra and Singh (2007) and are directly compiled from their data. In their work, they also carried out ROC analysis on their data, and they defined their true positives and true negatives exactly as done here. Using the ROCR software (R Development Core Team, 2008; Sing *et al.*, 2005), ROC analysis was carried out and the receiver operating characteristics curves plotted. Additionally, the area under the curve (AUC) was also calculated for each method, at false positive rate cutoff values of 0.1, 0.2 and 1.0.

4. Results

4.1 Descriptive Statistics

The number of protein multiple sequence alignments used in the analysis is 502, with an average of 99 sequences per alignment and 1530 annotated catalytic residues. The number of sequences in most of the alignments is between 20 and 220, with most alignments having between 1 and 4 catalytic residues (Fig. 4.1). The total number of columns in the entire 502 alignments which had positions with gaps occupying less than 50% of positions in the column was 166841. Fig. 4.2 provides a comparison of the distribution of conservation among the catalytic and non catalytic residues, using Entropy with independent count scheme. The entropy scores are sorted according to conservation level, with 0 being the most conserved while 1 is the least conserved. 709, 316 and 158 catalytic residues have conservation scores in the intervals [0,0.1], (0.1,0.2] and (0.2,0.3] respectively, while the number of non catalytic residues in the same conservation intervals are 7781, 6921 and 8590 respectively.

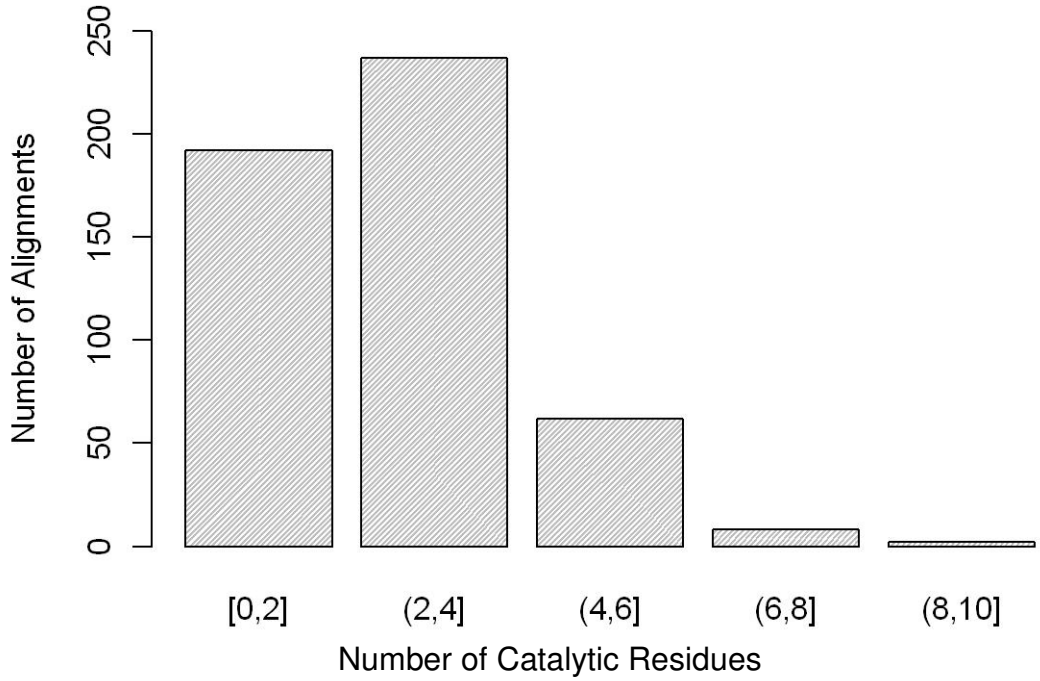

Figure 4.1: Distribution of Catalytic Residues in the Alignments

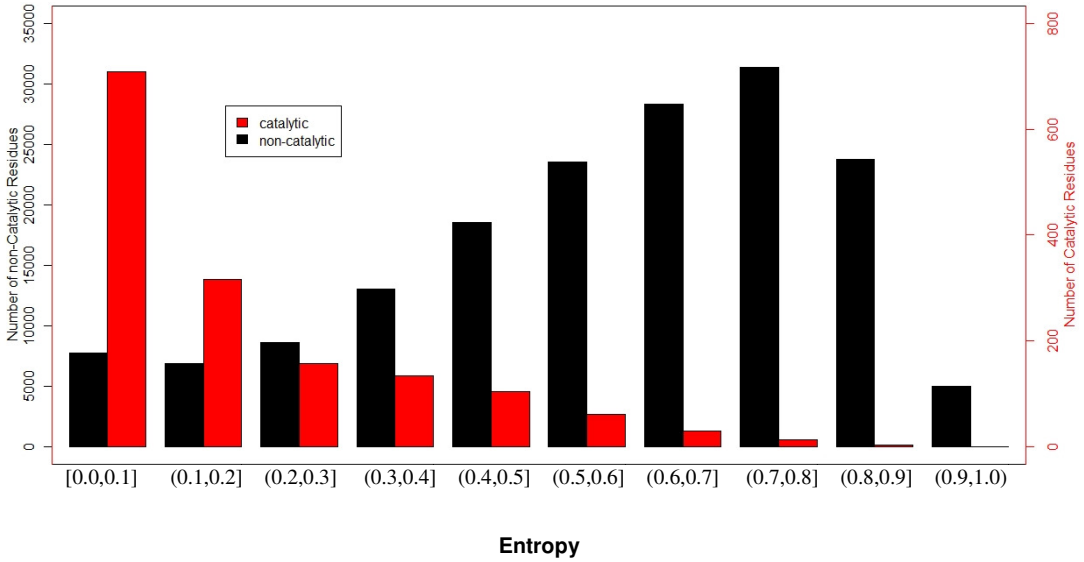

Figure 4.2: Distribution of Conservation in Catalytic and non-Catalytic Columns

4.2 Correlation Results

The results of the correlation analysis are given in Table 4.1. A previous study (Pei and Grishin, 2001) calculated the correlation matrix using 12 methods, derived from three main methods (Entropy, Sum of Pairs and Variance methods) and associated weighting schemes. They derived a minimum correlation of 0.85 between sum-of-pairs method using independent count and variance method using unweighted frequencies. The cells in Table 4.1 that have correlation values lower than 0.85 have been shaded. In the present study, and with specific focus on only the methods implemented by Pei and Grishin (2001), the lowest correlation of 0.81 is obtained between the same set of two methods, sum-of-pairs scoring using independent count-based frequencies and variance scoring using unweighted scoring. With the additional exception of the correlation between sum-of-pairs method using independent count scheme and variance method using position-based sequence weight, which was 0.83, all other methods shared between the two studies have correlation values greater than 0.85, as noted in the previous study.

Taking a more global look at the data, most of the correlation values of LogDf and Diff against other methods are below 0.85. There are also some low correlation values (below 0.85) between JSD method variants and some variants of entropy, relative entropy, Variance and Sum of Pairs methods. It should be noted, though, that most of

these low correlation values, except the ones including LogDf, Diff and variants, occur between variants which implement different weighting schemes. They may thus largely be attributed to the sensitivity of the method to the weighting schemes used.

The highest correlation of 0.99 is recorded between icJSD_Local and icREntropyL. Next on the highly correlated list comes Variance versus Entropy, icVariance versus icEntropy, and icJSD_Swiss versus icREntropyS, all with a correlation of 0.98. icJSD_Swiss versus icREntropyL, icVariance versus icJSD_Local, icVariance versus icJSD_Swiss, HVariance versus Entropy as well as HEntropy, and icJSD_Local versus icREntropyS all have a correlation value of 0.97, while icJSD_Swiss versus HREntropyS, icREntropyL versus icEntropyS, and icVariance versus icREntropyS and icREntropyL have correlation values of 0.96. icJSD_Swiss versus icEntropy as well as REntropyS, icREntropy versus icEntropy, icJSD_Local versus HREntropyL, SOP versus Entropy and HEntropy, as well as HSOP versus HEntropy all have correlation coefficient of 0.95. Correlation coefficient of 0.94 is discovered between icJSD_Local and HREntropyS, Variance and REntropyS, REntropyL as well as SOP, HVariance and HREntropyL, and between icSOP and icEntropy.

Table 4.1: Correlation between Conservative Indices Calculated by Different Methods

	Entropy	HEntropy	icEntropyS	REntropyS	HREntropyS	icREntropy	REntropyL	HREntropyL	icREntropyL	JSD_Swiss	HJSD_Swiss	icJSD_Swiss	JSD_Local	HJSD_Local	icJSD_Local	Variance	HVariance	icVariance	SOP	HSOP	icSOP	Diff	HDiff	icDiff	LogDf	HLogDf	icLogDf
Entropy	1.00	0.99	0.94	0.93	0.93	0.90	0.92	0.92	0.90	0.87	0.86	0.91	0.84	0.84	0.91	0.98	0.97	0.94	0.95	0.94	0.87	0.91	0.89	0.88	0.86	0.84	0.80
HEntropy		1.00	0.95	0.93	0.94	0.90	0.91	0.93	0.91	0.85	0.87	0.92	0.83	0.85	0.91	0.96	0.97	0.95	0.95	0.95	0.88	0.89	0.90	0.88	0.85	0.85	0.81
icEntropy			1.00	0.89	0.90	0.95	0.88	0.89	0.96	0.73	0.74	0.95	0.70	0.72	0.95	0.88	0.89	0.98	0.90	0.90	0.94	0.81	0.81	0.88	0.78	0.78	0.83
REntropyS				1.00	0.99	0.95	0.98	0.97	0.93	0.91	0.90	0.95	0.88	0.88	0.93	0.94	0.93	0.91	0.88	0.87	0.81	0.81	0.80	0.78	0.76	0.74	0.70
HREntropyS					1.00	0.96	0.97	0.98	0.94	0.90	0.91	0.96	0.87	0.89	0.94	0.93	0.94	0.92	0.88	0.88	0.82	0.80	0.81	0.79	0.76	0.75	0.71
icREntropyS						1.00	0.93	0.94	0.99	0.77	0.79	0.98	0.75	0.76	0.97	0.86	0.87	0.96	0.85	0.85	0.89	0.75	0.75	0.81	0.72	0.72	0.76
REntropyL							1.00	0.99	0.94	0.89	0.88	0.93	0.91	0.90	0.94	0.94	0.93	0.91	0.87	0.86	0.80	0.81	0.79	0.78	0.76	0.74	0.70
HREntropyL								1.00	0.95	0.88	0.90	0.94	0.89	0.91	0.95	0.93	0.94	0.92	0.87	0.87	0.81	0.80	0.80	0.78	0.76	0.75	0.71
icREntropyL									1.00	0.76	0.78	0.97	0.76	0.77	0.99	0.87	0.88	0.96	0.85	0.85	0.89	0.75	0.75	0.81	0.73	0.72	0.76
JSD_Swiss										1.00	0.98	0.83	0.98	0.96	0.82	0.92	0.91	0.80	0.83	0.81	0.67	0.81	0.79	0.70	0.73	0.70	0.59
HJSD_Swiss											1.00	0.85	0.95	0.98	0.83	0.91	0.92	0.82	0.83	0.83	0.69	0.79	0.80	0.71	0.72	0.72	0.61
icJSD_Swiss												1.00	0.80	0.82	0.99	0.89	0.90	0.97	0.87	0.87	0.89	0.77	0.77	0.82	0.74	0.73	0.76
JSD_Local													1.00	0.98	0.82	0.91	0.89	0.79	0.81	0.79	0.65	0.79	0.78	0.69	0.71	0.69	0.57
HJSD_Local														1.00	0.83	0.90	0.91	0.80	0.81	0.81	0.67	0.77	0.78	0.69	0.71	0.70	0.59
icJSD_Local															1.00	0.89	0.90	0.97	0.87	0.87	0.89	0.77	0.77	0.82	0.74	0.73	0.76
Variance																1.00	0.99	0.92	0.94	0.91	0.81	0.90	0.88	0.83	0.84	0.82	0.74
HVariance																	1.00	0.93	0.93	0.93	0.83	0.88	0.89	0.84	0.83	0.83	0.76
icVariance																		1.00	0.91	0.91	0.93	0.81	0.82	0.87	0.78	0.78	0.81
SOP																			1.00	0.99	0.91	0.91	0.90	0.90	0.92	0.91	0.88
HSOP																				1.00	0.93	0.88	0.89	0.89	0.91	0.92	0.89
icSOP																					1.00	0.77	0.79	0.89	0.81	0.82	0.90
Diff																						1.00	0.98	0.94	0.90	0.87	0.81
HDiff																							1.00	0.95	0.88	0.88	0.83
icDiff																								1.00	0.85	0.86	0.91
LogDf																									1.00	0.97	0.91
HLogDf																										1.00	0.93
icLogDf																											1.00

4.3 ROC Analysis Results

The ROC curves are given in Figs. 4.3 to 4.5. Interpreting the ROC curves cannot be done correctly without taking into consideration the sizes of the true positives and true negatives. In this dataset, the number of true positives was 1530 while that of true negatives was 165311. At false positive rate of 0.05, and while paying particular attention to the ROC curve drawn using position-based sequence weight methods, it could be

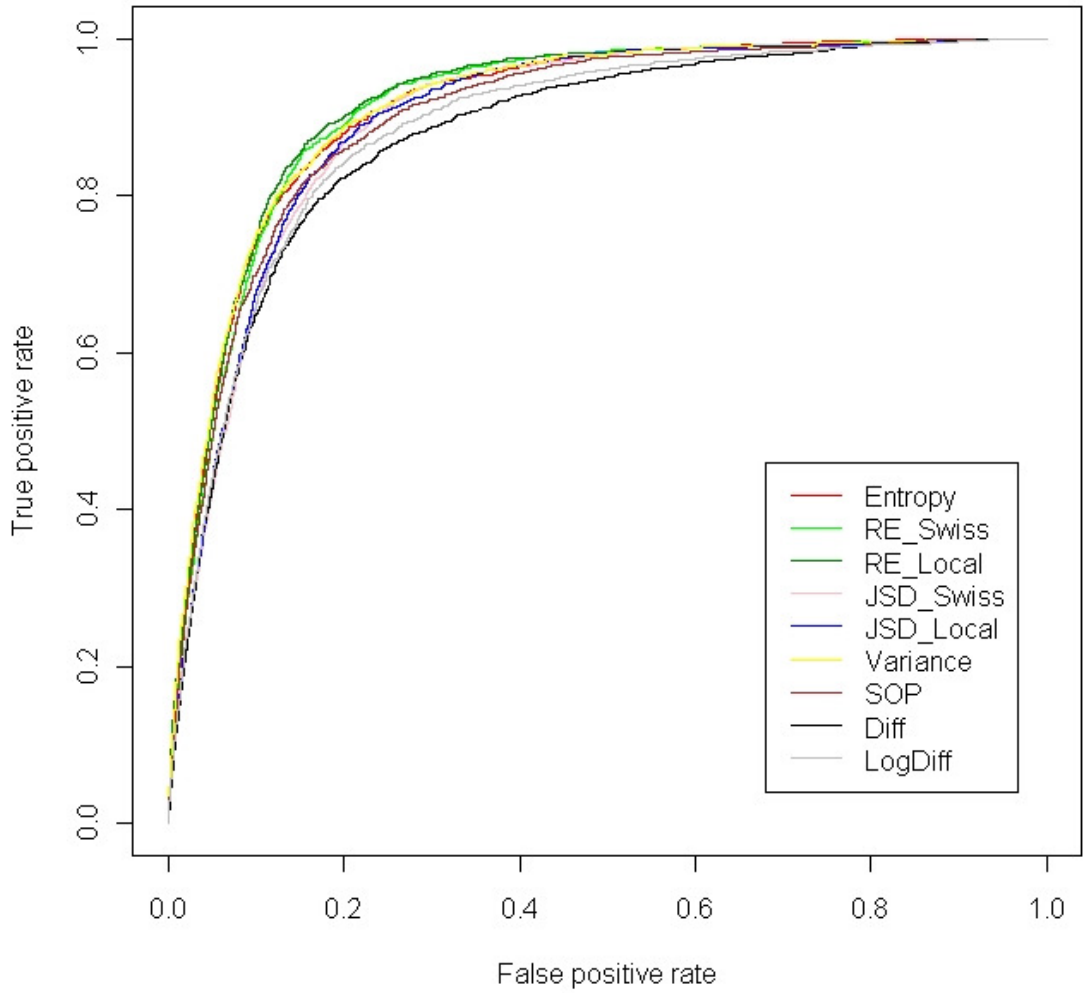

Figure 4.3: ROC Curves Showing the Performance of the Different Methods Using No Sequence Weights.

deduced by visual inspection that the approximate true positive rate for the best performing method at that point is 0.54. This could be interpreted to mean that, at this cutoff threshold, the best performing method has correctly classified 54% (826 residues) of all true positives as positives while at the same time it has incorrectly classified 5%

(8265) of all true negatives as being positives. Going up to false positive rate of 0.1, 76% (1170) of all true positives have been correctly classified, while 10% (16531) of true negatives has been incorrectly classified as positives. This seemingly poor performance is the same for the unweighted and independent count schemes too.

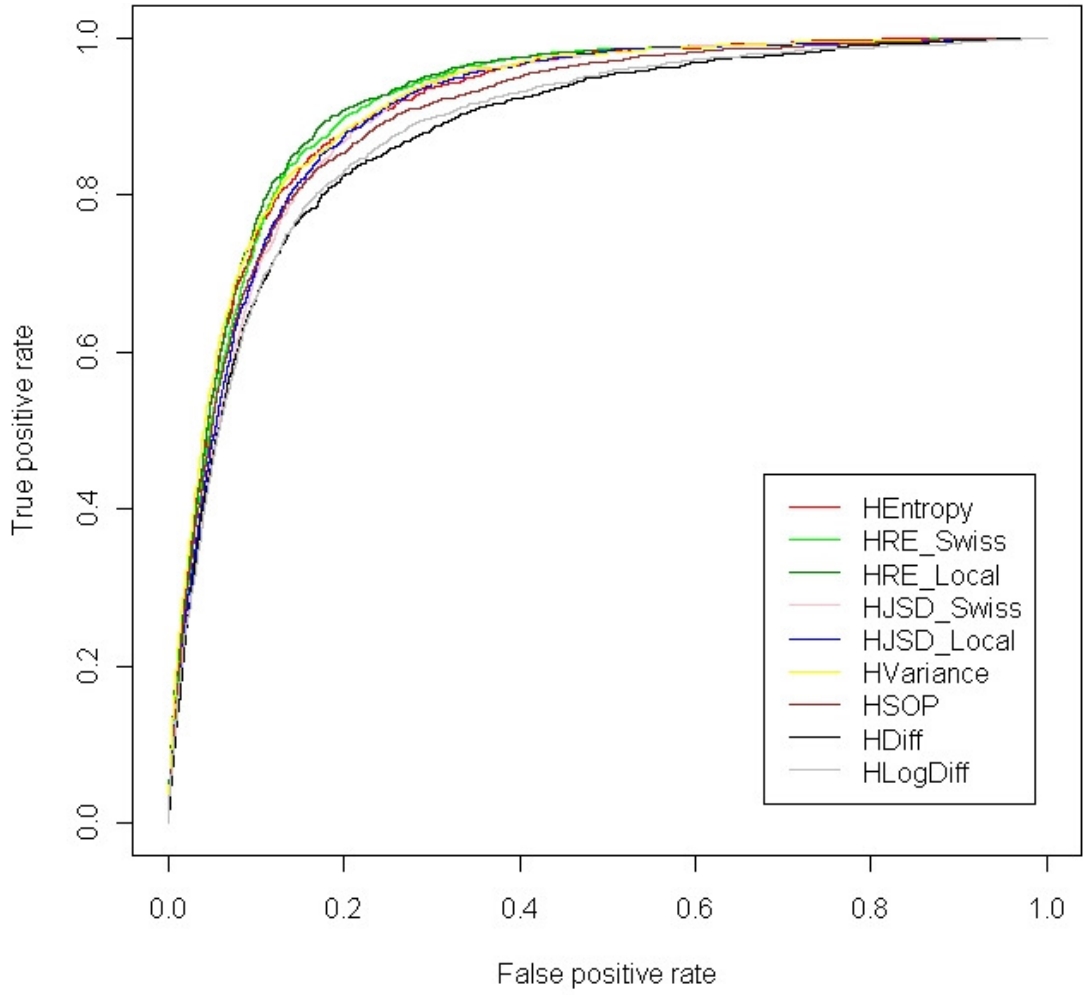

Figure 4.4: ROC Curves Showing the Performance of the Different Methods Using Position-Based Sequence Weights.

AUC values at 0.1, 0.2 and 1 false positive rate cutoff values, arranged in decreasing order, are given in tables 4.2, 4.3 and 4.4. Additionally, tables 4.5, 4.6 and 4.7 show the comparison of performance across different weighting schemes, at AUC values of 0.1, 0.2 and 1. From tables 4.2, 4.3 and 4.4, a cursory glance reveals that the three best-performing methods, at all levels of false positive rate cutoff and considering only those methods utilizing the independent count schemes, are the JSD method and relative entropy method, both using alignment-based background frequency, as well as the

variance method, with the performance decreasing in the order in which the methods are listed.

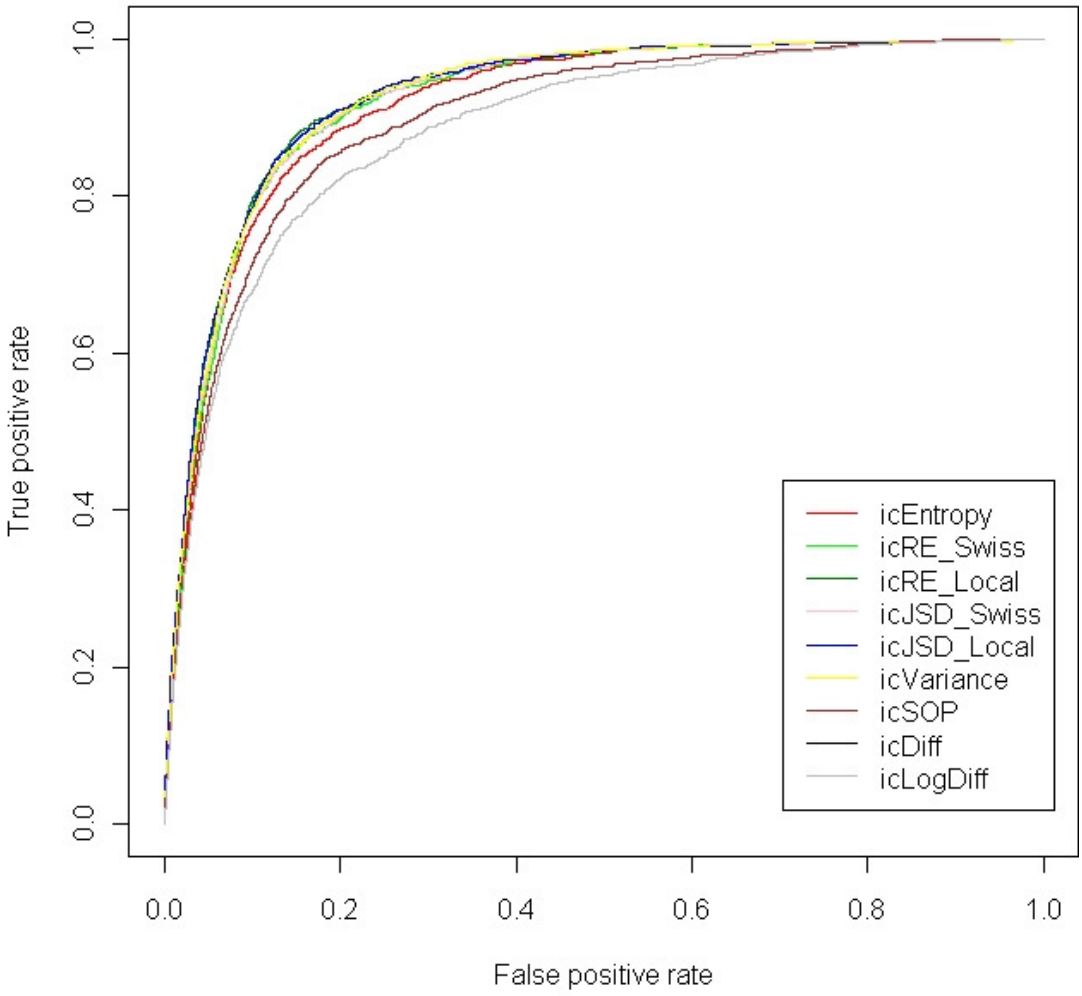

Figure 4.5: ROC Curve Showing the Performance of the Different Methods Using Independent Counts.

The trend is not so strong when considering the unweighted as well as the position-based sequence weight schemes. The best performing methods change as the false positive rates increase. At the false positive rate of 0.1, the best performing methods are Variance, Relative Entropy (using alignment-based background frequency) and Entropy, but moving down to false positive rate value of 0.2 and to 1, Entropy is outperformed by relative entropy (using SwissProt-based background frequency). It is also noteworthy that at all levels of false positive rate cutoffs, icJSD_Local, icRE_Local, icVar and icRE_Swiss have higher AUC than other methods using the position-based sequence weights as well as unweighted conservation measures.

Table 4.2: Area Under the Curve (AUC) at 0.1 False Positive Rate Cutoff Value

Unweighted Methods		Position-Based Sequence weights Method		Independent Count Methods	
Method	**$AUC_{0.1}$**	**Method**	**$AUC_{0.1}$**	**Method**	**$AUC_{0.1}$**
Var	0.049	HVar	0.051	icJSD_Local	0.055
RE_Local	0.048	HRE_Local	0.050	icRE_Local	0.054
Entropy	0.047	HEntropy	0.049	icVar	0.053
RE_Swiss	0.046	HRE_Swiss	0.048	icJSD_Swiss	0.053
SOP	0.045	HSOP	0.046	icRE_Swiss	0.052
JSD_Local	0.041	HJSD_Local	0.045	icEntropy	0.051
LogDf	0.041	HJSD_Swiss	0.044	icSOP	0.048
JSD_Swiss	0.040	HDiff	0.042	icDiff	0.046
Diff	0.040	HLogDf	0.042	icLogDf	0.044

Table 4.3: Area Under the Curve (AUC) at 0.2 False Positive Rate Cutoff Value

Unweighted Methods		Position-Based Sequence weights Method		Independent Count Methods	
Method	**$AUC_{0.2}$**	**Method**	**$AUC_{0.2}$**	**Method**	**$AUC_{0.2}$**
RE_Local	0.132	HRE_Local	0.135	icJSD_Local	0.141
Var	0.131	HVar	0.134	icRE_Local	0.141
Entropy	0.130	HEntropy	0.131	icVar	0.139
RE_Swiss	0.129	HRE_Swiss	0.131	icJSD_Swiss	0.138
SOP	0.125	HSOP	0.126	icRE_Swiss	0.138
JSD_Local	0.120	HJSD_Local	0.126	icEntropy	0.134
JSD_Swiss	0.118	HJSD_Swiss	0.123	icSOP	0.128
LogDf	0.117	HLogDf	0.118	icDiff	0.122
Diff	0.115	HDiff	0.118	icLogDf	0.120

Table 4.4: Total Area Under the Curve (AUC)

Unweighted Methods		Position-Based Sequence weights Method		Independent Count Methods	
Method	**AUC_1**	**Method**	**AUC_1**	**Method**	**AUC_1**
RE_Local	0.916	HRE_Local	0.919	icJSD_Local	0.925
Var	0.913	HRE_Swiss	0.915	icRE_Local	0.925
RE_Swiss	0.913	HVar	0.915	icVar	0.924
Entropy	0.911	HEntropy	0.912	icJSD_Swiss	0.922
SOP	0.900	HJSD_Local	0.906	icRE_Swiss	0.921
JSD_Local	0.899	HJSD_Swiss	0.904	icEntropy	0.916
JSD_Swiss	0.898	HSOP	0.900	icSOP	0.898
LogDf	0.885	HLogDf	0.882	icDiff	0.884
Diff	0.877	HDiff	0.879	icLogDf	0.878

Table 4.5: Comparison of Performance across Various Weighting Schemes at False Positive Rate Cutoff Value of 0.1.

Relative Entropy (Local Alignment Background)		Relative Entropy (SwissProt Background)		JSD (Local Alignment Background)		JSD (SwissProt Background)	
icRE_Local	0.054	icRE_Swiss	0.052	icJSD_Local	0.055	icJSD_Swiss	0.053
HRE_Local	0.050	HRE_Swiss	0.048	HJSD_Local	0.045	HJSD_Swiss	0.044
RE_Local	0.048	RE_Swiss	0.046	JSD_Local	0.041	JSD_Swiss	0.040
Entropy		**Variance**		**Sum of Pairs**		**Dev. From Max. Frequency**	
icEntropy	0.051	icVar	0.053	icSOP	0.048	icDiff	0.046
HEntropy	0.049	HVar	0.051	HSOP	0.046	HDiff	0.042
Entropy	0.047	Var	0.049	SOP	0.045	HLogDf	0.042
Log form of Dev. From Max. Freq.							
icLogDf	0.044						
Diff	0.040						
LogDf	0.041						

Table 4.6: Comparison of Performance across Various Weighting Schemes at False Positive Rate Cutoff Value of 0.2

Relative Entropy (Local Alignment Background)		Relative Entropy (SwissProt Background)		JSD (Local Alignment Background)		JSD (SwissProt Background)	
icRE_Local	0.141	icRE_Swiss	0.138	icJSD_Local	0.141	icJSD_Swiss	0.138
HRE_Local	0.135	HRE_Swiss	0.131	HJSD_Local	0.126	HJSD_Swiss	0.123
RE_Local	0.132	RE_Swiss	0.129	JSD_Local	0.120	JSD_Swiss	0.118
Entropy		**Variance**		**Sum of Pairs**		**Dev. From Max. Frequency**	
icEntropy	0.134	icVar	0.139	icSOP	0.128	icDiff	0.122
HEntropy	0.131	HVar	0.134	HSOP	0.126	HDiff	0.118
Entropy	0.130	Var	0.131	SOP	0.125	Diff	0.115
Log form of Dev. From Max. Freq.							
icLogDf	0.120						
HLogDf	0.118						
LogDf	0.117						

Table 4.7: Comparison of Performance across Various Weighting Schemes at False Positive Rate Cutoff Value of 1

Relative Entropy (Local Alignment Background)		**Relative Entropy (SwissProt Background)**		**JSD (Local Alignment Background)**		**JSD (SwissProt Background)**	
icRE_Local	0.925	icRE_Swiss	0.921	icJSD_Local	0.925	icJSD_Swiss	0.922
HRE_Local	0.919	HRE_Swiss	0.915	HJSD_Local	0.906	HJSD_Swiss	0.904
RE_Local	0.916	RE_Swiss	0.913	JSD_Local	0.899	JSD_Swiss	0.898
Entropy		**Variance**		**Sum of Pairs**		**Dev. From Max. Frequency**	
icEntropy	0.916	icVar	0.924	SOP	0.900	icDiff	0.884
HEntropy	0.912	HVar	0.915	HSOP	0.900	HDiff	0.879
Entropy	0.911	Var	0.913	icSOP	0.898	Diff	0.877
Log form of Dev. From Max. Freq.							
icLogDf	0.878						
HLogDf	0.882						
LogDf	0.885						

4.4 Execution Time

Tables 4.8 and 4.9 show the minimum, maximum, mean and standard deviation of the execution times (in seconds) of the different methods, tabulated separately based on the weighting scheme used, and sorted in increasing order on the basis of the mean time. Execution time included the time to calculate the different residue frequencies and the time to perform the actual conservation calculation. Sum of pairs method gave the worst performance in the case of the methods using the position-based sequence weights. This is as a result of the rapid increase in the number of computational steps in conducting the pairwise calculations as the number of pairs in the alignment column increase. The entropy method has the fastest computation time. The computation times for the methods utilizing the independent count scheme is very much higher than those utilizing the position-based sequence weights.

Table 4.8: Minimum, Maximum, Mean and Standard Deviation of the Execution Times of the Different Methods, Utilizing the Position-based sequence weights

	Method Time /sec.			
Conservation Methods	**Min**	**Max**	**Mean**	**SD**
HEntropy	0.15	2.97	0.63	0.5
HrEntropyS	0.16	3.12	0.65	0.52
HJSD_Swiss	0.18	3.07	0.68	0.52
HDiff	0.22	3.25	0.81	0.57
HVariance	0.19	4.46	0.87	0.75
HJSD_Local	0.22	4.41	0.9	0.74
HRE_Local	0.2	4.35	0.9	0.81
HSOP	0.24	3.83	1.05	0.73

Table 4.9: Minimum, Maximum, Mean and Standard Deviation of the Execution Times of the Different Methods Utilizing the Independent Counts Scheme

	Method Time /sec.			
Conservation Method	**Min**	**Max**	**Mean**	**SD**
icEntropy	2.55	325.11	47.03	61.25
icJSD_Swiss	2.55	337.24	47.18	62.32
icJSD_Local	2.6	327.16	47.3	61.31
icVariance	2.59	339.61	47.46	62.89
icSOP	2.66	333.64	47.7	62.6
icDiff	2.64	338.08	47.88	63.12
icRE_Local	2.55	329.77	48.22	63.32
icrEntropyS	2.52	344.34	48.27	63.51

5. Discussion

The independent count scheme is basically useful in situations in which it is not desirable to reduce the weights associated with highly similar sequences, but only to take into consideration the correlation between them (Pei and Grishin, 2001). This condition is true with the alignments used in this study, noting that the sequences in the alignments are orthologs. It may therefore not be too desirable to penalize for sequence similarity.

The relatively low correlation figures coming from all the variants of the LogDf and Diff methods in Table 4.1 as well as their lackluster performance across all levels of false positive rate cutoffs in tables 4.2, 4.3 and 4.4 is enough evidence to point to the fact that the basis of the method is faulty. The methods have relatively high correlation with SOP and Entropy variants, but this is so because, as pointed out in the literature review, the methods are actually stripped down version of the sum-of-pairs method, taking substitutions as being from the dominant amino acid at an alignment column to all others, while SOP considers substitution between all pairs of amino acids present at a column. SOP has high correlation with Entropy variants, and this is the reason for the relatively high correlation between Entropy and the variants of LogDf, Diff. The LogDf and Diff methods are based upon the argument that in an alignment column, evolution could be taken as being from the dominant amino acid in that column to the others. This cannot easily be substantiated, and the low performance of this method compared to others testifies to its shortcomings.

Most of the other low correlation figures in Table 4.1 occur in situations where one of the methods uses the independent counts scheme and the other uses no weighting method at all, or position-based sequence weight method. The low correlation may thus be partly due to the different weighting schemes used. Exceptions to this are the correlations between JSD_Local, HJSD_Local and Entropy, HEntropy, SOP and HSOP, as well as between JSD_Swiss, HJSD_Swiss and SOP and HSOP which have comparatively low correlation values (between 0.79 and 0.84) without independent count being involved in the schemes.

On the other hand, it is noteworthy that the Variance and Relative Entropy methods (using background frequency from the alignment) have quite good correlation with all other methods. They are also always part of the three top-performing methods in tables

4.2, 4.3, and 4.4, across all levels of false positive rate cutoff and weighting schemes. These two methods, along with the Jensen-Shannon divergence method (using background frequency from the alignment), which happens to be the best performing method across all cutoffs under the independent counts scheme, measure conservation as deviation from the overall amino acid frequency in the alignment. Their top performance may thus be an indication of the superiority of this approach to measuring conservation as compared to the one taken by entropy and sum-of-pairs methods.

The sum-of-pairs method exhibits very high positive correlation with Entropy (0.95) and Variance (0.95) methods. This is a pointer to the fact that it may not be inferior to other methods like Entropy, even though its computational process may not be biologically explainable (Valdar, 2000). The method is also the only high performing method that incorporates the substitution matrix directly into its computation.

The ROC analysis-based comparative performance of the different methods as discovered in this work differs from that of Capra and Singh (2007) who, using a superset of the data used in this work and position-based sequence weights (Henikoff and Henikoff, 1994), claimed that the order of performance (in increasing order) is Jensen-Shannon Divergence method, Relative Entropy method, Sum of Pairs method and Entropy method, at false positive rate cutoff values of 0.1 and 1. In their work, they used the background frequency from the overall amino acid distribution in the BLOSUM 62 alignments and also used pseudocounts in their scheme. In this work, amino acid distributions from all of the protein sequences in SwissProt database as well as amino acid distribution from the alignment itself were used as background frequencies. Pseudocounts were not used at all, neither is this feature available in the software implemented.

On the other hand, the comparative performance in this work agrees with that of Wang and Samudrala (2006). In their work, the comparative performance between the methods they considered (in decreasing order) was in the order: Relative Entropy, Variance, Entropy and Sum of Pairs. In particular, this work as well as Wang and Samudrala (2006) disagrees with the claim of Capra and Singh (2007) that the sum-of-pairs method outperforms entropy method.

The claim by Capra and Singh (2007) that the Jensen-Shannon divergence method performs better than all others across all ranges of false positive rates is refuted by this

study as well as by the study of Fischer *et al*. (2008). According to the figures available in Tables 4.2, 4.3 and 4.4, Relative Entropy, Variance, Entropy and Sum of Pairs methods outperform Jensen-Shannon Divergence method using both unweighted and the position-based sequence weight schemes, but they are all outperformed by Jensen-Shannon Divergence method if independent count is used.

The discrepancy in the results may be attributable to the inappropriate definition of the true positives and true negatives in the studies. The definition of catalytic residues as being the true positives and non-catalytic residues the true negatives is made with the assumption that catalytic residues mostly exhibit high conservation and non-catalytic residues low conservation. This definitely is not the case as residues could be conserved for structural reasons, too (Cheng *et al*., 2005; Sadowski and Jones, 2009; Wang and Samudrala, 2005). In the dataset used for this study, the number of non-catalytic but highly conserved residues conserved is far greater than the number of conserved catalytic residues (Table 4.2).

Sequence conservation is the most valuable single contributor to assigning function to protein structures (Sadowski and Jones, 2009), as well as being the most powerful criteria for discriminating between catalytic and non-catalytic residues (Gutteridge *et al*., 2003), but recognizing the inaccuracy inherent in predicting functionality from conservation alone, various studies have tried utilizing other forms of discriminatory information, in addition to residue conservation (e.g. Fischer *et al*., 2008; Gutteridge *et al*., 2003; Sodhi *et al*., 2004; Youn *et al.*, 2007). Fischer *et al*. (2008) combined information from the conservation at each column, the column's amino acid distribution, as well as its predicted secondary structure and relative solvent accessibility in order to predict catalytic as well as ligand-binding residues. Gutteridge *et al*., (2003), on the other hand, employed a neural network approach in predicting catalytic site as well as in predicting active sites. Structural parameters such as the solvent accessibility, type of secondary structure, depth, and cleft that the residue lies in, as well as the conservation score and residue type, are used as inputs for this neural network. Unfortunately, even with the number of parameters used in these studies and others, the success rate in predicting functional residues are still very much below optimal.

The ratio of catalytic to non catalytic residues in the dataset is approximately 1:109, while the ratio of highly conserved catalytic residues (entropy between 0 and 0.2) and

highly conserved non-catalytic residues (entropy in the same range) is 1:14 (Fig. 4.2). It is therefore very evident that conservation measures cannot differentiate between highly conserved catalytic residues and highly conserved non-catalytic residues. Even though a high percentage of catalytic residues are very well conserved, but some still have entropic conservation in the average to low conservation region (entropy 0.4 to 0.9) (Fig. 4.2) (Bartlett *et al.*, 2002). This could, for example, be due to the case in which the different enzymes in an alignment are truly homologous, but work on different substrates and therefore exhibit substrate specificity. Numerous other properties, apart from conservation, also affect the propensity of particular residues to be catalytic. All the aforementioned facts could be summarized by the statement of (Chelliah *et al.*, 2004) that sequence conservation measures do not distinguish evolutionary restraints that arise from function from those that arise from structure. Thus, the choice of catalytically active residues as the only true positives in a test meant to make generic statements about the performance of conservation measures is a bad one.

5.1 Future Direction

A dataset that would effectively serve as a gold standard for testing the performance of the methods in this study is one in which all the conserved residues have been annotated. This, though, is a big and difficult task. Another avenue to exploit is that of qualitative assessment, using a model alignment designed to test the differentiating power of different alignment measures on different combination of amino acids in alignment columns. This was the approach taken by Valdar (2002) in his seminal paper on measures of conservation.

The conservation modules developed in this work could still be enhanced. Pseudocounts could be incorporated into the variance, relative entropy and the Jensen-Shannon divergence methods. These are basically the three best methods according to the criteria used in this study, and possibly they may perform better with the incorporation of the pseudocounts.

The independent count scheme has shown quite remarkable performance, but the average execution time for the methods using the scheme is quite high (about ten times higher) than the other methods that do not make use of the scheme. This is due to the exponential increase in the number of calculations needed to compute the independent count for each

amino acid as the amino acid variability in columns of the alignment as well as the number of sequences in the alignment increase. This scheme may thus show a remarkable improvement in execution time if a dynamic programming algorithm is developed for it.

6. References

Ahola, V., Aittokallio, T., Vihinen, M., and Uusipaikka, E. (2006). A statistical score for assessing the quality of multiple sequence alignments. *BMC Bioinformatics, 7*, 484.

Ahola, V., Aittokallio, T., Vihinen, M., and Uusipaikka, E. (2008). Model-based prediction of sequence alignment quality. *Bioinformatics, 24*(19), 2165-2171.

Altschul, S. F., Madden, T. L., Schaffer, A. A., Zhang, J., Zhang, Z., Miller, W., *et al*. (1997). Gapped BLAST and PSI-BLAST: A new generation of protein database search programs. *Nucleic Acids Res., 25*(17), 3389-3402.

Armon, A., Graur, D., and Ben-Tal, N. (2001). ConSurf: An algorithmic tool for the identification of functional regions in proteins by surface mapping of phylogenetic information. *J. Mol. Biol., 307*(1), 447-463.

Armougom, F., Moretti, S., Keduas, V., and Notredame, C. (2006). The iRMSD: A local measure of sequence alignment accuracy using structural information. *Bioinformatics*, 22(14), e35-9.

Armougom, F., Moretti, S., Poirot, O., Audic, S., Dumas, P., Schaeli, B., *et al*. (2006). Expresso: Automatic incorporation of structural information in multiple sequence alignments using 3D-coffee. *Nucleic Acids Res., 34*(Web Server issue), W604-8.

Atkins, J.F. and Gesteland, R. (2002). The 22nd Amino Acid. *Science* 296 (5572), 1409–1410.

Attwood,T.K. (2002). The PRINTS database: a resource for identification of protein families. *Brief Bioinform.*, **3**, 252–263.

Bairoch,A., Bucher,P. and Hofmann,K. (1997). The PROSITE database, its status in 1997. *Nucleic Acids Res.*, 25, 217–221.

Bartlett, G. J., Porter, C. T., Borkakoti, N., and Thornton, J. M. (2002). Analysis of catalytic residues in enzyme active sites. *J. Mol. Biol., 324*(1), 105-121.

Beiko, R. G., Chan, C. X., and Ragan, M. A. (2005). A word-oriented approach to alignment validation. *Bioinformatics*, 21(10), 2230-2239.

Berman, H. M., Henrick, K., and Nakamura, H. (2003). Announcing the worldwide protein data bank. *Nat. Struct. Biol., 10*(12), 980.

Capra, J. A., and Singh, M. (2007). Predicting functionally important residues from sequence conservation. *Bioinformatics, 23*(15), 1875-1882.

Carrillo, H., and Lipman, D. (1988). The multiple sequence alignment problem in biology. *SIAM J Appl Math*, 48, 1073-1082.

Chelliah, V., Chen, L., Blundell, T. L., and Lovell, S. C. (2004). Distinguishing structural and functional restraints in evolution in order to identify interaction sites. *J. Mol. Biol., 342*(5), 1487-1504.

Cheng, G., Qian, B., Samudrala, R., and Baker, D. (2005). Improvement in protein functional site prediction by distinguishing structural and functional constraints on protein family evolution using computational design. *Nucleic Acids Res., 33*(18), 5861-5867.

Cone, B. E.; Del Rio, R. M.; Davis, J. N. and Stadtman, T. C. (1976). Chemical Characterization of the Selenoprotein Component of Clostridial Glycine Reductase: Identification of Selenocysteine as the Organoselenium Moiety. *PNAS* 73 (8), 2659–2663.

Conway, D. (2000). Object Oriented Perl. Manning Publications Co. 1 – 23,124-125.

Do, C. B., Mahabhashyam, M. S., Brudno, M., and Batzoglou, S. (2005). ProbCons: Probabilistic consistency-based multiple sequence alignment. *Genome Res., 15*(2), 330-340.

Dodge, C., Schneider, R., and Sander, C. (1998). The HSSP database of protein structure-sequence alignments and family profiles. *Nucleic Acids Res., 26*(1), 313-315.

Durbin, R., Eddy, S., Krogh, A., and Mitchison, G. (1998). Biological sequence analysis: Probabilistic models of proteins and nucleic acids. Cambridge (UK): Cambridge University Press.

Edgar, R. C. (2004). MUSCLE: Multiple sequence alignment with high accuracy and high throughput. *Nucleic Acids Res., 32*(5), 1792-1797.

Edgar, R. C., and Batzoglou, S. (2006). Multiple sequence alignment. *Curr. Opin. Struct. Biol., 16*(3), 368-373.

Elofsson, A. (2002). A Study on Protein Sequence Alignment Quality. *Proteins.* 46(3): 330-339.

Fawcett, T. (2006). An introduction to ROC analysis. *Pattern Recognition Letters, 27*(8), 861-874.

Feng, D. F., and Doolittle, R. F. (1987). Progressive sequence alignment as a prerequisite to correct phylogenetic trees. *J. Mol. Evol., 25*(4), 351-360.

Finn R.D.; Tate, J.; Mistry, J.; Coggill, P.C.; Sammut, S.J.; Hotz, H.R.; Ceric, G.; Forslund, K.; Eddy, S.R.; Sonnhammer, E.L.; Bateman, A. (2008). The Pfam protein families database. *Nucleic Acids Res.* 36:D281-8.

Fischer, J. D., Mayer, C. E., and Soding, J. (2008). Prediction of protein functional residues from sequence by probability density estimation. *Bioinformatics, 24*(5), 613-620.

Gutteridge, A., Bartlett, G. J., and Thornton, J. M. (2003). Using a neural network and spatial clustering to predict the location of active sites in enzymes. *J. Mol. Biol., 330*(4), 719-734.

Halterman, R. L. (2008). Object Oriented Programming in Java. Accessed at http://computing.southern.edu/halterman/OOPJ/index.html on 4th April, 2009.

Henikoff, J. G.; Greene, E. A.; Pietrokovski, S. and Henikoff, S. (2000). Increased coverage of protein families with the blocks database servers. *Nucl. Acids Res*. 28:228-230.

Henikoff, S., and Henikoff, J. G. (1994). Position-based sequence weights. *J. Mol. Biol., 243*(4), 574-578.

Henikoff, S.; Henikoff, J. G.; Pietrokovski, S. (1999). Blocks+: A non-redundant database of protein alignment blocks derived from multiple compilations. *Bioinformatics* 15(6):471-479.

Karlin, S., and Brocchieri, L. (1996). Evolutionary conservation of *RecA* genes in relation to protein structure and function. *J. Bacteriol., 178*(7), 1881-1894.

Katoh, K., Kuma, K., Toh, H., and Miyata, T. (2005). MAFFT version 5: Improvement in accuracy of multiple sequence alignment. *Nucleic Acids Res., 33*(2), 511-518.

Katoh, K., Misawa, K., Kuma, K., and Miyata, T. (2002). MAFFT: A novel method for rapid multiple sequence alignment based on fast fourier transform. *Nucleic Acids Res., 30*(14), 3059-3066.

Krzycki, J. (2005). The direct genetic encoding of pyrrolysine. *Curr. Opin. Microbiol.* 8 (6), 706–712.

Lafore, R. (1998). Object Oriented Programming in C++ (Third Edition). Macmillan Computer Publishing. 8 – 14.

Landan, G., and Graur, D. (2007). Heads or tails: A simple reliability check for multiple sequence alignments. *Mol. Biol. Evol.*, 24(6), 1380-1383.

Landan, G., and Graur, D. (2008). Local reliability measures from sets of co-optimal multiple sequence alignments. Pacific Symposium on Biocomputing. 15-24.

Lassmann, T., and Sonnhammer, E. L. L. (2002). Quality assessment of multiple alignment programs. *FEBS Lett*, 529, 126-130.

Lassmann, T., Frings, O., and Sonnhammer, E. L. (2009). Kalign2: High-performance multiple alignment of protein and nucleotide sequences allowing external features. *Nucleic Acids Res., 37*(3), 858-865.

Lassmann, T., and Sonnhammer, E. L. (2005). Kalign--an accurate and fast multiple sequence alignment algorithm. *BMC Bioinformatics, 6*, 298.

Lena, K. W. (2001). Receiver operating characteristic (ROC) analysis. evaluating discriminance effects among decision support systems. No. UMINF 01.08). Sweden: Department of Computer Science, Umea University.

Lichtarge, O., Bourne, H. R., and Cohen, F. E. (1996). An evolutionary trace method defines binding surfaces common to protein families. *J. Mol. Biol., 257*(2), 342-358.

Lipman, D. J., Altschul, S. F., and Kececioglu, J. D. (1989). A tool for multiple sequence alignment. *Proc. Natl. Acad. Sci. U.S.A., 86*(12), 4412-4415.

Liu, X., Li, J., Guo, W., and Wang, W. (2006). A new method for quantifying residue conservation and its applications to the protein folding nucleus. *Biochem. Biophys. Res. Commun., 351*(4), 1031-1036.

Liu, X. S., and Guo, W. L. (2008). Robustness of the residue conservation score reflecting both frequencies and physicochemistries. *Amino Acids, 34*(4), 643-652.

Lockless, S. W., and Ranganathan, R. (1999). Evolutionarily conserved pathways of energetic connectivity in protein families. *Science, 286*(5438), 295-299.

Marchler-Bauer, A., Anderson, J. B., Chitsaz, F., Derbyshire, M. K., DeWeese-Scott, C., Fong, J. H., *et al*. (2009). CDD: Specific functional annotation with the conserved domain database. *Nucleic Acids Res., 37*(Database issue), D205-10.

Marti-Renom, M.A.; Stuart, A.C.; Fiser, A.; Sanchez, R.; Melo, F. and Sali, A. (2000). Comparative protein structure modeling of genes and genomes. *Annu Rev Biophys Biomol Struct.* 29:291-325.

Mayrose, I., Graur, D., Ben-Tal, N., and Pupko, T. (2004). Comparison of site-specific rate-inference methods for protein sequences: Empirical Bayesian methods are superior. *Mol. Biol. Evol., 21*(9), 1781-1791.

Mevissen, H. T., and Vingron, M. (1996). Quantifying the local reliability of a sequence alignment. *Protein Eng.*, 9(2), 127-132.

Mihalek, I., Res, I., and Lichtarge, O. (2007). Background frequencies for residue variability estimates: BLOSUM revisited. *BMC Bioinformatics, 8*, 488.

Mirny, L. A., and Shakhnovich, E. I. (1999). Universally conserved positions in protein folds: Reading evolutionary signals about stability, folding kinetics and function. *J. Mol. Biol., 291*(1), 177-196.

Morrison, D. A., and Ellis, J. T. (1997). Effects of nucleotide sequencevalignment on phylogeny estimation: A case study of 18S rDNAs of Apicomplexa. *Mol. Biol. Evol.* 14: 428–441.

Mulder,N.J., Apweiler,R., Attwood,T.K., Bairoch,A., Barrell,D., Bateman,A., Binns, D., Biswas, M., Bradley, P., Bork, P. (2003). The InterPro Database, 2003 brings increased coverage and new features. *Nucleic Acids Res.*, 31, 315–318.

Needleman, S. B., and Wunsch, C. D. (1970). A general method applicable to the search for similarities in the amino acid sequence of two proteins. *J. Mol. Biol., 48*(3), 443-453.

Notredame, C., Higgins, D. G., and Heringa, J. (2000). T-coffee: A novel method for fast and accurate multiple sequence alignment. *J. Mol. Biol., 302*(1), 205-217.

Obuchowski, N. A. (2005). ROC analysis. *AJR Am. J. Roentgenol., 184*(2), 364-372.

O'Docherty, M. (2005). Object-Oriented Analysis and Design: Understanding System Development with UML 2.0. John Wiley & Sons Ltd., England. 9 – 48.

O'Sullivan, O., Suhre, K., Abergel, C., Higgins, D. G., and Notredame, C. (2004). 3DCoffee: Combining protein sequences and structures within multiple sequence alignments. *J. Mol. Biol., 340*(2), 385-395.

O'Sullivan, O., Zehnder, M., Higgins, D., Bucher, P., Grosdidier, A., and Notredame, C. (2003). APDB: A novel measure for benchmarking sequence alignment methods without reference alignments. *Bioinformatics*, 19 Suppl 1, i215-21.

Pei, J., and Grishin, N. V. (2001). AL2CO: Calculation of positional conservation in a protein sequence alignment. *Bioinformatics, 17*(8), 700-712.

Pei, J., and Grishin, N. V. (2006). MUMMALS: Multiple sequence alignment improved by using Hidden Markov Models with local structural information. *Nucleic Acids Res., 34*(16), 4364-4374.

Pei, J., and Grishin, N. V. (2007). PROMALS: Towards accurate multiple sequence alignments of distantly related proteins. *Bioinformatics, 23*(7), 802-808.

Pei, J., Sadreyev, R., and Grishin, N. V. (2003). PCMA: Fast and accurate multiple sequence alignment based on profile consistency. *Bioinformatics, 19*(3), 427-428.

Phillips, A., Janies, D. and Wheeler, W. (2000). Multiple sequence alignment in phylogenetic analysis. *Mol. Phylogenet. Evol.*, 16, 317–330.

Porter, C. T., Bartlett, G. J., and Thornton, J. M. (2004). The Catalytic Site Atlas: A resource of catalytic sites and residues identified in enzymes using structural data. *Nucleic Acids Res., 32*(Database issue), D129-33.

Pupko, T., Bell, R. E., Mayrose, I., Glaser, F., and Ben-Tal, N. (2002). Rate4Site: An algorithmic tool for the identification of functional regions in proteins by surface mapping of evolutionary determinants within their homologues. *Bioinformatics, 18 Suppl 1*, S71-7.

R. Development Core Team. (2008). R: A language and environment for statistical computing. Vienna, Austria: R Foundation for Statistical Computing.

Sadowski, M. I., and Jones, D. T. (2009). The sequence-structure relationship and protein function prediction. *Curr. Opin. Struct. Biol.* 19(3):357-362.

Shannon, C. E. (1948). A mathematical theory of communication. *The Bell System Technical Journal, 27*, 379-432; 623-656.

Shenkin, P. S., Erman, B., and Mastrandrea, L. D. (1991). Information-theoretical entropy as a measure of sequence variability. *Proteins, 11*(4), 297-313.

Shi, J., Blundell, T. L., and Mizuguchi, K. (2001). FUGUE: Sequence-structure homology recognition using environment-specific substitution tables and structure-dependent gap penalties. *J. Mol. Biol., 310*(1), 243-257.

Sigrist C.J.A., Cerutti L., Hulo N., Gattiker A., Falquet L., Pagni M., Bairoch A., Bucher P. (2002). PROSITE: a documented database using patterns and profiles as motif descriptors. *Brief Bioinform*. 3:265-274.

Sing, T., Sander, O., Beerenwinkel, N., and Lengauer, T. (2005). ROCR: Visualizing classifier performance in R. *Bioinformatics, 21*(20), 3940-3941.

Smith, T. F., and Waterman, M. S. (1981). Identification of common molecular subsequences. *J. Mol. Biol., 147*(1), 195-197.

Sodhi, J. S., Bryson, K., McGuffin, L. J., Ward, J. J., Wernisch, L., and Jones, D. T. (2004). Predicting metal-binding site residues in low-resolution structural models. *J. Mol. Biol., 342*(1), 307-320.

Taylor, W. R. (1999). Protein structure comparison using iterated double dynamic programming. *Protein Sci., 8*(3), 654-665.

The Universal Protein Resource (UniProt) (2008). *Nucleic Acids Res.* 36:D190-D195.

Thomas, M. C., and Joy, A. T. (1991). *Elements of information theory* John Wiley & Sons, Inc.

Thompson, J. D., Higgins, D. G., and Gibson, T. J. (1994). CLUSTAL W: Improving the sensitivity of progressive multiple sequence alignment through sequence weighting, position-specific gap penalties and weight matrix choice. *Nucleic Acids Res., 22*(22), 4673-4680.

Thompson, J. D., Plewniak, F., Ripp, R., Thierry, J. C., and Poch, O. (2001). Towards a reliable objective function for multiple sequence alignments. *J. Mol. Biol.*, 314(4), 937-951.

Thompson, J. D., Plewniak, F., Thierry, J., and Poch, O. (2000). DbClustal: Rapid and reliable global multiple alignments of protein sequences detected by database searches. *Nucleic Acids Res., 28*(15), 2919-2926.

Tomovic, A., and Oakeley, E. J. (2007). Quality estimation of multiple sequence alignments by Bayesian hypothesis testing. *Bioinformatics*, 23(18), 2488-2490.

Topsøe, F. (2003). Jensen-Shannon divergence and norm-based measures of discrimination and variation. Department of Mathematics, University of Copenhagen.

Valdar, W. S. (2002). Scoring residue conservation. *Proteins, 48*(2), 227-241.

Wang, K., and Samudrala, R. (2005). FSSA: A novel method for identifying functional signatures from structural alignments. *Bioinformatics, 21*(13), 2969-2977.

Wang, K., and Samudrala, R. (2006). Incorporating background frequency improves entropy-based residue conservation measures. *BMC Bioinformatics, 7*, 385.

Wang, L., and Jiang, T. (1994). On the complexity of multiple sequence alignment. *J. Comput. Biol., 1*(4), 337-348.

Williamson, R. M. (1995). Information theory analysis of the relationship between primary sequence structure and ligand recognition among a class of facilitated transporters. *J. Theor. Biol., 174*(2), 179-188.

Yona, G., and Levitt, M. (2002). Within the twilight zone: A sensitive profile-profile comparison tool based on information theory. *J. Mol. Biol., 315*(5), 1257-1275.

Youn, E., Peters, B., Radivojac, P., and Mooney, S. D. (2007). Evaluation of features for catalytic residue prediction in novel folds. *Protein Sci., 16*(2), 216-226.

Zinoni, F.; Birkmann, A.; Stadtman, T. C. and Bock, A. (1986). Nucleotide Sequence and Expression of the Selenocysteine-Containing Polypeptide of Formate Dehydrogenase (Formate-hydrogen-lyase-Linked) from Escherichia coli. *PNAS* 83 (13), 4650–4654.

7. Appendices

7.1 Comparison of Differences between Manually Calculated Results and the Program-Calculated Output

Calculation Using the Unweighted Scheme

Alignment Column	Entropy Score Program	Manual	Difference		Relative Entropy Score Program	Manual	Difference
1	0	0	0		4.207714	4.207714	4.98E-08
2	0.108515	0.108515	3.76E-08		3.707265	3.707265	3.03E-08
3	0.231378	0.231378	1.32E-08		3.050448	3.050448	2.00E-08
4	0.108515	0.108515	3.76E-08		3.786438	3.786438	3.83E-08
5	0.314897	0.314897	2.95E-08		2.768656	2.768656	4.01E-09
6	0.343707	0.343707	2.00E-08		2.723318	2.723318	1.43E-08
7	0.352141	0.352141	3.15E-09		2.231497	2.231497	4.05E-08
8	0.352141	0.352141	3.15E-09		3.615903	3.615903	1.07E-08
9	0.768622	0.768622	1.32E-08		1.204808	1.204808	6.19E-09
10	0.6	0.6	0		1.347928	1.347927	4.86E-08
11	0.6	0.6	0		1.347928	1.347927	4.86E-08

Alignment Column	Jensen-Shannon Divergence Score Program	Manual	Difference		Variance Score Program	Manual	Difference
1	0.3731238	0.373124	5.5306E-09		0.55102	0.55102	8.16E-09
2	0.3294435	0.329444	2.3105E-09		0.452201	0.452201	2.54E-08
3	0.2837933	0.283793	2.2054E-08		0.370873	0.370873	1.11E-09
4	0.3371666	0.337167	3.2635E-08		0.454887	0.454887	2.64E-08
5	0.2539266	0.253927	4.4297E-08		0.278532	0.278532	2.07E-08
6	0.2495406	0.249541	3.6923E-08		0.236532	0.236532	1.50E-08
7	0.1983488	0.198349	7.3753E-09		0.484499	0.484499	2.39E-08
8	0.3083006	0.308301	6.6291E-09		0.532027	0.532027	4.41E-08
9	0.0882623	0.088262	1.3581E-08		0.430632	0.430632	1.13E-08
10	0.1249308	0.124931	3.5556E-08		0.35102	0.35102	8.16E-09
11	0.1249308	0.124931	3.5556E-08		0.35102	0.35102	8.16E-09

Alignment Column	Sum of Pairs Score Program	Manual	Difference
1	1	1	0
2	0.900498	0.900498	4.72E-08
3	0.723607	0.723607	2.25E-09
4	0.709773	0.709773	3.84E-08
5	0.488658	0.488658	2.25E-10
6	0.318205	0.318205	7.73E-10
7	0.68	0.68	0
8	0.05467	0.05467	1.68E-08
9			
10	0.4	0.4	0
11	0.4	0.4	0

Calculation using Position-based Sequence Weights (Henikoff and Henikoff, 1994)

Alignment Column	Entropy Score				Relative Entropy Score		
	Program	Manual	Difference		Program	Manual	Difference
1	0	0	0		4.207714	4.207714	4.98E-08
2	0.171925	0.171925	3.17E-08		3.398356	3.398356	2.08E-08
3	0.230412	0.230412	4.07E-08		3.042663	3.042663	2.97E-08
4	0.171925	0.171925	3.17E-08		3.565268	3.565268	4.58E-08
5	0.350665	0.350665	4.16E-08		2.68985	2.68985	2.71E-08
6	0.3529	0.3529	1.77E-08		2.757995	2.757995	2.74E-08
7	0.364206	0.364206	4.51E-09		2.228656	2.228656	4.84E-08
8	0.364206	0.364206	4.51E-09				
9	0.748384	0.748384	3.66E-08		1.382622	1.382622	4.43E-08
10	0.608305	0.608305	1.00E-07		1.319941	1.319941	2.08E-07
11	0.608305	0.608305	1.00E-07		1.319941	1.319941	2.08E-07

Alignment Column	Jensen-Shannon Divergence Score				Variance Score		
	Program	Manual	Difference		Program	Manual	Difference
1	0.373124	0.373124	5.53E-09		0.584787	0.584787	3.34E-08
2	0.304729	0.304729	4.54E-09		0.381899	0.381899	3.80E-09
3	0.283374	0.283373	2.56E-08		0.407334	0.407334	9.58E-09
4	0.323639	0.323638	3.12E-08		0.397973	0.397973	6.16E-09
5	0.247296	0.247296	1.47E-08		0.241421	0.241421	5.54E-09
6	0.252808	0.252808	4.59E-08		0.25741	0.25741	4.07E-08
7	0.198146	0.198146	7.29E-09		0.480383	0.480383	3.02E-09
8	0.29271	0.292709	4.10E-08		0.517113	0.517113	1.90E-08
9	0.109425	0.109425	8.25E-09		0.42082	0.42082	1.20E-08
10	0.122337	0.122337	5.91E-08		0.348351	0.348351	7.67E-08
11	0.122337	0.122337	5.91E-08		0.348351	0.348351	7.67E-08

Alignment Column	Sum of Pairs Score		
	Program	Manual	Difference
1	1	1	0
2	0.81606	0.81606	3.05E-08
3	0.725206	0.725206	3.64E-09
4	0.463482	0.463482	3.00E-08
5	0.296363	0.296363	4.94E-08
6	0.194413	0.194413	7.20E-09
7	0.683539	0.654953	0.028586
8	0.052953	0.052953	2.34E-09
9			
10	0.391695	0.391695	1.00E-07
11	0.391695	0.391695	1.00E-07

Calculation using the Independent Count Scheme

Alignment Column	Entropy Score Program	Manual	Difference		Relative Entropy Score Program	Manual	Difference
1	0	0	0		4.207714	4.207714	4.98E-08
2	0.198396	0.198396	3.19E-08		3.261729	3.261729	2.67E-08
3	0.22697	0.22697	3.03E-08		3.043998	3.043998	1.26E-08
4	0.198396	0.198396	3.19E-08		3.48458	3.48458	2.12E-08
5	0.358735	0.358734	3.53E-08		2.644072	2.644072	4.97E-08
6	0.365044	0.365044	2.39E-08		2.702982	2.702982	3.19E-08
7	0.364537	0.364537	3.86E-08		2.164674	2.164674	1.96E-08
8	0.364537	0.364537	3.86E-08		3.46199	3.46199	2.43E-08
9	0.768622	0.768622	1.32E-08		1.204808	1.204808	6.19E-09
10	0.6	0.6	0		1.347928	1.347927	4.86E-08
11	0.6	0.6	0		1.347928	1.347927	4.86E-08

Alignment Column	Jensen-Shannon Divergence Score Program	Manual	Difference		Variance Score Program	Manual	Difference
1	0.373124	0.373124	5.53E-09		0.664381	0.664381	3.08E-08
2	0.295878	0.295878	7.50E-09		0.407922	0.407922	3.95E-08
3	0.283446	0.283446	2.35E-08		0.448071	0.448071	5.99E-09
4	0.319655	0.319655	4.81E-09		0.433563	0.433563	2.68E-08
5	0.24422	0.24422	2.66E-08		0.302544	0.302544	6.80E-09
6	0.248367	0.248367	4.85E-08		0.305251	0.305251	1.75E-08
7	0.192351	0.192351	2.90E-08		0.424846	0.424846	2.35E-08
8	0.297856	0.297856	3.32E-08		0.523847	0.523847	2.57E-08
9	0.088262	0.088262	1.36E-08		0.348235	0.348235	1.60E-08
10	0.124931	0.124931	3.56E-08		0.313379	0.313379	4.74E-09
11	0.124931	0.124931	3.56E-08		0.313379	0.313379	4.74E-09

Alignment Column	Sum of Pairs Score Program	Manual	Difference
1	1	1	0
2	0.776402	0.776402	1.80E-08
3	0.730875	0.730875	1.41E-08
4	0.347806	0.347806	3.79E-09
5	0.259402	0.259402	2.15E-08
6	0.155632	0.155632	4.88E-08
7	0.653493	0.684224	0.030732
8	0.043374	0.043374	7.40E-09
9			
10	0.4	0.4	0
11	0.4	0.4	0

Computation Using Relative Entropy Scheme with Position-Based Sequence Weight and Background Frequency Derived from the Alignment

Alignment Column	Relative Entropy Score		
	Program	Manual	Difference
1	1.2680774	1.268076481	9.19E-07
2	0.8703967	0.870396959	2.59E-07
3	1.1536614	1.153663623	2.22E-06
4	1.0467757	1.046776445	7.45E-07
5	0.8103157	0.810318286	2.59E-06
6	0.8828735	0.882876299	2.8E-06
7	2.6124935	2.612488871	4.63E-06
8			
9	2.3631043	2.363110416	6.12E-06
10	1.2439575	1.243958055	5.55E-07
11	1.2439575	1.243958055	5.55E-07

Computation Using Jensen-Shannon Divergence scoring with Independent Counts scheme and Background Frequency Derived from the Alignment

Alignment Column	Jensen Shannon Divergence Score		
	Program	Manual	Difference
1	0.124676	0.089821	0.034854766
2	0.068554	0.049119	0.019435586
3	0.107293	0.109138	0.00184514
4	0.095371	0.075764	0.019607897
5	0.07614	0.081987	0.005846829
6	0.084209	0.089534	0.005324653
7	0.203371	0.225068	0.021696403
8	0.308916	0.29986	0.009055856
9	0.29393	0.304658	0.010727888
10	0.078907	0.118212	0.039305203
11	0.078907	0.118212	0.039305203

NOTE:

- Smaller difference indicates better level of agreement between the computed scores.
- Not all the test results are displayed
- Some values were skipped (blank cells in the tables) because of the combinatorial calculations involved, or because they follow from the results in other calculations.
- Relatively high differences between the program-generated and manually calculated values in the calculations using background frequencies generated from the alignment itself is as a result of accumulated round-off errors mostly in the manual calculation of the background frequencies from the alignment.
- Most of the other differences could also be attributed to round-off errors in the manual calculations.

7.2 Background Frequencies Computed from SwissProt Sequences

Amino Acid	Frequency
A	0.08133
B	0.00000
C	0.01418
D	0.05412
E	0.06730
F	0.03888
G	0.07040
H	0.02285
I	0.05929
K	0.05878
L	0.09673
M	0.02412
N	0.04075
O	0.00000
P	0.04771
Q	0.03964
R	0.05506
S	0.06671
T	0.05360
U	0.00000
V	0.06813
W	0.01103
Y	0.02936
Z	0.00000

7.3 Documentation of the Program Modules

7.3.1 conservation::Conservation Module

SYNOPSIS

```
use conservation::Conservation;
use Bio::AlignIO;
#create the conservation object directly by passing the filename as an
argument
my $cons = conservation::Conservation->new(-file => 'filename', -format
=> 'alignment_format');
#alternatively, create the conservation object by first opening it with
Bio::AlignIO and then passing the result alignment object
my $in = new Bio::AlignIO(-file => 'filename');
my $aln = $in->next_aln;
my $cons = conservation::Conservation->new(-alnObj => $aln);
```

DESCRIPTION

```
The conservation::Conservation module calculates conservation based on
some of the various conservation measures that have been come up with
in the literature.
Particularly, it provides the following conservation measures:
```

- Entropy conservation measure
- Sum of Pairs conservation measure
- Variance conservation measures (all from Pei and Grishin (2001))
- Relative entropy conservation meausure (Wang and Samudrala, 2006)
- Jensen-Shannon divergence conservation measure (Capra and Singh, 2007),
- and the measure that considers conservation in a column as the deviation from the residue with maximum frequency (Liu *et al*., 2006; Liu and Guo, 2008).

The conservation::Conservation module enables the user to use two different alignment sequence weighting schemes apart from the third option of using no weighting at all. These options include:

- Position based sequence weights (Henikoff and Henikoff, 1994)
- The independent count scheme (Pei and Grishin, 2001).

The relative entropy as well as the Jensen-Shannon divergence conservation measures require the use of a suitable background frequencies. The module provides the ability to choose from two background frequencies viz:

- Background frequency calculated from SwissProt
- Background frequency calculated from the overall amino acid distribution in the alignment under consideration.

In calculating rate of substitution between two amino acids in the Sum of Pairs measure, this module provides for the use of only the Blosum 62 matrix. Other matrices could be made available in later releases. The work of Liu *et al*. (2006) Liu and Guo (2008) used a

transformed version of the blosum substitution matrix, and this is provided as a custom Blosum substitution matrix named Blosum_Liu.

METHODS

Title: new
Usage: $cons = conservation::Conservation->new(-alnObj => alignment_object);
$cons = conservation::Conservation->new(-file => 'filename', -format => 'alignment_format');#another form of creation, using file directly
Function: Creates a new conservation::Conservation object
Returns: conservation::Conservation object
Args:
alignment_object - Bio::Align::AlignI compliant object.
filename - Name of the file containing the alignment.
alignment_format - format of the alignment. If this is omitted, bioperl will try to guess the alignment format.
cutoff_value - optional value (between 0 and 1, both inclusive) specifying the limit to the percentage of gaps in a column above which the gaps will be considered absolute unconserved and conservation values will not be calculated for such column. Defaults to 0.5.

Title: entropyCIndex
Usage: $entropy = $cons->entropyCIndex(-wMethod => 'weight_method');
Function: calculates and returns the entropy conservation score of each column in the alignment
Returns: a hash reference, with the keys being the column indices and the values the entropies
Args: weight_method: This could be 'hh94' or 'indcount', for Henikoff and Henikoff (1994) and independent count methods respectively. If not specified, unweighted scheme is used.

Title: relEntropyCIndex
Usage: $rentropy = $cons->relEntropyCIndex(-bg => "bg_freq_source", -wMethod => 'weight_method');
Function: calculates and returns the relative entropy conservation score of each column in the alignment
Returns: a hash reference, with the keys being the column indices and the values the relative entropy conservation scores.
Args: weight_method: This could be 'hh94' or 'indcount', for Henikoff and Henikoff (1994) and independent count methods respectively. If not specified, unweighted scheme is used.
bg_freq_source - This can be 'swiss' or 'local'. This is the background frequency source to be used in this computation. 'local' means background frequency from the alignment itself while 'swiss' means background frequency calculated from SwissProt.

Title: jSDCIndex
Usage: $jsd = $cons->jSDCIndex(-bg => "bg_freq_source", -wMethod => 'weight_method', -lambda => lambda_value);
Function: calculates and returns the Jensen-Shannon divergence -based conservation score of each column in the alignment
Returns: a hash reference, with the keys being the column indices and the values the JSD conservation scores.

Args: weight_method: This could be 'hh94' or 'indcount', for Henikoff and Henikoff (1994) and independent count methods respectively. If not specified, unweighted scheme is used.
bg_freq_source - This can be 'swiss' or 'local'. This is the background frequency source to be used in this computation. 'local' means background frequency from the alignment itself while 'swiss' means background frequency calculated from SwissProt.
labmda_value - (optional, defaults to 0.5) Any value between 0 and 1 could be specified for this.

Title: varianceCIndex
Usage: $vari = $cons->varianceCIndex(-wMethod => 'weight_method');
Function: calculates and returns the relative entropy conservation score of each column in the alignment
Returns: a hash reference, with the keys being the column indices and the values the variance conservation scores.
Args: weight_method: This could be 'hh94' or 'indcount', for Henikoff and Henikoff (1994) and independent count methods respectively. If not specified, unweighted scheme is used.

Title: devFromMaxCIndex
Usage: $devia = $cons->devFromMaxCIndex(-subMatrix => substitution_matrix, -wMethod => 'weight_method');
Function: calculates and returns the divergence from amino acid with maximum frequency conservation score of each column in the alignment
Returns: A reference to a hash of hash. The first level has two keys, 'nonLogForm' and 'logForm', which can be used to access the non-log and log forms of the conservation scores respectively.
Args: weight_method: This could be 'hh94' or 'indcount', for Henikoff and Henikoff (1994) and independent count methods respectively. If not specified, unweighted scheme is used.
substitution_matrix - recommended value is "blosum_liu", which is the one used in the original publication.

Title: sumOfPairsCIndex
Usage: $sop = $cons->sumOfPairsCIndex(-subMatrix => "substitution_matrix", -normalize => normalize_value, -wMethod => 'weight_method');
Function: calculates and returns the sum of pairs conservation score of each column in the alignment
Returns: a hash reference, with the keys being the column indices and the values the sum-of-pairs conservation scores.
Args: substitution_matrix - name of the substitution matrix to use. Only option available now is 'blosum62'.
normalize_value - value indicating whether to normalize the matrix so that diagonals are normalized to 1. Any expression that could evaluate to true (eg 1) activates this feature.
weight_method: This could be 'hh94' or 'indcount', for Henikoff and Henikoff (1994) and independent count methods respectively. If not specified, unweighted scheme is used.

7.3.2 conservation::Statistics Module

SYNOPSIS

```
use conservation::Statistics;

#create a statistics object, using the alignment object
$stats = conservation::Statistics->new(-alnObj => $para->{'-alnObj'});

#another form of creation, using file directly
$stats = conservation::Statistics->new(-file => 'filename', -format =>
'alignment_format');

#get the unweighted frequencies for residues in the alignment columns
$uwFreqs = $stats->uwFrequencies();

#get the independent count based frequencies for the residues in the
#alignment
$icFreqs = $stats->independentcounts();

#get the Henikoff and Henikoff (1994) based frequencies for the residues #in
the alignment
$wFreqs = $stats->wFrequencies();

#get the distribution of each amino acid in the whole alignment using the
#unweighted scheme
$uwTFreqs = $stats->uwTotalFrequency();

#get the distribution of each amino acid in the whole alignment based on #the
ind. count scheme
$icTFreqs = $stats->icTotalFrequency();

#get the distribution of each amino acid in the whole alignment based on
#Henikoff and Henikoff (1994) scheme
$wTFreqs = $stats->wTotalFrequency();
```

DESCRIPTION

The statistics class provide the basic statistics needed for the calculation of various conservation measures. It directly relies on the conservation::AlnWrapper Object to make available various sub-parts of the alignment and the conservation::Weights objects to make available the calculated weights of various sequence in the alignment, as well as the independent counts of various amino acids in an alignment column.

METHODS

```
Title: new
Usage: $stats = conservation::Statistics->new(-alnObj =>
alignment_object);
$stats = conservation::Statistics->new(-file => 'filename', -format =>
'alignment_format');#another form of creation, using file directly
Function: Creates a new conservation::Statistics object
Returns: conservation::Statistics object
Args:   alignment_object - Bio::Align::AlignI compliant object.
        filename: Name of the file containing the alignment.
        alignment_format: format of the alignment. If this is omitted,
```

bioperl will try to guess the alignment format.

Title: gapChar
Usage: $gpchar = $stats->gapChar();
Function: Returns the character used to represent gaps in the current alignment
Returns: Character
Args:

Title: getBGFreqs
Usage: $bgfreq = $stats->getBGFreqs();
Function: Returns the background frequency of the amino acids as pre-calculated from amino acids in SwissProt.
Returns: Reference to a hash, in which the amino acids are the keys and the values are the frequency of each amino acid from SwissProt.
Args:

Title: uwFrequencies
Usage: $uwFreq = $stats->uwFrequencies();
Function: Calculates and returns the unweighted frequencies of each amino acid in each column of the alignment.
Returns: Reference to a hash of a hash, with the first hash having as its key indexes to column in the alignment and its value being a reference to another hash having as its key amino acids in the column and its values frequencies of those amino acids.
Args:

Title: uwPositionalAAFreq
Usage: $uwCount = $stats->uwPositionalAAFreq();
Function: Calculates the unweighted count of each amino acid at each position.
Returns: Reference to a hash of a hash, with the first hash having as its key indexes to column in the alignment and its value being a reference to another hash having as its key amino acids in the column and its values counts of those amino acids.
Args:

Title: uwTotalPositionalFreq
Usage: $uwCountSum = $stats->uwTotalPositionalFreq();
Function: Calculates the sum of the unweighted counts for each position.
Returns: A hash reference, with the keys being column index and the value the sum of the unweighted counts
Args:

Title: independentcounts
Usage: $indCounts = $stats->independentcounts();
Function: Calculates and returns the independent counts for the entire alignment.
Returns: A hash reference, with the keys being column index and the value being a reference to another hash, having the amino acids in the column as keys and their estimated independent count as values.
Args:

Title: wFrequencies
Usage: $wFreq = $stats->wFrequencies();
Function: Calculates and returns the weighted frequency for the whole alignment.

Returns: A hash reference, with the keys being column index and the value being a reference to another hash, having the amino acids in the column as keys and their weighted frequencies as values.
Args:

Title: wPositionalAAFreq
Usage: $wCounts = $stats->wPositionalAAFreq();
Function: Calculates the weighted count of each amino acid at each position.
Returns: Reference to a hash of a hash, with the first hash having as its key indexes to column in the alignment and its value being a reference to another hash having as its key amino acids in the column and its values weighted counts of those amino acids.
Args:

Title: wTotalPositionalFreq
Usage: $wCountSum = stats->wTotalPositionalFreq();
Function: Calculates the sum of the weighted counts for each position.
Returns: A hash reference, with the keys being column index and the value the sum of the unweighted counts
Args:

Title: uwTotalFrequency
Usage: $uwTFreq = $stats->uwTotalFrequency();
Function: Calculates the unweighted frequency of each amino acid in the whole alignment.
Returns: A hash reference, with the keys being amino acids and the values the associated alignment-wide frequency.
Args:

Title: icTotalFrequency
Usage: $icTFreq = $stats->icTotalFrequency();
Function: Calculates the frequency of each amino acid in the whole alignment based on the independent count scheme.
Returns: A hash reference, with the keys being amino acids and the values the associated alignment-wide ind. count frequencies.
Args:

Title: wTotalFrequency
Usage: $wTFreq = $stats->wTotalFrequency();
Function: Calculates the weighted frequency of each amino acid in the whole alignment.
Returns: A hash reference, with the keys being amino acids and the values the associated alignment-wide weighted frequency.
Args:

Title: getCutoffIndices
Usage: $cutoffInd = $stats->getCutoffIndices(-cutoff => cutoff_value, -method => 'method_value');
Function: Returns the column indices of those columns that have the percentage gaps in them less than that specified as cutoff.
Returns: A hash reference, containing as keys only the indices of the columns to be retained and the values set to 1.
Args: cutoff_value - figure between 0 and 1 specifying the limit above which the percentage gap in a column must be for the column conservation score to be discarded.
method_value - could be 'hh94' (for Henikoff and Henikoff (1994) weights), 'indcount' or it could be left out (unweighted schemes used for indcount and unweighted frequencies).

```
Title:  percentageGaps
Usage:  $pgaps = $stats->percentageGaps(-method => 'method_value');
Function: Calculates the weighted frequency of each amino acid in the
whole alignment.
Returns: A hash reference, containing as keys indices of each columns
and the percentage of gaps in the column as values.
Args: method_value - could be 'hh94' (for Henikoff and Henikoff (1994)
      weights), 'indcount' or it could be left out (unweighted schemes
      used for indcount and unweighted frequencies).
```

```
Title:  maxF
Usage:  $mFreq = $stats->maxF(-freq => $frequencies);
Function: Gets the residues having the maximum frequency in a column.
Returns: A reference to an array containing the residue(s) having the
maximum frequency.
Args: frequencies - A hash ref containing the residues in a column and
      their associated frequencies
```

DEPENDENCIES

```
conservation::Weights
conservation::AlnWrapper
Error
List::Util
Carp
```

7.3.3 conservation::SubstitutionMatrix Module

SYNOPSIS

use conservation::SubstitutionMatrix;

DESCRIPTION

This module loads a substitution matrix from disc. It is additionally able to normalise it and make its entries available to the calling programme.

METHODS

```
Title:  new
Usage:  $submat = conservation::SubstitutionMatrix->new(-matrix =>
'matrix_type');
Function: Creates a new conservation::SubstitutionMatrix instance
Returns: conservation::SubstitutionMatrix object
Args:   matrix_type - one of "blosum_62" or "blosum_liu"
```

```
Title:  getRate
Usage:  $rate = $submat->getRate(-aa1=> 'first_amino_acid', -aa2 =>
'second_amino_acid');
Function: get the substitution rate for one amino acid to the other
Returns: a float - the substitution rate
Args:   first_amino_acid, second_amino_acid - amino acids character
       representations
```

```
Title:  normalize
Usage:  $submat->normalize();
Function: Normalizes the matrix so that the diagonals become 1.
Returns:
Args:
```

```
Title:  getType
Usage:  $typeofsub = $submat->getType();
Function: Return the type of substitution matrix that this represents.
Returns: "BLOSUM62" or "BLOSUM_LIU"
Args:
```

DEPENDENCIES

```
Error Module
```

7.3.4 conservation::Weights Module

SYNOPSIS

```
use conservation::Weights;

#create a new weight object
$weight = conservation::Weights->new(-alnWrapper => AlnWrapperObj, -
method => weighting_method_name, -gapchar => gap_character);

#calculate the independent count for some combination of sequences
$weight->indCounts($seq_nos);

#get the Henikoff and Henikoff (1994) weight for the next sequence
$weight->next();

#get the Henikoff and Henikoff (1994) weight for a particular sequence
$weight->getSeqWeight($seqIndex);

#get the Henikoff and Henikoff (1994) weight for all the sequences
$weight->getWeights();
```

DESCRIPTION

The conservation::Weights module is one that implements different weighting scheme for sequences in an alignment. Currently available schemes include the method of Henikoff and Henikoff (1994) and the independent cound method of Pei and Grishin(2001)

METHODS

```
Title: new
Usage: $weight = conservation::Weights->new(-alnWrapper =>
AlnWrapperObj, -method => weighting_method_name, -gapchar =>
gap_character);
Function: Creates a new conservation::Weights object
Returns: conservation::Weights object
Args:   AlnWrapperObj: This is an alignment wrapper object.
```

```
        -method: This is the weighing scheme method. It could  be
       "hh94" (for Henikoff and Henikoff (1994) method) or 'indcount',
       for the independent count scheme
        gap_character: the gap character in the alignment
```

Title: indCounts
Usage: $weight->indCounts($seq_nos);
Function: Returns the independent count for an amino acid in a particular column, given the indexes of the sequences having that aa in the column.
Returns: The calculated independent count, a float.
Args: $seq_nos: This is a hash which contains as its keys the indexes of the sequences which have the particular aa in the column. The values of the keys should be any true value (eg 1).

Title: next
Usage: $weight->next();
Function: Returns the Henikoff and Henikoff (1994) calculated weight of the next sequence.
Returns: Henikoff and Henikoff (1994) calculated weight of the next sequence, undef if the last sequence's weight has already been returned.
Args:

Title: getSeqWeight
Usage: $weight->getSeqWeight($seqIndex);
Function: Returns the Henikoff and Henikoff (1994) calculated weight for a particular sequence, given the index.
Returns: Henikoff and Henikoff (1994) calculated weight of the sequence with the supplied index.
Args: $seqIndex - the index of the sequence whose weight is requested.

Title: getWeights
Usage: $weight->getWeights();
Function: Returns the weights of all the sequences, only available for Henikoff and Henikoff (1994) weights.
Returns: The weights, in a hash ref with the sequence numbers as the keys and the weights as the values.
Args:

DEPENDENCIES

```
conservation::Iterators::AlnIterator;
```

7.3.5 conservation::Iterators::SeqIterator

SYNOPSIS

```
use conservation::Iterators::SeqIterator;

#create new instance
$seqIt = conservation::Iterators::SeqIterator->new(-alnHash =>
alignment_hash_obj);

#get the integer that indexes the next sequence
$seqIt->next

#get the integer that indexes the previous sequence
```

```
$seqIt->previous
```

DESCRIPTION

The conservation::Iterators::SeqIterator class is a module that iterates over the sequences in an alignment, returning each sequence as an array, or a string, or a string with gaps removed.

METHODS

```
Title: new
Usage: conservation::Iterators::SeqIterator->new(-alnHash =>
alignment_hash_obj, -min => minimum_value, -max => maximum_value);
Function: Creates a new SeqIterator object
Returns: SeqIterator object
Args:   alignment_hash_obj: this is an alignment hash object. The
        alignment should be formatted as a hash of arrays, with each
        hash position containing a reference to an array which holds
        the individual AA in the sequence occupying that position
        minimum_value: (optional) minimum value for the iterator
        maximum_value: (optional) maximum value for the iterator
```

```
Title: next
Usage: $seqIt->next(-join => join_value,-remgap => gap_rem_value);
Function: Returns the next sequence, as an array reference or as a
string, depending on the parameters supplied (see Args below).
Returns: Next sequence as an array reference, or string.
Args:   -join => join_value :   (optional) Specifies that a string
         should be returned, and not an array of characters. Any true
         value would do.
        -remgap => gap_rem_value:  (optional) Specifies that gap should
         be removed. Any true value would do
```

```
Title: previous
Usage: $seqIt->previous(-join => join_value,-remGap => gap_rem_value);
Function: Returns the previous sequence, as an array reference or as a
string, depending on the parameters supplied. This does not decrement
the pointer (as in the case of next which increments the pointer to the
next sequence). It just returns the sequence previous to this one.
Returns: Previous sequence as an array reference, or string.
Args:   -join => join_value :   (optional) Specifies that a string
         should be returned, and not an array of characters. Any true
         value would do.
        -remgap => gap_rem_value:  (optional) Specifies that gap should
         be removed. Any true value would do
```

DEPENDENCIES

```
conservation::Iterators::AlnIterator module
```

7.3.6 conservation::Iterators::ColumnIterator

SYNOPSIS

```
use conservation::Iterators::ColumnIterator;

#create new instance
$colIt = conservation::Iterators::ColumnIterator->new(-alnHash =>
alignment_hash_obj);

#returns next column
$colIt->next();

#returns previous column
$colIt->previous():
```

DESCRIPTION

The conservation::Iterators::ColumnIterator class is a module that iterates over the columns of an alignment, returning each column as an array, or a string, or a string with gaps removed.

METHODS

Title: new
Usage: conservation::Iterators::ColumnIterator->new(-alnHash =>
alignment_hash_obj, -colData => colData, -gapChar => gap_character);
Function: Creates a new conservation::Iterators::ColumnIterator object
Returns: conservation::Iterators::ColumnIterator object
Args: alignment_hash_obj: this is an alignment hash object (which is a hash reference). This contains the alignment formatted as a hash of arrays, with each hash position containing a reference to an array which holds the individual AA in the sequence occupying that position. The keys are integers starting from 0.
colData: Reference to a hash of arrays, the key to each position in this hash ref is the alignment column number, starting from zero, and the content of each position is a reference to an array, with each array having the individual amino acids in each cell.
gap_character: (optional) This is a character that is used to represent gaps in the alignment. If not supplied the default "-" is used.
minimum_value: (optional) minimum value for the ColumnIterator
maximum_value: (optional) maximum value for the ColumnIterator

---

Title: next
Usage: $colIt->next(-join => join_value,-remgap => gap_rem_value);
Function: Returns the next column, as an array reference or as a string, depending on the parameters supplied (see Args below).
Returns: Next column as an array reference, or string.
Args: join_value : (optional) Specifies that a string should be returned, and not an array of characters. Any true value would do.
gap_rem_value: (optional) Specifies that gap should be removed. Any true value would do

---

```
Title: previous
Usage: $colIt->previous(-join => join_value,-remgap => gap_rem_value);
Function: Returns the previous column, as an array reference or as a
string, depending on the parameters supplied (see Args below).
Returns: Previous column as an array reference, or string.
Args:   join_value :   (optional) Specifies that a string should be
        returned, and not an array of characters. Any true value would
        do.
        gap_rem_value:  (optional) Specifies that gap should be
        removed. Any true value would do
```

DEPENDENCIES

```
conservation::Iterators::AlnIterator module
```

7.3.7 conservation::Iterators::AlnIterator

SYNOPSIS

```
use conservation::Iterators::AlnIterator;

#create an conservation::Iterators::AlnIterator object
$it = conservation::Iterators::AlnIterator->new(-min => min_value, -max
=>max_value, -reverse =>reverse_value);

#get the next value
$it->next;

#get the previous value
$it->previous;
```

DESCRIPTION

The conservation::Iterators::AlnIterator class is a module that provides counters that could easily be used to navigate both the columns and rows of an alignment. Or, on a wider scale, that could be used to iterate the numbers between min and max, incrementing by one at each iteration step.

METHODS

```
Title: new
Usage: $it = conservation::Iterators::AlnIterator->new(-min =>
min_value, -max =>max_value, -reverse =>reverse_value);
Function: Creates a new conservation::Iterators::AlnIterator object
Returns: conservation::Iterators::AlnIterator object
Args:   -min => minimum value (optional, 0 if not supplied)
        -max => maximum value
        -reverse => optional value specifying whether to start from
maximum and iterate down to min. 1 is recommended in order to activate
this value, but can be any value that evaluates to true in perl
```

```
Title: next
Usage: $it->next;
```

Function: Returns the next integer, increments the internal value point to the next integer
Returns: Integer value, max inclusive. Returns undef if max is exceeded
Args:

Title: previous
Usage: $it->previous;
Function: Returns the previous integer
Returns: Integer value, min inclusive. Returns undef if min is exceeded
Args: