UTILISATION DES ALGORITHMES GENETIQUES POUR L'ANALYSE DE SEQUENCES BIOLOGIQUES

Cédric Notredame

Doctorat en Bio−informatique

Février 1998

Université Paul Sabatier
France

Directeur De Thèse: Prof. François Amalric

Acknowledgement

I wish to thank the EMBL for their funding through an EMBL grant.

This work was carried out in the lab of Des Higgins, first at the European Molecular Biology Laboratory in Heidelberg,Germany and later at the EMBL outstation, the European Bioinformatics Institute, in Hinxton, U.K.
Des has been a constant source of support. I wish to thank him and express him all my gratitude for teaching me most of what I know in bioinformatics, and so much more about being a scientist. I also wish to thank Ture Etzold who gave me a place in his team when Des had to leave for Ireland. Thanks to his expertise in the field, Thure has had a lots of influence on my work.
A special thanks to the system managers, namely Roy Omond, Rodrigo Lopez and Pettere Jokinen who have always been supportive, allowing me to overload the machines any time I needed it. Without their help, this work could not have been achieved. I also wish to thank Miguel Andrade, Inge Jonassen and Burkhard Rost for stimulating discussions and frendship. Chris Sander and Liisa Holm have always been available to share with me their extensive experience of the field. I wish to express them my gratitude.

Michelle Magraine has proved an unvaluable allie for fighting against my english grammar weaknesses. I wish to thank her for that. I also wish to thank Rob Harper for joining me in my everyday fight against post script files. My stay at the EBI has been extremly enjoyable.

This work is dedicated to my friends and family for their constant support over all these years.

RESUME DE THESE

Une grande partie de la recherche fondamentale en biologie moléculaire repose sur l’étude des protéines et des acides nucléiques. Ces molécules extrêmement complexes résultent de la combinaison d’éléments plus simples: les acides aminés et les nucléotides. Vingt acides aminés constituent la grande majorité des protéines, cinq nucléotides constituent la plupart des acides nucléiques. Le termes séquence est utilisé pour désigner l’enchaînement de nucléotides constituant un acide nucléique ou l’enchaînement d’acides aminés constituant une protéine. La plupart des protéines sont codées par les gènes contenus dans l’ADN des chromosomes.

Au cours de ces dernières années, de nombreux progrès techniques ont rendu possible le séquençage à grande échelle du génome de plusieurs espèces bactériennes ou eucaryotes. Ces séquences d’ADN sont entreposées dans des banques de données spécialisées (Swiss Prot, Gene Bank, EMBL nucléotides database...) dont la croissance (en taille) est aujourd’hui exponentielle.

La bio−informatique est une sous−discipline de la biologie ayant pour objet l’analyse de ces données par des moyens informatiques. Le principe de base d’une telle approche est la notion de relation entre fonction et séquence. Le but est d’extrapoler des données obtenues de façon expérimentales sur certaines séquences à d’autres séquences pour lesquelles aucune donnée expérimentale n’est disponible.

Alors qu’il est clair que deux protéines (ou acide nucléiques) ayant la même séquence ont probablement la même fonction, une corrélation devient plus difficile à établir lorsque les séquences ne présentent qu’une homologie partielle. La nécessitée d’utiliser ce type d’information est la principale motivation derrière le développement des méthodes de comparaison aujourd’hui utilisées. Le travail présenté dans cette thèse est essentiellement consacré à cet aspect de la bio−informatique.

L’un des moyens les plus utilisés pour la comparaison de séquences est l’alignement. Un alignement permet d’identifier les zones conservées entre deux séquences. Ces zones peuvent correspondre à des motifs structuraux ou fonctionnels dont l’identification permet de faire des prédictions quand à la fonction putative des séquences analysées. De façon plus générale, un alignement permet l’identification de régions sur lesquelles existent des contraintes diverses imposant le maintient de certaines propriétés. D’autre part, un alignement de qualité permet l’évaluation de la distance évolutive séparant deux organismes, ou deux protéines.

Cependant, dans les cas complexes, la quantité d’information contenue dans deux séquences n’est pas suffisantes, et il devient nécessaire d’étendre la comparaison à plusieurs séquences. C’est là l’objet des alignements de séquences multiples. Leur problématique est double.

Il s’agit tout d’abord d’un problème biologique. Etant donné un groupe de séquences, les propriétés de l’alignement optimal doivent être définies. La règle la plus simple est de tenter de générer autant d’identités que possible dans les colonnes tout en limitant le nombre d’insertions/délétions (gaps). En pratique néanmoins, les règles utilisées sont plus complexes et peuvent prendre en compte la nature des acides aminés alignés (protéines) ou la structure secondaire des séquences (ARN). L’usage est de donner à cette liste de règles une forme mathématique associant un score à chaque alignement. On parle alors de fonction objective. Un nombre important de fonctions de ce types ont été décrites au cours de ces dernières années. Globalement, elles peuvent être divisées en deux groupes: les fonctions basées sur des matrices de substitutions et des penalitées d’insertion/délétion et les fonctions globales telles que les HMM (Hidden Markov Models). Une des propriétés les plus importantes d’une fonction objective est sa signification biologique. De façon idéale, une fonction doit assigner à un alignement optimal un score traduisant l’intérêt biologique de l’information qu’il contient.

Le second aspect est purement informatique. Il ne suffit pas d’avoir une fonction objective, il faut aussi être capable d’optimiser le score de cette fonction (i.e. produire l’alignement ayant le meilleur score). Ce problème est loin d’être trivial. L’optimisation de la plupart des fonctions objectives appartient à la classe des problèmes dits NP complets. En conséquence, l’optimisation ne peut être réalisée qu’en utilisant des méthodes dites heuristiques qui ne garantissent pas une solution optimale.

Le travail présenté dans cette thèse englobe l’ensemble de ces problématiques. Dans la première partie, une méthode d’optimisation globale par algorithme génétique est proposée. Cette

méthode est intégrée dans un logiciel nommé SAGA (Sequence Alignment by Genetic Algorithm). Les algorithmes génétiques sont des stratégies d'optimisation basées sur une analogie avec le phénomène de sélection naturelle. Cette méthode peut en théorie être appliquée à n'importe quel type de fonction objective.

Le second aspect du travail a consisté à définir une nouvelle fonction objective (COFFEE: Consistency based Objective Function for alignEment Evaluation) et à optimiser cette fonction en utilisant SAGA de façon à prouver que COFFEE peut induire la création de meilleurs alignements que des méthodes alternatives.

La troisième application a été axée sur l'alignement d'ARNs ribosomiques avec définition d'une fonction objective adaptée à la prise en compte des interactions secondaires. Ce programme, adapté de SAGA a été nommé RAGA (RNA Alignment by Genetic Algorithm). L'une des principales limitations de RAGA réside dans la simplicité de la fonction objective utilisée. Afin de remédier à ce problème, un travail d'analyse a été réalisé sur des alignements de référence afin de déterminer les paramètres pouvant aider à la définition d'une fonction objective plus réaliste dans la prise en compte des contraintes à modéliser dans l'alignement. Ce travail constitue la quatrième application présentée dans cette thèse.

Dans l'ensemble, ce travail a permis d'établir l'utilité des algorithmes génétiques dans le contexte des problèmes d'alignement de séquences multiples. SAGA est à l'heure actuelle l'algorithme le plus performant pour l'optimisation des fonctions objectives couramment utilisées pour les alignements de séquences multiples. Dans le cas de séquences protéiques, SAGA est le seul algorithme capable de réaliser l'alignement global de plus de dix séquences. Pour ce qui est de l'ARN ribosomique, RAGA est le seul programme capable d'aligner des séquences ayant une longueur supérieure à deux mile paires de bases, tout en prenant en compte les pseudo−noeux. D'autre part, la fonction COFFEE est l'une des rares fonctions capable de permettre la génération d'alignements biologiquement plus exacts que ceux obtenus par ClustalW (ClustalW est une des méthodes d'alignement les plus populaires).

RESUME DES ANNEXES

Document Numéro 1
SAGA: Sequence Alignement by Genetic Algorithm

Dans cet article, une nouvelle approche est proposée pour la résolution du problème des alignements de séquences multiples. Un algorithme génétique a été conçu et intégré dans un logiciel nommé SAGA. La méthode implique l'évolution d'une population d'alignements. Dans ce contexte, évolution signifie que la qualité des alignements est graduellement améliorée au gré d'une succession de cycles (générations) contenant des étapes de modifications aléatoires (opérateurs) ainsi que des étapes de sélection basée sur le score. Le degré d'amélioration est jugé par l'évaluation du score de chaque alignement à l'aide de la fonction objective. SAGA utilise une technique de contrôle automatique pour réguler l'utilisation simultanée de vingt opérateurs destinés à recombiner entre eux des alignements (crossing overs), ou bien à les modifier individuellement (mutation). Afin de tester SAGA, nous avons utilisé comme référence le programme M.S.A. (Multiple Sequence Alignment) capable d'optimiser une des fonction objectives le plus couramment utilisée (somme des paires avec pénalités affines de délétion/insertions).

Utilisé dans ce contexte, SAGA fournit de meilleurs résultats que M.S.A. en terme d'optimisation (score de l'alignement obtenu). De plus les alignements produits par SAGA sont biologiquement plus exacts s'il on en juge par leur similarité avec l'alignement des mêmes séquences réalisés par comparaison de structures. Au total, SAGA a été teste sur treize groupes de séquences pour lesquelles un alignement de référence basé sur les structures est disponible dans la banque de données 3D_ali.

Document Numéro 2
COFFEE: A New Objective Function for Multiple Sequence Alignmnents

Dans ce travail, nous présentons un nouveau mode d'évaluation des alignements de séquences multiples. Cette fonction est nommée COFFEE. COFFEE est une mesure du degré de consistance existant entre un alignement de séquences multiples et un bibliothèque de référence contenant les mêmes séquences alignées deux par deux. Il est montré que le score COFFEE peut être efficacement optimisé par SAGA. La fonction a été utilisé sur onze groupes de séquences pour lesquels un alignement de référence est disponible dans la banque de données 3D_ali. Dans neuf cas sur onze, SAGA, utilisé avec COFFEE, produit des alignements meilleurs que ceux obtenus avec

ClustalW (s'il on en juge par leur similarité avec les alignements de références). Nous avons aussi montré que le score assigné par COFFEE peut être utilisé pour évaluer la qualité d'un alignement multiple, de façon locale ou globale. Finalement, la bibliothèque de référence peut être constituée d'alignement en paire obtenus par comparaison de structure (par exemple, des alignements extraits de FSSP). Dans ce cas là, SAGA−COFFEE est capable de produire des alignements structuraux multiple de très haute qualité. En théorie, COFFEE devrait permettre d'appliquer aux alignements multiples n'importe quelle méthode adaptée à l'alignement de paires de séquences.

Document Numéro 3
RAGA: RNA Sequence Alignment by Genetic Algorithm.

Cet article décrit une nouvelle approche pour aligner deux séquences d'ARN homologues lorsque la structure secondaire de l'une des deux molécules est connue. A cette fin, deux programmes ont été développés, RAGA (RNA séquence alignment by Genetic algorithm) et PRAGA (Parallel RAGA). Ces deux programmes sont essentiellement basés sur le programme SAGA. La parallélisation est réalisée par la synchronisation d'un nombre défini de copies actives de RAGA. Celles−ci échangent une partie de leur population suivant une topologie définie comme étant un arbre à branches multiple et profondeur variable.

Cette méthode permet d'optimiser une fonction objective prenant en compte les informations primaires et secondaires contenues dans les deux séquences. Une des propriétés les plus intéressantes de RAGA réside dans le fait qu'il est possible de prendre en compte aussi bien les tiges boucles classiques que les pseudo−noeux présents dans l'ARN ribosomique.

RAGA à été testé à l'aide de neuf alignements de référence extraits à partir d'alignements d'experts. Ces alignements, constitués d'ARNs de la petite sous unité ribosomique, ont servi de référence. Dans chacun des cas, PRAGA est capable de surpasser en exactitude les méthodes alternatives basées sur la programmation dynamique. Ceci est vrai même lorsque la distance phylogénétique séparant les deux séquences à aligner est très importante (comme entre l'humain et saccharomyces cerevisiae).

Document Numéro 4
Optimisation of RNA Profile Alignments

Ce projet fait pendant au projet numéro 3 et s'intègre dans un contexte plus large visant à la création des outils nécessaires à la maintenance automatique des banques de données d'ARN ribosomiques. De part le monde, plusieurs banques de données de ce type existent. Elles sont essentiellement maintenues de façon manuelle. A long terme, la création de méthodes automatiques va devenir une nécessité absolue. Dans le document numéro 3, nous avons proposé une procédure destinée à l'alignement de deux séquences. N'utiliser que deux séquences revient à ignorer la vaste quantité d'information contenue dans les alignements multiples établis par des groupes d'experts. Le but de ce projet a été de définir certaines des modalités d'utilisation de cette information.

Différentes méthodes de pondérations ont été testées et implémentée dans un contexte de programmation dynamique. L'évaluation des méthodes a été réalisée en testant la qualité d'alignement obtenue lorsqu'une séquence est extraite puis réintroduite dans un alignement multiple. Cette stratégie ne prend en compte que les contraintes primaires.

Les résultats montrent que par l'utilisation d'un mode de pondération adéquat et d'un système de pénalités d'insertion/délétion adapté, il est possible d'améliorer considérablement la qualité de l'alignement entre un profil et une séquence.

TABLE OF CONTENTS

1−INTRODUCTION

1.1 BIOINFORMATICS AND BIOLOGY

Life as we know it is a complex arrangement of biological structures designed to interact with each other. As complex as they may appear, even the most elaborated living structures can be described as arrangements of smaller less complex building blocks (such as cells), which are themselves the result of the combination of even smaller basic blocks (such as metabolites, proteins, nucleic acids).

Identifying these structures and characterizing their functions is a major aim of biology. Questions can be addressed at any level of organization one may wish to study (from populations to atoms in fact). In such a top to bottom approach, molecular biology is almost at the bottom. It deals with the biological structures at the molecular level, trying to understand how these are created and interact with one another to perform the basic functions of life.

The search for ordered systems is an important part of the biological methodology. Ordered systems usually make it possible to establish general rules allowing a global understanding of otherwise disparate collections of facts. In this respect, the discovery of DNA structure and the understanding of RNA and protein synthesis have been two of the most important milestones of modern molecular biology. They have allowed a deep and precise understanding of some of the most central cellular mechanisms. We now know that proteins and RNA molecules are involved at virtually all the steps of biological processes. We also know that these key components of cellular life are coded in DNA sequences, in an almost universal manner. These DNA sequences are contained in the genomes of living organisms.

At the lowest level, a genome can be described as a long string of nucleotides. It could be compared to a very long text made of four letters. As in a text, the letters are not distributed at random but organized in words. In the context of a genome, a word will be any sequence having a function. One of the difficulties when trying to identify these ’words’ stems from the fact that nature uses spaces and punctuation in a very personal way. This mean that not only do we ignore beforehand the function of the ’words’, we also do not know where they start and finish. Add to this the fact that there are many different classes of ’words’. Some allow the binding of other molecules on the DNA, others are translated into RNAs, that can in turn be translated into proteins. A protein or an RNA molecule may contain motifs that will have functions (binding, catalytic site....). The genome also contains some very specific combinations of words such as genes (enhancer, promoter, introns, exons...). Bioinformatics and molecular biology are two complementary techniques with similar aims: identification of biological structures and sub−structures at a molecular level and characterization of their function.

’Function’ is a very general concept. If we look at it from an experimental point of view such as genetics, a function can be defined with respect to a gene and will often be deduced from what happens when this gene is inactivated/modified by a mutation. Often, this is not enough to gain a real deep understanding of the mechanisms involved. To do so, one will have to know whether this function is performed by a protein, an RNA, or a regulatory sequence. If it is a protein then the next question is ’how does the protein perform its function?’. If it is an enzyme we will want to know where is the catalytic site, does it look like any other known site, what are the residues involved in the site and how they perform their function? The protein (enzyme or not) may also interact with other proteins, nucleic acids or metabolites.

Here again we will want to know what are the portions of the protein involved in these operations and what are the potential partners. In most of the cases, this information will be much easier to understand/predict when a 3D model is available for the protein. In a broad sense, the function of a protein (or a nucleic acid or a DNA/RNA regulatory element) is defined by the sum of all these elements.

Until recently, the only way available for gathering together these pieces of information was to use wet lab techniques. These involve genetic analysis, cloning, sequencing, interaction experiments..... Although the results obtained that way can usually be regarded as strong biological evidence, they suffer from a major drawback. The cost of wet lab techniques is extremely high, in terms of time and money. This means that there is a limit on the number of functions that can be thoroughly investigated through such techniques. The problem has become especially severe now that the improvement of sequencing techniques gives us access to far more sequences than it will ever be possible to analyse in the wet lab. It is this situation that has promoted the massive development of bioinformatics techniques over the last few years.

Bioinformatics could be regarded as an approach diametrically opposed to the traditional experimental ones. Instead of starting from a phenotype, one will start with a sequence and try to gather as much information as possible by comparing this sequence to others for which experimental evidence is available. But the difference between the two approaches is much less acute than it seems. Bioinformatics relies on the same basic assumptions as classic biology. It is a method of inquiry based on a series of comparisons that lead to classifications/predictions. The main paradigm of bioinformatics is that sequence conservation is correlated to function conservation. Under this framework, the aim is to extrapolate, as much as possible the information acquired experimentally. The process follows a traditional feed back scheme where models are built and validated or invalidated by experiments made in wet lab or in silico.

Darwinian laws of evolution, and the notion of parsimony often underlie the bioinformatics approach. The assumption is that biological systems have evolved from the same origin, constantly reusing some basic building blocks (such as metabolic pathways) and adapting them to respond to their environmental constraints. If each time a new constraint appeared, a new biological system was created from scratch, the bioinformatics approach would probably be bound to fail. Fortunately, in most of the cases, this is not what happens. Through the cycles of mutation/selection that constitute evolution, new functions have been created by reusing pieces of already existing machinery, and existing functions have evolved to become more adapted to the environment in which they are needed. If we consider this problem in terms of sequences, this means that two sequences responsible for similar functions may be different, depending on how long they have been diverging (i.e. how long ago the original sequence was duplicated, or how long ago the two organisms containing these sequences started diverging). Nevertheless, if the distance separating them is small enough, an evolutionary scenario can be reconstructed that will show how related these sequences are. Depending on what is known for one of these sequences (or for other sequences of the same category), it will then become possible to make assumptions about the function. On the other hand, if the sequences are evolutionarily too far apart, accurately analysing their relationship may prove difficult by simply comparing the sequences. The signal they contain may have to be enhanced using other techniques such as structure prediction.

Sequences are only conceptual objects. As such, they have no function in a cell. In fact, even the distinction between RNA, proteins and DNA is artificial. As far as the cell is concerned, all these elements only exist as complex 3D arrangement of atoms. It is because of its precise 3D structure that a molecule has the mechanical and chemical properties it needs to perform its function (catalytic activity, interactions...). The relation between structure and function is probably one of the oldest paradigms of molecular biology. We also accept that broadly speaking, structures are induced by sequence although we know for a fact that very different amino acid sequences can code for similar 3D folds.

This last point helps understand why proteins with different sequences can have similar function and structure, since natural selection is applied on the active 3D structures rather than on the sequences (i.e. evolution gives more freedom to the sequence than to the structure). As a consequence, relationships between proteins (or RNAs) are usually easier to analyse when the structures are known. Unfortunately, structures are difficult to determine experimentally and prediction from sequence alone (ab initio folding, threading) is still one of the main challenges of computational biology. It is true however that useful tools exist that can help supplement weak sequence identity.

Developing new techniques for automatically analysing sequences is one of the main purposes of research in bioinformatics. It is a point of crucial importance. Today, all the major databases of nucleotide or protein sequences are growing in size at an exponential rate (doubling every year or so). It means that the proportion of sequences for which experimental data are available is decreasing. For this reason, targeting the points at which experiments are needed has become more important than ever. Such a goal will only be achieved by gaining some more understanding on the ways in which information can be extrapolated from one sequence (or a set of sequences) to another. This is the only way available for making any use of the DNA sequencing results (at least in a reasonable amount of time). It is for this reason that sequence comparison tools are at the heart of the bioinformatics approach.

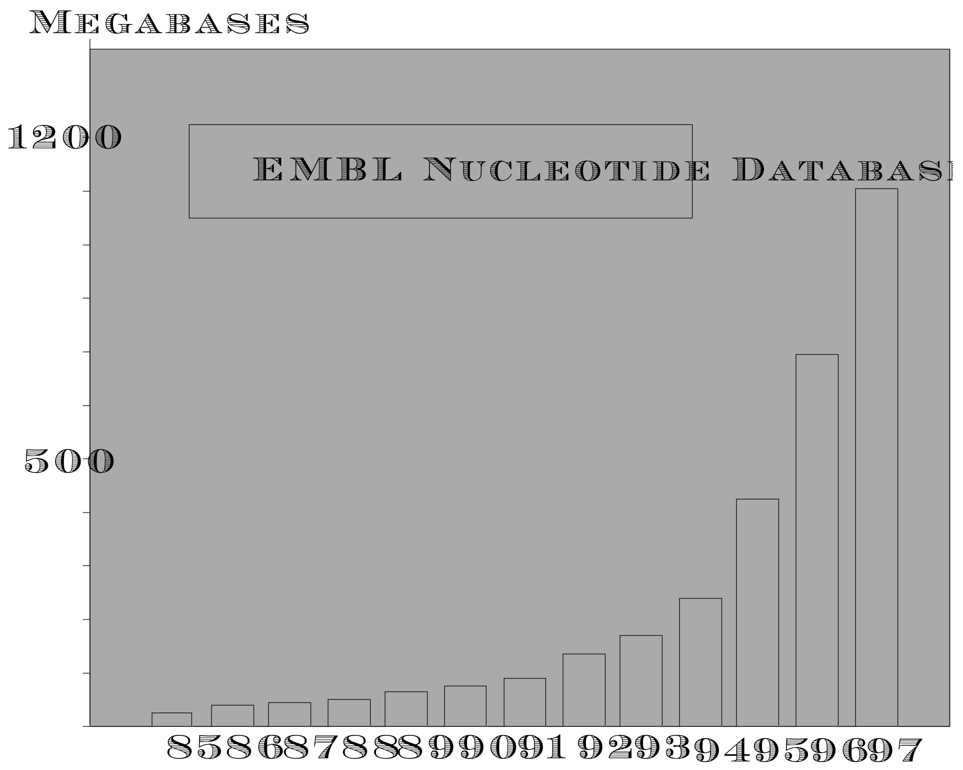


Figure 1. Growth of the EMBL Nucleotide database over the 1985−1997 period. The last release of the EMBL nucleotide sequence database (Rel. 52, October 1997) contained 1,181,167,498 nucleotides. The last release of the Swiss−Prot Database (Rel. 32 October 1996) contained 21 210 389 amino acids in 59021 entries. For comparison, the last PDB release (proteins with known structures) of December 1997 contained a total of 6731 entries.

1.2. COMPARING SEQUENCES

In most cases the problem facing the user takes the following form: a new sequence is available and it is desirable to search the database and find out whether one or more close relatives of this sequence have already been reported. If so, one may wish to extrapolate some of the experimental data gathered that way to the new sequence. In such a case, the solution is to compare the sequences of interest to all the sequences contained in the database, keeping track of the most similar. Two very popular tools are used to perform basic database similarity searches: FASTA(1) and BLAST(2).

Sequence comparison can also be much more complex. For instance, by combining experimental data contained in the databases and sequence analysis, one may want to know if a specific motif is sufficient for a protein to bind zinc. If several types of alternative motifs emerge when doing such analysis, one may want to build a classification. Further experimental data may also be available that allows the establishment of functional differences between the zinc−binding motifs (some may be associated with RNA−binding proteins and others with DNA−binding proteins). This for instance, is one of the ideas developed in the Prosite database(3). Once such results have been established, new sequences can be scanned for the known motifs they contain.

As simple and trivial as they may seem, such strategies present various very complex difficulties that need to be overcome. First of all, one has to define what ’comparing’ means. This is typically a biological problem. The features one is interested in when looking at two sequences or more will obviously depend on the aim and the scope of the comparison. Do we want to compare two proteins for having the same sequence, the same function, for having related functions, for being expressed in the same circumstances, for having similar folds? Are these questions equivalent? When it comes to making detailed analysis, sequence alignments appear as one of the most powerful solutions. In the simplest case, they only involve two sequences (pairwise alignments) but they can be extended to a larger number (multiple sequence alignments).

Having decided that we want to use sequence alignments, the problem of defining what is a ’good’ sequence alignment remains. It is a difficult question that requires a deep understanding of the biological information one wishes to extract from such alignments. In most of the cases, the criterion that will allow evaluation of the quality of an alignment will take the form of a mathematical function(objective function) associating a value to an alignment. But even so, the problem is not solved. Having a criterion for alignment quality is not enough. One also needs to be able to build the best scoring alignment according to this quality criterion. In most of the cases, this is far from easy. Many of the problems in bioinformatics and more specifically in sequence alignment are said to be NP complete. This means that the number of potential solutions rises exponentially with the number of sequences and their length (i.e. the solution cannot be found in polynomial time and space). The need to overcome such severe limitations requires the development of powerful algorithms.

1.3 OUR APPROACH

In the work presented here, the problem of sequence alignment was approached through the two aspects mentioned above:

- −defining new objective functions for sequence alignment.
- −developing new ways to optimize these functions.

One of the main concerns of the approach was the fact that there is no use in defining new sequence comparison schemes if no tool is available to use them and allow judgment to be made on their potential relevance. To test an objective function, one must be able to optimize it and compare the quality of the alignments it provides with other methods.

The new optimization scheme proposed here is a genetic algorithm named SAGA (Annex 1) for Sequence Alignment by Genetic Algorithm. This algorithm was used to evaluate the biological relevance of COFFEE (Consistency Based Objective Function for alignmEnt Evaluation) ,a new objective function designed for protein multiple sequence alignment(Annex 2). SAGA was also adapted to RNA alignments (RAGA, RNA Alignment by Genetic Algorithm) using an objective function that takes into account secondary structure interactions in RNA (Annex 3). In order to improve RAGA's objective function, a new function was designed for aligning an RNA sequence to a large multiple RNA sequence alignment (Annex 4). This function was only tested using a traditional optimization method (Dynamic Programming).

The following sections will deal with the three main concepts associated with sequence alignment: what it is useful for, how to define a sequence alignment and finally how to build a sequence alignment. The last section will put the four contributions in their relative context.

2−THE SCOPE OF SEQUENCE ALIGNMENTS

2.1 WHAT IS A SEQUENCE ALIGNMENT ?

A sequence alignment is the representation of two sequences in a way that reflects their relationship. If the alignment is correct, two residues aligned with one another are homologous. The definition of homology depends on the criterion used for the alignment. For instance if the aim is to identify the relationship between two structures, two residues will be aligned because they are equivalent in the 3D structures. If the alignment is designed to reflect phylogenetic relationships, two residues will be aligned when they originate from the same residue in the common ancestor. The definition of a pairwise alignment can be extended to multiple sequence alignments. In this case, several sequences are aligned together and each column contains homologous residues. However, homologous residues do not necessarily exist in each sequence for each position of the alignment. If a given sequence lacks one residue, a gap will be inserted in its place at the corresponding position. Gaps usually take the form of strings of nulls. In an evolutionary context, a null sign means that a residue was inserted in one of the sequences or deleted in the other while the sequences were diverging from their common ancestor.

There are two types of alignments: global and local. In a local alignment, the only portions that are aligned are those which are clearly homologous. The rest of the sequence is ignored. In a global alignment, whole sequences are aligned, regardless of the local level of similarity. The scope of global and local alignment is usually different. Local alignments are more appropriate when the sequences analysed are remotely related and may share only a few domains. Global alignments are mostly designed to analyse sequences that are known to be homologous to one another. In this thesis, I will mostly consider global sequence alignments.

It is also important to realize that, given a set of sequences, there are lots of alternative alignments. For instance, given two sequences of 1000 residues each. there are about 10^{764} different possible alignments. This rules out any naive enumeration strategy for identifying the correct one! Instead, we will see that several strategies have been developed that allow more or less efficient computation in polynomial time.

2.2 WHAT IS THE USE OF A SEQUENCE ALIGNMENT ?

Quantitatively, the most widely used application of sequence alignment is in database searching. When doing so, the aim is to find for a given query sequence, all the related sequences contained in a database. The principle is very straightforward. The query sequence is aligned in turn to all the members of the database and results are ranked according to some similarity criterion. FASTA(1) and BLAST (2)are the most popular tools of this type. They rely on local alignments rather than global.

However important the results obtained in this way, there is a clear limit on the quality of the alignments that can be deduced from these searches. Firstly, the sequence alignment algorithms implemented in these programs are only crude approximations of the standard sequence alignment algorithm. This is necessary in order to search very large databases in a reasonable amount of time. Secondly, in most of the cases, the searches are based on pairwise alignments. This means that they only contain a limited amount of information.

Although database searching is at the heart of many approaches, pushing the analysis further may require important refinements. Such refinements can involve structure prediction, identification of new motifs or domains, generalization of the family properties (i.e. combining the information contained in the known sequences in order to identify distant members), phylogenetic analysis. For all these applications, pairwise alignments are of limited use. A way to simultaneously combine the information contained in several sequences is needed. Such a need is the main motivation for building multiple sequence alignments.

Multiple alignments are very important for phylogenetic analyses because they provide a way to compute evolutionary distances and phylogenetic trees. Trees are computed from sets of pairwise distances using some clustering algorithms such as the neighbor joining method(4). When computing a tree, it is very important to have accurate pairwise distances hence the use of a multiple alignment in which the pairwise alignment of two sequences depends on the information contained in all the sequences of the set.

Another fundamental application of multiple sequence alignments is the identification of motifs or domains. In a multiple alignment, these elements often appear as portions on which constraints exist that limit divergence. If some of the sequences are experimentally characterized, these motifs can be used for function prediction. This is, for instance, one of the aims of the Prosite(3) or the ProDom databases(5). The information contained in a multiple sequence alignment can also be generalized in order to produce a profile(6)or a hidden Markov model (7)that can be used for identifying new family members.

The other important use of multiple alignments is structure prediction. In a given protein, residues do not evolve in the same fashion, depending on their role in the structure (buried/exposed, helix/beta strand/loop...). It is very hard to extract such information from a sequence alone while it can be accessed through analysis of multiple alignments by looking at the distribution of the substitutions. Using multiple sequence alignments instead of sequences alone has had a dramatic effect on this area of sequence analysis, boosting the accuracy of protein secondary structure predictions from 55%(8) (9)to 75%(10) accuracy. One can also go further and try to identify correlated mutations in multiple sequence alignments. This has been done on many occasions in proteins with limited success(11−13). On the contrary, in RNA analysis, the identification of correlated mutations has been of great help, allowing accurate prediction for secondary structure analysis and even tertiary structures(14−16).

Finally, a less challenging but very important application of multiple sequence alignments is the localization of highly conserved area for the design of efficient PCR primers, in order to clone new members of a family. All these examples reflect the importance of multiple sequence alignments in the domain of sequence analysis. We will show here that making good multiple sequence alignments is a multi−step task.

2.3 WHAT IS A 'GOOD' ALIGNMENT ?

A scoring function associates a score to an alignment. Ideally, the better the score, the more biologically accurate the alignment. An alignment with the best possible score is said to be optimal, whether it is biologically relevant or not. An optimal alignment always exists. Being able to distinguish between biologically relevant and non relevant alignments is an important issue, especially when analyzing databases. Powerful statistical tools have been developed for this purpose, allowing the discrimination of hits for their potential biological relevance(2). This problem remains when aligning sequences known to be related. For instance, two homologous domains may surround a loop that is different in the two structures. In this case, any alignment of the residues contained by these loops will be meaningless even if a mathematical optimum exists.

Another important problem, common to many areas of computational biology has to do with the choice of parameters. Most of the objective functions come with complex sets of parameters. In many cases, one has to rely on empirical values, known to lack robustness (i.e. small changes of the parameter values may induce very different alignments) which may lead to inaccurate alignments. This explains the fact that a large amount of the work dedicated to objective function definition focused on parameter elimination. Ironically, so much work has been done in this field that the choice of a scoring scheme can almost be regarded as one more parameter requiring optimization. Among the countless number of existing methods, we will only describe some of the most important schemes, focusing on those related to the work carried out for this thesis.

There are two types of alignments: sequence and structure alignments. Sequence alignments do not require any non local interactions to be taken into account and are therefore less complex (algorithmically speaking) than structural alignments. For the problem of sequence alignments we will mostly talk about protein sequences while the problem of structure alignment will be addressed through the example of RNA secondary structures. The problem of protein structure alignment will not be analysed in depth. Its complexity and the amount of literature available on this subject put it beyond the scope of this dissertation which is mostly oriented toward sequence analysis rather than structure.

2.4 HOW TO BUILD A ’GOOD’ SEQUENCE ALIGNMENT

In many cases, building a sequence alignment takes the form of a compromise between biological relevance, mathematical optimality and efficiency. Given an objective function, it may be very hard to produce the mathematically optimal alignment. We already mentioned earlier that done in a naive way through enumeration, sequence alignment is beyond the scope of any computer. For this reason, trade−offs need to be made on both side. Objective functions need to be defined in such a way that they fit already existing optimization techniques, and optimization techniques need to be improved in order to accommodate the complexity of the problem.

For instance, in its more general form, the problem of aligning two sequence is NP complete(17). However, if formulated under certain constraints, it can be solved with a technique known as dynamic programming(18) but becomes NP complete again with the number of sequences (i.e. when trying to align more than two sequences simultaneously)(19). Structure alignments (i.e. sequence alignments taking into account non local interactions) are also NP complete, even for two sequences, and can only be addressed in a simplified form(i.e. using an objective function that does not reflect all the known constraints)(20).

Because of this NP completeness most of the algorithms developed in the context of sequence alignments are heuristics(21−24). It means that they do not guarantee a mathematically optimal solution, but rather a good approximation. In many cases, this trade off is reasonable and allows the computation of multiple sequence alignments in an efficient manner. In this thesis, I will describe some of the optimization methods currently used, with a special emphasis on the genetic algorithms.

3 EVALUATING ALIGNMENTS

3.1 PROTEIN PAIRWISE ALIGNMENTS

3.1.1 Substitution Matrices

The twenty amino acids commonly found in proteins have very specific physico−chemical properties such as size, charge and hydrophobicity. The role of a residue in a protein mostly depends on these properties. For this reason, substitutions do not occur at random but in a way that reflects physico−chemical constraints in the 3D structure. It is therefore a very intuitive idea to try to associate with each possible substitution a cost depending on its probability. This information can be stored in what is known as a substitution matrix, a 20 by 20 table giving the relative cost or the probability of each possible amino acid substitution. Although many types of matrices have been proposed, the more successful are those derived empirically (25, 26). The principle often involves statistical analysis of a large set of alignments. Interestingly, these matrices tend to be in general agreement with what would be expected, knowing the physico−chemical properties of the residues (e.g. substitutions conserving charge, size or hydrophobicity have lower costs). We will review here the most popular of these matrices and their relative strengths/weaknesses.

The simplest possible substitution matrices are those only rewarding identities in an alignment. Considering their simplicity, they do remarkably well in a variety of cases, probably owing to the fact that they put a very drastic threshold on background noise, only allowing the identification of very strong signals(27). On the other hand, these matrices disregard a large part of the information and this proves a big disadvantage when doing pairwise alignments between sequences with a low level of identity but a clear homology. The need to show that two sequences with a low level of identity can still be significantly similar has been one of the main motivations behind the development of more complex substitution matrices that take into account similarity as well as identity.

The Dayhoff matrices(25), also known as PAM matrices, are among the most widely used. The principle on which they are built is quite straightforward. Alignments of closely related proteins are made (more than 85% identity). When such a high level of identity is shared by the sequences, alignments are usually straightforward and accurate. In these alignments, the frequency of each possible substitution is measured. The table of frequencies obtained in this way is turned into a probability model (log−odds matrix). This model can be used to define weight matrices, appropriate for comparing sequences of any degree of divergence The distance is measured in PAM, Point Accepted Mutations per 100 residues and one can have matrices from 1 PAM up to 500 PAM.

```
C 11.5
S  0.1  2.2
T −0.5  1.5  2.5
P −3.1  0.4  0.1  7.6
A  0.5  1.1  0.6  0.3  2.4
G −2.0  0.4 −1.1 −1.6  0.5  6.6
```

```
N −1.8  0.9  0.5 −0.9 −0.3  0.4  3.8
D −3.2  0.5  0.0 −0.7 −0.3  0.1  2.2  4.7
E −3.0  0.2 −0.1 −0.5  0.0 −0.8  0.9  2.7  3.6
Q −2.4  0.2  0.0 −0.2 −0.2 −1.0  0.7  0.9  1.7  2.7
H −1.3 −0.2 −0.3 −1.1 −0.8 −1.4  1.2  0.4  0.4  1.2  6.0
R −2.2 −0.2 −0.2 −0.9 −0.6 −1.0  0.3 −0.3  0.4  1.5  0.6  4.7
K −2.8  0.1  0.1 −0.6 −0.4 −1.1  0.8  0.5  1.2  1.5  0.6  2.7  3.2
M −0.9 −1.4 −0.6 −2.4 −0.7 −3.5 −2.2 −3.0 −2.0 −1.0 −1.3 −1.7 −1.4  4.3
I −1.1 −1.8 −0.6 −2.6 −0.8 −4.5 −2.8 −3.8 −2.7 −1.9 −2.2 −2.4 −2.1  2.5  4.0
L −1.5 −2.1 −1.3 −2.3 −1.2 −4.4 −3.0 −4.0 −2.8 −1.6 −1.9 −2.2 −2.1  2.8  2.8  4.0
V −0.0 −1.0  0.0 −1.8  0.1 −3.3 −2.2 −2.9 −1.9 −1.5 −2.0 −2.0 −1.7  1.6  3.1  1.8  3.4
F −0.8 −2.8 −2.2 −3.8 −2.3 −5.2 −3.1 −4.5 −3.9 −2.6 −0.1 −3.2 −3.3  1.6  1.0  2.0  0.1  7.0
Y −0.5 −1.9 −1.9 −3.1 −2.2 −4.0 −1.4 −2.8 −2.7 −1.7  2.2 −1.8 −2.1 −0.2 −0.7  0.0 −1.1  5.1
7.8
W −1.0 −3.3 −3.5 −5.0 −3.6 −4.0 −3.6 −5.2 −4.3 −2.7 −0.8 −1.6 −3.5 −1.0 −1.8 −0.7 −2.6 −3.6
4.1 14.2
...C    S    T    P    A    G    N    D    E    Q    H    R    K    M    I    L    V    F    Y
W
```

Figure 2. A log odds matrix computed from mutation data. This is a PAM 250 matrix, extrapolated from the original PAM 15. Each entry indicates the cost for aligning two residues. The worst substitution costs are usually associated with the tryptophan (W).

Originally, the Dayhoff matrices were established using 71 sets of aligned protein sequence pairs with 1572 point mutations (amino acid substitutions). The main limit of this approach is the fact that the content of information in alignments having 85% sequence identity is low. Therefore, it appears risky to extrapolate such limited information to large evolutionary distances such as PAM 250. Furthermore, the extrapolation of the PAM model to any PAM distance is based on the assumption that mutations are independent events. This hypothesis was challenged by several alternative methods.

The most popular alternative to PAM based scoring functions is the BLOSUM scheme (26). It is based on a library of blocks, extracted from sequences of related proteins. A block is a local multiple alignment that does not contain any gaps. About 2000 blocks were used for establishing the matrices. Given a set of substitution frequencies, BLOSUM and PAM matrices are computed in a similar fashion. The main difference is that in BLOSUM the frequencies are measured in a way that takes into account sequence identity. This way, using the collection of blocks, several sets of matrices can be generated without having to do any extrapolation. They range from 80 to 45 % average identity. BLOSUM matrices have been shown on various occasions to outperform PAM or other matrices(26−28).

The main reproach that can be made to these two types of matrices, is that they attempt to be general while only relying on alignments with a low information content (sequences more than 85% identical), or domains that can be aligned without gaps (blocks). The question of the existence of bias in these matrices has often been discussed. For instance, let us consider alpha helices and beta strands. The type of substitutions observed on these two types of structural elements are known to be slightly different (this is the basis of some efficient secondary structure prediction algorithms(29)). As a consequence, if a matrix is built using a data set that contains more helices than beta sheets, it will be biased and will perform poorly when aligning portions of sequences coding for the beta−sheets. On the other hand, if the data set is equilibrated for the two types of structures, the matrix will simply be an average, and will not be as good as it could be in either of the two cases. In an attempt to compensate for this type of potential problem, several alternative matrices were built for helices(30), beta strands(30) or transmembrane domains(31). The problem is that the way in which such matrices should be used is far from being obvious. Solving this problem often amounts to solving structure prediction, unless a structure is available for some of the homologous sequences in which one is interested.

Overall, about 40 different substitution matrices have been proposed, using no less than 20 different methods with different training sets in most cases. Recently, two generic studies have been made in an attempt to understand the fundamental differences between these schemes(27, 32). The main motivation in the work of Vogt et al. was to assess the behavior of these matrices for the accuracy of the alignments they induce using dynamic programming (See Section 4.2.1). Correctness was judged using the structural alignments contained in 3D_ali(33). Their work shows that there is very little difference between the best matrices (Gonnet(34), BLOSUM(26) and Benner(35)). They also concluded that matrices which are able to identify remote homologues in database searches were the ones leading to the more correct alignments. Finally, from an algorithmic point of view, (see 4.2.1) they concluded that these matrices are more suited for global alignments than local alignments. The second study(32) focused on understanding the way these matrices reflect amino acids properties. The authors found that PAM units are significantly correlated to volume and hydrophobicity while other matrices are much more biased toward identity. Interestingly their results indicate that despite the different methods and different data sets used for their construction, most of the substitution matrices based on sequence analysis are highly correlated with the Dayhoff's. When trying to group these matrices according to their level of correlation, the global matrices fall into the same cluster (i.e. are very similar), while structure−specific matrices fall into separate clusters. This is further evidence supporting the idea that matrices should be applied in a way that takes into account structural information. We will see later that the Dirichlet mixtures(36) provide an interesting alternative to this problem (see section 3.2.4).

The matrix comparison problem was also addressed in a different context, and with a smaller set of matrices, by Henikoff et al. who compared matrices for their ability to discriminate sequences in a database search using FASTA or BLAST (i.e. local alignments). The results obtained that way confirm that matrices based on the alignment of distantly related sequences (such as BLOSUM) or structures (37) perform much better than PAM matrices.

Finally, a point on which all these studies agree is the necessity of using appropriate gap penalties when scoring an alignment with a matrix. Most of the results obtained using substitution matrices can be dramatically affected, depending on the set of penalties used to score gaps. In the next section, we review some of the concepts underlying the definition and the scoring of gaps when aligning two sequences.

3.1.2 Gap penalties

Substitutions are not the only events affecting sequences while they diverge. Insertions and deletions also occur. This means that two sequences may contain unrelated portions that should not be aligned. Such an event is represented by a gap in the sequence that did not receive the insertion, or where the deletion occurred.

```
          Deletion                          Terminal gap
XXXXXXX−−−−−−−−−−−−−−−−−XXXXXXXXXX−−−−−−−−
XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX
          or Insertion
```

It is obvious that a cost should be given to these events when scoring an alignment. Unfortunately the choice of a scoring scheme often implies some evolutionary model, and we still lack a good understanding of the underlying biology of insertions/deletions. For instance, unless a very reliable phylogenetic scenario is available, it is often very hard to distinguish between insertion and deletion(38, 39), hence the word indel that describes a position where an insertion OR a deletion occurred.

Some useful information can be gathered about indels by looking at structural alignments(39). As one would expect, indels do not happen at random, but tend to concentrate on the portions of the structure with less steric constraints such as loops. It is in theory possible to extrapolate this information to sequences with unknown tertiary structure. For instance, Chou and Fassman(40) proposed using secondary structure propensity in order to derive some local gap penalties. Another method proposed by Pascarella and Argos (39) involves measuring, on reference structural alignments, the probability of having a gap after a given residue. This scoring scheme has been implemented in ClustalW(21) where a gap can be given a cost depending on the residue after which it occurs. Similarly, some heuristics have been implemented in ClustalW that attempt to locate areas more prone to gap insertion such as stretches of hydrophilic residues, usually exposed in the loops. However, these methods are both empirical and very general, they do not take into account the specificity of the sequences one is interested in aligning (for instance due to the structural constraints, some positions may be more conserved than others...). We will later discuss the way information derived from a multiple sequence alignment can be used in order to established more reliable local gap propensity (cf. the Profile section, 3.2.3).

The question "where do gaps occur?" is only one part of the problem. When scoring a gap, one also has to ask: "Is this gap long enough?". There is clear evidence that gap should be penalized according to their sizes. Analyses made on mammals(41) suggest that a logarithmic scheme could be quite appropriate, with a gap opening penalty and an extension penalty that is a function of the length of the gap (i.e. a penalty per residue decreasing with the length of the gap). These results confirm those obtained by Benner et al. (38) and suggest that linear gap penalties are less than realistic (penalty cost increasing linearly with the gap length). Results also suggest that insertion and deletion should be distinguished from one another(38, 41). However, even if there is a wide agreement on the fact that non−linear schemes would probably be biologically more relevant(42), alternative solutions suffer from a major drawback: their implementation poses significant algorithmic problems when it comes to optimizing alignments(43, 44). For this reason, in practice, gap penalties are usually optimized under a simplified form known as "affine gap penalty". It can be formalised as follow:

cost = Gap Opening Penalty + Gap Extension Penalty * Length
(eq. 1)

This gives penalties as a linear function of gap length and an efficient algorithm for optimizing this was introduced by Gotoh(45). There is no real justification for using this type of model apart from the fact that it performs reasonably well, especially when the gaps are small (less than 20 residues). Since the size of indels is known to be a function of the evolutionary distance(38), this means that linear gaps will be acceptable when aligning closely related sequences. However, since affine gap penalties are not empirically based, they raise the problem of defining the values of the two parameters: the gap opening penalty (GOP) and the gap extension penalty (GEP). There is no guaranteed way to choose these values so that they fit the sequences one is interested in. A popular practice is to give to the opening penalty a value equal to the average of the values contained in the substitution matrix used for comparison (excluding the main diagonal). The extension penalty is then set to a tenth of the opening value. It is also common practice not to penalize terminal gaps, at least for opening.

When making an alignment there is a competition between gap insertion and substitution. To some extent, gap penalties can be regarded as thresholds used to decide whether a stretch of residues has a homologue or not in the other sequences. In this context, it makes sense to modulate the penalties with some local information (secondary structure propensity, profile information....). But even so, a major problem remains: when the sequences aligned are only remotely related, the gap penalties lack robustness. A study made by Vingron and Waterman showed that slight variations of values for the GOP and the GEP can induce very different optimal alignments(46). Under such conditions, it may be hard to decide which alignment is biologically the most relevant. An attempt to increase the robustness of the penalty parameters has been proposed by Taylor: "Score Run−Length Enhancement"(47). It originates from the observation that in biologically relevant alignments, gaps are usually clustered in a few parts of the sequences and separated by long uninterrupted blocks. The technique proposed by Taylor involves enhancing the score of long ungapped portions in order to avoid them being interrupted by gaps. This work shows that under this scoring scheme, a correct guess for the penalty values becomes much less critical than previously reported.

There is little doubt that the correct treatment of gaps and a deep understanding of their biology are critical parameters when making accurate sequence alignments. However, as shown here, the problem is mostly algorithmic. It is possible to define gap penalties that describe sequence relations in a realistic way, it is simply not practical to optimize them. The problem of practicality becomes even more acute when it comes to scanning databases containing hundreds of thousands of sequences.

3.1.3 Database Searches

An exhaustive treatment of database search methods is beyond the scope of this thesis. The most commonly used methods will be briefly described here as respect to the fact that they involve specific scoring schemes designed for evaluating the statistical significance of an alignment. The principle of a database search is quite straightforward. A query sequence is aligned in turn with all the other sequences of the database. Depending on their score, alignments are kept as relevant or discarded.

BLAST(2) is probably one of the most popular methods for database searches. Given two proteins, the method involves finding the High Scoring Pairs of residues (i.e. short stretches of aligned residues, with no gap, which have high scores). The scores are evaluated using a substitution matrix (PAM120 for instance). These segments are found by looking for words of a specified size(48), and by extending these words. Since the method does not allow gaps, it will in many cases be restricted to more or less small segments. In such a context, the score alone will not be informative enough to decide whether or not a high scoring pair is significant. These scores will need to be normalized in order to become comparable from a statistical point of view. This normalization takes into account the size of the database and the size of the sequences and gives the probability that the match arose by chance. This score is called the E value and is used to rank the hits.

The other popular tool for database searches is FASTA(49). As in BLAST, FASTA starts by looking for high scoring segments using the Wilbur and Lipman method(48). The segments the method is interested in are those having a high proportion of identities. They are scored by using the main diagonal of a substitution matrix such as PAM 250 (i.e. only considering identities, but giving them a score that depends on the amino acids). The ten best diagonals found that way are then re−scored using a full substitution matrix. In a second step, non collinear segments are joined by dynamic programming, using a segment joining penalty (analogous to a gap penalty). The resulting scores are then measured and used to rank the matches. In order to prevent this chaining step from decreasing selectivity, it is only applied when the best scoring segment is above some empirical threshold. The mean and the standard deviation of this distribution are then evaluated and used to decide on a final threshold that is used to separate spurious hits from real ones.

Of course, both these methods lead to false negatives and false positives, partly because they do not use the most accurate method for local alignments (50) that have been shown to significantly outperform FASTA(51), and also because some background noise is difficult to avoid especially when dealing with very large databases. The reason why the statistics behind FASTA are less evolved than for BLAST has to do with the fact that assessing the statistical significance of a gapped alignment is much harder than for an ungapped segment as in BLAST. This may change soon. Vingron and Waterman have recently proposed a scheme that allows the estimation of probabilities for gapped alignments(52). Furthermore, very recently, a new version of BLAST, gBLAST(53), has been published, that incorporates some of these results and allows statistical ranking of gapped alignments when scanning a database. Other statistical scoring systems include the one described by Bucher(54).

3.2 EVALUATING PROTEIN MULTIPLE SEQUENCE ALIGNMENT

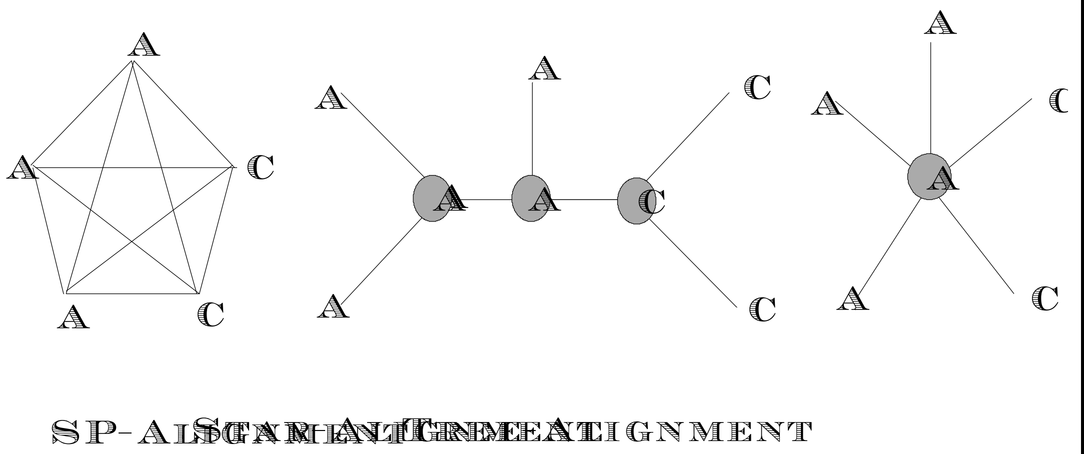
3.2.1 Using Substitution Matrices and Gap Penalties: SP Alignments

Extending the definition of pairwise costs to multiple sequence alignments can become complicated. To keep within the framework used for pairwise alignments, we seek a multiple alignment cost which is the sum of the substitution costs (cost based on substitution matrices and gap penalties). Nonetheless, based on different evolutionary scenario.

Because genetic mutations are binary events which change one protein or nucleic acid sequence into another, substitution costs for multiple alignments are generally defined in terms of those for pairwise alignments(42). Two different approaches have been described. The first is to define the substitution cost for a set of elements as the sum of the substitution cost for all pairs of elements chosen from the set(55−57). Thus for three sequences i, j and k the cost of the alignment i, j, k will be equal to the sum of the cost of each pairwise alignment induced by the multiple alignment (cost (i,j)+cost(i,k)+cost (j,k)). These induced pairwise alignments are also called pairwise projections. An alignment defined that way is called SP for "sum−of−pairs" alignment. In such a context the evaluation of a multiple alignment amounts to measuring the dissimilarities within a set of letters. Although this approach has the advantage of being straightforward and intuitive, its main disadvantage is that it has no clear foundation in the theory of molecular evolution. Sankoff(58) proposed an approach in closer agreement with biological intuition. In his model, an evolutionary tree is assumed where each sequence is a leaf. The nodes are occupied by reconstructed sequences. If the tree has k nodes, then substitution costs are defined on k−tuples and equal the sum of the pairwise substitution costs associated with each edge of the tree. An alignment defined that way is a tree alignment. It must not be confused with progressive alignments (see Section 4.2.2) which often rely on estimated phylogenetic trees for the computation of an approximated SP alignment.

In the cases where the tree alignment has only one central node, it is named star alignment, being based on star phylogeny.

Despite the fact that tree alignments are biologically more realistic than SP alignments they have not become very popular, mostly because of the fact that their construction presents serious algorithmic difficulties. A majority of the multiple alignment methods based on pairwise substitutions attempt to produce optimal SP alignments. In an SP context, defining gap costs is not necessarily straightforward, as can be gathered from the variety of alternative schemes (55, 56, 59, 60).



Sequences			
	A	A	A
	A	A	A
	A	A	A
	C	C	C
	C	C	C
Reconstructed Sequences (Nodes)			
	/	A	A
	/	A	
	/	C	
Cost			
	6	1	2

Figure 3. SP, Tree and Star alignment substitution costs for five 1 letter sequences (from Altschul, (42)). The reconstructed sequences are indicated by circles at the tree nodes. Dashed lines indicate substitution cost of 0 while plain lines indicate a cost of 1.

The simple implementation of pairwise costs in SP evaluation is known as 'natural gap costs'. The idea is to consider the costs of the gaps in each pairwise projection of a multiple alignment (any column of nulls is removed from the pairwise projection). Although these gap costs seem to be the obvious companions of an SP alignment, Altschul was the first one to formally propose them(42). Most of the alternative schemes have been introduced for aligning simultaneously three sequences. For instance, Gotoh(56) proposed to define a gap as a set of columns having a null in identical positions. For Murata(55), a gap is a set of adjacent columns, each containing at least a null. The main motivation behind these schemes was algorithmic. These approximations were made in order to make gap cost evaluation easier when computing an SP alignment. It is for this same reason that Altschul proposed a simplified version of the natural gap cost named ’quasi natural gap costs’(42).

Quasi natural gap costs are very similar to natural ones. The main difference is that when a pairwise projection is considered, columns of nulls are not removed, and an extra gap is counted as opening when a null run in one sequence starts after and finishes before a null run in the other sequence, such as:

```
Sequence 1 XXXXXX−−−−−−−−−−XXXXXXX
Sequence 2 XXXXXXXX−−−−−XXXXXXXXXX
```

that will lead to considering two gaps when in practice there is only one between the two sequences. The motivations behind this approximation are purely algorithmic and have to do with efficiency requirements when implementing the Carillo and Lipman algorithm for multiple sequence alignment(57) (see Section 4.3.1). This scheme induces a bias that favors similar gaps in aligned sequences, Nonetheless, this approximation is reasonable because indels seem to be rare events that tend to be kept unchanged through evolution(38). As a consequence, in most multiple alignments, the number of times where the quasi natural scheme induces alignments different from the natural one should be fairly limited. Natural and semi−natural gap costs can also be applied to tree or star alignments.

This type of gap penalties in the context of an SP alignment constitutes one of the most widely used objective function for multiple sequence alignments. It is often referred to as "sums of pairs with affine gap penalties". Some of the drawbacks of this method have already been discussed in the previous section. The main ones stem from the fact that substitution matrices are general descriptions that do not take into account local constraints, this is true as well of gap penalties that should incorporate local information. An attempt to do so has been made in ClustalW where penalties are locally reassessed, trying to use some information from the other sequences being aligned(21). Another weakness of the SP function has to do with the fact that sequences are considered in pairs only. It probably makes more sense to consider a column in a multiple alignment as a distribution of amino acids generated by evolution. In the next sections about hidden Markov models and profiles (Sections 3.2.3 and 3.2.4), we will present some methods that attempt to take this fact into account when scoring multiple sequence alignments.

Finally, a potential weakness inherent in any scoring scheme is the problem of non−representative information. The sequences used for building an alignment rarely constitute a representative set. They are often biased by the composition of the database used for collecting these sequences. In such a case, the alignment of the sequences that are in an isolated minority will suffer from the fact that the information they contain may be buried among the rest. We will see that several weighting schemes have been designed in order to overcome this problem.

3.2.2 Weights

With most of the multiple sequence scoring systems, weighting of the sequences is necessary. Weights are designed in order to correct for unequal representation among a set of sequences. For instance, when aligning together globin sequences found by querying a database like Swiss−Prot(61), large numbers of sequences will be identical, while others will be quite different from the rest of the set and will have no close relative. However, if we want our alignment to be representative of the globin family in general, it is important to avoid a complete domination by the vertebrate myoglobin and hemoglobin sequences, simply because the database contains far more of them. Such an alignment, made by giving each sequence the same weight would be biased. Weights are used to avoid this type of bias. Several methods have been proposed that can be separated into two groups whether they depend on an alignment or an estimated phylogenetic tree.

Alignment based weighting methods do not require the sequences to be related at all. Therefore, complex issues of tree topology and root placement are avoided. These methods can be based on pairwise distances (62) or on the distances from some average generalized sequence (63). Whatever the method, the general trend is similar and results in an up−weighting of the sequences which are poorly related to the rest of the set while sequences which, on average, are more similar to the other sequences have their weights accordingly lowered.

The tree−based weights assume that sequences are related through evolution and that a reasonably correct tree can be deduced from pairwise distances (64). Two schemes of this type have been proposed: branch−length proportional weights (65).and the Altschul Carol Lipman (ACL) weights that are based on a statistical analysis of the tree topology(66).

In the ACL scheme, a sequence receives a low weight if it is far from the tree root or if it has close neighbors in the tree. ACL weights have the advantage of correcting duplicated information without biasing the alignment toward very divergent sequences. The underlying assumption is that although a very divergent sequence contains a lots of information, it is hard to exploit such information without bringing in too much extra noise. The main weakness of the ACL method is that when the topology of the tree is hard to establish, mistakes can be made regarding the position of the root. The weights proposed by Thompson et al. (65) provide an alternative solution. They also rely on a phylogenetic tree, but under the scheme, sequences only get down weighted for having closely related neighbors.

In an SP context, applying pairwise or sequence weights to score a multiple alignment is straightforward. The weighted sums of pairs alignment score can be formulated as follows. Given N sequences $S_i..S_n$ a weight($W_{i,j}$) can be estimated for each possible pair of sequence S_i, S_j. This pairwise weight will be obtained directly through computation, or will result from the combination of individual sequence weights. Each pairwise projection $A_{i,j}$ of the sequences Si and Sj in the alignment has a cost (COST ($A_{i,j}$))evaluated using a substitution matrix and a set of gap penalties. Given these definitions, the global weighted SP score is equal to:

$$\text{SCORE} = \sum_{i=1}^{N-1} \sum_{j=i+1}^{N} W_{i,j} * \text{COST}\,(A_{i,j}) \qquad \text{(eq. 2)}$$

As with matrices, an important issue for weights is to decide which scheme should be applied. Each of these weights have desirable properties and unwanted side effects. Vingron and Sibbald proposed a systematic way of comparing five different methods(62). Their conclusion was that when sequences are related through a robust phylogenetic tree, the ACL do better than alignment based methods. On the other hand, when the relationship between the sequences is harder to estimate, leading to an inaccurate phylogenetic tree, the Sibbald and Argos(63) or related methods(67) are preferable. Similar results were more recently established by Henikoff and Henikoff(68) using an empirical evaluation method. These authors found that for phylogenetically related sequences, tree based methods are preferable and that the Thompson’s scheme slightly outperforms ACL. It is not clear however, to what extent these findings are method dependent, especially if one considers that most of these weights are used in different heuristic alignment making strategies.

Gotoh pointed out the fact that weighting schemes are very likely to be method dependent(69). For instance, an important difference between the ACL weights and the Thompson’s ones is that the ACL method produces pairwise weights while the other gives individual sequence weights. As a consequence, the ACL weights contain more information since the pairwise weights they depend on are not necessarily correlated (i.e. there is not always a set of sequence weights corresponding to a set of pairwise weights). On the other hand, as we will see later, these two types of weights are used in very different contexts. Thompson’s weights are mostly used for progressive alignments, where they are probably very appropriate since remotely related sequences usually have little effect on the overall multiple alignment(21). ACL weights are used in the program MSA (Multiple Sequence Alignment(22)) which does global simultaneous alignments where each sequence is given a chance to affect the overall alignment.

We will now see that weights can also be useful for the construction and the use of sequence profiles (i.e. generalized alignments used to describe a protein family or a domain).

3.2.3 Profiles

Multiple sequence alignments can be used to provide position specific scoring matrices known as profiles(6). The procedure of turning an alignment into a profile is fairly straightforward. It involves counting the residue frequencies in each column of the multiple alignment, and deducing from these measures, a table of substitution costs for each position of the profile. A local cost is also evaluated for gap insertion and extension. The term profile refers to the collection of costs associated with each position of the alignment. A profile can be treated as a single sequence and aligned to any other sequence (or profile), using the profile substitution costs and penalties instead of a single matrix.

POS	ALN	A	C	D	E	F	G	H	I	K	L	M	P	Q	...	Gap Penalty
1	EGVL	3	−2	3	4	0	4	−1	3	−1	4	4	1	1	...	9
2	LLSP	2	−2	−2	−1	3	0	−1	3	−1	6	5	−1	3	...	9
3	VVVV	2	2	−2	−2	2	2	−3	11	−2	1	−2	−2	0	...	9
.		.	.	.	.	.	.	.	.	.	.	.	.	.		.
.		.	.	.	.	.	.	.	.	.	.	.	.	.		.
.		.	.	.	.	.	.	.	.	.	.	.	.	.		.
21	SS−D	3	2	5	4	−4	5	0	−1	2	−3	−2	4	3	...	4
22	S−−S	2	3	1	1	−2	3	−1	0	1	−2	−1	2	2	...	4
.		.	.	.	.	.	.	.	.	.	.	.	.	.		.
.		.	.	.	.	.	.	.	.	.	.	.	.	.		.
.		.	.	.	.	.	.	.	.	.	.	.	.	.		.
49	SSNY	2	5	2	1	1	2	1	0	1	−2	−2	5	1	...	9

Figure 4. Example of profile (adapted from Gribskov et al.(6)). For each position (POS) of a multiple alignment (ALN, presented in a vertical format with each line corresponding to a column) a substitution cost is calculated for any amino acid that would be aligned to this position. A gap penalty is also evaluated. Note that at positions 21 and 22 of the profile, the

gap penalty is lowered because the alignment used for the profile contains gaps at this position.

A profile is specific for a family (or a domain). One of the main usage of profiles is to search databases for new members of a family. In such a context the desirable properties are sensitivity and selectivity (i.e. recognize very remote homologues and discriminate false positives). Such a result can only be achieved if the profile induces very good alignments. This in turns depends on the quality of the profile itself and can be affected by several factors: (i) choice of the sequences, (ii) method used for building the multiple alignment, (iii) method used to turn the multiple alignment into a profile (especially the treatment of the gaps and the method used to describe background frequencies).

In many cases, the choice of the sequences is directly imposed by the database and one of the best way to remove this type of bias is to use a weighting scheme when aligning the sequences (see previous section). However if one wishes to build some very specific profile, it is also possible to select the appropriate sequences, using techniques such as the one described by Neuwald et al. (70). Weights also need to be applied when the alignment is turned into a profile. On various occasions, it has been shown that many of the schemes used for sequence alignments do as well when used to build profiles, and help to increase the level of generalization(65, 71). Accumulation of gaps is another side effect that appears when many sequences are used to build a profile. Because the number of gaps tends to increase with the number of sequences, schemes have to be used for down weighting the effect of their occurrence(65, 72, 73).

Profiles are not only important for database searches, their computation is also a crucial step for some multiple sequence alignment strategies based on a progressive approach (67, 74, 75). In this context, to build the full multiple sequence alignment, partial multiple alignments need to be aligned in intermediate steps. The best way to do so is often to turn these alignments into profiles, and to align these profiles with one another. Methods for doing so have been extensively described by Higgins(76, 77) and Gotoh(69, 73).

Finally, since profiles are involved in database searches, some significant work has been done on establishing the statistical meaning of alignment scores(78, 79). This important aspect of the problem has received much more attention in the context of the hidden Markov models based approaches that will now briefly be reviewed.

3.2.4 hidden Markov models

A hidden Markov model (HMM) describes a series of observations generated by a "hidden" stochastic process (a Markov process)(80). They have been used extensively in speech recognition. HMMs designed for sequence alignments are related to profiles. Their aim is to provide a statistical model representative of a given family of proteins(7). In theory, one of the main advantages of HMMs (as opposed to profiles) is that they provide a way to estimate a model directly from unaligned sequences. However, in practice, most of the methods available for HMM optimization require the computation of multiple alignments. Nevertheless, HMMs have some interesting features that distinguish them from profiles.

A key concept in HMMs is the notion of states. A HMM is a chain of elements with different possible states. The number of possible states is arbitrarily defined. In SAM(81) for instance, there are three states: align, insert and delete. When going through a model, probabilities are given to each possible transition. The values of these transition probabilities are evaluated by training the model using known members of a protein family. Sequences can be aligned to a trained model using a variant of dynamic programming(18) known as the Viterby algorithm(80). An alignment between a sequence and a HMM is called a path, in the sense that it joins different states in order to produce the path with the highest probability. To a large extent, aligning a sequence to a model can be regarded as equivalent to aligning a sequence to a profile. There is however a fundamental conceptual difference. A new sequence is not 'aligned' to a HMM. What is measured is the probability for a given HMM to generate the optimally aligned sequence (i.e. the sequence with the right pattern of gaps/unaligned residues/aligned residues).

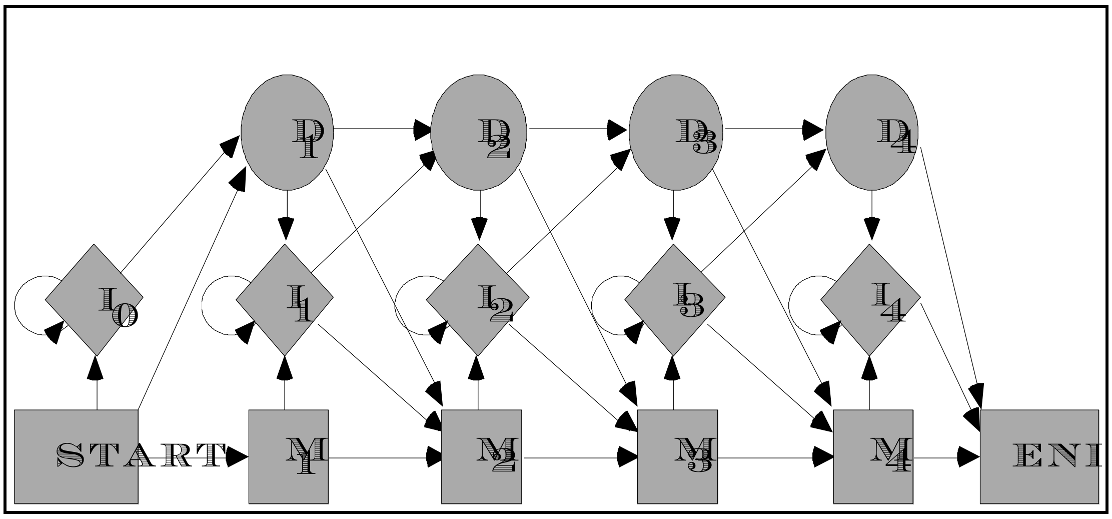


Figure 5. A linear hidden Markov model (from Hughey and Krogh, (82). This model has three different states (M, I, D). Each state is connected to the others by a transition probability (arrows). Assigning the weights to each transition is the purpose of the training.

The number of sequences, and their range of identities are critical factors that will influence the model. In their simplest expression, HMMs do not require any prior information (as opposed to profiles that required a multiple alignment made using a substitution matrix). In HMMs, residues are described as 'letters' and the training only relies on identities to establish the parameters. If there are enough sequences in the training set, this will result in a sensible model since the substitution constraints will be 'discovered' by the model on a position per position basis. The accuracy and the sensitivity of a model trained that way will be highly dependent on the number of sequences and their information content. To overcome possible lack of information (missing data), pseudo counts are incorporated in order to simulate background frequencies (82, 83). Their influence on the model decreases with the amount of information present in the sequences.

The actual values of the pseudocounts can be estimated using various methods. One can measure the probability of each amino acid in the training set, or other background probabilities from a standard substitution matrix. Generally speaking, the smaller the number of sequences used for the training, the more critical the values of the pseudocounts. In this context, Dirichlet mixtures(36) proved extremely useful. A Dirichlet mixture is a mathematical tool that, given an observed amino acid distribution and a set of reference distributions allows the computation of a probability for the observed distribution. In a hidden Markov model context, these mixtures can be regarded as the equivalent of a substitution matrix. They have been shown to be more sensitive to sequence conservation or variation than traditional substitution matrices(36).

As with profiles, HMMs can be used to generate multiple sequence alignments or for scanning databases. Scoring can be made by combining the probabilities of all the different alignments of a sequence to a model, which is equivalent to calculating the total probability of a sequence given the model. This can be done efficiently using the Viterby algorithm (80). Such a score is called the NLL score for Negative Log Likelihood score. The NLL scores measures how far a sequence is from its model (in other words, the statistical cost of forcing a given model to produce an aligned sequence). The problem with NLL score is that they depend on the size of both the sequence and the model. One way to overcome this is to measure the Z score, the number of standard deviations a NLL score is away from the average NLL score of unrelated sequences of the same length. A complete algorithm for the computation of Z scores is described in (82).

This study of multiple sequence alignment scoring systems is far from exhaustive. A large variety of alternative methods have been described that roughly fall into two distinct categories: those relying on SP evaluation and those (like HMMs) that consider distributions of amino acids rather than pairs. Although it seems that considering distributions is a more realistic approach, the main reason why SP schemes have so far been more popular has mostly to do with algorithmic problems associated with distribution based methods.

Both methods only deal with one aspect of the problem: the use of local information. We know that since proteins fold into active 3D structures, there must be more information in the sequences (i.e. tertiary structure interactions...) than what we discussed so far. If used in an appropriate manner there is no doubt it could help to improve the quality of the alignments. Few methods have been proposed to deal with these non−local interactions. There are two good reasons for that: this type of signal is usually very weak in proteins and the algorithmic problem is even more complex than when taking into account primary sequences. We will now see that the problem is different with RNA hence the use of more complex types of objective functions.

3.3 RNA ALIGNMENTS: TAKING INTO ACCOUNT NON LOCAL INTERACTIONS

When reliable non local interaction information is available, it makes sense to incorporate it into the scoring scheme. This is rarely the case with proteins, hence the difficulties encountered when doing structure threading (threading a protein sequence onto a known structure)(84). Fortunately, in the case of RNA the problem is different and the rules that govern the formation of non local interactions are better understood(85, 86). A large part of the secondary structures encountered in RNAs are due to Watson and Crick interactions. The existence of mathematical models that describe these interactions on a thermodynamic basis make RNAs good candidates for ’ab−initio’ folding predictions. However, so many parameters influence the structure of an RNA molecule (local conditions, interacting proteins, unknown tertiary interactions...) that accurate predictions based on thermodynamic models are still very imperfect despite some recent improvements(87, 88). This does not mean that the thermodynamic approach is wrong, but it may mean that the information it uses is not sufficiently detailed considering our knowledge of the RNA folding process. However when combined with phylogenetic information (such as can be taken from multiple sequence alignments) predictions can be realized to a high level of accuracy(16).

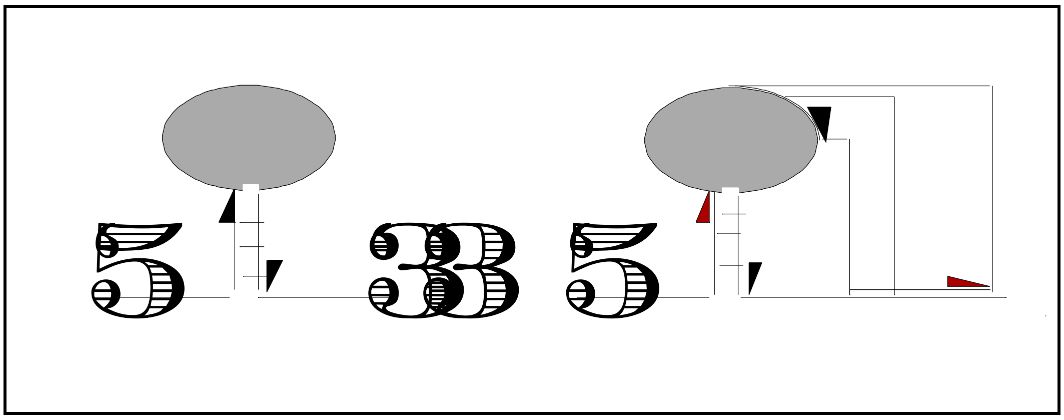


Figure 6. Some of the motifs commonly found in RNA secondary structure. Base pairings are usually made through Watson and Crick interaction although non canonical pairs have often been reported.

Multiple sequence alignments have the advantage of revealing constraints without the need of hypotheses on their origin. Such analysis, performed on ribosomal RNA sequences(16), made it possible to predict the secondary structures of these molecules with an accuracy outperforming traditional energy minimization schemes like Zuker(89). Unfortunately, these types of RNA alignments are difficult to build automatically due to complex algorithmic problems.

Several functions have been proposed that incorporate RNA secondary structure information into their evaluation of multiple sequence alignment quality(90−92). In their approach, Kim et al. (90) proposed a function that computes the probability for all the potential secondary structures contained in a multiple sequence alignment to be taken into account. It is a scheme that has the advantage of being very flexible (for instance, it allows pseudo−knots) and not requiring the actual computation of a secondary structure (structures are not assessed for compatibility). Its main drawback is that evaluation time is quadratic with the length of the alignment. This can prove a severe limit when dealing with very long sequences (like ribosomal RNAs) while using an iterative or stochastic optimization method such as simulated annealing. Eddy and Durbin(91) proposed a different type of approach. In their method, the secondary structure is expressed as a binary tree in which each node stands for a column in a multiple sequence alignment. This tree can be seen as a path through a generalized HMM (HMM with bifurcation) named Covariance Model (CV). A CV can be trained like an HMM. Once this has been done, the sequences are then aligned to the model in order to produce the multiple sequence alignment. This approach is very similar to the stochastic context free grammar (SCFG) methods(93, 94), where the aim is to express the structure using a special type of regular expression. The alignment is made by parsing the sequences through the proper expression that has been obtained by training on the sequences. CV and SCFG methods suffer from the same drawback. They can only allow nested structures and cannot take into account pseudoknots(95, 96) as opposed to the method proposed by Kim. Furthermore, their computation is very expensive and restricts these methods to small sequences (<200 nucleotides). An option for decreasing the complexity of the problem is to do threading: assume that some master structure is known and thread the sequences onto it. In several important cases, like ribosomal RNAs, this is a realistic assumption.

This approach can of course be taken using a SCFG based objective function. Alternatively, one can consider a simpler function such as the one described by Corpet and Michot(92). In this case, the evaluation of the alignment is split in two steps: (i) evaluating the primary sequence alignment (ii) evaluating the quality of the fold induced by the sequence of known structure onto the sequence of unknown structure. To generate the overall score, these two terms are combined with one another. Although this scheme has no real theoretical justification as opposed to those previously described, it has the merit of being conceptually simple. It can also accommodate a range of interactions such as pseudo−knots and other non nested structures. In this case, the alignment problem is known to be NP−complete(17). It is in order to provide a reasonable heuristic solution with long sequences (>1000 nucleotides) that we developed the package RAGA(97) (See section 5.3 and Annex 3).

4 MAKING MULTIPLE SEQUENCE ALIGNMENT

4.1 COMPLEXITY OF THE PROBLEM

So far, we have focused on reviewing some of the objective functions needed for evaluating the quality of sequence alignments. However, as pointed out earlier, this is only one side of the coin. The other one, that constitutes in fact the main bottleneck, is optimization. In other words, given an objective function, is it possible to optimize it by producing the best scoring alignment? There are at least two good reason for designing efficient and accurate optimization strategies. The first one is obvious: making the alignments that are needed for whatever purpose.... The second reason is less direct but of extreme importance. The evaluation schemes described above are only theoretically justified using phylogenetic or structural. criteria. As such, they do not constitute any proof and must therefore be validated through empirical analysis (i.e. how well do they perform).

The optimization methods required for these two reasons do not necessarily need to be equivalent. One can, for instance, use a very robust but expensive (computer time and memory) method to compare and validate alternative scoring schemes. If one of these schemes proves useful it may later become appropriate to develop a very specific heuristic method that approximates the optimization reasonably well while being efficient enough for production purpose. Needless to say, whatever direction one wishes to take, the design of an optimization technique will always prove to be a very demanding problem.

We already mentioned that even for two sequences of moderate length, a naive approach to alignment computation can lead to impractical enumeration problems. Fortunately, the situation does not have to be that bad and will depend on the scoring scheme one wishes to optimize. In many cases, there are short cuts that allow efficient computation of an alignment, given some specific objective functions. We will see in section 4.2.1 that dynamic programming(18) is one of these techniques that allows the computation of pairwise alignments in time proportional to the product of the length of the two sequences. This essential technique constitutes the core of many alignment methods. In theory, it is not restricted to two sequences, but since its complexity is a function of the product of the length of the sequences to align, it can hardly be used for more than three sequences at a time(55). This does not mean multiple sequence alignments cannot be computed automatically with dynamic programming, but it means that to do so, one has to rely on heuristic algorithms.

Heuristic methods do not guarantee an optimal solution but may perform well and sometimes even guarantee the solution to be inside given boundaries. Generally speaking, multiple sequence alignment algorithms can be divided into two classes: (i) the greedy algorithms that usually rely on sequence clustering algorithms and dynamic programming for making progressive alignments (ii)the non−progressive algorithms that attempt to simultaneously align all the sequences. These non progressive algorithms themselves fall into two distinct sub−categories:

−deterministic heuristics.
−stochastic heuristics.

In the following sections, the underlying principles of these algorithms and their main differences will be briefly explained. More emphasis will be given to the genetic algorithm techniques on which the SAGA package is based(98)(see section 5.1 and annex 1).

4.2 DETERMINISTIC GREEDY APPROACHES

4.2.1 Aligning Two Sequences

The main algorithm for aligning two sequences, often referred to as the Needleman and Wunsch(18) or dynamic programming (DP) algorithm, is one of the oldest and most important tools in bioinformatics. Over the last 30 years, it has been used under in form or another in most of the methods developed for sequence comparison.

When applying dynamic programming to two sequences, it is possible to compute the best scoring alignment between these sequences using an amount of memory and time proportional to the product of the lengths of the two sequences to align. This provides a dramatic improvement over the naive approach that would require enumerating all the possible alignments. An important advantage of dynamic programming is that it is a very general scheme. Given substitution costs (e.g. matrix or profile...) and a scheme for scoring gaps, the algorithm can compute the alignment with the best score. In practice, DP can accommodate any context−independent scoring scheme.

The algorithm is based on extending recursively the best scoring alignment until all the residues of each sequence have been aligned. In practice, this means finding the best path through a matrix constructed from the scores of all pairs of elements between the two sequences. Let us consider two sequences, A of length m and B of length n, a matrix that assigns a score $d_{i,j}$ to the substitution of residue i in A by the residue j in B and a gap penalty g. Computation of the optimal score is achieved by incrementally extending each path with a locally optimal step. For example; element $d_{i,j}$ can extend any path terminating in the preceding row (i−1, m:m<j) and/or the preceding column (n, j−1:n−1), less any gap penalty. If the extension that produces the highest new path score is always taken, then applying the condition repeatedly (to every element in the sequence matrix), transforms a sequence matrix of pairwise match−scores into a sequence matrix of path−scores (S). This can be expressed more formally by the recursive expression:

$$S_{ij}=d_{ij}+\max \{ (S_{i-1,j-1}), (S_{i-1,m} -g, (m<j-1)), S_{n,j-n} -g, (n<i-1))\}$$
(eq. 3)

Once the global score S_{ij} has been found, the optimal alignment can be retrieved by doing a traceback from the cell with the highest score. This algorithm is slightly more elaborate than the original Needleman and Wunsch that did which used simpler gap penalties. Computation is quadratic in time and space for the score. Gotoh proposed a modified version of this algorithm that allows computation of score in linear space, while using affine gap penalties(45). Finally, a recursive algorithm originally described by Hirschberg(99) and refined by Myers and Millers(100) allows computation of both score and alignment in linear space, using affine gap penalties. A vast number of variations have been proposed around the original algorithm. For instance, since it is known that the mathematical optimal is rarely biologically optimal methods have been described that allow computation of sub optimal alignments(101−103). Such computations can prove useful when trying to assess the reliability of a sequence alignment(104). Another variation, proposed by Taylor, involves biasing the DP toward motifs, in order to minimize the effect of the gap penalty choice(47).

DP is not restricted to global alignments but can also be used for finding the longest sub−sequences shared between two otherwise unrelated sequences. This algorithm (50) is extremely important for database searches. It is mostly an adaptation of the Needleman and Wunsch algorithm and involves extending a path as long as the alignment it contains is improving (hence the 0 in the following equation):

$$S_{ij}=\max \{ (S_{i-1,j-1})+d_{ij}, (S_{i-1,m} - g, (m<j-1))+d_{ij}, S_{n,j-n} - g, (n<i-1))+d_{ij}, 0\} \quad \text{(eq. 4)}$$

The score of a path is set to 0 if this path gets a negative score (the substitution matrix needs to give negative values to unfavored substitutions). Best segments are identified by finding the best scores in the score matrix and making tracebacks, using 0 as a boundary for the path.

The use of DP is not restricted to sequences. For instance, the Viterby algorithm, used for aligning a sequence to a HMM(80) is an adaptation of the classic DP algorithm. Similarly, modified versions of DP can be used to align structures. This was shown by Corpet and Michot who described a DP strategy for aligning two RNA molecules taking into account potential secondary structures in one of the sequences while knowing the structure of the other(92). Taylor and Orengo(105) also described a heuristic based on dynamic programming designed to align two protein structures (double dynamic programming). Finally, the algorithm used to align an RNA molecule to a covariance model(91) is based on DP and is very similar to the one described by Zuker for predicting RNA folds(89). Most of these algorithms are computationally very demanding (higher complexity than regular DP). For this reason, alternative techniques (like stochastic optimization techniques) can be sensible alternatives or even necessities when dynamic programming cannot be used.

Nonetheless, in the case of pairwise sequence alignments DP may provide the most efficient and accurate way to compute a solution, as long as traditional gap penalty schemes are used. We will now see that how this pairwise method can be extended to multiple sequences.

4.2.2 Aligning Two Alignments : Progressive Alignment Methods

Although it is hard to align more than two sequences at a time by regular dynamic programming, it is possible to align two alignments (i.e. two sets of prealigned sequences) with a pairwise algorithm. This is due to the fact that an alignment (pairwise or multiple) can be regarded as a generalized sequence where instead of having one residue per position, one has a residue vector. Such a definition allows the alignment of two alignments, treating them like normal sequences(69). It makes sense when doing so to treat each alignment as a profile.

These methods for aligning multiple alignments are very important because they are central to one of the most popular multiple alignment strategies: the progressive alignment algorithms described by Feng and Dolittle(74) and Taylor(75). The principle of a progressive alignment is quite straightforward. Since it is impossible to produce a true multiple sequence alignment by DP, using all the sequences at the same time, one will begin by aligning pairs of closely related sequences. Such pairwise alignments, dealing with very similar sequences are likely to be correct if the sequences are similar enough. In the second stage, pairs of closely related pairwise alignments will then be aligned two by two, and so on until the whole multiple sequence alignment has been produced. Ideally, the order in which the sequences are aligned depends on their phylogenetic relationship (i.e. a phylogenetic tree) and the procedure will follow the topology of some estimated tree. Several methods exist that allow the computation of such trees using various clustering algorithms that rely on an estimates of the pairwise distances between the sequences(4, 106).

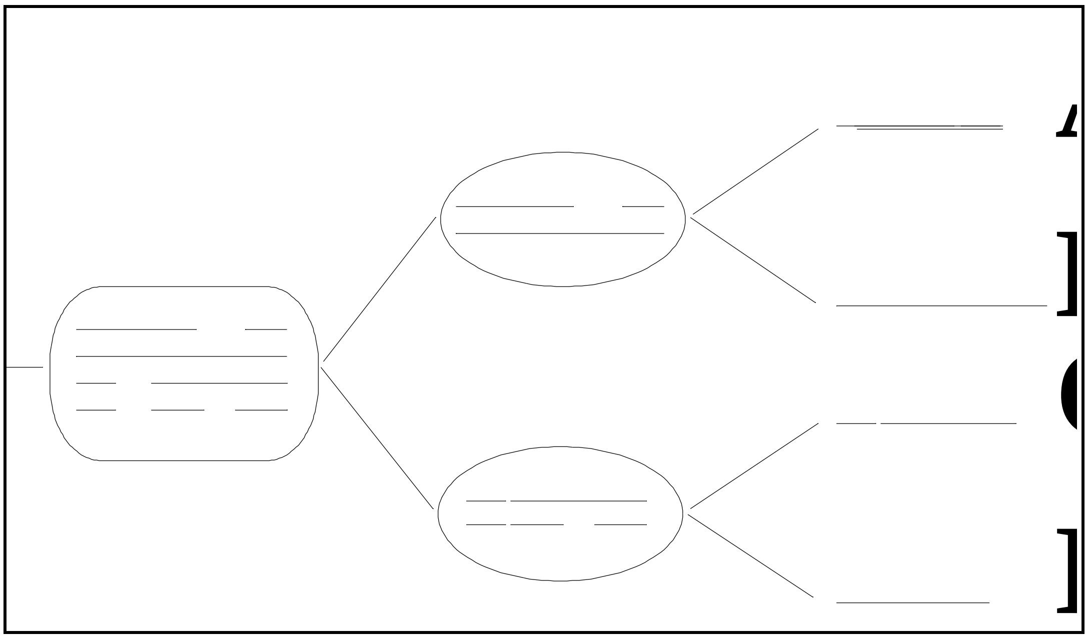


Figure 7. Progressive alignment strategy(74, 75). A phylogenetic tree is estimated for the four sequences A, B, C and D (gray lines). Its topology is used to decide the order for aligning the sequences by dynamic programming.

The advantage of progressive alignments is that they provide an efficient heuristic for overcoming the problem of computational complexity posed by global optimization techniques. However, if the aim is, for instance, to optimize the sums of pairs with a substitution matrix and a pair of gap penalties, one has to be aware that the progressive method does not offer any guarantee. Its performances may be unpredictable and will depend on the quality of the intermediate alignments. These will depend in turn on the relations between the sequences and the accuracy and the density of the tree. Intermediate alignments are not necessarily correct (in terms of optimization) because they are made using only a subset of the information available. This may have serious consequences since early mistakes will never be corrected (’once a gap always a gap’) and may also propagate by inducing more mistakes in further intermediate alignments. This problem is often referred to as a ’local minimum problem’, in the sense that the method by its greediness is pushed toward satisfying short term constraints that may lead to a non global solution.

Nevertheless, in many cases, this progressive strategy leads to convincing alignments at a low computational cost. There are two good reasons for that. First of all, in many cases, the sequences are well fitted to a progressive approach, and secondly because mathematical optimality does not necessarily mean better biological alignments. This explains why some of the most popular multiple sequence alignment packages are based on these algorithm. They include Clustal V(77), Multal(75),ClustalW(21),Barton and Stenberg(107), etc. In a wide variety of cases these methods have been shown to do well when their performances are assessed using sets of reference alignments. Many of the techniques used for improving multiple sequence alignments such as position specific gap penalties, tree based weights, secondary or tertiary structure information can be implemented in a progressive alignment strategy as shown in ClustalW(21) and in the work of Barton and Stenberg(107).

One of the problems of progressive alignments is that their accuracy will clearly depend on the relation between the sequences aligned. If these are closely related, intermediate alignments are likely to be accurate and to induce a correct overall alignment. On the other hand, if the sequences are only remotely related, it may be necessary to simultaneously use all the information they contain to build a realistic model (i.e. most of the pairs of sequences needed to build the intermediate alignments will not contain enough information for inducing a correct pairwise alignment). The only way to improve the relation between the sequences in a set is to incorporate some more sequences into this set. In most of the cases this is not an option, simply because the sequences are not present in the databases and may not even exist. Simultaneous sequence alignment methods are therefore necessary. We will now describe methods that attempt to produce this type of multiple sequence alignment.

4.3 DETERMINISTIC APPROACHES FOR NON PROGRESSIVE MULTIPLE ALIGNMENTS

4.3.1 The Carrillo and Lipman Algorithm

The basic DP algorithm is hard to apply to more than two sequences at a time and the computation of an SP multiple alignment using DP has been shown to be NP−complete(19). In order to compensate for these limitations,. an algorithm was designed by Carrillo and Lipman(57) to perform such operations on a larger number of sequences (up to 10). It relies on the idea that the whole alignment space does not necessarily need to be explored when aligning sequences. Bounds can be derived that guarantee finding the optimal alignment inside a portion of the hyper space defined by the sequences of the set. Several methods have been described for deriving guaranteed boundaries (57, 108). Unfortunately, these boundaries are often too loose to be practical (i.e. they define a space that is too large to be explored). To be of any use, the Carrillo and Lipman algorithm needs to rely on tighter heuristic boundaries that do not guarantee optimality. These are implemented in the package MSA(22, 109). Their main property is that they depend on the level of similarity of the sequences. For very closely related sequences, the boundaries will be extremely tight and define a very small portion of the hyper space. If the sequence similarity decreases, the boundaries become looser, until the computation of an alignment requires the exploration of a space beyond computational ability.

The limit on the number of sequences and on the level of similarity shared by these sequences constitutes the major drawback of MSA. It means that in most cases the only alignments that are within the scope of the program are those for which an accurate progressive alignment can be made because the sequences are related enough. The second important drawback of MSA is that it is restricted to a very specific type of objective function. This is especially drastic with regard to the type of gap penalties MSA is able to use ('semi natural gap penalties' instead of 'natural gap penalties', see section 3.1.2).

4.3.2 Other Approximation Techniques

More recently, non−dynamic programming based methods have been proposed that attempt to generate multiple sequence alignments by solving graph theory problems. One of these techniques is known as the MWT (Maximum Weight Trace) formulation, introduced in 1993 by Kececioglu(110), (111). Like dynamic programming, the solving of MWT is known to be NP complete. But even so, using a branch and bound algorithm, a MWT based algorithm has been described(111) that manages to align up to 15 sequences in a reasonable amount of time. Unfortunately, this approach suffers from the same drawback as MSA: it requires tight boundaries to be established, which is not always possible, especially when the sequences have very low similarity.

Considering all this evidence, it is clear that in the years to come, our ability to build multiple sequence alignments will increase along with computer power. This means that these methods will be able to handle more and more sequences. On the other hand, the problem, that in most cases has been shown to be NP complete is likely to remain unsolvable in polynomial time. This implies that the number of sequences a deterministic strategy is able to handle will always remain severely limited. Another serious drawback is the fact that in many cases these algorithms are very specific for a given objective function and difficult to adapt. We will now focus this review on stochastic techniques that have proven to be much more flexible and versatile tools.

4.4 STOCHASTIC HEURISTICS

4.4.1 What is a stochastic Method ?

A stochastic optimization strategy (also known as Monte Carlo simulation) is a way of finding a solution to a problem through some form of random sampling. The extent to which the search is random mostly depends on the heuristic one uses. The advantage of randomness is that there is a non null probability for any potential solution to be sampled, regardless of the solution space size. Of course, the randomness also implies that all the solutions may not be looked at (including the global optimum). In order to correct for this problem, a large number of heuristics have been designed that attempt to bias the way in which the solution space is sampled. The aim is to improve the chances of sampling the optimal solution. As a consequence, most stochastic strategies can be regarded as trade offs between greediness and randomness.

This can best be explained considering a solution landscape with hills and valleys(19, 112). The optimal solution is the top of the highest hill (or the bottom of the deepest valley). If there is only one hill, one can find this point by simply climbing along the steepest gradient. Such a method does not have to be stochastic and is known as hill climbing. However, if there are several small hills and a high one, the outcome of hill climbing will be restricted by the point where it was started. The action of climbing the hill will not necessarily lead to the optimum, but to the first local optimum. This is the price to pay for the excessive greediness of a hill climbing strategy that always goes up. If we add an element of randomness that decreases the greediness of the strategy, it may sometimes be possible to take alternative routes (going down, for instance, which in turn may allow the search to encounter the highest hill). This less greedy technique is known as Stochastic Hill Climbing. Ultimately, its success (as for any stochastic strategy) will mostly depend on the fitness landscape. The more complex this landscape, the hardest it is to find the global optimum.

Stochastic methods are quite well suited for solving sequence alignment problems where the landscape is very complex(113) and the number of solutions much too large to be exhaustively analysed(19). Many of these techniques are inspired from physical (Simulated Annealing), biological (Genetic Algorithms, Evolutionary strategies), or even social (Taboo search) phenomena, others are simply based on iterative statistical analyses like the Gibbs sampling (stochastic expectation−maximization strategy). In this section, we will review some of the applications of stochastic heuristics to the problem of sequence alignment. Since one of these techniques (Genetic Algorithm) is central to the work presented here, it will be given a more detailed presentation in the next section.

A very important advantage of many Monte Carlo strategies over more traditional heuristics is the fact than they often allow conceptual separation between optimization and objective function. Ideally, the optimization strategy will play the role of a black box into which any objective function can theoretically be plugged. The main drawback of these strategies is that they rarely give any indication of how complete the optimization is. In fact, since most of these methods are iterative, it is usual practice to stop them if the search stabilizes for a specific number of iterations. The general lack of guaranteed criteria for stopping the search is usually the main problem. It can sometimes be overcome if the score obtained with the objective function contains some information regarding the completeness of the optimization.

4.4.2 Iterative alignments and Expectation−Maximization Strategies

Iterative strategies are heuristic approaches, very similar in spirit to EM methods (Expectation maximization). Expectation−Maximization is a Hill climbing strategy that involves two steps: (i) given a model and some data, estimation of new parameters for the model, (ii) given the new parameters and the ancient model, estimation of a new model. The procedure is carried on iteratively until the model and the parameters stabilize (good fit between the model and the parameters). Depending on the implementation, EM can incorporate some stochastic steps as do iterative alignment strategies. The scope of EM strategies is quite large. The optimization of RNA covariance models is made using EM(91), as well as the training of HMMs in SAM(81). Several multiple sequence alignments strategies have also been described based on EM(114, 115)

To perform an iterative multiple sequence alignment, two components are needed: a set of prealigned sequences and an algorithm for aligning pairs of sequences (or pairs of alignments) such as DP using Gotoh's operators for aligning profiles and alignments(69, 73). An iterative alignment algorithm usually takes the following steps:

1−split the multiple sequence alignment into two groups of prealigned sequences
2−apply the DP algorithm to the two multiple alignments
3−if the score is not stabilized, go to 1

Since the number of possible partitioning in step 1 increases factorially with the number of sequences, exhaustive analysis cannot be done, and one has to use random partitioning or other types of partitions that use for instance the phylogenetic tree. Several alternative schemes of this type have been described (for a review, see (116)) These methods do better on average than progressive strategy (in terms of scores) but are also much slower (100 times on average). Recently, Gotoh presented such a technique named double−nested randomized iterative strategy(117). He uses it to optimize the weighted sums of pairs with a reevaluation of the weights after each iteration. According to the author, as judged from reference structural alignments, this technique is significantly more accurate than progressive alignments strategies.

Gibbs sampling is a more sophisticated example of stochastic EM(118). In Gibbs sampling applied to multiple sequence alignment(83), a random solution is generated and used to guide the generation of the next solution (ungapped multiple alignment), assigning probabilities to all the neighboring solutions, and randomly choosing one of these (or more exactly, choosing a new solution according to their probabilities, given the current model). The search is iterative and has been successfully implemented for identifying ungapped motifs conserved among a set of sequences. In such a context, given a set of non−aligned sequences, the aim is to generate the more informative model (the less likely to have been obtained by chance). The objective function described in that context by Lawrence et al. is purely based on statistical analysis of the data and can be made fully independent of any prior information. In many respects it is very similar to HMMs.

4.4.3 Simulated Annealing

One of the first stochastic techniques described was simulated annealing (SA)(119, 120). SA relies on an analogy with physics. The idea is to compare the solving of an optimization problem to the cooling of a metal (i.e. finding the best position of each atom in the cooled metal is equivalent to solving a multi constraints problem). The principle is very straightforward. Given a function, a random solution will be generated, and a temperature chosen. Every iteration, a new solution is created based on the previous one (using some random modification algorithm). If the new solution is better than the old one, it will be accepted. Otherwise, it will be accepted/rejected with a probability that depends on the temperature and the level of disimprovement (i.e. the higher the temperature, the more likely a solution is to be accepted, regardless of its quality). Temperature will gradually be cooled down. This means that at the beginning, almost any new solution is accepted, while in the last phases (low temperature) only the solutions bringing an improvement are accepted. The SA algorithm can be summarized as follow given that Sc is the current solution at the nth iteration:

```
1) Generate an initial  solution Sc=So and an initial Temperature (To), Tc=To
2) Modify the solution Sc into Sn
3)      IF F(Sn) is better than F(Sc) then keep the new solution: Sc=Sn
        ELSE
                keep Sc with a probability $P=e^{-(|F(Sn) - F(Sc)|)/Tc}$
4) Change the value of Tc according to the annealing schedule
5) Go to 2 as long as the termination criterion has not been met.
```

The strategy depends on three parameters: the initial temperature, the cooling schedule and the acceptance function. The cooling and the acceptance function define the nature of the SA. The most common are those using Boltzman distributions, but one can also use Gaussian or other ad−hoc distributions(121). A remarkable property of SA is that given a cooling function slow enough, the search is guaranteed to reach the optimal solution. Of course this argument remains theoretical since in most of the cases an adequate cooling schedule would require too much time to be of any practical use (hence the slowness of SA).

Despite this intrinsic limitation, SA has been applied to multiple sequence alignment on several occasions(122−124). The conclusion of these studies has mostly been that although it does reasonably well, because of its slowness, SA has to be restricted to being an alignment improver method rather than a full alignment method. In all these studies, SA is applied to sequence alignment in a similar fashion: a multiple alignment is created (randomly or using some heuristic) and modified using specific sub−routines that move gaps around or insert them. We will later see that these 'modification procedures' are in many ways similar to the mutations used in a genetic algorithm strategy.

As opposed to the iterative methods previously discussed, SA is not restricted to making SP multiple alignments using DP. On the contrary, the method displays most of the properties of a black box. If it was not so slow, SA would possibly be the ideal method for any optimization problem. It has successfully been applied to the alignment of RNA molecules using potential secondary structure information(90). Similarly SA based methods(125) have been described for predicting the fold of very long RNA molecules that are beyond the scope of traditional methods like Zuker's. Finally, a variant of expectation−maximization based on SA is used for the training of HMMs(126). In the work presented here (SAGA and RAGA), SA has been an important source of inspiration, because of the similarities that exist between SA and GAs.

GAs are probably some of the most interesting stochastic optimization tools available today. They can best be described as a very flexible framework in which most of the available methods (deterministic or not) can be integrated in order to constitute a general tool. One of the reason why GAs have received so little attention in the context of multiple sequence alignment is probably due to the fact that the implementation of a genetic algorithm specialised for multiple alignment is much less straightforward than with simulated annealing. In other areas of computational biology, GAs have already been established as powerful tools. These include protein 3D structure prediction or RNA secondary structure prediction

4.5 GENETIC ALGORITHMS

4.5.1 What is a Genetic Algorithm?

Genetic algorithms are based on a loose analogy with the phenomenon of natural selection. They have existed in one form or another for quite a while, but were formally introduced by Holland in 1975(127). The principle is quite straightforward. Given a problem, a set of potential solutions (population) compete with one another (selection). These solutions can be modified (mutations), or combined with one another (crossovers). The idea is that acting together, selection and evolution will lead to an overall improvement of the population. Most of the ideas developed here about GAs are taken from (128, 129)

There are two essential concepts at the heart of the GA strategy. The first one is selection. Selection is established in order to lead the search toward improvement. It means that the best solutions to the problem (as judged with the objective function one is interested in) will be selected according to their quality. If one was to do so in a deterministic way, for instance, by only selecting the best solution every generation, the search would very rapidly converge toward the first local optimum it encounters. In such a case, GAs would mostly behave like Hill Climbing. To avoid this major problem (called premature convergence), the selection procedure in a GA is not absolute but statistical. Making statistical selection is a very straightforward process.

In a first step, each individual is evaluated using the objective function. The score obtained that way is turned into some fitness measure. In the selection round, an imaginary wheel is spun where each individual has a number of slots proportional to its fitness. This means that individuals with a very high score are the most likely to be selected, while those with a low score are less likely. However, everything is possible and it is this uncertainty that prevents a GA from converging prematurely to the closest local minimum.

The second key concept in a GA is the concept of evolution. The aim of the operators is to create new solutions based on those present in the population. By analogy to real life, there are two sets of operators: the crossover (combining two individuals) and the mutations. Since crossovers can be regarded as a very primitive interpretation of sexual reproduction, it is current practice to apply selection when choosing the two individuals that are going to be combined. When doing so, one hopes that the child produced that way will combine the qualities of both parents.

In practice, a genetic algorithm follows a series of cycles known as generations. During each cycle, individuals are evaluated and used to create the next generation through selection and operations. The variations one can put inside such a scheme are virtually infinite. Different schemes can be used for turning a score into a fitness measure. Selection can be made stronger or weaker. Many of the effects of such parameters have been studied in detail. However there is no solid argument for one model of GA to outperform the others. In fact the choice of a model seems to be mostly problem dependent. We found in our experiments that if the problem is suited to combinatorial optimization, any model of GA does reasonably well. However, some do better than others and the appropriate tuning needs to be done in what is mostly a trial−and error−strategy.

There is very little theory around the reasons why GAs perform efficient optimizations. For instance, as compared to SA, there is no formal proof that a GA will reach the optimal solution given enough time. The most popular concept to explain the performance of GAs is the notion of building blocks. The concept is best understood if we consider a GA using a coding system that consists of binary strings (i.e. each individual is a chromosome where each gene can have two allelic values 0 or 1). In such a context, a block is defined as any stretch of 0s and 1s present in a chromosome. The GA strategy through its cycles of selection and combination makes it possible for important blocks to be selected (i.e. increase the number of their copies in the population). This in turn will increase the chances for these blocks to be extended through mutations and crossovers. The more stable a block (i.e. the better the score it induces), the more likely it is to spread in the population (this would be the equivalent of the fixation of a group of alleles in an animal population). In this context mutations help create new blocks, or restore blocks that may have been lost. The fact that the number of existing blocks is much larger than the population itself induces a phenomenon known as implicit parallelism.

This GA procedure can also be looked at through another angle. If we consider each solution as a vector in a hyperspace, with each piece of the chromosome being one of the coordinates, we can view the search as a form of multivariate analysis. In this context, building blocks are sets of coordinates that restrict the search to some smaller portion of the hyperspace that has a higher concentration of good solutions than the rest. While the search goes on, this hyperspace becomes smaller and smaller until it cannot be reduced anymore.

The theory of the building blocks was initially proposed for the simple Genetic Algorithm developed by Goldberg(128). For this reason, it is restricted to binary chromosomes, where each gene only has two allelic values (1 or 0). It is hard to extrapolate this theory to the more complex representations required by sequence alignment problems. However, empirical evidences suggest that the building block theory constitutes a reasonable approximation when analysing the behavior of more complex representation systems. It should also be stressed that although it provides an elegant way to answer the question of how a GA works, the theory of the building blocks is useless for predicting with which type of problem the GA approach will fail or succeed.

Deceptivity is an area of GA research that has received some significant attention over the last few years. It amounts to defining the conditions in which a GA will fail at finding the correct solution. What these studies have shown is that like any optimization method, the GA is sensitive to the fitness landscape of the function one is interested in. These results established what would intuitively be assumed. When there is very little continuity in the function one is interested in, the optimization becomes much harder or impossible. On the other hand, given a complex problem, there are often alternative ways of defining a landscape. In fact, the landscape will depend on the representation of a solution and on the means one uses to walk along the landscape (operators in our case). When using binary coding, methods that attempt to reshape the fitness landscape in order to make it more continuous are known as gray coding. More generally, whatever the problem and the representation, efforts should be made so that neighboring solutions should have comparable fitness. Defining the neighborhood of two solutions is a complex problem that depends on two factors: the representation system and the operators used. There is no use having a representation that induces a very smooth fitness landscape if the operators are not as well designed to sample close neighbors...

For instance, in the case of a multiple sequence alignment, there are many ways one can define such neighbors. The neighbor of a given alignment will be another alignment created when inserting a gap or when shifting an already existing gap. The continuity of the landscape explored during the search will depend on the shape of the fitness landscape these operators define (i.e. do they induce any gradient that can be followed?). It is clear that a mutation that inserts a gap at a random position (thus potentially generating a shift for a whole sequence) does not explore the same neighborhood as a mutation that simply shifts a gap by one residue. This is a serious problem, and in practice defining the appropriate operators constitutes the main difficulty when designing a Genetic Algorithm.

4.5.2 Applications of GAs in Sequence Analysis

Although the concept of GA is relatively new in the context of sequence alignments, GAs themselves have been applied to sequence analysis on several occasions, mostly to solve structure predictions problems for DNA, RNA and proteins. This makes sense since structure predictions problems fit the concept of GAs very well. A protein structure can be described as a list of amino acids associated with some conformation values. These conformation values may be angles of chemical bonds. In such a context, the chromosomal representation can be the list of the bonds that need to be characterized (genes) with the allelic values being the actual angles in a given conformation. These problems are difficult ones, mostly because it is hard to create an objective function that accurately mirrors the protein folding process. Several attempts have been made that suggest GAs are among the best suited optimization strategies for ab initio folding(130−136). Recently a strategy has been proposed for docking analysis using a GA(137). Finally, RNA folding has also received some attention(87, 88, 138, 139). Generally speaking, the use of genetic algorithms in sequence and structure analysis is rapidly expanding. For instance, Medline records about 110 publications describing GA based methods for sequence analysis over a period of 8 years [1989−1997]. Out of these 110 publications, 45 appeared during 1997 alone.

In terms of sequence alignment, the only work we are aware of is an attempt to perform iterative alignments using a genetic algorithm in order to speed up the process(115). Apart from SAGA, no method has been described which is able to directly perform operations on the alignment and to accommodate non−standard objective functions. We will now briefly review the motivations behind SAGA as well as the work done to validate our approach and extend its scope to a larger variety of objective functions and problems.

5 COMPARISON OF THE METHODS

We are only aware of two general studies made to compare systematically different alignment methods(117, 140) .The work by McClure et al. was made by comparing the performances of seven different methods(22, 23, 74, 77, 107, 110, 141) when aligning four different protein families (Hemoglobin, RibonucleaseH, Kinases and aspartic proteases). The methods were evaluated for their ability to correctly align some motifs of known structural or functional importance in each family. The conclusions were that proteins with more than 50% identity can usually be correctly aligned, regardless of the methods. When identity decreases, all the methods become affected by the number of sequences (more sequences do not necessarily improve the results). The general conclusion is that progressive methods doing global alignments are usually the best. However, the test sets were not large enough to really allow strong distinctions to be made between the methods ability and the non progressive methods tested were still in an early development stage.

A main drawback of the approach taken by McClure was the limited set of reference alignments on which their comparison was based, and the fact that these alignments had not been established by structural analysis. It has now become standard procedure to evaluate new methods by comparing their output with reference alignments available in some of the structural databases (33, 142−144). Gotoh recently presented such a comparison (117)based on structural databases(33, 142). In his study, he compared four methods: one based on progressive alignments (21) , two iterative alignment strategies that he had previously described(73, 145) and a new one that involves iterative alignment with dynamic reevaluation of the weights and the phylogenetic tree for optimizing the weighted sums of pairs with affine gap penalties. The methods were applied to 54 protein families. The conclusion was that iterative alignments methods do on average a bit better than ClustalW(21). It should be noted that the study did not make any use of the program MSA. Previous results obtained with the MSA program(22) indicate that MSA does not outperform ClustalW(98),(Barton, personal communication). This suggests that rather than the quality of the sums of pair optimization provided by the iterative alignment strategies (i.e. global optimization) the improvement is probably due to the dynamic weighting scheme used by Gotoh that allows a better use of the information. The weights used in this strategy are similar to those described for the MSA program(66). In MSA, these weights are computed from an initial progressive alignment which is probably less accurate than those obtained by Gotoh on optimized alignments and used for reevaluation of the tree. It may also be that the iterative strategy used by Gotoh does not lead to mathematical optimality, but to sub−optimal alignments that are biologically more accurate.

6 SUMMARY OF THE CONTRIBUTIONS

6.1 SAGA: MAKING MULTIPLE SEQUENCE ALIGNMENT BY GENETIC ALGORITHM

SAGA is a package designed to perform multiple sequence alignments using a genetic algorithm. The method involves evolving a population of alignments in a quasi evolutionary manner and gradually improving the fitness of the population as judged by an objective function which measures multiple alignment quality. SAGA uses an automatic scheduling scheme to control the usage of 22 different operators for combining alignments or mutating them between generations. When used to optimize the sums of pairs objective function, SAGA performs better than some of the widely used alternative packages like MSA. This is seen with respect to the ability to achieve an optimal solution. The general attraction of the approach is the ability to optimize any objective function that one can invent.

This last point was the main motivation behind the development of SAGA. A lot of alternative packages exist that can provide reasonable optimizations of SP multiple alignments, for instance. However, it is known that in cases where the sequences are very remotely related, these methods fail, not necessarily because of an optimization problem, but also because traditional SP objective functions may not be well adapted to these alignments. Another potential application for a good quality optimization black box is hidden Markov model training. The EM based algorithms used at present are known to be very sensitive to local minimum problems. This problem could easily be overcome by using a GA based training scheme.

The validation of SAGA as an optimization tool was made using the program MSA. Remarkably, starting from unaligned sequences, SAGA was always able to reproduce MSA results or to outperform them. As far as we know, SAGA is the only optimization method that can achieve this result without using dynamic programming. However, it is fair to say that SAGA is not yet ready for systematic use and does not appear as a challenger to programs like ClustalW or other progressive alignment packages. On the other hand, SAGA is a powerful tool for analysing new objective functions and evaluating the quality of faster heuristic methods. In the long term, we hope that SAGA will become much more practical, thanks to the increasing power of the computers and to some improvements made in the algorithm (parallelisation, better seeding, optimized evaluation).

The original version of SAGA has been available for just over a year (http://www.ebi.ac.uk/~cedric/saga_hp.html). The package now regroups a community of known users of about 30 people. The software is supported and a new release is scheduled for the beginning of 1998.

6.2 COFFEE: IMPROVING ON EXISTING OBJECTIVE FUNCTIONS

COFFEE is a natural extension of SAGA. It is an attempt to design a new type of objective function for evaluating the quality of multiple sequence alignments. There is already a number of objective functions described for evaluating multiple sequence alignments. They all have qualities and drawbacks. With COFFEE, the aim is not to add one more objective function to the list, but to propose a scheme that makes it possible to combine any existing scoring schemes. The COFFEE score reflects the level of consistency between a multiple sequence alignment and a library containing pairwise alignments of the same sequences. We show that multiple sequence alignments can be optimized for their COFFEE score with the genetic algorithm package SAGA. The function is tested on 11 test cases made of structural alignments extracted from 3D_ali(33). On 9 of these test cases, SAGA with the COFFEE function is able to outperform ClustalW (progressive multiple sequence alignments) as judged by comparison with the structural references. We also show that the COFFEE score can be used as a reliability index on multiple sequence alignments. Finally, we show that given a library of structure based pairwise sequence alignments extracted from FSSP, SAGA can produce high quality multiple sequence alignments.

An important issue in COFFEE is the validation of an objective function. This process should not be confused with the evaluation of an optimization strategy, as described for SAGA. An objective function may be optimized correctly but lead to biologically irrelevant alignments. Verifying the correctness of an alignment is a complex process, that requires the use of biologically correct references. Since the closest thing to a biologically correct alignment is a structural alignment, we used such alignments to validate the COFFEE approach (3D_ali).

COFFEE is powerful mostly because of its openness. It can accommodate virtually any type of modification, including position specific weights and different types of sequence weights. Furthermore any alignment making technique (pairwise, multiple, global, local) can be integrated into the COFFEE framework and several otherwise incompatible techniques can be combined together.

6.3 RAGA: THREADING RNA SECONDARY STRUCTURES

The distinction between sequence and structure alignments has already been developed here. It seemed like an interesting question to ask whether SAGA would be able to accommodate a structure based objective function. The fitness landscape of such a function is likely to be very different from those used for protein alignments. We applied the technique to RNA secondary structure because this is a much simpler problem than protein threading. Even if the algorithmic complexity is the same, the problem can at least be formulated in a fairly accurate way, thanks to the fact that RNA secondary structures are mostly based on Watson and Crick base pairing.

In RAGA, we describe a new approach for accurately aligning two homologous RNA sequences, when the secondary structure of one of them is known. To do so we developed two software packages called RAGA and PRAGA which use a genetic algorithm approach to optimize the alignments. RAGA is mainly an extension of SAGA. In PRAGA, several genetic algorithms run in parallel and exchange individual solutions. This method allows us to optimize an objective function that describes the quality of a RNA pairwise alignment, taking into account both primary and secondary structure, including pseudoknots. We report the results obtained using PRAGA on nine test cases of pairs of eukaryotic small subunit ribosomal RNA sequence (nuclear and mitochondrial).

The parallel implementation is described in detail in the corresponding publication. It involves a set of synchronized RAGA processes running on different machines on the same set of sequences. Every N generations (typically 5), the processes exchange some of their fittest individuals. The main originality of this implementation is its relative simplicity. It only involves a population sharing scheme between several GAs and does not require any low level modification of the GAs. Of course, the parallel program is very different from the original GAs (i.e. it induces a very different population structure). Despite this fact the parallel version has properties very close to those that would have been expected with RAGA parallelized at a lower level (speed and accuracy). The parallelization module was made very general. It can be used with SAGA and does not depend in any way on the objective function. We plan to maintain this module so that it remains compatible with other developments made on SAGA or RAGA. This type of parallelisation (known as island parallelisation) is not completely new, but we are not aware of any previous descriptions of a model strictly identical to ours. Although no systematic work has yet been done for accurate characterization, we found that on the RAGA objective function, the speedup is much more significant than when the parallel module is applied to SAGA. This may have to do with the fact that our parallelisation makes it easier for a GA to analyse very complex fitness landscape, such as the one required by a structure based objective function. RAGA and PRAGA have been made available over the WWW (http://www.ebi.ac.uk/~cedric/raga_hp.html)

In its present form, RAGA is mostly a prototype. There is little interest in attempting to align two ribosomal RNA sequences at a time. Large accurate alignments exist that should be used in order to guide the characterization of a new sequence. This has been the main motivation for the last project presented here: analysing large multiple alignments in order to optimally use the information they contain. However, out of a ribosomal RNA context, the requirement of knowing the guide structure of the sequences to align is quite heavy. In many interesting cases, such prior information will not be available, and will be difficult to extract when sequence identity is very low. For this reason, we plan to extend the algorithm in order to use an objective function similar to the one described in (90)or in(146) that allows the alignment of two RNA sequences using primary and potential secondary structure. Finally, applying RAGA to the problem of Protein Threading is one of the natural continuations of this work. We plan to do so in the context of the program phDthreader(20).

6.4 OPTIMIZING RIBOSOMAL RNA PROFILE ALIGNMENTS

The purpose of this project is to show that the use of the information contained in a profile can be optimized when trying to introduce (align) a new sequence. The main question is to decide how each sequence of the alignment should contribute to this new alignment.

It is quite obvious that sequences closely related to the one of interest should receive a higher weight. For this reason, we developed a weighting scheme based on the inverse of the distances between the new sequence and the rest of the sequences. This can be seen as a pairwise weighting scheme (cf. section 3.2.3) quite similar in fact to the one proposed for COFFEE. On the other hand, there is also a need to avoid bias due to unequal representation of different taxa. This effect is achieved by using tree based weights that attempt to correct for unequal representation.

We found that the ideal weighting scheme, in a dynamic programming context, appears to be a combination of these two. Attempts were also made to use position specific gap penalties. In a DP context , it is hard to take into account secondary structures. This should be done in a second step that will involve implementing a profile based objective function in the RAGA/PRAGA framework. In the long term the alignment method is meant to be made available over the WWW through a JAVA−based server.

CONCLUSION

This work provides further evidence for the usefulness of genetic algorithms in a sequence analysis context. GAs mimic very efficiently the human approach that is made of trial and error and of a continual attempt to combine partial results in order to globally improve a solution. The work done here is only preliminary, and there is still a lot to be done before SAGA or RAGA come into everyday use. However, this may happen if the algorithm is improved in terms of speed and if the developments of COFFEE keep the promises of the preliminary results.

Putting aside the slowness that constitutes their main drawback, GAs have a lot of desirable properties. They can accommodate a large variety of problems and are easy to hybridize with other alternative methods. They also allow a conceptual barrier to be put between biological and computational problems. Optimizing a function is purely a formal problem that does not teach us much about biology. On the other hand, designing and testing an objective function can be extremely informative about the problem analysed and can help understanding the type of constraints that occur during evolution.

It is along these lines that I plan to extend the genetic algorithm strategy to a much wider range of problems, namely genome alignments and motif discovery. In a period where new complete genome sequences are published every month or so, there is a surprisingly small number of tools allowing one to compare these genomes. It is unfortunate because such comparisons can be extremely informative in understanding the way genomes evolve, the way functionally related sequences get clustered or not. In terms of computation, this is a difficult problem, because of the nature of the events that occur during evolution (inversion, transpositions, deletions, insertion) that are almost impossible to model using traditional techniques. I believe a GA approach is very likely to provide a convincing solution to such a problem.

REFERENCES

1. Lipman, D. J. and Pearson, W. R., *Rapid and sensitive protein similarity searches.* Science, 1985. 227: p. 1435−1441.

2. Altschul, S. F., Gish, W., Miller, W., Myers, E. W., and Lipman, D. J., *Basic local alignment search tool.* Journal of Molecular Biology, 1990. 215: p. 403−410.

3. Bairoch, A., Bucher, P., and Hofmann, K., *The PROSITE database, its status in 1997.* Nucleic Acids Research, 1997. 25: p. 217−221.

4. Saitou, N. and Nei, M., *The neighbor−joining method: a new method for reconstructing phylogenetic trees.* Mol. Biol. Evol., 1987. 4: p. 406−425.

5. Sonnhammer, E. L. L. and Kahn, D., *Modular Arrangement of Proteins as Infered from Analysis of Homology.* Protein Science, 1994. 3: p. 482−492.

6. Gribskov, M., McLachlan, M., and Eisenberg, D., *Profile analysis: Detection of distantly related proteins.* Proceedings of the National Academy of Sciences, 1987. 84: p. 4355−5358.

7. Haussler, D., Krogh, A., Mian, I. S., and Sjölander, K. *Protein Modeling using Hidden Markov Models: Analysis of Globins*. in *Proceedings for the 26th Hawaii International Conference on Systems Sciences*. 1993. Wailea, HI, U.S.A.: Los Alamitos, CA: IEEE Computer Society Press.

8. Garnier, J., Gibrat, J.−F., and Robson, B., *GOR method for predicting protein secondary structure from amino acid sequence.* Methods in Enzymology, 1996. 266: p. 540−553.

9. Garnier, J. and Robson, B., *The GOR method for predicting secondary structure in proteins,* in *Prediction of protein structure and the principles of protein conformation,* F.G. D., Editor. 1989, Plenum Press: New York. p. 417−465.

10. Rost, B., Sander, C., and Schneider, R., *PHD − an automatic server for protein secondary structure prediction.* CABIOS, 1994. 10: p. 53−60.

11. Göbel, U., Sander, C., Schneider, R., and Valencia, A., *Correlated mutations and residue contacts in proteins.* Proteins, 1994. in press: p. 000−000.

12. Neher, E., *How frequent are correlated changes in families of protein sequences?* Proceedings of the National Academy of Sciences, 1994. 91: p. 98−102.

13. Shindyalov, I. N., Kolchanov, N. A., and Sander, C., *Can three−dimensional contacts in protein structures be predicted by analysis of correlated mutations?* Protein Engineering, 1994. 7: p. 349−358.

14. Woese, C. R., Gutell, R., Gupta, R., and Noller, H. F., *Detailed analysis of the higher−order of 16S−like ribosomal ribonucleic acids.* Microbiol. Rev., 1983. 47: p. 621−669.

15. Gautheret, D. and Gutell, R. R., *Inferring the conformation of RNA base pairs and triples from patterns of sequence variation.* Nucleic Acids Res, 1997. 25(8): p. 1559−64.

16. Gutell, R. R., Weiser, B., Woese, C. R., and Noller, H. F., *Comparative anatomy of 16S−like ribosomal RNA.* Prog. Nucleic Acid Res. Mol. Biol., 1985. 32: p. 155−216.

17. Lathrop, R. H., *The protein threading problem with sequence amino acid interaction preferences is NP−complete.* Protein Engineering, 1994. 7: p. 1059−1068.

18. Needleman, S. B. and Wunsch, C. D., *A general method applicable to the search for similarities in the amino acid sequence of two proteins.* J. Mol. Biol., 1970. 48: p. 443−53.

19. Wang, L. and Jiang, T., *On the complexity of multiple sequence alignment.* Journal of computational biology, 1994. 1(4): p. 337−348.

20. Rost, B., Schneider, R., and Sander, C., *Protein fold recognition by prediction−based threading.* Journal of Molecular Biology, 1996. : p. in press.

21. Thompson, J., Higgins, D., and Gibson, T., *CLUSTAL W: improving the sensitivity of progressive multiple sequence alignment through sequence weighting, position−specific gap penalties and weight matrix choice.* Nucleic Acids Res., 1994. 22: p. 4673−4690.

22. Lipman, D. J., Altschul, S. F., and Kececioglu, J. D., *A tool for multiple sequence alignment.* Proc. Natl. Acad. Sci. USA, 1989. 86: p. 4412−4415.

23. Taylor, W. R., *Multiple sequence alignment by a pairwise algorithm.* Computer Applications in Biological Science, 1987. 3: p. 81−87.

24. Corpet, F., *Multiple sequence alignment with hierarchical clustering.* Nucleic Acids Res., 1988. 16: p. 10881−10890.

25. Dayhoff, M. O., Schwarz, R. M., and Orcutt, B. C., *A model of evolutionary change in proteins. Detecting distant relationships: computer methods and results,* in *Atlas of Protein Sequence and Structure,* M.O. Dayhoff, Editor. 1979, National Biomedical Research Foundation: Washington, D.C. p. 353−358.

26. Henikoff, S. and Henikoff, J. G., *Amino acid substitution matrices from protein blocks.* Proc. Natl. Acad. Sci., 1992. 89: p. 10915−10919.

27. Vogt, G., Etzold, T., and Argos, P., *An assessement of amino acid exchange matrices in aligning protein sequences: the twilight zone revisited.* J. Mol. Biol. 1995, 1995. 299(4): p. 816−831.

28. Henikoff, S. and Henikoff, J. G., *Performance evaluation of amino acid substitution matrices.* Proteins: Structure, Function, and Genetics, 1993. 17: p. 49−61.

29. Rost, B. and Sander, C., *Prediction of protein secondary structure at better than 70% accuracy.* Journal of Molecular Biology, 1993. 232: p. 584−599.

30. Lüthy, R., McLachlan, A. D., and Eisenberg, D., *Secondary structure−based profiles: use of structure−conserving scoring tables in searching protein sequence*

databases for structural similarities. Proteins: Structure, Function, and Genetics, 1991. 10: p. 229−239.

31. Jones, D. T., Taylor, W. R., and Thornton, J. M., *A mutation data matrix for transmembrane proteins.* FEBS Letter, 1994. 339: p. 269−275.

32. Tomii, K. and Kanehisa, M., *Analysis of amino acid indices and mutation matrices for sequence comparison and structure prediction of proteins.* Protein Engineering, 1996. 9: p. 27−36.

33. Pascarella, S. and Argos, P., *A data bank merging related protein structures and sequences.* Protein Eng., 1992. 5: p. 121−137.

34. Gonnet, G. H., Cohen, M. A., and Benner, S. A., *Exhaustive matching of the entire protein sequence database.* Science, 1992. 256: p. 1443−1445.

35. Benner, S. A., Cohen, M. A., and Gonnet, G. H., *amino acid substitution during functionnally constrained evolution of protein sequences.* Protein Engineering, 1994. 7(11): p. 1323−1332.

36. Sjolander, K., Karplus, K., Brown, M., Huguey, R., Krogh, A., Saira, M., and Haussler, D., *Dirichlet mixtures: a method for improved detection of weak but significant protein sequence homology.* Comp. App. in Biosci., 1996. 12(4): p. 327−345.

37. Overington, J., Donnelly, D., Johnson, M. S., Sali, A., and Blundell, T. L., *Environment−specific amino acid substitution tables: tertiary templates and prediction of protein folds.* Protein Science, 1992. 1: p. 216−226.

38. Benner, S. A., Cohen, M. A., and Gonnet, G. H., *Empirical and structural models for insertions and deletions in the divergent evolution of proteins.* J. Mol. Biol., 1993. 229: p. 1065−1082.

39. Pascarella, S. and Argos, P., *Analysis of insertions/deletions in protein structures.* J. Mol. Biol., 1992. 224: p. 461−471.

40. Chou, P. Y. and Fasman, G. D., *Empirical predictions of protein conformation.* Ann. Rev. Biochem., 1978. 47: p. 251−276.

41. Gu, X. and Wen−Hsiung, L., *The size distribution of insertions and deletions in human and roent pseudogenes suggests the logarithmic gap penalty for sequence alignmnent.* J. Mol. Evol., 1995. 40: p. 464−473.

42. Altschul, S. F., *Gap costs for multiple sequence alignment.* J. Theor. Biol., 1989. 138: p. 297−309.

43. Waterman, M. S., *Efficient sequence sequence alignment.* Journal of Theoretical Biology, 1984. 108: p. 333−337.

44. Miller, W. and Myers, E. W., *Sequence comparison with concave weighting functions.* Bull. Math. Biol., 1988. 50: p. 97−120.

45. Gotoh, O., *An improved algorithm for matching biological sequences.* J. Mol. Biol., 1982. 162: p. 705−708.

46. Vingron, M. and Waterman, M. S., *Sequence alignment and penalty choice.* Journal of Molecular Biology, 1994. 235: p. 1−12.

47. Taylor, W. R., *Motif−Biased Protein Sequence Alignment.* Journal of Computational Biology, 1994. .

48. Wilbur, W. J. and Lipman, D. J., *Rapid similarity searches of nucleic acid and protein data banks.* Proceedings of the National Academy of Sciences, 1983. 80: p. 726−730.

49. Pearson, W. R. and Lipman, D. J., *Improved tools for biological sequence comparison.* Proceedings of the National Academy of Sciences, 1988. 85: p. 2444−2448.

50. Smith, T. F. and Waterman, M. S., *Identification of common molecular subsequences.* J. Mol. Biol., 1981. 147: p. 195−197.

51. Pearson, W. R., *Searching protein sequence libraries: comparison of the sensitivity and selectivity of the Smith−Waterman and FASTA algorithms.* Genomics, 1991. 11: p. 635−650.

52. Waterman, M. S. and Vingron, M., *Rapid and accurate estimates of statistical significance for sequence database searches.* Proc. Natl. Acad. Sci. USA, 1994. 91: p. 4625−4628.

53. Altschul, S. F., Madden, T. L., Schaffer, A. A., Zhang, J., Zhang, Z., Miller, W., and Lipman, D., *Gapped BLAST and PSI−BLAST: a new generation of protein database search programs.* Nucleic Acids Research, 1997. .

54. Bucher, P. and Hofmann, K., *A sequence similarity search algorithm based on a probabilistic interpretation of an alignment scoring system.* Ismb, 1996. 4(44): p. 44−51.

55. Murata, M., Richardson, J. S., and Sussman, J. L., *Simultaneous comparison of three protein sequences.* Proceedings of the National Academy of Sciences, 1985. 82: p. 3073−3077.

56. Gotoh, O., *Alignment of three biological sequences with an efficient traceback procedure.* J. Theor. Biol., 1986. 121: p. 327−337.

57. Carrillo, H. and Lipman, D. J., *The multiple sequence alignment problem in biology.* SIAM J. Appl. Math., 1988. 48: p. 1073−1082.

58. Sankoff, D., *Minimal mutation trees of sequences.* SIAM J. Appl. Math., 1975. 28: p. 35−42.

59. Fredman, M. L., *Algorithms for computing evolutionary similarity measures with length−independent gap penalties.* Bull. Math. Biol., 1984. 46: p. 553−566.

60. Altschul, S. F. and Erickson, B. W., *Optimal sequence alignment using affine gap costs.* Bull. Math. Biol., 1986. 48: p. 603−616.

61. Bairoch, A. and Apweiler, R., *The SWISS−PROT protein sequence data bank and its new supplement TrEMBL.* Nucleic Acids Research, 1997. 25: p. 31−36.

62. Vingron, M. and Sibbald, P., *Weighting in sequence space: a comparison of methods in terms of generalized sequences.* Proceedings of the National Academy of Sciences, 1993. 90: p. 8777−8781.

63. Sibbald, P. R. and Argos, P., *Weighting aligned protein or nucleic acid sequences to correct for unequal representation.* Journal of Molecular Biology, 1990. 216: p. 813−818.

64. Felsenstein, J., *Inferring evolutionary trees from DNA sequences*, in *Statistical analysis of DNA sequences*, B.S. Weir, Editor. 1983, Marcel Dekker Inc.: New York. p. 133−150.

65. Thompson, J. D., Higgins, D. G., and Gibson, T. J., *Improved sensitivity of profile searches through the use of sequence weights and gab excision.* Computer Applications in Biological Science, 1994. 10: p. 19−29.

66. Altschul, S. F., Carroll, R. J., and Lipman, D. J., *Weights for data related by a tree.* Journal of Molecular Biology, 1989. 207: p. 647−653.

67. Sander, C. and Schneider, R., *Database of homology−derived structures and the structurally meaning of sequence alignment.* Proteins: Structure, Function, and Genetics, 1991. 9: p. 56−68.

68. Henikoff, S. and Henikoff, J. G., *Position−based sequence weights.* Journal of Molecular Biology, 1994. 243: p. 574−578.

69. Gotoh, O., *Optimal alignment between groups of sequences and its application to multiple sequence alignment.* Comput Appl Biosci, 1993. 9(3): p. 361−70.

70. Neuwald, A. F., Liu, J. S., Lipman, D. J., and Lawrence, C. E., *Extracting protein alignment models from the sequence database.* Nucleic Acids Res, 1997. 25(9): p. 1665−77.

71. Gibson, T. J., Thompson, J. D., and Heringa, J., *The KH domain occurs in a diverse set of RNA−binding proteins that include the antiterminator NusA and is probably involved in binding nucleic acid.* FEBS Letters, 1993. 324: p. 361−366.

72. Taylor, W. R., *A non−local gap−penalty for profile alignment.* Bull Math Biol, 1996. 58(1): p. 1−18.

73. Gotoh, O., *Further improvement in methods of group−to−group sequence alignment with generalized profile operations.* Comput Appl Biosci, 1994. 10(4): p. 379−87.

74. Feng, D.−F. and Doolittle, R. F., *Progressive sequence alignment as a prerequisite to correct phylogenetic trees.* Journal of Molecular Evolution, 1987. 25: p. 351−360.

75. Taylor, W. R., *A flexible method to align large numbers of biological sequences.* Journal of Molecular Evolution, 1988. 28: p. 161−169.

76. Higgins, D. G. and Sharp, P. M., *CLUSTAL: a package for performing multiple sequence alignment on a microcomputer.* Gene, 1988. 73: p. 237−244.

77. Higgins, D. G., Bleasby, A. J., and Fuchs, R., *CLUSTAL V: improved sofware for multiple sequence alignment.* Computer Applications in Biological Science, 1992. 8: p. 189−191.

78. Bucher, P., Karplus, K., Moeri, N., and Hofmann, K., *A flexible motif search technique based on generalized profiles.* Comput Chem, 1996. 20(1): p. 3−23.

79. Luthy, R., Xenarios, I., and Bucher, P., *Improving the sensitivity of the sequence profile method.* Protein Sci, 1994. 3(1): p. 139−46.

80. Rabiner, L. R., *A tutorial on hidden Markov Models and selected applications in speech recognition.* Proc. IEEE, 1989. 77: p. 257−286.

81. Krogh, A., Brown, M., Mian, I. S., Sjölander, K., and Haussler, D., *Hidden Markov Models in Computational Biology: Applications to Protein Modeling.* J. Mol. Biol., 1994. 235: p. 1501−1531.

82. Hughey, R. and Krogh, A., *Hidden Markov models for sequence analysis: extension and analysis of the basic method.* Computer Applications in Biological Science, 1996. 12: p. 95−107.

83. Lawrence, C. E., Altschul, S. F., Boguski, M. S., Liu, J. S., Neuwald, A. F., and Wootton, J. C., *Detecting Subtle Sequence Signals: A Gibbs Sampling Strategy for Multiple Alignment.* Science, 1993. 262: p. 208−214.

84. Bryant, S. H. and Altschul, S. F., *Statistics of sequence−structure threading.* Current Opinion in Structural Biology, 1995. 5: p. 236−244.

85. Crick, F. H. C., *Central dogma of molecular biology.* Nature, 1970. 227: p. 561−563.

86. Richards, R. G., *5 S RNA An analysis of Possible Base Pairing Schemes.* Eur J Biochem, 1969. 10: p. 36−42.

87. Gultayaev, A. P., van Batenburg, F. D. H., and Pleij, C. W. A., *The computer Simulation of RNA Folding Pathways Using a Genetic Algorithm.* J. Mol. Biol., 1995. 250: p. 37−51.

88. Shapiro, B. A. and Wu, J. C., *Predicting RNA HType pseudoknots with the massively parallel genetic algorithm.* Comp. Applic. in Biosci., 1997. 13(4): p. 459−471.

89. Zuker, M., *Computer prediction of RNA structure.* Meth. Enzymol., 1989. 180: p. 262−288.

90. Kim, J., Cole, J. R., and Pramanik, S., *Alignment of possible secondary structures in multiple RNA sequences using simulated annealing.* Comp. Applic. Biosci., 1996. 12(4): p. 259−267.

91. Eddy, S. R. and Durbin, R., *RNA aligment using covariance models.* Nucleic Acid Res., 1994. 22(11): p. 2079−2088.

92. Corpet, F. and Michot, B., *RNAlign program: alignment of RNA sequences using both primary and secondary structures.* Comp. Applic. Biosci., 1994. 10(4): p. 389−99.

93. Sakakibara, Y., Brown, M., Underwood, R. C., Mian, I. S., and Haussler, D. *Stochastic Context−Free Grammars for Modeling RNA*. in *27th Hawaii International Conference on System Sciences*. 1994. Wailea, HI, U.S.A.: Los Alamitos, CA: IEEE Computer Society Press.

94. Lefebvre, F. *An optimized parsing algorithm well suited to RNA folding*. in *ISMB−95*. 1995. Cambridge, England: AAAI Press.

95. Pleij, C., *Pseudoknots: a new motif in the RNA GAme.* Trends Biochem. Sci., 1990. 15: p. 143−147.

96. Westhof, E. and Jaeger, L., *RNA pseudoknots.* Current Opinion In Structural Biology, 1992. 2: p. 327− 333.

97. Notredame, C., O'Brien, E. A., and Higgins, D. G., *RAGA: RNA Sequence Alignment by Genetic Algorithm.* Nucleic Acids Res.(in press), 1997. .

98. Notredame, C. and Higgins, D. G., *SAGA: sequence alignment by genetic algorithm.* Nucleic Acids Res., 1996. 24: p. 1515−1524.

99. Hirschberg, D. S., *A linear space algorithm for computing maximal common subsequences.* Comm. ACM, 1975. 18: p. 341−343.

100. Myers, E. W. and Miller, W., *Optimal alignments in linear space.* Comput. Applic. Biosci., 1988. 4: p. 11−17.

101. Naor, D. and Brutlag, D. L., *On Near−Optimal Alignmnets of Biological Sequences.* Journal of Computational Biology, 1994. 1(4): p. 349−366.

102. Saqi, M. A. S. and Sternberg, M. J. E., *A simple method to generate non−trivial alternate alignments of protein sequences.* Journal of Molecular Biology, 1991. 219: p. 727−732.

103. Chao, K. M., *Computing all the suboptimal alignments in linear space.* Lecture Notes In Computer Science, 1994. 807: p. 31−42.

104. Vingron, M. and Argos, P., *Determination of reliable regions in protein sequence alignment.* Protein Eng., 1990. 3: p. 565−569.

105. Taylor, W. R. and Orengo, C. A., *Protein structure alignment.* Journal of Molecular Biology, 1989. 208: p. 1−22.

106. Sneath, P. H. A. and Sokal, R. R., *Numerical Taxonomy*. 1973, San Francisco: Freeman, W.H.

107. Barton, G. J. and Sternberg, M. J. E., *A strategy for the rapid multiple alignment of protein sequences: confidence levels from tertiary structure comparisons.* Journal of Molecular Biology, 1987. 198: p. 327−337.

108. Gusfield, D., *Efficient methods for multiple sequence alignment with guaranteed error bounds.* Bull. Math. Biol., 1993. 55: p. 141−154.

109. Gupta, S. K., Kececioglu, J. D., and Schaffer, A. A., *Improving the practical apace and time efficiency of the shortest−paths approach to sum−of−pairs multiple sequence alignment.* Journal of Computational Biology, 1995. 2(3): p. 459−472.

110. kececioglu, J. D., *The maximum weight trace problem in multiple sequence alignmnet.* Lecture Notes in Computer Science, 1993. 684: p. 106−119.

111. Reinert, K., Lenhof, H. P., Mutzel, P., Melhorn, K., and Kececioglu, J. D., *A branch−and−cut Algorithm for multiple sequence alignmnet.* Recomb97, 1997. : p. 241−249.

112. Kauffman, S. and Levin, S., *Toward a General Theory of Adaptative Walks on Rugged Landscapes.* J. Theor Biol, 1987. 128(1): p. 11−45.

113. Charleston, M. A., *Toward a characterisation of landscapes of combinatorial optimisation problems, with special attetion to the phylogeny problem.* Journal of Computational Biology, 1995. 2(3): p. 439−450.

114. Berger, M. P. and Munson, P. J., *A novel randomized iterative strategy for aligning multiple protein sequences.* Comput. Appl. Biosci., 1991. 7: p. 479−484.

115. Ishikawa, M., Toya, T., and Tokoti, Y. *Parallel Iterative Aligner with Genetic Algorithm*. in *Artifificial Intelligence and Genome Workshop, 13th International Conference on Artificial Intelligence*. 1993. Chambery, France:

116. Hirosawa, M., Totoki, Y., Hoshida, M., and Ishikawa, M., *Comprehensive study on iterative algorithms of multiple sequence alignments.* Comp. App. Biosci., 1995. 11(1): p. 13−18.

117. Gotoh, O., *Significant Improvement in Accuracy of Multiple Protein Sequence Alignments by Iterative Refinements as Assessed by Reference to Structural Alignments.* J. Mol. Biol., 1996. 264(4): p. 823−838.

118. Geman, D. and Geman, D., *Gibbs Sampling.* Trans. Pattern Anal. Mach. Intell, 1984. 6: p. 721.

119. Metropolis, N., Rosenbluth, A. W., Rosenbluth, M. N., Teller, A. H., and Teller, E., *Equation of state calculations by fast computing machines.* J. Chem. Phys., 1953. 21: p. 1087−1092.

120. Kirkpatrick, S., Gelatt, C. D. J., and Vecchi, M. P., *Optimization by Simulated Annealing.* Science, 1983. 220: p. 671−680.

121. Ingber, L. and Rosen, B., *Genetic Algorithm and Very Fast simulated Reannealing: a comparison.* Mathematical Computer Modeling, 1993. 16: p. 87−100.

122. Ishikawa, M., Toya, T., Hoshida, M., Nitta, K., Ogiwara, A., and Kanehisa, M., *Multiple sequence alignment by parallel simulated annealing.* Comp. Applic. Biosci., 1993. 9: p. 267−273.

123. Kim, J., Pramanik, S., and Chung, M. J., *Multiple Sequence Alignment using Simulated Annealing.* Comp. Applic. Biosci., 1994. 10(4): p. 419−426.

124. Godzik, A. and Sander, C., *Conservation of residue interactions in a family of Ca−binding proteins.* Prot. Eng., 1989. 2: p. 589−96.

125. Schmitz, M. and Steger, G., *Description of RNA Folding by Simulate Annealing.* J. Mol. Biol, 1996. 255: p. 254−266.

126. Eddy, S. R. *Multiple alignment using hidden Markov models*. in *Third International converence on Intelligent Systems for Molecular Biology (ISMB)*. 1995. Cambridge, England: Menlo Park, CA: AAAI Press.

127. Holland, J. H., *Adaptation in natural and artificial systems*. 1975, Ann Arbour, MI: University of Michigan Press.

128. Goldberg, D. E., *Genetic Algorithms in Search, Optimization, and Machine Learning*. ed. D.E. Goldberg. 1989, New York: Addison−Wesley.

129. Davis, L., *The handbook of Genetic Algorithms*. ed. L. Davis. 1991, New York: Van Nostrand Reinhold.

130. Rabow, A. A. and Scheraga, H. A., *Improved genetic algorithm for the protein folding problem by use of a cartesian combination operator.* Protein Science, 1996. 5: p. 1800−1815.

131. Pedersen, J. T. and Moult, J., *Ab initio structure prediction for small polypeptides and protein fragments using genetic algorithms.* Proteins: Structure, Function, and Genetics, 1995. 23: p. 454−460.

132. Legrand, S. M. and Merz, K. M., *The genetic Algorithm and the conformational search of polypeptides and proteins.* Molecular Simulation, 1994. 13: p. 299−320.

133. Unger, R. and Moult, J., *Genetic Algorithms for Protein Folding Simulations.* J. Mol. Biol., 1993. 231: p. 75−81.

134. Schulze−Kremer, S., *Genetic algorithms for protein tertiary structure prediction*, in *Proceedings of the 2nd Conference on Parallel Problem Solving from Nature*, R. Männer and B. Manderick, Editor. 1992, Elsevier Science Publishers: Amsterdam. p. 391−400.

135. Sun, S., *Reduced representation model of protein structure prediction: statistical potential and genetic algorithms.* Protein Science, 1993. 2(5): p. 762−785.

136. Dandekar, T. and Argos, P., *Potential of genetic algorithms in protein folding and engineering simulations.* Protein Eng., 1992. 5: p. 637−645.

137. Verkhivler, G. M., Rejto, P. A., Gelhaar, D. K., and Freer, S. T., *Exploring the energy landscape of molecular recognition by a genetic algorithm: analysis of the*

requirement for robust docking of HIV−1 Protease and FKBP−2 Complexes. Proteins:Structure, Function and Genetics, 1996. 250: p. 342−353.

138. Shapiro, B. A. and Wu, J. C., *An annealing mutation operator in the genetic algorithm for RNA folding.* Comp. Applic. Biosci., 1996. 12(3): p. 171−180.

139. Ogata, H., Yutaka, A., and Minoru, K., *A genetic algorithm based molecular modeling technique for RNA stem−loop structures.* Nucleic Acids res., 1995. 23(3): p. 419−426.

140. McClure, M. A., Vasi, T. K., and Fitch, W. M., *Comparative analysis of multiple protein−sequence alignmnent methods.* Molecular Biology Evolution, 1994. 11(4): p. 571−592.

141. Subbiah, S. and Harrison, S. C., *A method for multiple sequence alignments with gaps.* J. Mol. Biol., 1989. 209: p. 539−548.

142. Sali, A. and Overington, J. P., *Derivation of Rules for comparative protein modeling from a database of protein structure alignments.* Protein Sci., 1994. 3: p. 1582−1596.

143. Holm, L. and Sander, C., *The FSSP database: fold classification based on structure−structure alignment of proteins.* Nucleic Acids Res., 1996. 24: p. 206−210.

144. Barton, G. J., *Scop: structural classification of proteins.* Trends Biochem Sci, 1994. 19(12): p. 554−5.

145. Gotoh, O., *A weighting System and Algorithm for aligning many phylogenetically related Sequences.* Comput. App. in Biosci., 1995. 11(543−551).

146. Tabaska, E. J. and D., S. G. *Automated alignment of RNA sequences to pseudoknotted structures*. in *ISMB−97*. 1997. Halkidiki, Greece: AAAI Press.