CS 594 : Introduction to Computational Biology

Applications of Hidden Markov Models (HMMs) to Computational Biology problems

- *A Report*

Group 4

Shalini Venkataraman

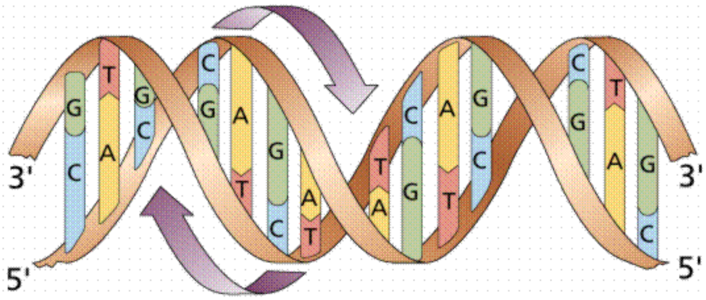
Vidya Gunaseelan

Thursday, Apr 25

This report examines the role of a powerful statistical model called **Hidden Markov Models** (HMM) in the area of computational biology. We will start with an overview of HMMs and some concepts in biology. Next, we will discuss the use of HMMs for biological sequences and finally conclude with a discussion on the advantages and limitations of HMMs and possible future work.

1. Biological Background

1.1. DNA - Deoxyribonucleic Acid



In humans, as in other higher organisms, a DNA molecule consists of two strands that wrap around each other to resemble a twisted ladder whose sides, made of sugar and phosphate molecules, are connected by rungs of nitrogen containing chemicals called bases. Four different bases are present in DNA: adenine (A), thymine (T), cytosine (C), and guanine (G). The particular order of the bases arranged along the sugar-phosphate backbone is called the DNA sequence; the sequence specifies the exact genetic instructions required to create a particular organism with its own unique traits. The two DNA strands are held together by weak bonds between the bases on each strand, forming base pairs (bp). Genome size is usually stated as the total number of base pairs; the human genome contains roughly 3 billion bp. A gene is a segment of a DNA molecule (ranging from fewer than 1 thousand bases to several million), located in a particular position on a specific chromosome, whose base sequence contains the information necessary for protein synthesis.

1.2. RNA

RNA has the same structure as DNA. The primary differences between RNA and DNA are:

RNA has a hydroxyl group on the second carbon of the sugar and instead of using nucleotide thymine, RNA uses another nucleotide called uracil (U). Since RNA has extra hydroxyl group on it's sugar strand, RNA is too bulky to form a stable double helix therefore it exists as a single-stranded molecule. In addition to that, because the RNA molecule is not restricted to a rigid double helix, it can form many different structures. There are several different kinds of RNA made by the cell. They are mRNA, tRNA, rRNA and snRNA.

1.3. Proteins

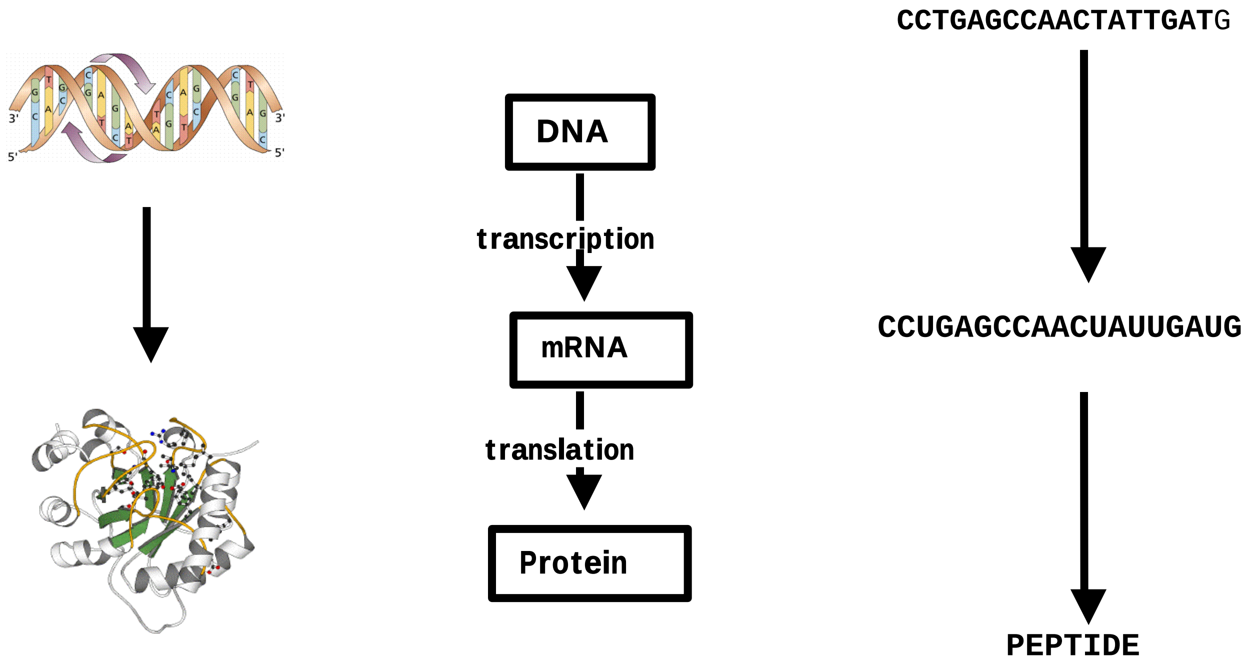
Proteins are involved in almost all biological activities, structural or enzymatic. A protein is made by arranging amino acids together in a specific sequence (the sequence of every protein is different). These amino acids are held together by a special bond called a peptide bond. There are altogether 20 different amino acids.

1.4. The Central Dogma Of Molecular Biology

How does the sequence of a strand of DNA correspond to the amino acid sequence of a protein? This concept is explained by the **central dogma of molecular biology**, according to which

- The DNA replicates its information in a process called **replication** that involves many enzymes.
- The DNA codes for the production of messenger RNA (mRNA) during **transcription.** In eukaryotic cells, the mRNA is processed (essentially by **splicing**) and migrates from the nucleus to the cytoplasm.
- Messenger RNA carries coded information to ribosomes. The ribosomes "read" this information and use it for protein synthesis. This process is called **translation**.

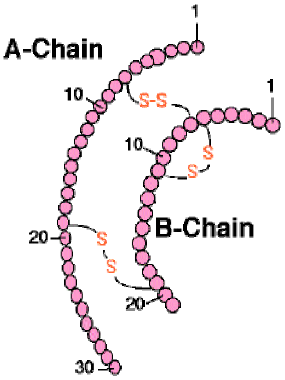
Diagrammatically,



1.5. Protein structure

A striking characteristic of proteins is that they have very well defined 3-D structures. A stretched-out polypeptide chain has no biological activity, and protein function arises from the conformation of the protein, which is the 3-D arrangement or shape of the molecules in the protein. The native conformation of a protein is determined by a number of factors, and the most important are the 4 levels of structure found in proteins. **Primary**, **secondary** and **tertiary** refer to the molecules in a single polypeptide chain, and the fourth (**quaternary**) refers to the interaction of several polypeptide chains to form a multi-chained protein. In this paper, we limit our discussion to just the primary and secondary structure.

Primary Structure

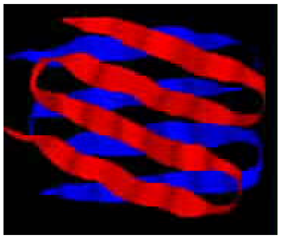


The primary structure of a protein is determined by the number and order of amino acids within a polypeptide chain. A polypeptide is a sequence of two or more amino acids joined together by peptide bonds. Determination of primary structure is an essential step in the characterization of a protein.

Secondary Structure

Protein secondary structure refers to regular, repeated patters of folding of the protein backbone. The two most common folding patterns are the alpha helix and the beta sheet. Patterns result from regular hydrogen bond patterns of backbone atoms.

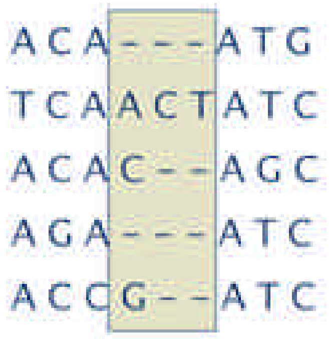
In the alpha helix, the polypeptide folds by twisting into a right handed screw so that all the amino acids can form hydrogen bonds with each other. This high amount of hydrogen bonding stabilizes the structure so that it forms a very strong rod-like structure.



The beta-pleated sheet is substantially different from the alpha-helix in that it is a sheet rather than a rod and polypeptide chain is fully stretched rather than tightly coiled as in helix. It is called a beta-pleated sheet because of zig zag appearance when viewed from the side.

The **tertiary structure** of a protein is formed when the attractions of side chains and those of the secondary structure combine and cause the amino acid chain to form a distinct and unique 3-dimensional structure. It is this unique structure that gives a protein its specific function.

1.6. Multiple Sequence Alignment



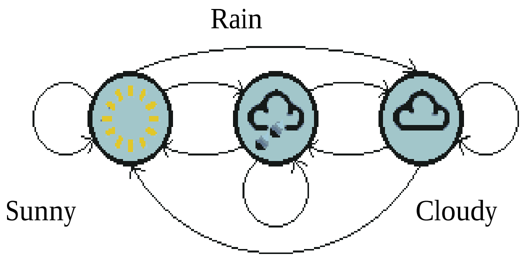
```
ACA---ATG
TCAACTATC
ACAC--AGC
AGA---ATC
ACCG--ATC
```

Multiple alignment is the process of aligning two or more sequences with each other in order to determine any evolutionary relationships. For aligning two sequences the dynamic programming approach is the most suitable. This approach can be generalized for multiple sequence alignment also. But for a large number of sequences this approach becomes impractical. There are heuristic methods available to speed up the dynamic programming approach like the local multiple alignment using the Sum of Pairs scoring function. In our treatise, we will show how HMMs can be effective in solving this problem.

2. Hidden Markov Model (HMM) Architecture

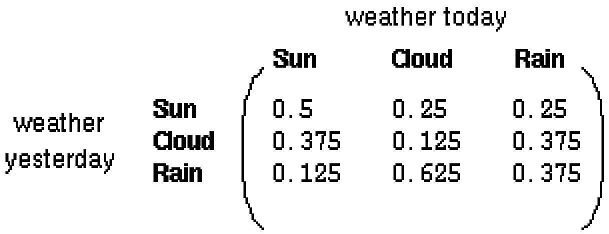
2.1. Markov Chains

Let the three states of weather be Sunny, Cloudy and Rainy. We cannot expect these three weather states to follow each other deterministically, but we might still hope to model the system that generates a weather pattern. One way to do this is to assume that the state of the model depends only upon the previous states of the model. This is called the **Markov assumption** and simplifies problems greatly. When considering the weather, the Markov assumption presumes that today's weather can always be predicted solely given knowledge of the weather of the past few days.



A Markov process is a process, which moves from state to state depending (only) on the previous *n* states. The process is called an *order n* model where *n* is the number of states affecting the choice of next state. The simplest Markov process is a first order process, where the choice of state is made purely on the basis of the previous state. This figure shows all possible first order transitions between the states of the weather example.

The state transition matrix below shows possible transition probabilities for the weather example;



$$\begin{array}{llccc}
 & & & \text{weather today} & \\
 & & \textbf{Sun} & \textbf{Cloud} & \textbf{Rain} \\
\text{weather yesterday} & \textbf{Sun} & 0.5 & 0.25 & 0.25 \\
 & \textbf{Cloud} & 0.375 & 0.125 & 0.375 \\
 & \textbf{Rain} & 0.125 & 0.625 & 0.375
\end{array}$$

that is, if it was sunny yesterday, there is a probability of 0.5 that it will be sunny today, and 0.25 that it will be cloudy or rainy.

To initialize such a system, we need to state what the weather was (or probably was) on the day after creation; we define this in a vector of initial probabilities, called the π vector.

$$\begin{array}{ccc} \textbf{Sun} & \textbf{Cloud} & \textbf{Rain} \\ \left(1.0\right. & 0.0 & \left.0.0\right) \end{array}$$

So, we know it was sunny on day 1.

We have now defined a **first order Markov process** consisting of :

- **states** : Three states - sunny, cloudy, rainy.
- **πvector** : Defining the probability of the system being in each of the states at time 0.
- **state transition matrix** : The probability of the weather given the previous day's weather.

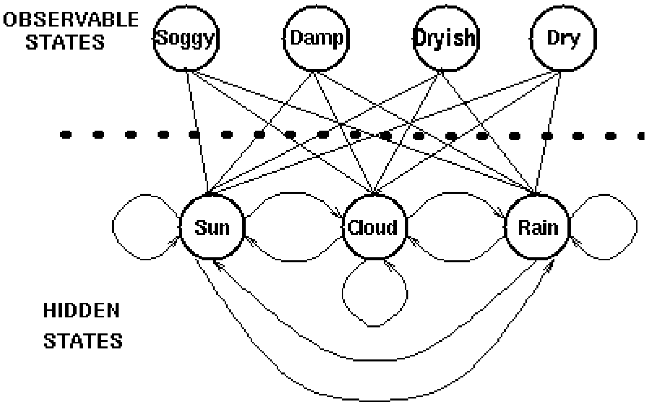
Any system that can be described in this manner is a Markov process.

2.2. Hidden Markov Models

In some cases the patterns that we wish to find are not described sufficiently by a Markov process. Returning to the weather example, a hermit for instance may not have access to direct weather observations, but does have a piece of seaweed. Folklore tells us that the state of the seaweed is probabilistically related to the state of the weather - the weather and seaweed states are closely linked. In this case we have two sets of states

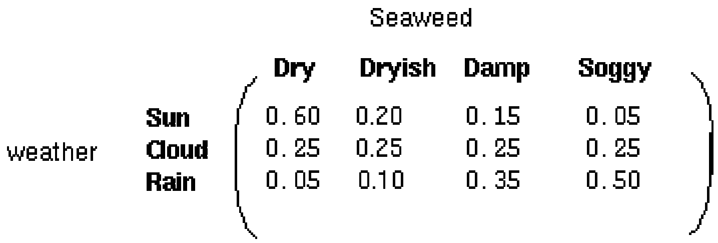
- **observable** states (the state of the seaweed) and
- **hidden** states (the state of the weather).

We wish to devise an algorithm for the hermit to forecast weather from the seaweed and the Markov assumption without actually ever seeing the weather. The diagram below shows the hidden and observable states in the weather example. It is assumed that the hidden states (the true weather) are modeled by a simple first order Markov process, and so they are all connected to each other.



The connections between the hidden states and the observable states represent the probability of generating a particular observed state given that the Markov process is in a particular hidden state. It should thus be clear that all probabilities `entering' an observable state will sum to 1, since in the above case it would be the sum of *Pr*(Obs|Sun), *Pr*(Obs|Cloud) and *Pr*(Obs|Rain).

In addition to the probabilities defining the Markov process, we therefore have another matrix, termed the **output matrix**, which contains the probabilities of the observable states given a particular hidden state. For the weather example the output matrix might be;



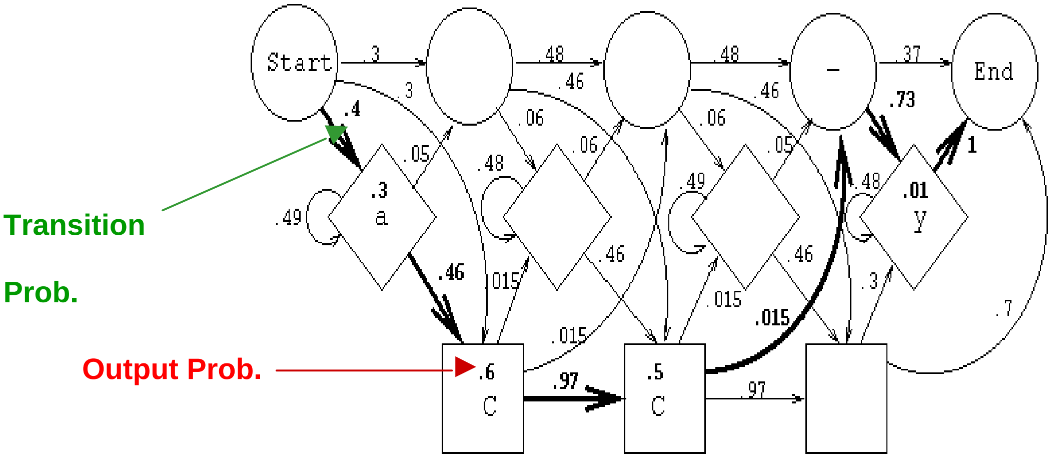
$$\text{weather}\quad \begin{array}{c|cccc} & \multicolumn{4}{c}{\text{Seaweed}} \\ & \textbf{Dry} & \textbf{Dryish} & \textbf{Damp} & \textbf{Soggy} \\ \textbf{Sun} & 0.60 & 0.20 & 0.15 & 0.05 \\ \textbf{Cloud} & 0.25 & 0.25 & 0.25 & 0.25 \\ \textbf{Rain} & 0.05 & 0.10 & 0.35 & 0.50 \end{array}$$

So, this is a model containing three sets of probabilities in addition to the two sets of states

- **πvector** : contains the probability of the hidden model being in a particular hidden state at time t= 1.
- **state transition matrix** : holding the probability of a hidden state given the previous hidden state.
- **output matrix** : containing the probability of observing a particular observable state given that the hidden model is in a particular hidden state.

Thus a hidden Markov model is a standard Markov process augmented by a set of observable states, and some probabilistic relations between them and the hidden states.

2.3. An example of a HMM for Protein Sequences



This is a possible hidden Markov model for the protein ACCY. The protein is represented as a sequence of probabilities. The numbers in the boxes show the probability that an amino acid occurs in a particular state, and the numbers next to the directed arcs show probabilities, which connect the states. The probability of ACCY is shown as a highlighted path through the model. There are three kinds of states represented by three different shapes. The squares are called **match states**, and the amino acids emitted from them form the conserved primary structure of a protein. These amino acids are the same as those in the common ancestor or, if not, are the result of substitutions. The diamond shapes are **insert states** and emit amino acids that result from insertions. The circles are special, silent states known as **delete states** and model deletions. These type of HMMs are called Protein Profile-HMMs and will be covered in more depth in the later sections.

Scoring a Sequence with an HMM

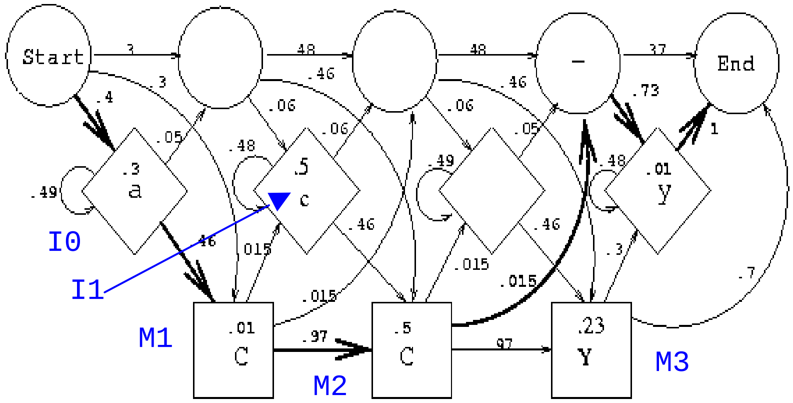
Any sequence can be represented by a path through the model. The probability of any sequence, given the model, is computed by multiplying the **emission** and **transition** probabilities along the path. A path through the model represented by ACCY is highlighted. For example, the probability of A being emitted in position 1 is 0.3, and the probability of C being emitted in position 2 is 0.6. The probability of ACCY along this path is

.4*.3*.46*.6*.97*.5*.015*.73*.01*1 = 1.76x10^{-6}.

2.4. Three Problems Of Hidden Markov Models

1) Scoring Problem

We want to find the probability of an observed sequence given an HMM. It can be seen that one method of calculating the probability of the observed sequence would be to find each possible sequence of the hidden states, and sum these probabilities. We use the Forward Algorithm for this.

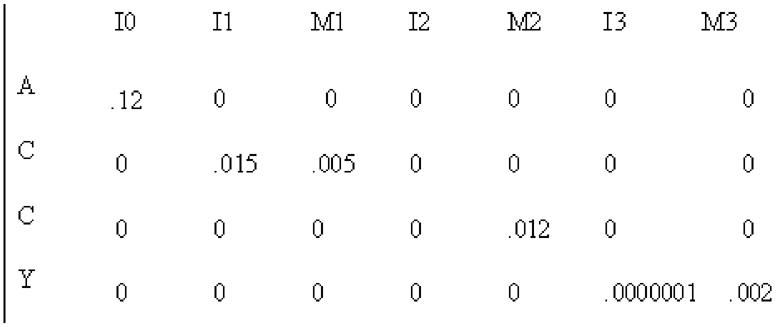


Consider the HMM shown above. In this figure several paths exist for the protein sequence ACCY.

The **Forward algorithm** employs a matrix, shown below. The columns of the matrix are indexed by the states in the model, and the rows are indexed by the sequence. The elements of the matrix are initialized to zero and then computed with these steps:

1. The probability that the amino acid *A* was generated by state I0 is computed and entered as the first element of the matrix. This is .4*.3 = .12
2. The probabilities that C is emitted in state M1 (multiplied by the probability of the most likely transition to state M1 from state I0) and in state I1 (multiplied by the most likely transition to state I1 from state I0) are entered into the matrix element indexed by C and I1/M1.
3. The sum of the two probabilities, sum(I1, M1), is calculated.
4. A pointer is set from the winner back to state I0.
5. Steps 2-4 are repeated until the matrix is filled.

The probability of the sequence is found by summing the probabilities in the last column.



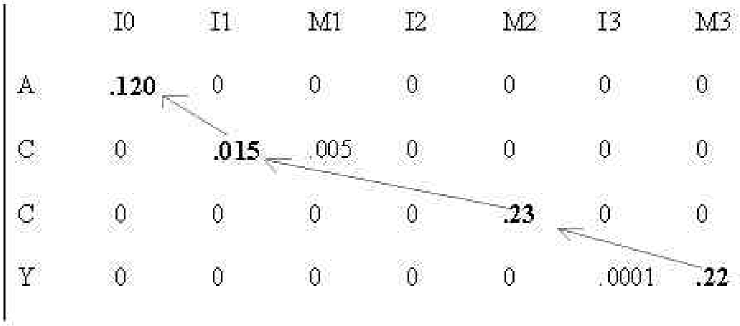
	I0	I1	M1	I2	M2	I3	M3
A	.12	0	0	0	0	0	0
C	0	.015	.005	0	0	0	0
C	0	0	0	0	.012	0	0
Y	0	0	0	0	0	.0000001	.002

Matrix for the Forward algorithm

2) Alignment Problem

We often wish to take a particular HMM, and determine from an observation sequence the most likely sequence of underlying hidden states that might have generated it. This is the alignment problem and the **Viterbi Algorithm** is used to solve this problem.

The Viterbi algorithm is similar to the forward algorithm. However in step 3, maximum rather than a sum is calculated. The most likely path through the model can now be found by following the back-pointers.



	I0	I1	M1	I2	M2	I3	M3
A	**.120**	0	0	0	0	0	0
C	0	**.015**	.005	0	0	0	0
C	0	0	0	0	**.23**	0	0
Y	0	0	0	0	0	.0001	**.22**

Matrix for the Viterbi algorithm

Once the most probable path through the model is known, the probability of a sequence given the model can be computed by multiplying all probabilities along the path.

3) Training Problem

Another tricky problem is how to create an HMM in the first place, given a particular set of related training sequences. It is necessary to estimate the amino acid emission distributions in each state and all state-to-state transition probabilities from a set of related training sequences. This is done by using the Baum-Welch Algorithm or the Forward Backward Algorithm.

The algorithm proceeds by making an initial guess of the parameters (which may well be entirely wrong) and then refining it by assessing its worth, and attempting to reduce the errors it provokes when fitted to the given data. In this sense, it is performing a form of gradient descent, looking for a minimum of an error measure.

3. Applications of HMM’s

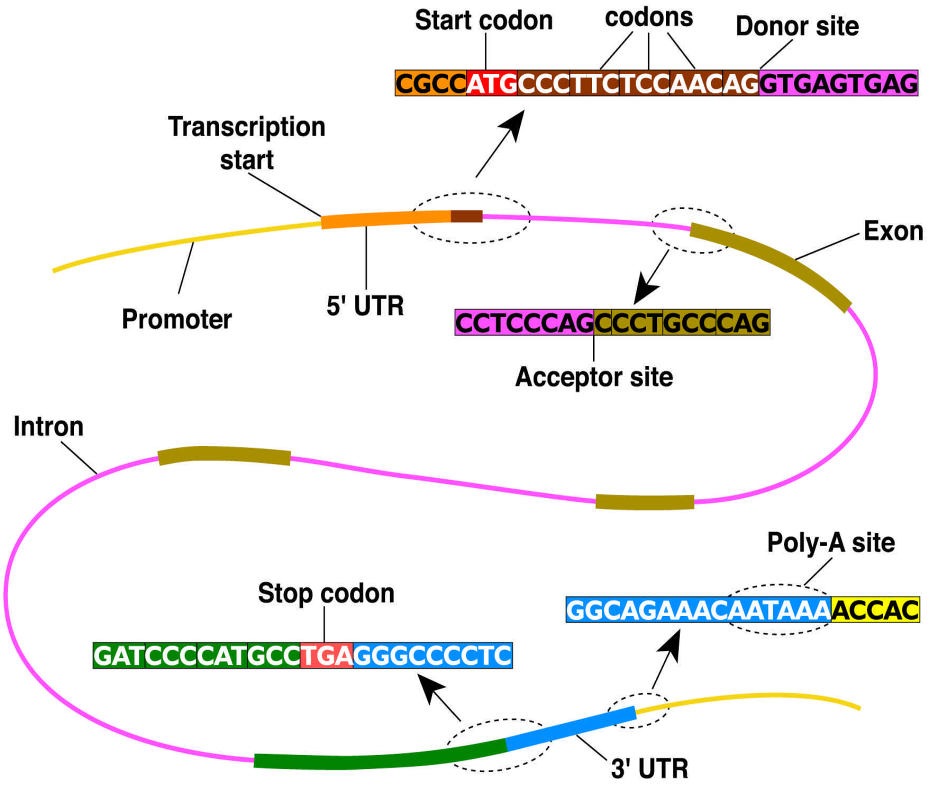
In this section, we will delve into greater depth at specific problems in the area of computational biology and examine the role of HMM's.

3.1. Gene finding and prediction

We introduce here the gene-prediction HMMs that can be used to predict the structure of the gene. Our objective is to find the coding and non-coding regions of an unlabeled string of DNA nucleotides.

The motivation behind this is to

- assist in the annotation of genomic data produced by genome sequencing methods
- gain insight into the mechanisms involved in transcription, splicing and other processes



As shown in the diagram above, a string of DNA nucleotides containing a gene will have separate regions

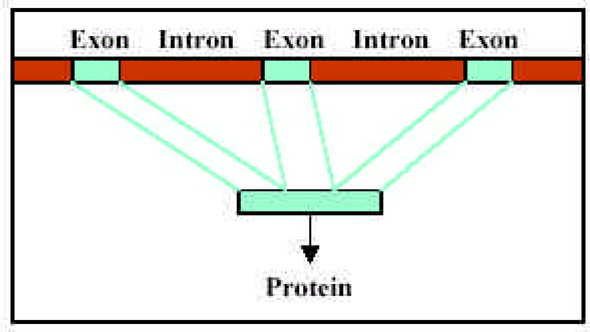
- Introns – non-coding regions within a gene

- Exons – coding regions

These regions are separated by functional sites

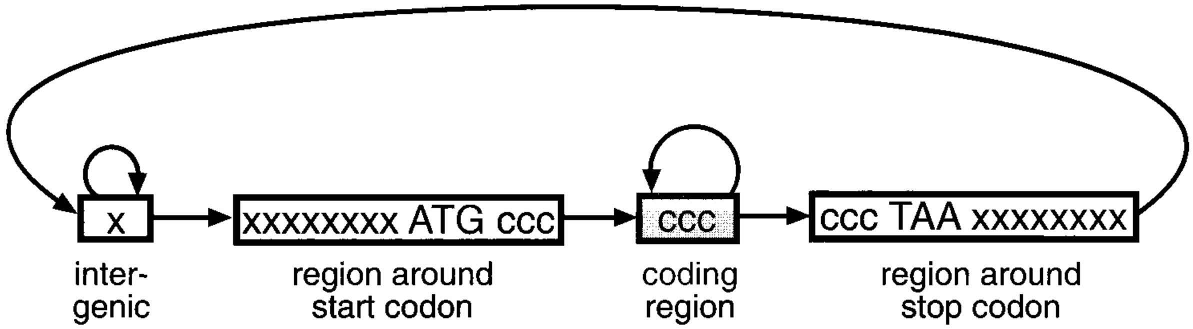
- Start and stop codons
- Splice sites – acceptors and donors

In the process of transcription, only the exons are left to form the protein sequence as depicted below.



Many problems in biological sequence analysis have a grammatical structure . HMMs are very useful in modeling grammar. The input to such a HMM is the genomic DNA sequence and the output, in the simplest case is a parse tree of exons and introns on the DNA sequence.

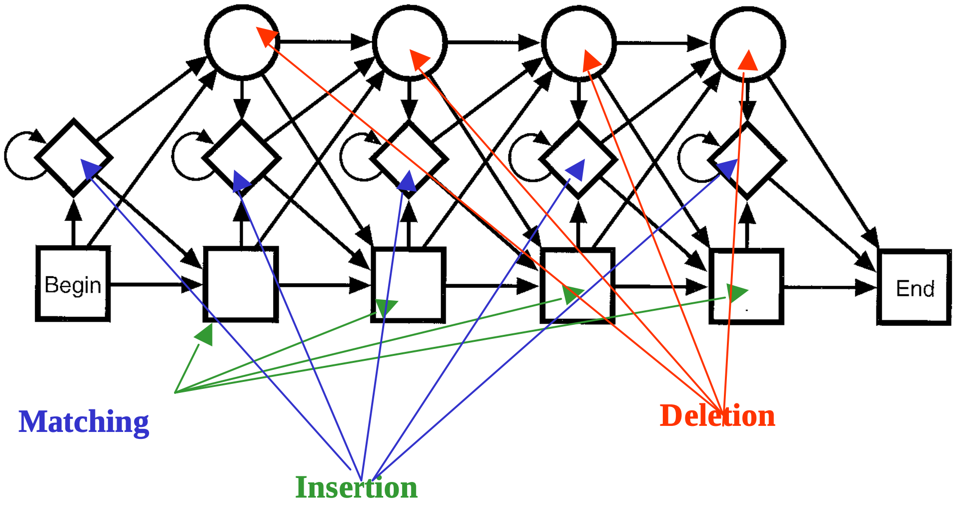
Shown below is a simple model for unspliced genes that recognizes the start codon, stop codon (only one of the three possible stop codons are shown) and the coding/non-coding regions. This model has been trained with a test set of gene data.



Having such a model, how can we predict genes in a sequence of anonymous DNA ? We simply use the Viterbi algorithm to find the most probable path through the model

3.2. Protein- Profile HMMs

As we have seen earlier, protein structural similarities make it possible to create a statistical model of a protein family which is called a **profile**. The idea is, given a single amino acid target sequence of unknown structure, we want to infer the structure of the resulting protein. The profile HMM is built by analyzing the distribution of amino-acids in a training set of related proteins. This HMM in a natural way can model positional dependant gap penalties.



The basic topology of a profile HMM is shown above. Each position, or module, in the model has three states. A state shown as a rectangular box is a **match** state that models the distribution of letters in the corresponding column of an alignment. A state shown by a **diamond**-shaped box models insertions of random letters between two alignment positions, and a state shown by a circle models a **deletion**, corresponding to a gap in an alignment. States of neighboring positions are connected, as shown by lines. For each of these lines there is an associated `transition probability', which is the probability of going from one state to the other.

The match state represents a consensus amino acid for this position in the protein family. The delete state is a non-emitting state, and represents skipping this consensus position in the multiple alignment. Finally, the insert state models the insertion of any number of residues after this consensus position.

A repository of protein profile HMMs can be found in the PFAM Database (**http://pfam.wustl.edu**). Building profiles from a family of proteins(or DNA) a profile HMM can be made for searching a database for other members of the family. As we have seen before in the section on HMM problems, profile HMM's can also be used for the following

Scoring a sequence

We are calculating the probability of a sequence given a profile by simply multiplying emmision and transition probabilities along the path.

Classifying sequences in a database

Given a HMM for a protein family and some unknown sequences, we are trying to find a path through the model where the new sequence fits in or we are tying to 'align' the sequence to the model. Alignment to the model is an assignment of states to each residue in the sequence. There are many such alignments and the Vitterbi's algorithm is used to give the probability of the sequence for that alignment.

Creating Multiple sequence alignment

HMMs can be used to automatically create a multiple alignment from a group of unaligned sequences. By taking a close look at the alignment, we can see the history of evolution. One great advantage of HMMs is that they can be estimated from sequences, without having to align the sequences first. The sequences used to estimate or **train** the model are called the training sequences, and any reserved sequences used to evaluate the model are called the test sequences. The model estimation is done with the forward-backward algorithm, also known as the Baum-Welch algorithm. It is an iterative algorithm that maximizes the likelihood of the training sequences.

3.3. Prediction of protein secondary structure using HMM's

Prediction of secondary structures is need for the prediction of protein function. As an alternative method to direct X-ray analysis, a HMM is used to

- Analyze the amino-acid sequences of proteins
- Learn secondary structures such as helix, sheet and turn
- Predict the secondary structures of sequences

The method is to train the four HMMs of secondary structure – helix, sheet, turn and other – by training sequences. The Baum-Welch method is used to train the HMMs. So, the HMM of helix is able to produce helix-like sequences with high probabilities. Now, these HMMs can be used to predict the secondary structure of the test sequence. The forward-backward algorithm is used to compute the probabilities of these HMMs outputting the test sequence. The sequence has the secondary structure whose HMM showed the highest probability to output the sequence.

4. HMM implementation

These are the two publicly available HMM implementation software.

HMMER - http://hmmer.wustl.edu/

SAM system - http://www.cse.ucsc.edu/research/compbio/sam.html

5. Advantages of HMMs

- HMM's can accommodate variable-length sequence.

Because most biological data has variable-length properties, machine learning techniques which require a fixed-length input, such as neural networks or support vector machines, are less successful in biological sequence analysis

- Allows position dependant gap penalties.

HMM's treat insertions and deletions is a statistical manner that is dependant on position.

6. Limitations of HMMs

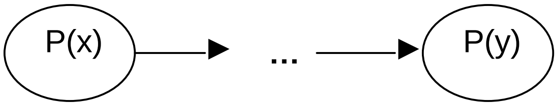
- Linear Model

So, they are unable to capture higher order correlations among amino-acids.

- Markov Chain assumption of independent events

Probabilities of states are supposed to be independent which is not true of biology

Eg, P(y) must be independent of P(x), and vice versa



- Standard Machine Learning Problems

In the training problem, we need to watch out for local maxima and so model may not converge to a truly optimal parameter set for a given training set. Secondly, since the model is only as good as your training set, this may lead to over-fitting.

7. Open areas for research in HMMs in biology

- Integration of structural information into profile HMMs.

Despite the almost obvious application of using structural information on a member protein family when one exists to better the parameterization of the HMM, this has been extremely hard to achieve in practice.

- Model architecture

The architectures of HMMs have largely been chosen to be the simplest architectures that can fit the observed data. Is this the best architecture to use? Can one use protein structure knowledge to make better architecture decisions, or, in limited regions, to learn the architecture directly from the data? Will these implied architectures have implications for our structural understanding?

- Biological mechanism

In gene prediction, the HMM's may be getting close to replicating the same sort of accuracy as the biological machine (the HMM's have the additional task of finding the gene in the genomic DNA context, which is not handled by the biological machine that processes the RNA). What constraints does our statistical model place on the biological mechanism— in particular, can we consider a biological mechanism that could use the same information as the HMM?

8. References

1. L. R. Rabiner and B. H. Juang, *An Introduction to Hidden Markov Models,* IEEE ASSP Magazine, January 1986, pp. 1-16.
2. K. Asai, S. Hayamizu and H. Handa, *Prediction of protein secondary structures by hidden Markon models*, Computer Application in the Biosciences (CABIOS), Vol 9, No 2, 1993, pp. 141-146.
3. Krogh, A., M. Brown, I.S. Mian, K. Sjolander, and D. Haussler. *Hidden Markov Models in Computational Biology: Applications to Protein Modeling*, J. Mol. Biol., Vol. 235, pp.1501-1531, 1994.
4. S. Eddy. *Profile hidden Markov model*s. Bioinformatics, 14:755--763, 1998.
5. L. R. Rabiner, *A Tutorial on Hidden Markov Models and Selected Applications in Speech Recognition* , Proceedings of the IEEE, 77 , no. 2, 257--285, February 1989.
6. Baldi, P., Chauvin, Y., Hunkapiller, T. and McClure, M.A. (1993) *Hidden Markov Models in Molecular Biology: New Algorithms and Applications*, In Advances in Neural Information Processing Systems 5, Eds. S.J. Hanson, J.D. Cowan and C. Lee Giles, (Morgan Kaufmann) pp 747-754
7. Baldi, P., Chauvin, Y., Hunkapiller, T., and McClure, M.A. (1994) *Hidden Markov Models of Biological Primary Sequence Information*, Proceedings of the National Academy of Science, USA 91: 1059-1063.
8. David Kulp, David Haussler, Martin G. Reese, and Frank H. Eeckman, *A generalized hidden markov model for the recognition of human genes in DNA*, Procedings of the Fourth International Conference on Intelligent Systems for Molecular Biology (Menlo Park, CA), AAAI Press, 1996.

9. R. Hughey and A. Krogh. Hidden Markov models for sequence analysis: extension and analysis of the basic method. *Computer Applications in the Biosciences,* 12:95-107.1996 http://www.cse.ucsc.edu/research/compbio/html_format_papers/hughkrogh96/cabios.html

10. Henderson,J., Salzberg,S. and Fasman,K. *Finding genes in human DNA with a hidden Markov model*. Journal of Computational Biology 1997, 4, 127--141.

11. *An Introduction to Hidden Markov Models for Biological Sequences* by A. Krogh. In S. L. Salzberg et al., eds., Computational Methods in Molecular Biology, 45-63. Elsevier, 1998.

12. E Birney. Hidden Markov models in biological sequence analysis. Deep computing for the life sciences. Volume 45, Numbers ¾ 2001. http://www.research.ibm.com/journal/rd/453/birney.html

13. HMMer - http://hmmer.wustl.edu/

14. SAM system - http://www.cse.ucsc.edu/research/compbio/sam.html