Hidden Markov Models in Bioinformatics

Alexander J Ohrt - UPC BarcelonaTech

December 3, 2021

Contents

1 Introduction

These two first parts should be reviewed (perhaps written when done with the rest of the report) This report will present the uses of Hidden Markov Models (HMMs) in Bioinformatics. It will explain and highlight why they are populaar alternative tools for solving a wide range of problems in the field. In order to understand why HMMs are useful, and why they can be used effectively, some background on stochastic processses and Markov Chains is needed. Moreover, for completeness, a brief theoretical construct of some different HMMs will be done. In the remaining part of the report, a range of problems in Bioinformatics will be presented, first solved without the use of HMMs, then contrasted with the HMM based solutions. This will hopefully highlight why HMMs are effective alternatives. Note that this report is by no means extensive; there exists a plethora of theory and problems where the HMMs can be used.

2 Objectives

The main objective of this report is to explain why HMMs are very widely used in Bioinformatics. Several problems in the field of study will be presented, with common solutions. First of all, solutions that are not based on HMMs will be shown, with their strengths and weaknesses. These will be contrasted with solution based on HMMs, which in all cases yield alternative methods which oftentimes perform better in some respect. For completeness, the reader should get a quick introduction to the vast theoretical background for HMMs, which includes a quick look at Markov Chains, the Hidden Markov Models and different types of them. Furthermore, the most important algorithms for working with HMMs in practice are presented. For the experienced reader, the section on the theoretical background may be skipped. Finally, some practical examples in the open-source programming language R will be developed.

3 Theoretical Background

Basic knowledge in statistics and probability theory is assumed. Any introductory book on probability and statistics will do, but a quick refresher can be found in the two first chapters of [7].

3.1 Markov Chains

A Markov Chain (MC) $\mathbf{X} = \{X_t\}$ is a stochastic process which is characterized by one important property; given the value of X_t, the values of X_b, $b > t$ are not influenced by the values of X_a, $a < t$, where $a, b \in \mathbb{Z}$ decide what *order* of MC we are working with. Choosing $b = t + 1$ and $a = t - 1$ yields the type of *Markov property* we will be concentrating on in this case, which is referred to as a *first-order* Markov property. More precisely, we have that

$$\mathrm{P}(X_{t+1} = j | X_0 = i_0, X_1 = i_1, \ldots X_{t-1} = i_{t-1}, X_t = i) = \mathrm{P}(X_{t+1} = j | X_t = i) =: \mathrm{P}_{ij}^{t,t+1},$$

where the superscripts highlight an important point; the *one-step transition probability* does not only depend on the state, but also on the time when the transition occurs. In our case, we will concentrate on *time-homogeneous* MCs, which means that the transition probabilites are stationary in time. Thus, we drop the superscripts in the notation and denote the transition probabilities as

$$\mathrm{P}_{ij} = \mathrm{P}(X_{t+1} = j | X_t = i).$$

In a MC, the random variable X can only take values from a pre-specified set called *states*. We will concentrate on the simplest type of state space, which yields a so-called *discrete-time* MC. This is the case when the set of states is finite or countable set and whose time index is $T = (0, 1, 2, \ldots)$. Thus,

we have described the assumptions made when working with the simplest form of MC, which we call a *discrete-state time-homogeneous first-order MC* [7]. These lay the foundation of the Hidden Markov Models that we will study in the remainder of the report.

Let us define the notation that will be used in the remainder. First of all, let the random vector $\mathbf{X} = (X_0, X_1, \ldots, X_L)^T$ represent a MC of length $L+1$. In the following we will always assume that $\mathbf{X}$ is of finite length $L+1$. A realization of the random vector will be denoted in lower case, i.e. $\mathbf{x} = (x_0, x_1, \ldots, x_L)^T$, following standard notation in most probability or statistics courses. What exactly this means will become clear later. A discrete-time MC consists of a finite set of states, which will be denoted by $\mathcal{H} = \{h_1, \ldots, h_m\}$, where $m = \text{card}(\mathcal{H})$ are the amount of states in the model. Why the letter $\mathcal{H}$ is chosen will hopefully become clear when studying the Hidden Markov Models. The MC transitions between each of the states in $\mathcal{H}$, based on the set of *transition probabilities*, which where denoted by P_{ij} above. In the following, the notation $\mathcal{T}_{ij}$ will be adopted instead, for ease of recalling that they are transition probabilities. These are summarized in a $n \times n$-matrix $\mathcal{T}$, where each position $\mathcal{T}_{ij}$ represents the probability of a transition from state i to state j, where $i, j \in \mathcal{H}$. Written in mathematical notation

$$\mathcal{T}_{ij} = \mathrm{P}(X_{t+1} = j | X_t = i), \quad i, j \in \mathcal{H}.$$

As noted, the term *time-homogeneous* refers to the fact that $\mathcal{T}$ does not change as time goes on, i.e. as the MC is generated, but stays fixed during the entire simulation of the chain. In mathematical notation, this assumption means that

$$\mathcal{T}_{ij} = \mathrm{P}(X_{t+1} = j | X_t = i), \quad i, j \in \mathcal{H}, \forall t \in [0, L].$$

Moreover, the term *first-order* refers to an important assumption in this model, which makes it possible to define $\mathcal{T}$ as we did, which is that the transition probability from the previous state to the current state only depends on the previous state. For example, a time-homogeneous second-order MC would depend on the two previous states when transitioning to the current state, which makes the model more general, but also more complicated. Note that the initial state of a MC is determined based on the initial distribution of the states, commonly denoted by $\pi = (\pi_1, \ldots, \pi_n)$. The assumptions make calculations of probabilities in the MCs very simple, since they are strong independence assumptions.

A MC is completely specified by π and $\mathcal{T}$ (where $\mathcal{H}$ is indirectly given in both these two quantities). Once these quantities are specified, one can make calculations concerning the MC. The probability of a realization $\mathbf{x} = (x_0, x_1, \ldots, x_L)^T$ taking place can be calculated recursively as

$$\begin{aligned}
\mathrm{P}(X_0 = x_0, X_1 = x_1, \ldots, X_L = x_L) &= \mathrm{P}(X_0 = x_0, X_1 = x_1, \ldots X_{L-1} = x_{L-1}) \\
&\quad \cdot \mathrm{P}(X_L = x_L | X_0 = x_0, X_1 = x_1, \ldots X_{L-1} = x_{L-1}) \\
&= \mathrm{P}(X_0 = x_0, X_1 = x_1, \ldots X_{L-1} = x_{L-1}) \\
&\quad \cdot \mathrm{P}(X_L = x_L | X_{L-1} = x_{L-1}) \\
&= \mathrm{P}(X_0 = x_0, X_1 = x_1, \ldots, X_{L-1} = x_{L-1}) \cdot \mathcal{T}_{L-1,L} \\
&= \mathrm{P}(X_0 = x_0, X_1 = x_1, \ldots, X_{L-2} = x_{L-2}) \cdot \mathcal{T}_{L-2,L-1} \cdot \mathcal{T}_{L-1,L} \\
&\vdots \\
&= \pi(x_0) \cdot \mathcal{T}_{0,1} \cdot \ldots \cdot \mathcal{T}_{L-2,L-1} \cdot \mathcal{T}_{L-1,L}
\end{aligned}$$

where the second inequality holds because of the Markov property. Also, note that $\pi(x_0)$ refers to the probability that the initial state is x_0. Hence, it is apparent that calculating the probability of a specific realization of the MC is very simple.

Could discuss stationarity and other properties also, if I think it is relevant for the rest

For a more rigorous treatment of stochastic processes and MCs, the reader is referred [7].

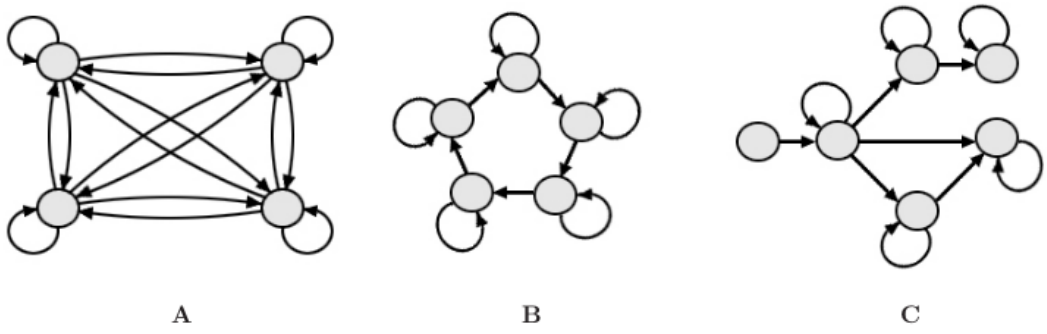

Fig. 2 Some existing HMM topologies. **A.** a fully connected HMM; **B.** a circular HMM; **C.** a left-right HMM.

Figure 1: Examples of the three highlighted topologies stolen from Choo.

3.1.1 Markov Chain Topologies

Perhaps not that important. The topology of a MC refers to which state transitions are permitted and prohibited, i.e. what the values in the transition probability matrix are. There exists many different ways of constructing MCs. As inspired by Choo and colleagues, three different topologies will be highlighted here [2]. Figure 1 gives some simple graphical explanations of these models.

A *fully connected model*, as the name suggests, yields a complete directed graph. This means there are no zero entries in the transition probability matrix, except the possibility of zero entries in the diagonal, which would mean that a loop in the given state is not possible.

A *circular model* is

A *left-right model* is

3.2 Hidden Markov Models

Yoon gives a very nice first introduction to Hidden Markov Models (HMMs): "A hidden Markov model (HMM) is a statistical model that can be used to describe the evolution of observable events that depend on internal factors, which are not directly observable." [3]. A HMM, simply put, is a model that comprises of two different parts. The first part is the sequence or the result that can be observed. A possible observed sequence could be a sequence of nucleotides or it could be a chain of events, like 20 dice throws in a row. The second part is a MC, which is *hidden*, i.e. it cannot be observed. This very simple idea gives a very flexible model, which can be used for many different types of problems. The main goal is often to infer the characteristics of the hidden MC from the observed sequence.

In a HMM, a hidden sequence of states is generated, according to the MC. This sequence will not be observable and will in (almost) all practical cases be considered as unknown. Each of the states creates, or *emits*, an observed value. All these observed values are what make up the observed sequence. The emission follows a multinomial distribution, where each state follows such a distribution with different parameters. An important assumption for the HMM we will deal with is that the parameters of each multinomial only depends on the state to which it belongs [2]. This means that the emitted value only depends on its respective hidden state and not on other hidden states. A HMM thus represents a doubly stochastic process: the current underlying state in the MC is stochastic, with an added layer of stochasticity in the observed value. Note that each of the hidden states should be able to produce the same symbols, i.e. the same observable values, but in different frequencies [4].

Following the same notation as earlier, let $\mathbf{X}$ be the random vector that represents the underlying MC of length $L+1$, let $\mathcal{T}$ be the transition probability matrix, let $\mathcal{H}$ denoted the hidden states,

where $m = \text{card}(\mathcal{H})$, and let π be the initial distribution of states. In addition to this, we need to define some notation for the observable sequence in the HMM. First of all, let the random vector $\mathbf{Y} = (Y_0, Y_1, \ldots, Y_L)^T$ represent an observable sequence of length $L+1$. As earlier, $\mathbf{y} = (y_0, y_1, \ldots, y_L)^T$ will represent a realization of $\mathbf{Y}$. Note that both the hidden and the observed sequence are of length $L+1$, since it is assumed that each state emits exactly one symbol. The *symbol alphabet*, i.e. the set of symbols that can be emitted from each state, will be denoted by $\mathcal{S} = \{s_1, \ldots, s_M\}$, where $M = \text{card}(\mathcal{S})$. The *emission probabilities*, i.e. the probabilities used in each multinomial distribution in each state, are usually arranged in a matrix as well. Denote by $\mathcal{E}$ the emission probability matrix. $\mathcal{E}$ is of dimension $m \times M$, where each row in the matrix contains the emission probabilities (of each of the symbols) for a given state. Written in mathematical terms

$$\mathcal{E}_{ij} = \text{P}(Y_t = j | X_t = i), \quad i \in \mathcal{H}, j \in \mathcal{S}.$$

Time-homogeneity is also assumed in the stochastic process of emission, which means that these emission probabilities are the same $\forall t \in [0, L]$.

A HMM is completely determined by the following four components: (i) A symbol alphabet, K different symbols (e.g. $A = \{A, C, T, G\}, \text{card}(A) = K$), (ii) number of states in the model M, (iii) emission probabilities $e_i(x)$ for each state i and (iv) transition probabilities $t_i(j)$ for each state i to another state j [1].

CALCULATIONS: This independence assumption, together with the independence assumptions of the "standard" Markov chain, makes the calculation of the probabilities very easy, since most of the conditional probabilites are reduced to "regular" probabilities. The total likelihood of the hidden sequence is simply the product of the individual probabilities of the states at each position in the sequence, as noted in the Markov chains earlier. Similarly, this idea applies to the likelihood of the emitted sequence given the hidden state at each position. The theorem of total probability can be used to compute the probability of the observed values. Since this probability is infeasible (why?), it is solved via the forward algorithm, which is based on the dynamic programming paradigm [4].

We need some notation and perhaps an example.

Let $\mathcal{H} = \{h_1, \ldots, h_N\}$ be the set of hidden states in the underlying Markov chain and let $N = \text{card}(\mathcal{H})$. Similarly,A hidden sequence of states in the Markov chain will be denoted by $\mathbf{X} = X_1 X_2 \ldots X_n$, where n is the number of states that the Markov chain has traversed. An observed sequence will be denoted by $\mathbf{Y} = Y_1 Y_2 \ldots Y_n$, where n are the number of symbols in the observed sequence. The emission and transition probabilities are presented as matrices. Denote by $\mathcal{T}$ the transition probability matrix of $N \times N$. Remember that each position $\mathcal{T}_{ij}$ in $\mathcal{T}$ represents the probability of a transition from state i to state j, where $i, j \in \mathcal{H}$.

Calculating probabilities in HMMs is simple. The first-order time-homogeneous assumptions for the underlying Markov chain make it possible to find the probability of $\mathbf{X}$ as the product of all appropriate transition probabilities. In mathematical notation, the probability of $\mathbf{X}$ is

$$\text{P}(\mathbf{X}) = \pi_{x_1} \prod_{t=2}^{n} \mathcal{T}_{x_{t-1}, x_t},$$

where π_{x_1} denotes the initial state probability of x_1, which is the first state in the hidden MC. In a similar fashion the independence assumptions in the rest of the HMM Specify above what these assumptions are! make it possible to find the probability of the observed sequence simply by taking the product of all appropriate emission probabilities. In mathematical terms, the probability of seeing the sequence $\mathbf{Y}$, given the state sequence $\mathbf{X}$, is

$$\text{P}(\mathbf{Y} | \mathbf{X}) = \prod_{t=1}^{n} \mathcal{E}_{x_t, y_t}.$$

Finally, combining these results, the joint probability of the sequences $\mathbf{X}$ and $\mathbf{Y}$ is

$$\mathrm{P}(\mathbf{X}, \mathbf{Y}) = \mathrm{P}(\mathbf{X})\mathrm{P}(\mathbf{Y}|\mathbf{X}).$$

Note that we here assume that the hidden state sequence $\mathbf{X}$ is known, which is usually not the case in practice. The state sequence needs to be inferred from the observed sequence, which is one of the main problems of a HMM, which will be explained in the following. For a more rigorous treatment of this topic, the reader is referred to ... Henvis til kilder som utleder disse resultatene her!!

Transitional probabilities, emission probabilities and initial state probabilities completely specify a HMM [3].

A more technical article with more examples is [3].

The model parameters can be estimated given some training data, where both the hidden and observed states are known, using the maximum likelihood principle and the EM (expectation maximization) algorithm [4].

As with the Markov models described in an earlier section, we need to declare some initial probabilities for the state of the Markov process.

3.3 Algorithms (or Basic Problems)

http://jedlik.phy.bme.hu/ gerjanos/HMM/node6.html

As Yoon points out in his article, there are three issues that need to be resolved in order to be able to use HMMs in practical applications [3]. In the following, these issues will be presented, together with algorithms that are used to address the problems.

3.3.1 Forward Algorithm (The Evaluation Problem)

The first issue that needs to be addressed is: how can the probability $\mathrm{P}(\mathbf{y})$ be calculated? That is, how can one find the probability of the observed sequence $\mathbf{y}$? If the underlying state sequence was known, this would have been easily calculated based on the emission probabilities in each state. Refer to some formula from earlier perhaps, if possible However, the underlying state sequence cannot be observed, which means that there may exist many different underlying state sequences that can yield the same observed sequence $\mathbf{y}$. Thus, the first solution that comes to mind would be to use the law of total probability to calculate

$$\mathrm{P}(\mathbf{y}) = \sum_{\forall \mathbf{x}_i \in \mathcal{H}^n} \mathrm{P}(\mathbf{y}, \mathbf{x}_i) = \sum_{\forall \mathbf{x}_i \in \mathcal{H}^n} \mathrm{P}(\mathbf{x}_i)\mathrm{P}(\mathbf{y}|\mathbf{x}_i), \tag{1}$$

where $\mathcal{H}^n$ is the space of all possible state sequences that can yield $\mathbf{y}$. Of course, $\mathcal{H}^n$ can be very large, which makes this calculation computationally infeasible. In fact, $\mathcal{H}^n$ may consist of N^n sequences, since these are all possible state sequences. Luckily, the *forward algorithm* exists, which uses dynamic programming to solve this issue in a computationally feasible manner. This algorithm has time complexity $\mathcal{O}(nN^2)$ Check that this is correct!, which is a significant improvement from the exponential solution proposed by equation (1). The algorithm defines the forward variable

$$\phi(n, i) = \mathrm{P}(\mathbf{X}, h_n = i). \tag{2}$$

Continue... This is read from Yoon2009.

Posterior decoding - confidence in the best scoring path.

Now, note that the forward algorithm is used to verify if the observed sequence could have been generated by the given HMM. If the calculated probability is lower then some "background distribution" of similar sequences, then the probability of the sequence being emitted by the HMM is low and one would conclude that the HMM is not a good model for the observed sequence. However, if the probability is high, then we can conclude that the HMM can be used in the situation. When this is verified, we tackle the *decoding problem*.

3.3.2 Viterbi Algorithm (The Decoding Problem)

The second issue that needs to be addressed is: what state path $\mathbf{X}$ maximizes the probability of emitting the observed sequence $\mathbf{H}$, i.e. how can the *optimal state path* be found? This follows the principle of maximum likelihood, which says that we should choose the hidden state path that maximizes the likelihood of yielding the observed sequence. Our best bet when trying to infer the hidden state path from the observed sequence is exactly the one that gives the maximum probability of obtaining the observed sequence. In other words, the hidden state sequence that best explains the observed sequence is the optimal state path

$$\mathbf{H}^* = \arg\max_{\mathbf{H}} \mathrm{P}(\mathbf{H}|\mathbf{X}). \tag{3}$$

Notice that this is equivalent to finding the state sequence that maximizes $\mathrm{P}(\mathbf{H}, \mathbf{X})$, because of the definition of conditional probability

$$\mathrm{P}(\mathbf{H}|\mathbf{X}) = \frac{\mathrm{P}(\mathbf{H}, \mathbf{X})}{\mathrm{P}(\mathbf{X})}.$$

Again, the first solution that comes to mind would be to compare all possible state sequences S^n, but this is still computationally infeasible in most cases. Therefore, the *Viterbi algorithm*, which also is based on dynamic programming, exists. The time complexity of the Viterbi algorithm is $\mathcal{O}(nS^2)$, which is the same as for the forward algorithm. This is a large improvement over the first, straightforward solution that was proposed, which has a time complexity yields an exponential increase with the size of the sequence n. Continue... Reading from Yoon2009.

Finding the best state path, i.e. inferring the hidden state path from the given sequence. More specifically, the Viterbi algorithm is used to find the hidden sequence with the highest likelihood to have produced the observed sequence.

The algorithm is given below. Notation needs to be consistent throughout the report!! I will define some notation below right away, but should be moved to the theoretical backgorund on HMMs after.

Let $\mathcal{H}$ be the set of hidden states in the underlying Markov chain and let $N = \mathrm{card}(\mathcal{H})$. Let $\mathcal{E}$ be the symbol alphabet, i.e. the set of symbols that can be emitted from each state. Let $\mathcal{T}$ be the transition matrix in the underlying Markov chain, which has dimensions $N \times N$.

Given an observed sequence $\mathbf{s}$ of length n, the Viterbi algorithm is

Algorithm 1 Viterbi Algorithm

$V \leftarrow \mathrm{array}(N \times (n+1))$
$i \leftarrow 0$
$V(0,0) \leftarrow 1$
$V(k,0) \leftarrow 0, \forall k > 0$
for $i = 1, 2, \ldots, n$ **do**
  $V(l,i) \leftarrow \mathcal{E}(l, \mathbf{s}(i)) \max_k \{V(k, i-1)\mathcal{T}(k,l)\}$
  $\mathrm{pointer}(i,l) \leftarrow \arg\max_k \{V(k, i-1)\mathcal{T}(k,l)\}$
end for

Missing output (traceback using pointers) etc

Note that in practice it is common to work with the logarithm of these probabilities, because the product of these probabilities quickly become very small. Thus working with addition, as a consequence of the log-transform, yields larger numerical stability.

3.3.3 Backward Algorithm (The Training Problem)

The Viterbi algorithm finds the optimal path for obtaining the entire sequence, i.e. it finds the sequence that maximizes the joint probability of obtaining the entire observed sequence. In some

cases, one might be interested in finding the path that gives the largest probability of observing each symbol of the sequence individually. This can be done using the *backward algorithm*. The problem here is to find the state h_j that has the highest likelihood of being the hidden state of x_j, i.e.

$$\hat{h}_j = \arg\max_i \mathrm{P}(h_j = i|\mathbf{X}).$$

As before, searching all the possible states is not a viable option, which is why the backward algorithm is used. It works as ... Continue... Reading from Yoon2009.

But why do we care about finding the sequence of individually optimal states? This will maximize the expected number of correctly predicted states [3] Hvorfor det egentlig? Well, if this is the case, why does one even bother using the Viterbi algorithm? The problem with the sequence that maximizes the probability of each observation individually is that it in general will be suboptimal, i.e. that there exists some other sequence $\mathbf{h}^*$ such that $\mathrm{P}(\mathbf{X}, h_1 h_2 \dots h_n) \leq \mathrm{P}(\mathbf{X}, \mathbf{h}^*)$. Moreover, the sequence may not even be a possible path for the structure of the HMM, for example if some of the transitions in the predicted sequence are not possible with the given state transition probabilities of the model. For this reason, the Viterbi algorithm is preferred when our goal is to find the optimal path for the entire sequence, while the backward algorithm is preferred when our goal is to infer which state is optimal in a specific position in the sequence. Moreover, the backward algorithm can be used to conclude about the confidence of a state prediction Continue... Reading from Yoon2009.

Få med dette også, da det virker som et begrep som brukes ofte!Posterior decoding - confidence in the best scoring path.

3.3.4 The Training Problem

The third issue that needs to be addressed is: how can the HMM parameters be reasonably chosen based on a set of observed sequences? Not sure what to call this, because there exists many algorithms it seems like. For example, we have a set of sequences $\chi = \{\mathbf{X}_1, \dots, \mathbf{X}_G\}$, where each $\mathbf{X}_i$, $i \in [1, G]$ is an observed sequence of symbols, that we want to represent with a HMM. The difficulty that needs to be solved is how the parameters of the HMM can be estimated based on the set χ. This is what is typically called the *training* or *learning* problem, analogously to the need for training any other machine learning model. There exists no optimal way to train the HMM from a limited number of finite observation sequences, but there exists algorithms that can find local maximums in the obsercation probability. Some examples are the Baum-Welch algorithm, standard gradient based methods from optimization or simulation with Monte Carlo expectation maximization (MCEM).

Continue... Reading from Yoon2009. And also can read some from the website linked in the beginning of "Algorithms".

Introduction to Mathematical Methods in Bioinformatics - Alexander Isaev.

The parameters of the HMM are estimated, or the model is trained, based on a larger set of sequences, what is often referred to as a training set. In this training data it it essential that the hidden states are also known, in order for the model to be able to "learn" from the data. After all the probabilities are estimated, the model can be used on new data, which is often referred to as the testing set, and we can use the results for predict or infer something about the data.

3.3.5 Estimation of Emission Probabilities

[2]

3.3.6 HMM Models

In addition to the standard HMM that was described in the previous section, there exists many other variants of the HMM. This section presents several different types, but note that this list is by no means exhaustive.

Standard, Generalized, Pair, Generalized Pair, Profile [2].

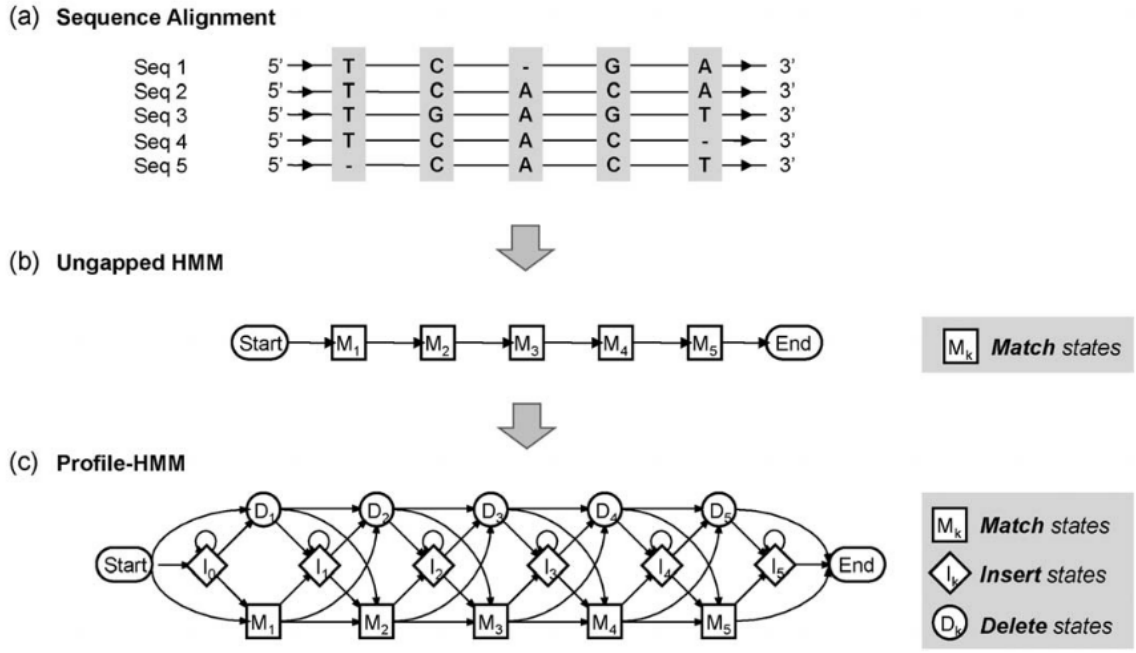

Fig. (2). Profile hidden Markov model. **(a)** Multiple sequence alignment for constructing the profile-HMM. **(b)** The ungapped HMM that represents the consensus sequence of the alignment. **(c)** The final profile-HMM that allows insertions and deletions.

Figure 2: Example from [3]

3.3.7 Generalized HMMs

Changes the distribution of time until state transition (to another state) from geometric to another generalized distribution. Let's see if I have to add it here (depending on if it is used in some of the applicatins in biology. I think this is a continuous time Markov Chain (with sojourn times etc) as learned in StokMod, but not sure.

3.3.8 Profile-HMMs

A profile HMM have a specific architecture, which make them suitable for modeling sequence profiles [3]. Because of this, the topology of the profile HMM will be explained with multiple sequence alignment in mind. Two simple ways to think of a profile HMM is as an abstract description of a protein family or a statistical summary of multiple sequence alignment [4]. A profile HMM is constricted to not contain any cycles. Moreover, a profile HMM consists of three different types of hidden states: match states (M_k), insert states (I_k) and delete states (D_k). These are used to describe symbol frequencies, insertions and deletions, respectively, at a specific position in a sequence. A great example on how to build a profile HMM, which helps to clarify the ideas, can be found in [3]. Can I just steel this example?

Kanskje dette skal flyttes til eksempelt på Multiple Sequence Alignment? The example shows how a profile HMM can be built from a multiple sequence alignment. The M_k's are used to indicate match states, i.e. they represent the case where a symbol in the new observation matches with the state of the profile HMM. The emission probabilities from each match state are easily estimated by using the frequencies of each symbol in the match state. The ungapped HMM can represent ungapped sequences. The states I_k and D_k are added so that the HMM can represent sequences that contain

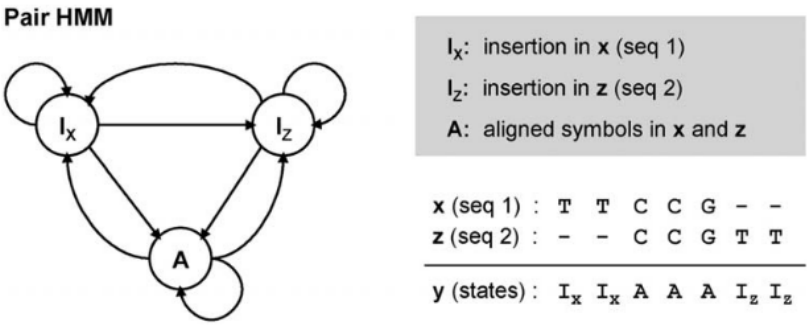

Fig. (3). Example of a pair hidden Markov model. A pair-HMM generates an aligned pair of sequences. In this example, two DNA sequences **x** and **z** are simultaneously generated by the pair-HMM, where the underlying state sequence is **y**. Note that the state sequence **y** uniquely determines the pairwise alignment between **x** and **z**.

Figure 3: Example from [3]

gaps, which makes the profile HMM a much more powerful model compared to the simpler models that describe a multiple sequence alignment, like regular expressions and PSSM. It is important to note that the delete states D_k are silent states; they are placeholders for missing symbols in the consensus sequence of the alignment when compared to a shorter new observed sequence.

3.3.9 Pair-HMMs

In the context of bioinformatics, a pair HMM is especially useful for finding sequence alignments and evaluating the significant of the alignments [3]. To contrast a regular HMM, a pair HMM generates an aligned pair of sequences, instead of only one sequence. An example is shown in [3] Kan jeg bare stjele dette eksempelet også!?

Since there is a one-to-one relationship between the hidden state sequence $\mathbf{H}$ of a pair HMM and the alignment between the two observed sequences, the alignment problem reduces to finding the optimal state path in the hidden Markov chain. This can be found by a simple variation of the Viterbi algorithm and has time complexity $\mathcal{O}(n_{\mathbf{x}} n_{\mathbf{y}})$, where $\mathbf{x}, \mathbf{y}$ are the two sequences that are aligned and $n_{\mathbf{x}}, n_{\mathbf{y}}$ are their respective lengths. The pair HMM model is an improvement over the classical sequence alignment methods, since it can be used to compute the alignment probability of the pair of sequences. A problem in the classical methods is how one should choose a punctuation scheme in order to find a biologically meaningful optimal alignment of the sequences. In cases where this is difficult to choose, it is more meaningful to compute the probability that the sequences are related, which a pair HMM makes possible [3]. This probability can be calculated by a slight modification of the forward algorithm.

3.3.10 Context-Sensitive HMMs (csHMMs)

In the context of bioinformatics, the ordinary HMM has limitations that make it non-suitable for dealing with RNA sequences. In order to deal with RNA sequences, one needs to extend the HMM to allow for pairwise correlations between non-adjacent symbols in the observed sequence. This is something that the ordinary HMM cannot model, since we assume that each state in the Markov chain depend only on the previous state and since each emission probability depends solely on the current state in the Markov chain. For this reason, the *context-censitive HMM*s can be used.

The main difference between the context-censitive HMMs and the ordinary HMMs is that the former can use information about earlier emissions to adjust the emission probabilities at future states.

This information is referred to as the "context". Thus, this makes it possible to model correlation between non-adjacent states. These HMMs use three different types of states: single-emission states (ξ_i), pairwise-emission states (ϕ_i) and context-sensitive states (γ_i). ξ_i are very similar to the normal states in ordinary HMMs, since they have usual emission probabilities and do not use any additional information. ϕ_i have fixed emission probabilities also, but, in addition, they store the symbols that are admitted in memory. Then, when the Markov chain reaches the corresponding state γ_i, the memory is queried for the symbol that was emitted in ϕ_i. The emitted symbol at this context-sensitive state h_j is adjusted based on the retrieved symbol h_k. Thus, we can state the emission probability for this context-sensitive state as Probs need to change the index notation etc here a bit. Do it after I fully understand the model!

$$\epsilon(h_j|h_k, \gamma_i, \phi_i) = \mathrm{P}(h_j \text{ emitted at } \gamma_i \text{ given that } h_k \text{ was emitted at } \phi_i).$$

Using the fact that h_k is independent of γ_i (as always is the case for emission probabilities) the join emission probability of h_k and h_j can be calculated

$$\mathrm{P}(h_j, h_k|\gamma_i, \phi_i) = \mathrm{P}(h_k|\gamma_i)\mathrm{P}(h_j|h_k, \gamma_i, \phi_i) = \epsilon(h_k|\gamma_i)\epsilon(h_j|h_k, \gamma_i, \phi_i).$$

Thus, using the pair of states (ϕ_i, γ_i) pairwise symbol correlation can be modeled for any pair by specifying these emission probabilities. A simple example of a csHMM is shown in [3] Kan jeg bare stjele dette eksempelet også!?.

An example of a profile csHMM is shown in [3] Kan jeg bare stjele dette eksempelet også!?

Remember that profile HMMs were used to make profiles that represent multiple alignments of sequences, which in the context of biology can be DNA or proteins (sequences of aminoacids). Imagine that we want to make profiles that take column-wise correlations in the multiple alignment into account. This can be done relatively easily by combining the ideas of the profile HMM and the csHMM, into what we call a profile csHMM. Like for the profile HMMs studied earlier, the profile csHMMs contain match states (M_k), insert states (I_k) and delete states (D_k). The difference is that the match states can be of three different types, to account for symbols that are pairwise correlated. As expected, this model has the match states single-emission match states ($\xi_{m,i}$), pairwise-emission match states ($\phi_{m,i}$) and context-sensitive match states ($\gamma_{m,i}$). The single-emission match states are used to represent columns in the multiple alignment that are uncorrelated with the others, while the pairing of $(\phi_{m,i}, \gamma_{m,i})$ are used to model pairwise correlation between symbols in different columns. An example, taken from [3], is shown in figure 5.

3.3.11 Advantages of HMMs (could be moved somewhere else)

3.3.12 Disadvantages of HMMs (could be moved somewhere else)

HMMs do not deal well with correlations between residues, since they assume that each residue depends only on one underlying state [1], i.e. it is assumed that the sequence is generated from the hidden state path, where each letter in the sequence only depends on the emission probabilities of its corresponding node in the state path (hidden Markov chain). Since the Markov property is assumed, an HMM has no way of "remembering" any distant correlation between the letters in the sequence alphabet.

As already noted, Choo et. al writes that "the linear nature of HMM also makes it difficult to capture higher-level information or correlations among amino acids" [2].

4 Applications in Bioinformatics

Introduction to Computational Genomics: A Case Studies Approach, Chapter 4.

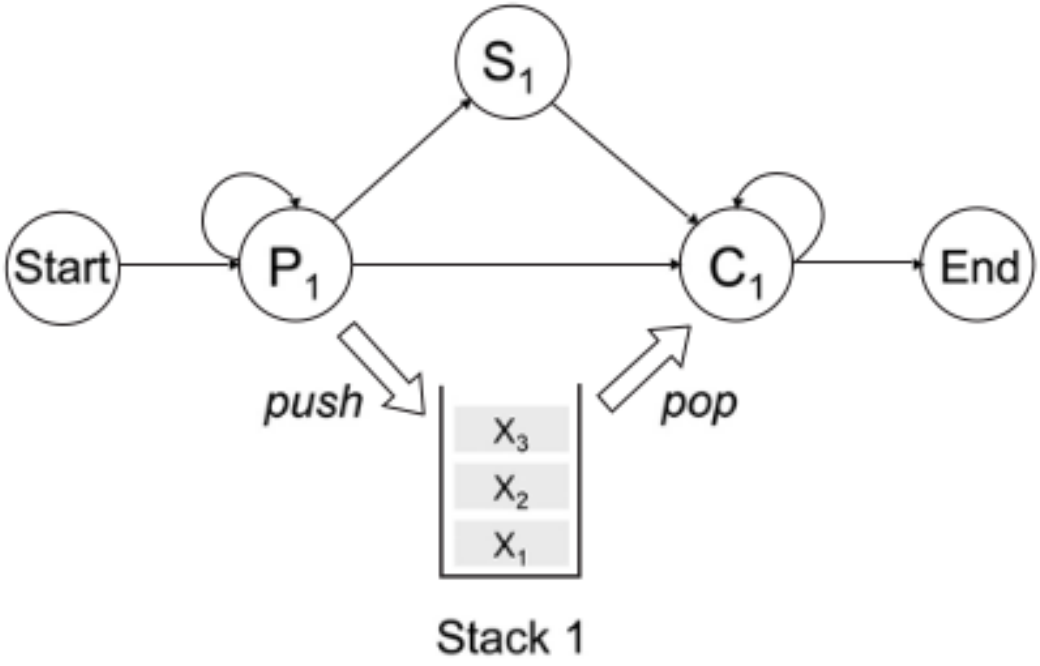

Fig. (4). A context-sensitive HMM that generates only symmetric sequences, or palindromes.

Figure 4: Example from [3]. I should make a similar example, but for repeating sequences instead of for palindromes, as Yoon did. This would be almost the same, but the result is slightly different.

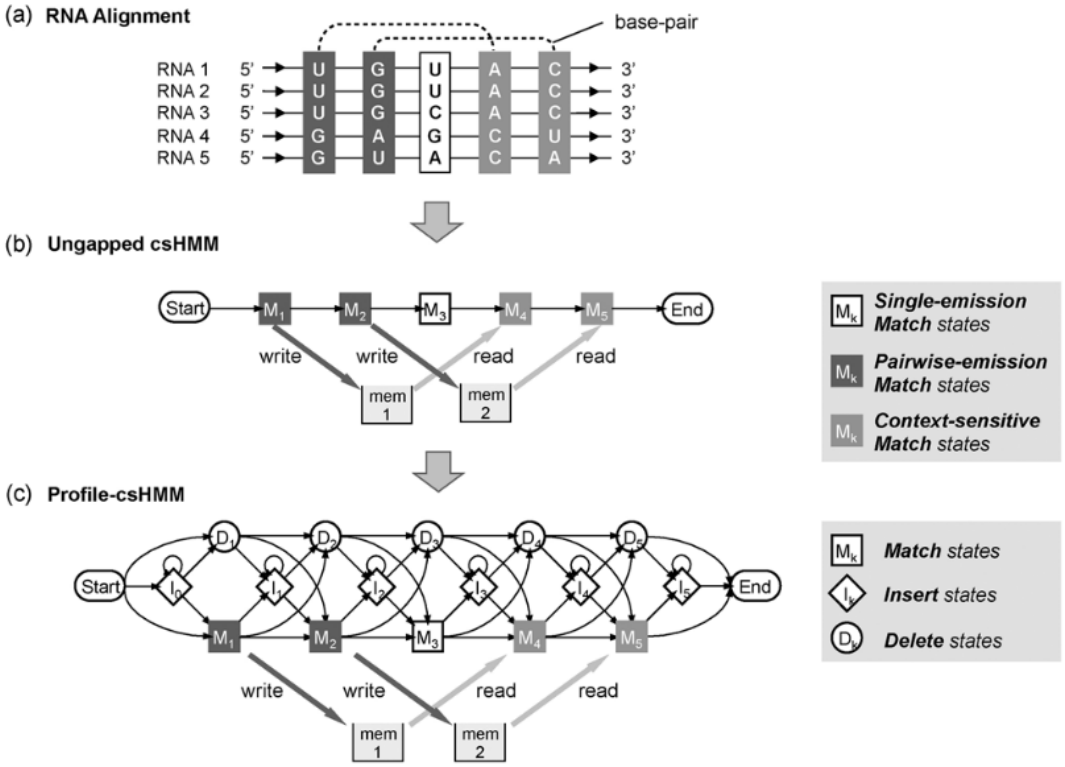

Fig. (5). Constructing a profile-csHMM from a multiple RNA sequence alignment. (**a**) Example of an RNA sequence alignment. The consensus RNA structure has two base-pairs. (**b**) An ungapped csHMM constructed from the given alignment. (**c**) The final profile-csHMM that can handle symbol matches, insertions, and deletions.

Figure 5: Example from [3]. Tenker at jeg stjeler dette eksempelet, da det er veldig godt (eller bare henviser), men lager de to andre litt selv etter hvert!

Gapped motifs of variable length can be found using HMM (As we did in task5, but more complicated, since we do not restrict ourselves to the most simple cases, i.e. we allow gaps with variable lengths).

Some applications listed on the Wikipedia page:

4.1 Pairwise Sequence Alignment

Pairwise sequence alignment is done to infer functional, structural and/or evolutionary relationships between two sequences. Such an alignment can be done both locally and globally and there exists a variety of methods of doing it. Optimal algorithms exist, such as the Needleman-Wunsch algorithm for optimal global alignment and the Smith-Waterman algorithm for optimal local alignment. These algorithms are based on dynamic programming and can be time consuming to use. Therefore, algorithms based on heuristics were developed, where the two most popular algorithms are called BLAST and FASTA. Proteins or DNA? Only proteins? Sjekk for alle algoritmene her! All the mentioned algorithms work well for highly similar sequences, but produce mediocre results for highly divergent sequences [2]. Profile based analysis was developed to improve these results, in which HMMs play a crucial role.

Pair-HMMs can be used when tackling the pairwise sequence alignment problem as a stochastic process, as used by Smith et al [5] Skumles artikkelen og sjekk at det faktisk stemmer! Pairwise sequence alignment is the typical example of use of a pair-HMM in bioinformatics, as also mentioned in the theoretical part concerning the model. Complete this after completing theory about Pair-HMMs, because I think it will be easier then!

Goal: Infer functional similarity Could be merged with "Prediction of Function" perhaps?. Can use a Pair HMM [2] (go to source after Choo).

As noted, pair HMMs are a great alternative to using classical sequence alignment techniques and evaluating the statistical significance of the alignment.

Pair Hidden Markov models on tree structures (PHHMTS) can be used for aligning trees. Since most RNA secondary structures can be represented as trees, this provides a useful framework for aligning RNA sequences.

4.2 Multiple Sequence Alignment

Multiple sequence alignment is commonly used to find conserved regions in a group of sequences and/or predicting protein structures. In contrast to the pairwise sequence alignment problem, there does not exist any optimal algorithms to solve the multiple alignment problem. However, there exists a variety of heuristic methods for solving it. One possible solution is to reduce the multiple alignment problem down to a set of pairwise comparisons between the sequences, in an orderly fashion, in order to end up with the multiple alignment in the end. Here one can choose to use either the optimal algorithms or the heuristic algorithms when performing the pairwise alignment. One type of methods that employs this paradigm is referred to as progressive sequence alignment. More details on a progressive algorithm that uses Needleman-Wunsch for the pairwise alignment can be found in [6]. Some commonly available implementations of this solution include the T-Coffee and the Clustal package [8]. Another type of methods that uses this paradigm is referred to as iterative alignment. These methods are slightly different than the progressive methods, since they allow realignment of the pairwise sequences during multiple iterations, where the progressive methods depends highly on the initial pairwise alignment of the first two sequences [9]. One popular openly available implementation of an iterative method is MUSCLE [8].

As in the case of pairwise alignment, HMMs provide powerful alternatives to these other methods for multiple alignment. In this problem, profile-HMMs have been particularly successful [2]. Continuing to read in Choo.

Profile HMMs have been applied to this problem with much success . Is connected to the Viterbi algorithm also.

Må virkelig få strukturert dette skikkelig! Vanskelig å dele opp alt slik jeg tenkte opprinnelig! Mulig jeg bør samkjøre teori og applications i større grad, selv om jeg egentlig ville ha dem separert!

`http://pfam.xfam.org/`

Profile HMMs are defined in order to solve the problem of multiple alignment of sequences. All new sequences can efficiently be aligned against this profile HMM. They also facilitate quick assignment of protein function. These profile HMMs are commonly regarded as a summary of a multiple alignment of sequences or as a model for a family of such sequences [4].

Since pHHMs are an abstract representation of a multiple alignments, they can be used to produce pairwise or multiple alignments as well. Thus, aligning a sequence with a pHHM is equivalent to aligning the sequence to many, many other sequences, which where used to establish the pHMM in the first place [4]. E.g. PFAM is a free online repository, that store pHMMs of many known protein families. PROSITE is another database that stores profile HMMs.

The profile HMMs are very useful in the context of searching for homologues. Given a profile HMM that represents a family of sequences, one can use this model to search in a database of sequences, in order to find additional homologues to the family. Similary, given a database of pre-built profile HMMs (like PROSITE or PFAM), we can look for matching profiles to a symbol sequence. Thus, we can use this database to classify and annotate the given sequence.

Profile HMMs can also be used to compare two different multiple sequence alignments (sequence profiles). This can be beneficial for detecting remote homologues [3]. "These profile HMMs are also what makes it possible to assign protein function quickly, and can be regarded both as a summary of a multiple alignment and as a model for a family of sequences" [4].

Many multiple sequence alignment algorithms also use pair HMMs [3]. The most widely used approach based on pair HMMs is called progressive alignment.

4.3 Motif Representation (or Identification?)

A motif is a recurring pattern in DNA or proteins, which is assumed to be related to biological function. Thus a motif can be a sequence of nucleotides or of amino-acids. Finding such motifs is interesting to a biologist because *transcription factor binding sites* (TFBS) appear as such motifs in sequences. These are regulatory sequences that control gene transcription, which is important information for a biologist because they repress or promote the expression of many other genes [4].

There are several ways of representing such sequence motifs, where the performance of each method generally depends on what type of motifs one wants to represent. For shorter, ungapped motifs of fixed length, methods like *consensus sequences*, *regular expressions* (RegEx), *sequence logos* and *position specific scoring matrix* (PSSM), in ascending order of complexity, are commonly used. Note that these can also be seen as different ways of visualizing or representing multiple sequence alignments, which are useful tools in practice, since the alignments themselves are not very easy in use.

The drawbacks of these methods are that they do not work well for longer motifs with variable length gaps. This is where the profile-HMMs shine in comparison. The HMMs work well for all types of motifs, including the short and fixed length motifs, but they are more complicated models. Thus, even though the HMMs always will produce good results, given that they can be built correctly in the specific case, one should not always use these models because of their complexity. Always keep the principle of parsimony in mind; for competing explanations, or models, where all are reasonable, one should always choose the simplest, with the least amount of assumptions. Hence, even though the HMMs are great tools also for motif identification, they should be used when appropriate.

But how can the HMMs be used to represent motifs with variable length gaps? The profile-HMMs have gained much traction for this problem, since it yields very reliable results. How they work can be seen easily from the example presented about the profile-HMMs in the section on theoretical background, since the example talks about how to represent a multiple sequence alignment as a profile-HMM. The same principle explained in the example can be used when working with much larger and more complicated alignments.

After constructing these representations, they can be used to search for sequences that belong to the same family as the aligned sequences, since sequences that have the same motifs may share the same functions. Good idea to have this as a different section compared to the multiple sequence alignment-section?

Note that the problem of finding the motifs is not discussed in detail here, as this is a much more complicated problem compared to simply representing the motifs. However, this problem is highly similar to multiple local alignment, which has been explained in detail earlier. An example of such an algorithm is given in chapter 10.3 in [4], which is a variant of the Gibbs sampling algorithm based on PSSMs.

EXTRA, if needed: Wikipedia: When a motif appears in the exon of a gene, it can encode a "structural motif" of a protein. Outside of gene exons, there exists regulatory sequence motifs

4.4 Prediction of Function

HMMs are used to make probabilistic statements about the function of proteins and thus, they can also be used to assign proteins to families of unknown function [4]. Is this a step in Genome Annotation? Perhaps they should be merged then! Or multiple sequence alignment?

4.5 Segmentation

Segmentation is about defining exact boundaries between distinct regions with different chemical properties. Moreover, segmentation is about defining larger sequences of heterogeneous nucleotide in genome and to identify biological features that are responsible for the heterogeneity that is observed [4]. Some classical methods are ...

HMMs can be used effectively for segmentation as well... They can help to define regions of gene and protein sequences with various chemical properties [4].

Example 4.2 in [4] shows an example of segmentation using HMMs.

In the setting of segmentation, the hidden states are interpreted as different types of the sequence and the hidden alphabet is typically very small. The underlying MC is cyclic in this case, allowing returns to the same state, i.e. to the same type of sequence, many times during the simulation [4].

4.6 Protein Homology Detection

Goal: Determine which proteins are derived from a common ancestor.

Not sure if the following fits in here. Characterize sets of homologous proteins (gene families) based on common patterns in their sequence. This allows us to determine if a new protein belongs to a certain family or not [4]. In thi case HMMs can be used to provide a more flexible characterization of sequence patterns. This, in comparison to the simpler way of using multiple alignment to construct a PSSM, also works well for cases which include gaps of variable length, which is a case where the multiple alignment method does not work well [4]. This type of homology detection is done with profile HMMs. "profile HMMs encode position-specific information about the frequency of particular amino acids as well as the frequency of insertions and deletions in the alignment. They are constructed from multiple alignments of homologous sequences" [4]. Since pHHMs are an abstract representation of a multiple alignments, they can be used to produce pairwise or multiple alignments as well. Thus, aligning a sequence with a pHHM is equivalent to aligning the sequence to many, many other sequences, which where used to establish the pHMM in the first place.

Feature-based Profile HMMs can be used to improve the performance of remote protein homology detection [3].

This is just an application of multiple sequence alignment, no? So perhaps those two can be merged?

4.7 Gene Prediction / Genomic Annotation (?) / Gene Finding (?)

Is Gene Prediction and Genomic Annotation (and Gene Finding) the same problem? Two different words for the same name or are they slightly different problems, with different goals in mind?

HMMs are employed to find eukaryotic genes and to find pseudogenes, which look like functioning genes except for some misplaced stop codons. They are very useful for these problems because of their flexibility [4].

If the hidden states in the HMM can be inferred, the genome can be better annotated or one can understand the dynamics of the genome better [4].

Genomic annotation: Generalized HMMs [2].

Eukaryotic genes can be modeled using HMMs [3].

Pair HMMs can be used for gene prediction [3].

4.8 Protein Sequence Classification

Profile HMMs (analagously to multiple sequence alignment).

4.9 Protein Structure Prediction

Is connected to homology detection. More in [2].

Profile HMMs can be used to "model sequences of protein secondary structure symbols: helix (H), strand (E) and coil (C)" [3]. This model can be used to recognize the three-dimensional fold of new protein sequences based on their secondary protein structure predictions.

Also page 79 in Introduction to Mathematical Methods in Bioinformatics.

4.10 Base-calling

[3]

4.11 Modeling DNA Sequencing Errors

[3]

4.12 ncRNA Identification

[3]

4.13 RNA Structural Alignment

As explained when talking about context-censitive HMMs (csHMMs), profile-csHMMs can be used to perform structural alignment of RNA and performing similarity searches, analagously to how the profile-HMMs can be used to perform multiple alignment of DNA or proteins and performing similarity searches in these cases. RNA sequence analysis is often of high computational complexity, because the alignment algorithms have to deal with base-pair correlations in the sequences that may be very complicated. More about this problem in [3].

5 Examples in R

5.1 CG-islands and the "Fair Bet Casino"

"An introduction to bioinformatics algorithms" - Jones and Pevzner Page 388.

Hay una tarea sobre esto tambien, descargado en pdf en "Books/bioinfo".

The reference list should be alphabetically ordered later!

References

[1] Sean R. Eddy (2004) *What is a hidden Markov model?*, Nature Biotechnology, Volume 22, Number 10, 1315-1316.

[2] Khar H. Choo, Joo C. Tong, Louxin Zhang (2004) *Recent Applications of Hidden Markov Models in Computational Biology*, Geno. Prot. Bioinfo, Vol. 2, No. 2, 84-96.

[3] Byung-Jun Yoon (2009) *Hidden Markov Models and their Applications in Biological Sequence Analysis*, Bentham Science Publishers Ltd., Current Genomics, Vol. 10, No. 6, 402-415.

[4] Nello Christianini, Matthew W. Hahn (2006) *Introduction to Computational Genomics: A Case Studies Approach*, Cambridge University Press.

[5] Smith, L., et al. (2003) *Hidden Markov models and op- timized sequence alignments*, Computational Biology and Chemistry, Vol. 27, 77-84.

[6] Feng DF, Doolittle RF (1987) *Progressive sequence alignment as a prerequisite to correct phylogenetic trees*, J Mol Evol. Vol. 25, No. 4, 351-360. doi: 10.1007/BF02603120. PMID: 3118049.

[7] Mark A. Pinsky, Samuel Karlin (2011) *An Introduction to Stochastic Modeling*, Fourth Edition, Elsevier

[8] DENNE MÅ FØRES OPP ORDENTLIG!

[9] DENNE MÅ FØRES OPP ORDENTLIG!

KANSKJE DENNE NEDENFOR SKAL BRUKES OGSÅ? Rabiner, L.R. 1989. A tutorial on hidden Markov models and selected applications in speech recognition