Hidden Markov Models in Bioinformatics

Valeria De Fonzo[1], Filippo Aluffi-Pentini[2] and Valerio Parisi[*,3]

[1]*EuroBioPark, Università di Roma "Tor Vergata", Via della Ricerca Scientifica 1, 00133 Roma, Italy*

[2]*Dipartimento Metodi e Modelli Matematici, Università di Roma "La Sapienza", Via A. Scarpa 16, 00161 Roma, Italy*

[3]*Dipartimento di Medicina Sperimentale e Patologia, Università di Roma "La Sapienza", Viale Regina Elena 324, 00161 Roma, Italy*

Abstract: Hidden Markov Models (HMMs) became recently important and popular among bioinformatics researchers, and many software tools are based on them. In this survey, we first consider in some detail the mathematical foundations of HMMs, we describe the most important algorithms, and provide useful comparisons, pointing out advantages and drawbacks. We then consider the major bioinformatics applications, such as alignment, labeling, and profiling of sequences, protein structure prediction, and pattern recognition. We finally provide a critical appraisal of the use and perspectives of HMMs in bioinformatics.

Keywords: Hidden markov model, HMM, dynamical programming, labeling, sequence profiling, structure prediction.

INTRODUCTION

A Markov process is a particular case of stochastic process, where the state at every time belongs to a finite set, the evolution occurs in a discrete time and the probability distribution of a state at a given time is explicitly dependent only on the last states and not on all the others.

A Markov chain is a first-order Markov process for which the probability distribution of a state at a given time is explicitly dependent only on the previous state and not on all the others. In other words, the probability of the next ("future") state is directly dependent only on the present state and the preceding ("past") states are irrelevant once the present state is given. More specifically there is a finite set of possible states, and the transitions among them are governed by a set of conditional probabilities of the next state given the present one, called transition probabilities. The transition probabilities are implicitly (unless declared otherwise) independent of the time and then one speaks of homogeneous, or stationary, Markov chains. Note that the independent variable along the sequence is conventionally called "time" also when this is completely inappropriate; for example for a DNA sequence, the "time" means the position along the sequence.

Starting from a given initial state, the consecutive transitions from a state to the next one produce a time-evolution of the chain that is therefore completely represented by a sequence of states that a priori are to be considered random.

A Hidden Markov Model is a generalization of a Markov chain, in which each ("internal") state is not directly observable (hence the term hidden) but produces ("emits") an observable random output ("external") state, also called "emission", according to a given stationary probability law. In this case, the time evolution of the internal states can be induced only through the sequence of the observed output states.

If the number of internal states is N, the transition probability law is described by a matrix with N times N values; if the number of emissions is M, the emission probability law is described by a matrix with N times M values. A model is considered defined once given these two matrices and the initial distribution of the internal states.

The paper by Rabiner [1] is widely well appreciated for clarity in explaining HMMs.

SOME NOTATIONS

For the sake of simplicity, in the following notations we consider only one sequence of internal states and one sequence of associated emissions, even if in some cases, as we shall see later, more than one sequence is to be considered.

Here are the notations:

U	the set of all the N possible internal states
X	the set of all the M possible external states
L	the length of the sequence
k	a time instant, where $k \in [1,\cdots,L]$
s_k	internal state at time k, where $s_k \in U$
$S \equiv (s_1, s_2, s_3, \ldots, s_L)$	a sequence of L internal states
e_k	emission at time k, where $e_k \in X$

*Address correspondence to this author at the Dipartimento di Medicina Sperimentale e Patologia, Università di Roma "La Sapienza", Viale Regina Elena 324, 00161 Roma, Italy; Tel: +39 06 4991 0787; Fax: +39 338 09981736; E-mail: Valerio.Parisi@uniroma1.it

1574-8936/07 $50.00+.00 **© 2007 Bentham Science Publishers Ltd.**

$E \equiv (e_1, e_2, e_3, \ldots, e_L)$	a sequence of L external states
$a_{u,v} = P(s_k = v \mid s_{k-1} = u)$	the probabilities of a transition to the state v from the state u
A	the $N \times N$ matrix of elements $a_{u,v}$
$b_u(x) = P(e_k = x \mid s_k = u)$	the probabilities of the emission x from the state u
B	the $N \times M$ matrix of elements $b_u(x)$
$\pi_u \equiv P(s_1 = u)$	the probability of the initial state u
Π	the N-vector of elements π_u
$\lambda = (A, B, \Pi)$	the definition of the HMM model

A SIMPLE EXAMPLE

We propose an oversimplified biological example of an HMM (Fig. **1**), inspired by the toy example in Eddy [2] with only two internal states but with exponential complexity. The model is detailed in Fig. **1a**.

The set of internal states is $U \equiv \{'c','n'\}$ where $'c'$ and $'n'$ stand for the coding and non-coding internal states and the set of emissions is the set of the four DNA bases:

$$X \equiv \{'A','T','C','G'\}$$

As emitted sequence, we consider a sequence of 65 bases (Fig. **1b**).

It is important to note that in most cases of HMM use in bioinformatics a fictitious inversion occurs between causes and effects when dealing with emissions. For example, one can synthesise a (known) polymer sequence that can have different (unknown) features along the sequence. In an HMM one must choose as emissions the monomers of the sequence, because they are the only known data, and as internal states the features to be estimated. In this way, one hypothesises that the sequence is the effect and the features are the cause, while obviously the reverse is true. An excellent case is provided by the polypeptides, for which it is just the amino acid sequence that causes the secondary structures, while in an HMM the amino acids are assumed as emissions and the secondary structures are assumed as internal states.

MAIN TYPES OF PROBLEMS

The main types of problems occurring in the use of Hidden Markov Models are:

A) Evaluation problem (Direct problem): compute the probability that a given model generates a given sequence of observations.

The most used algorithms are:

1. the forward algorithm: find the probability of emission distribution (given a model) starting from the beginning of the sequence.
2. the backward algorithm: find the probability of emission distribution (given a model) starting from the end of the sequence.

B) Decoding problem: given a model and a sequence of observations, induce the most likely hidden states.

More specifically:

1. find the sequence of internal states that has, as a whole, the highest probability. The most used algorithm is the Viterbi algorithm.
2. find for each position the internal state that has the highest probability. The most used algorithm is the posterior decoding algorithm.

C) Learning problem: given a sequence of observations, find an optimal model.

The most used algorithms start from an initial guessed model and iteratively adjust the model parameters. More specifically:

1. find the optimal model based on the most probable sequences (as in problem B1). The most used algorithm is the Viterbi training (that uses recursively the Viterbi algorithm in B1).
2. find the optimal model based on the sequences of most probable internal states (as in problem B2). The most used algorithm is the Baum-Welch algorithm (that uses recursively the posterior decoding algorithm in B2).

A) THE EVALUATION PROBLEM

The probability of observing a sequence E of emissions given an HMM λ (likelihood function of λ), is given by

$$P(E \mid \lambda) = \sum_S P(E \mid S; \lambda) \cdot P(S \mid \lambda)$$

We note that the logarithm of the likelihood function (log-likelihood) is more often used.

The above sum must be computed over all the N^L possible sequences S (of length L) of internal states and therefore the direct computation is too expensive; fortunately there exist some algorithms which have a considerably lower complexity, for example the forward and the backward algorithms (of complexity $O(N^2 L)$, see below).

A1) The Forward Algorithm

This method introduces auxiliary variables φ_k (called forward variables), where

$\varphi_k(u) = P(e_1, \ldots, e_k; s_k = u \mid \lambda)$ is the probability of observing a partial sequence of emissions $e_1 \ldots e_k$ and a state $s_k = u$ at time k.

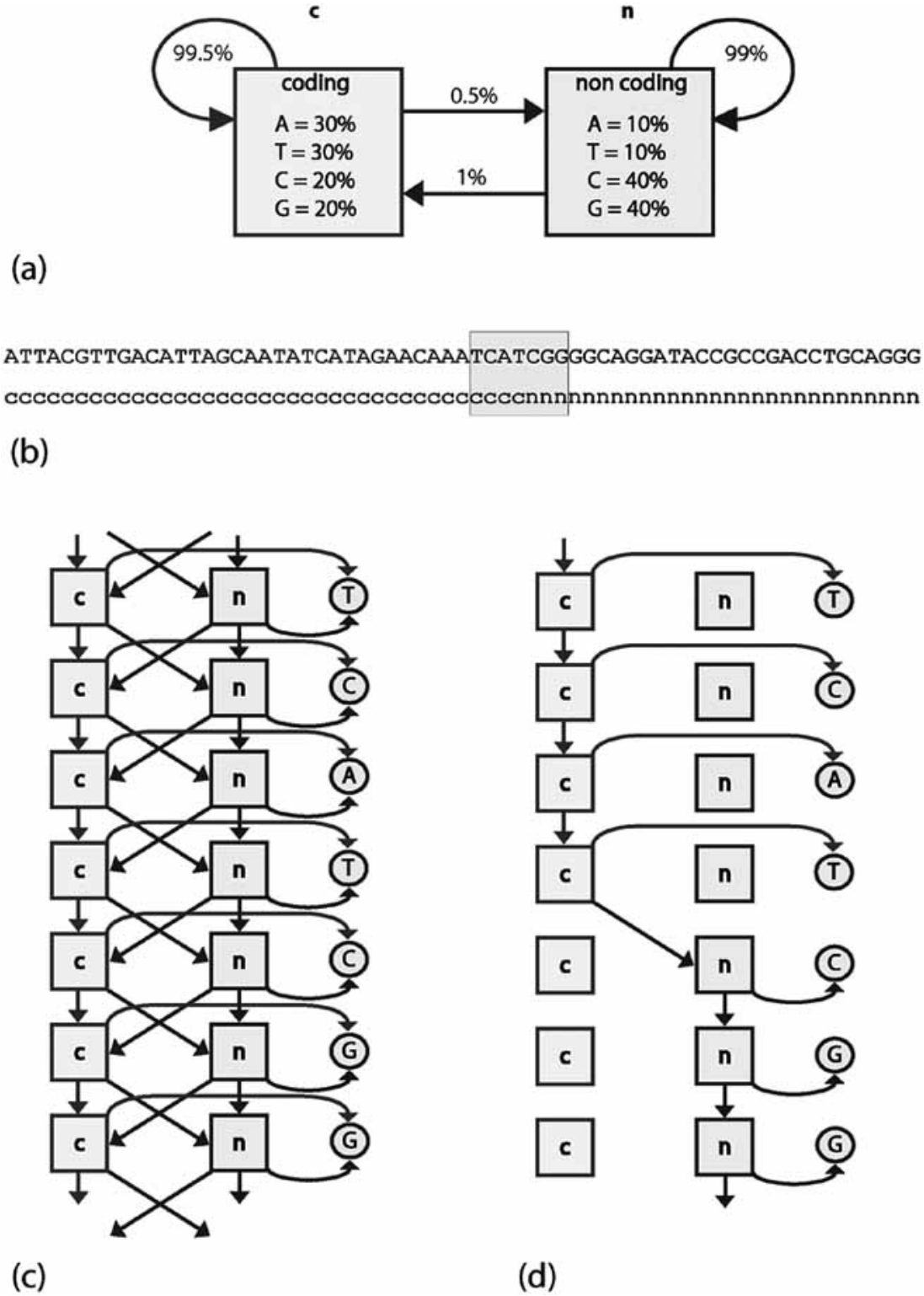

Fig. (1). An example of HMM.

(**a**) The square boxes represent the internal states $'c'$ (coding) and $'n'$ (non coding), inside the boxes there are the probabilities of each emission ($'A'$, $'T'$, $'C'$ and $'G'$) for each state; outside the boxes four arrows are labelled with the corresponding transition probability.

(**b**) The first row is a sample sequence of 65 observed emissions and the second row is one of the likely sequences of internal states. The boxed part is dealt with in (**c**) and (**d**).

(**c**) The right-hand side column represents the boxed tract of bases in (**b**). The other columns represent, for each circled base, the two possible alternatives for the internal state ($'c'$ or $'n'$) that emitted the base. Each row refers to the same position along the sequence. The arrows represent all possible transitions and the emissions.

(**d**) The figure shows a possible likely sequence of choices between the alternative internal states producing the sequence of internal states in (**b**). Such a sequence of choices of internal state transitions amounts to choosing a path in (**c**).

Detailed equations of the algorithm follow:

Initialisation:	$\varphi_1(u) = \pi_u b_u(e_1)$
Recursion: (for $1 \le k < L$)	$\varphi_{k+1}(u) = b_u(e_{k+1}) \sum_v \varphi_k(v) \cdot a_{v,u}$
Termination:	$P(E \mid \lambda) = \sum_u \varphi_L(u)$

Note that the calculation requires $O(N^2 L)$ operations.

A2) The Backward Algorithm

Also this method introduces auxiliary variables β_k (called backward variables), where $\beta_k(u) = P(e_{k+1} \ldots e_L \mid s_k = u; \lambda)$ is the probability of observing a partial sequence of emissions $e_{k+1} \ldots e_L$ given a state $s_k = u$ at time k.

Detailed equations of the algorithm follow:

Initialisation:	$\beta_L(u) = 1$
Recursion: (for $L > k \ge 1$)	$\beta_k(u) = \sum_v \beta_{k+1}(v) \cdot a_{v,u} \cdot b_v(e_{k+1})$
Termination:	$P(E \mid \lambda) = \sum_u \beta_1(u) \cdot \pi_u \cdot b_u(e_1)$

Note that the calculation requires $O(N^2 L)$ operations.

B) THE DECODING PROBLEM

In general terms, a problem of this type is to induce the most likely hidden states given a model and a sequence of observations. The two most common problems of this type, each one requiring an appropriate algorithm, are detailed in the next two paragraphs.

B1) Viterbi Algorithm

The Viterbi algorithm solves the following decoding problem.

Given a model λ and a sequence E of observed states, find the sequence S^* of internal states that maximises the probability $P(E,S \mid \lambda)$, i.e. the sequence S^* such that $p^* \equiv P(E,S^* \mid \lambda) \equiv \max_S (P(E,S \mid \lambda))$ or, more briefly, $S^* \equiv \arg\max_S (P(E,S \mid \lambda))$

The Viterbi algorithm has been designed in order to avoid the overwhelming complexity of a direct approach in the search of the maximum; it is an interesting example of Dynamic Programming (DP), a technique devised by Bellman to optimise multistage decision processes.

As shown in Fig. **1**, a sequence of internal states can be represented as a path; and the DP method applied to path optimisation includes two successive phases: a first phase optimises a number of subproblems, by storing suitable pointers that indicate promising (suboptimal) state transitions, and a second (reverse) phase obtains the optimal path by following the pointers. Detailed equations follow.

Initialisation:	$\gamma_1(u) = b_u(e_1) \cdot \pi_u$ $\psi_1(u) = 0$
Recursion: (for $1 < k \le L$)	$\gamma_k(u) = b_u(e_k) \cdot \max_v (\gamma_{k-1}(v) \cdot a_{v,u})$ $\psi_k(u) = \arg\max_v (\gamma_{k-1}(v) \cdot a_{v,u})$
Termination:	$p^* = \max_v (\gamma_L(v))$ $s_L^* = \arg\max_v (\gamma_L(v))$
Backtracking: (for $L > k \ge 1$)	$s_k^* = \psi_{k+1}(s_{k+1}^*)$

Fig. **2** illustrates the action, on the same tract of the sequence in Fig. **1b**, of the Viterbi algorithm used to decode the whole sequence by means of the model described in Fig. **1a**.

B2) Posterior Decoding

The problem is the following: given a model λ and a sequence E of observed states, find for each k among all the possible internal states u, the most probable internal state s_k^*.

The algorithm computes the probability of each possible internal state using the forward φ and backward β variables derived from A1 and A2 and select the state with highest probability, for each position of the sequence. Detailed equations follow.

$$P(s_k = u \mid E) = \frac{\varphi_k(u) \cdot \beta_k(u)}{P(E \mid \lambda)} \qquad 1 < k \le L$$

$$s_k^* = \arg\max_u (\varphi_k(u) \cdot \beta_k(u)) \qquad 1 < k \le L$$

Note that in the last equation the (irrelevant) denominator has been omitted.

C) THE LEARNING PROBLEM

We know the set of possible internal states, the set of possible external states, and a number of sequences of emissions. We hypothesise that the emissions originate from the same underlying HMM, and more specifically that each sequence of external states has been emitted from an associated sequence of internal states following the laws of the model.

The problem is to estimate the model, i.e. the transition and emission probabilities (for the sake of simplicity we often omit to consider the probabilities of initial states).

Let $E^j \equiv (e_k^j, k = 1, \ldots, L^j)$ $1 \le j \le R$ be the given sequences of emissions, and $S^j \equiv (s_k^j, k = 1, \ldots, L^j)$

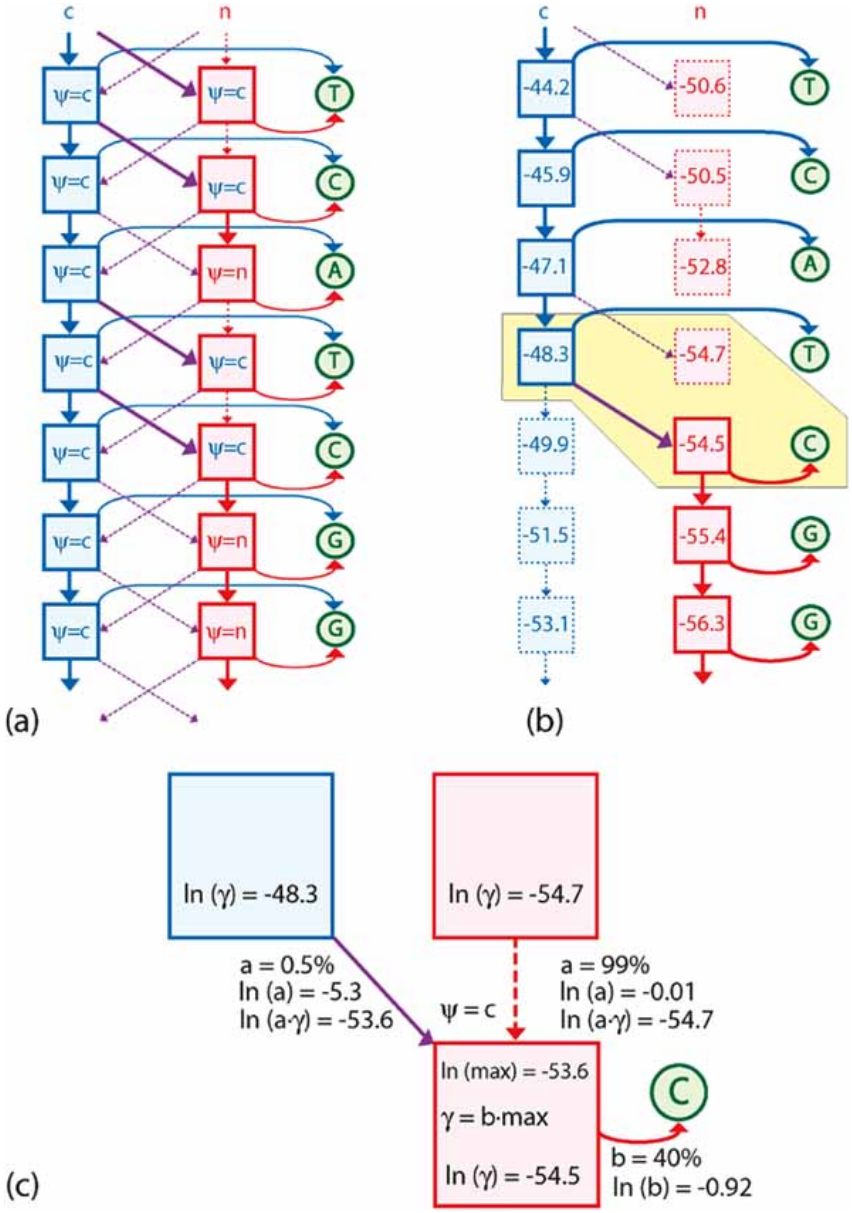

Fig. (2). The action is illustrated, on the same tract of the sequence, of the Viterbi algorithm used to decode the whole sequence by means of the model described in Fig. **1a**. More specifically (**a**) and (**b**) illustrate the transition from Fig. **1c** and **1d** (with the same meanings of the graphics).

(**a**) In each square box, there is the value of the ψ pointer, computed, as illustrated in (**c**), in the first phase. More specifically "$\psi = c$" means that we discard the hypothesis of the transition from the previous state $'n'$ (as indicated also by dashing the corresponding incoming arrow).

(**b**) In each square box, there is the value of the logarithm of the probability γ calculated and used in the first phase. Dashed lines represent the transitions discarded in the second phase. We note that for practical reasons we use the logarithms of the probabilities in order to avoid troubles due to too small numbers.

(**c**) A zoom of the marked zone in (**b**), where the computation of a recursion step of the Viterbi algorithm is detailed.

$1 \leq j \leq R$ the associated (unknown) sequences of internal states.

Usually one starts from an initial guess of the transition and emission probabilities and iteratively one improves them until a suitable stopping criterion is met. More in detail, one recursively gets (from the emissions and from the current model parameters) a suitable estimate of the internal states and, using it, one re-estimates the probabilities (from counts of transition and emission, i.e. one uses as probabilities the relative frequencies). Note that it is useful [3] to somehow regularize the counts often by adding to each count a suitable offset, called pseudocount. The most naïve but usually satisfactory choice is to use the Laplace's rule that sets all the pseudocounts to one. The use of the pseudocounts can seem bizarre but improves the algorithm performances, for example by avoiding considering unusual events as absolutely impossible.

The two most common algorithms used to attack problems of this type are detailed in the next two paragraphs.

C1) Viterbi Training Algorithm

An approach to model parameter estimation is the Viterbi training algorithm. In this approach, the most probable internal state sequence (path) associated to each observed sequence is derived using the Viterbi decoding algorithm. Then this path is used for estimating counts for the number of transitions and emissions, and such counts are used for recalculating the model parameters.

In more detail:

Initialisation:	choose somehow model parameters (initial guess)	$A,\ B,\ \Pi$
	and the pseudocounts (the values to be added to the frequency counts)	$\tilde{A},\ \tilde{B}$
Recursion: (for each iteration)	calculate the most probable internal state sequences S^j (omitting the star) using for each one the Viterbi decoding algorithm	
	calculate the matrices of the observed frequency counts of transitions and of emissions, $\hat{A}$ and $\hat{B}$	$\hat{a}_{u,v} = \sum_j \sum_k \delta\left(u, s_k^j\right) \cdot \delta\left(v, s_{k+1}^j\right)$ $\hat{b}_u(x) = \sum_j \sum_k \delta\left(u, s_k^j\right) \cdot \delta\left(x, e_k^j\right)$ where δ is the usual Kronecker delta
	calculate the regularized frequency counts:	$\bar{A} = \hat{A} + \tilde{A}$ $\bar{B} = \hat{B} + \tilde{B}$
	update the matrices A and B	$a_{u,v} = \frac{\bar{a}_{u,v}}{\sum_w \bar{a}_{u,w}}$ $b_u(x) = \frac{\bar{b}_u(x)}{\sum_y \bar{b}_u(y)}$
	apply, if necessary, a similar updating to Π	
Termination:	stop, if the model parameters do not change for adjacent iterations	

C2) Baum-Welch Algorithm

A different approach to model parameter estimation is the Baum-Welch algorithm. In this approach, the probability distribution of the internal states for each observed sequence is derived using the posterior decoding algorithm. Then these distributions are used for estimating counts for the number of transitions and emissions, and such counts are used for recalculating the model parameters.

More in detail:

Initialisation:	choose somehow model parameters (initial guess)	$A,\ B,\ \Pi$
	and the pseudocounts (the values to be added to the frequency counts)	$\tilde{A},\ \tilde{B}$
Recursion: (for each iteration)	calculate backward and forward coefficients from algorithm A1 and A2 for each sequence	
	calculate the observed (weighted) frequency counts of transitions and of emissions, $\hat{A}$ and $\hat{B}$	$\hat{a}_{u,v} = \sum_j \frac{1}{P\left(E^j \mid \lambda\right)} \sum_{k=1} \varphi_k^j(u) \cdot a_{u,v} \cdot b_v\left(e_{k+1}^j\right) \cdot \beta_{k+1}^j(v)$ $\hat{b}_u(x) = \sum_j \frac{1}{P\left(E^j \mid \lambda\right)} \sum_{k=1} \varphi_k^j(u) \cdot \beta_k^j(u) \cdot \delta\left(x, e_k^j\right)$ where δ is the usual Kronecker delta
	calculate the regularized frequency counts	$\bar{A} = \hat{A} + \tilde{A}$ $\bar{B} = \hat{B} + \tilde{B}$
	update the matrices A and B	$a_{u,v} = \frac{\bar{a}_{u,v}}{\sum_w \bar{a}_{u,w}}$ $b_u(x) = \frac{\bar{b}_u(x)}{\sum_y \bar{b}_u(y)}$
	apply, if necessary, a similar updating to Π	
Termination:	stop, if the convergence is too slow, or if the given maximum number of iterations is reached.	

COMPARISONS

A) Evaluation Problem (Direct Problem)

The backward and forward algorithms use different sets of auxiliary variables, but, being exact methods, they obviously find identical final results on the same problem. We introduced both algorithms since the different sets of auxiliary variables are both needed in the posterior decoding algorithm.

B) Decoding Problem

We recall that the two approaches to the decoding problem are quite different: the approach B1 (Viterbi algorithm) looks for the sequence of internal states that is the most probable, while the approach B2 (Posterior decoding algorithm) looks for the internal state that is the most probable in each position.

It is therefore only natural that the two approaches, attacked with different algorithms, give results that may be quite different, and it is therefore important to stress that, rather than blindly compare the results, one should carefully select a priori the approach that is more appropriate to what one is looking for.

Otherwise, one can easily risk accepting results that may be quite unreliable. On one hand, taking as the most probable internal state in a given position the corresponding internal state in the optimal sequence given by B1, one may take instead an internal state that is rather unlikely. On the other hand, taking as the optimal sequence the sequence having in each position the optimal internal state given by B2, one may take instead a sequence that is unlikely or even impossible.

For the sake of clarity, we consider in some detail another oversimplified biological example, especially designed to illustrate the last circumstance.

We consider an HMM with three possible internal states: 'c' (coding), 't' (terminator), 'n' (non coding), where the possible transitions are shown in Fig. **3**; we note that in order to go from coding to non coding at least a terminator is needed.

We assume that there are only three admissible sequences with given probabilities, as indicated in Fig. **4**, which shows also that, unlike the best sequence "$ccctn$" provided by the Viterbi algorithm, the sequence of most probable states "$cccnn$" provided by the Posterior Decoding algorithm is meaningless, since it is not consistent with the assumption that a coding subsequence must be followed by a terminator.

C) Learning Problem

Similar considerations apply to the comparison between the Viterbi Training and Baum-Welch algorithms, since they are respectively based on the Viterbi algorithm and on the Posterior Decoding algorithm. Both algorithms have the drawback that they can possibly remain trapped in a local attractor. As for the number of iteration steps (in the absence of stopping criteria) the first algorithm converges rapidly (in a few steps) to a point after which there is no further improvement, while the second algorithm goes on converging with progressively smaller improvements.

MAJOR BIOINFORMATICS APPLICATIONS

The HMMs are in general well suited for natural language processing [4, 5], and have been initially employed in speech-recognition [1] and later in optical character recognition [6] and melody classification [7].

In bioinformatics, many algorithms based on HMMs have been applied to biological sequence analysis, as gene finding and protein family characterization. As pioneer applications, we recall the papers of Lander and Green [8] and of Churchill [9]. An excellent critical survey, up to 2001, on HMMs in bioinformatics is provided by Colin Cherry (http://www.cs.ualberta.ca/~colinc/projects/606project.ps). A technical description of HMMs and their application to bioinformatics can be found in the Eddy's paper [10], in the book of Durbin *et al.* [3] and more recently in the survey of Choo *et al.* [11] containing also many software references.

Several HMM-based databases are available: we cite, for example, Pfam [12], SAM [13] and SUPERFAMILY [14]. A method for constructing HMM databases has been proposed by Truong and Ikura [15].

In what follows, we briefly schematise the main works about applications of HMMs in bioinformatics, grouped by kind of purpose.

A detailed description of all applications would be, in our opinion, outside the scope and the size of a normal survey paper. Nevertheless, in order to give a feeling of how the models described in the first part are implemented in real-life bioinformatics problems, we shall describe in more detail, in what follows, a single application, i.e. the use, for multiple sequence alignment, of the profile HMM, which is a powerful, simple, and very popular algorithm, especially suited to this purpose.

Multiple Sequence Alignment

A frequent bioinformatic problem is to assess if a "new" sequence belongs to a family of homologous sequences, using a given multiple alignment of the sequences of the family.

In this framework, a frequently used concept is the consensus sequence, i.e. the sequence having in each position the residue that, among those of the multiple alignment, occurs most frequently in that position.

A related concept is that of a profile: instead of assigning to each position the most frequent residue, assigning a profile to a sequence amounts to assign to each position of the sequence a set of "scores", each one to a residue that can occur in that position. More formally, the profile is a matrix, whose dimensions are the number of positions and the number of possible residues, and that for each position along the multiple alignment, assigns a score to each possible element in such position.

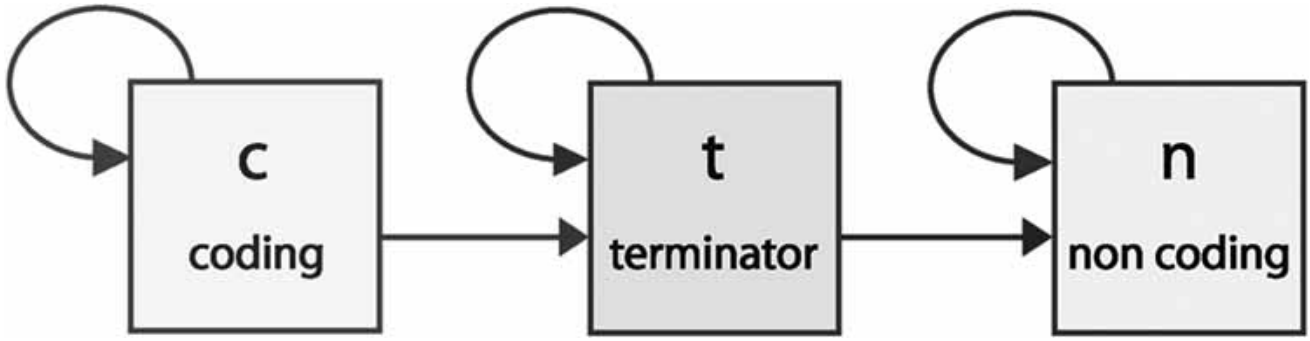

Fig. (3). An example of HMM with three internal states. The square boxes represent the internal states 'c' (coding), 't' (terminator) and 'n' (non coding). The arrows indicate the possible transitions.

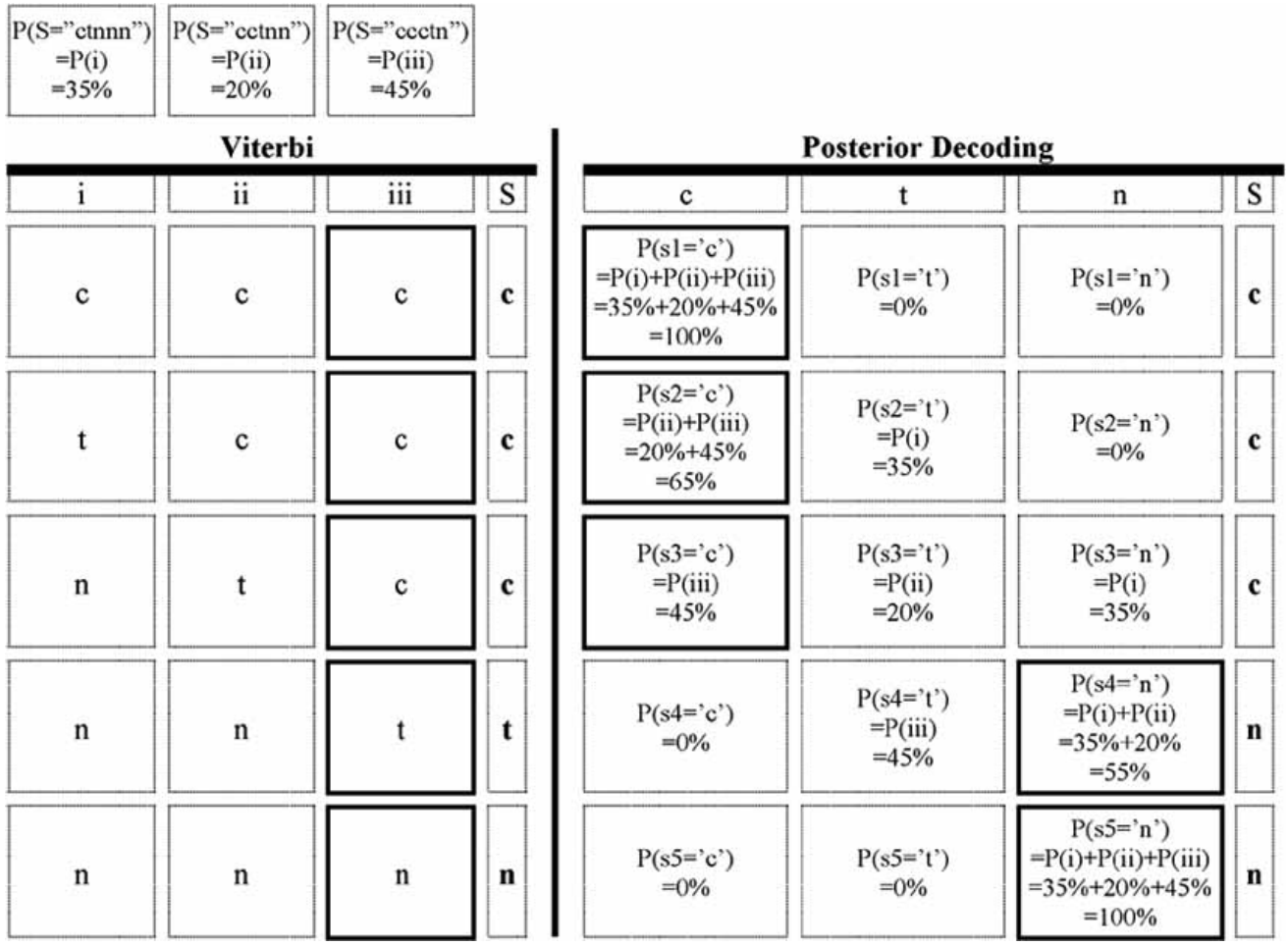

Fig. (4). A simple toy example especially designed to stress the differences between the results of the Viterbi algorithm and of the posterior decoding algorithm in the decoding problem. We consider an HMM with three possible internal states: $'c'$ (coding), $'t'$ (terminator) and $'n'$ (non coding), where the possible transitions are shown in Fig. **3**; we note that in order to go from coding to noncoding at least a terminator is needed. We assume that the admissible sequences of internal states (i.e. sequences with non-negligible probabilities) are those indicated as i, ii, iii (with their probabilities) in the top. The columns i, ii, iii of the Viterbi table are the three admissible sequences while the column S is the sequence that we would have found by the Viterbi algorithm, as the best sequence among all the possible sequences (as it is clear by inspection in this simple case). In the Posterior Decoding (PD) table, the first three columns are relative to the internal states $'c'$, $'t'$, $'n'$, and each one of the positions $s1, s2, \ldots, s5$ of a column contains the probability of finding the corresponding internal state in that position. The probabilities are those that we would have found by the PD algorithm (but that have computed here for the sake of simplicity from the probabilities of the admissible sequences, as indicated in each position). A bold frame shows the most probable state in each position: the column S contains the sequence of the most probable states that the PD algorithm would have selected.

To solve the above mentioned problem, a first technique to judge the total score obtained by aligning (using a suitable choice of the score matrix and of the gap penalties) the new sequence to the consensus sequence obtained from the multiple alignment.

A better technique is to judge the total score obtained by aligning (using the score matrix inside the profile and a suitable choice of the gap penalties) the new sequence to the profile obtained from the multiple alignment.

An even better technique is to use a "profile HMM", an implementation of the HMM which combines the idea of the profile [16] with the idea of the HMM, and has been specially designed for dealing with multiple sequence alignment.

The major advantages of the profile HMM with respect to profile analysis are that in profile analysis the scores are given heuristically, while HMMs strive to use statistically consistent formulas, and that producing a good profile HMMs requires less skill and manual intervention than producing good standard profiles.

A brief description of a profile HMM follows, while the use of a profile HMM is described later on.

We neglect for the moment, for sake of simplicity, insertions and deletions. A no-gap profile HMM is a linear chain of internal states (called match states), each one with unit transition probability to the next match state. Each internal state emits an external state, i.e. an emission, chosen among all possible residues, according to the profile, where

the score is in this case the corresponding emission probability.

However a multiple alignment without gaps is of limited and infrequent utility, and in this case, the profile HMM hardly exhibits its power. Difficulties arise when modelling gaps becomes mandatory; in this case HMM become more complicated but start exhibiting a power greater than in usual profile analysis. For modelling gaps, new features are added to the simple no-gap model.

To account for insertions (exhibited in the new sequence with respect to the consensus sequence) an internal state of a new kind, called insertion state, is added for each match state. Each insertion state emits a residue, in a way analogous to a match state.

Transitions are possible from each match state to the corresponding insertion state, from each insertion state to itself, and from each insertion to the next match state in the chain.

To account for deletions (exhibited in the new sequence with respect to the consensus sequence) an internal state of a new kind, called deletion state, is added for each insertion state. A deletion state does not emit, and therefore is called silent. Transitions from each delete state are possible to the corresponding insertion state, and to the next (in the chain) deletion state and match state; while transitions to each delete state are possible from the preceding deletion, insertion and match states.

The Fig. **5** shows the internal states of a section of the profile HMM, spanning over three positions $(k-1,k,k+1)$ along the multiple alignment (where squares M, diamonds I, and circles D represent match, insertion and deletion states).

It can be seen that for example if a transition occurs from M_k to I_k, and then to I_k, and then again to I_k and finally to M_{k+1} we have an insertion of three residues between the residue emitted by M_k and the residue emitted by M_{k+1}; if instead transitions occur from M_{k-1} to D_k and then to M_{k+1} we have the deletion of the residue that should have been emitted by M_k.

In order to build a complete model, the numerical values of all the emission and transition probabilities of the HMM must be computed from the numbers of occurrences, usually improved by means of pseudocounts. We illustrate, with a simple numerical example, the procedure for computing, by means of Laplace rule (all pseudocounts equal to 1), the emission probabilities in a given position of a multiple alignment. If, in an alignment of 6 DNA sequences, we have the following numbers of occurrences in a given position: 3 occurrences of $'T'$, 2 of $'A'$, 1 of $'C'$ and 0 of $'G'$, we obtain the emission probabilities:

$$b('T') = 40\% = \frac{3+1}{6+4}$$

$$b('A') = 30\% = \frac{2+1}{6+4}$$

$$b('C') = 20\% = \frac{1+1}{6+4}$$

$$b('G') = 10\% = \frac{0+1}{6+4}$$

The numerators of all fractions are the number of occurrences augmented by the pseudocount (equal to 1), while the denominator (the same for all fractions) in the total number of occurrences, plus the 4 pseudocounts.

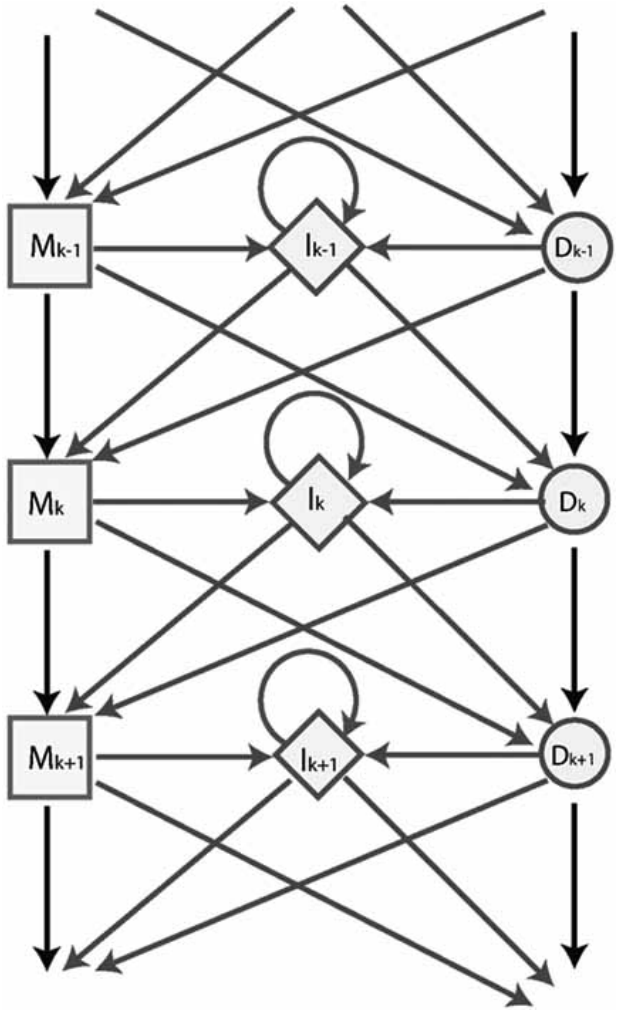

Fig. (5). A tract of a profile HMM.

The internal states are shown of a tract of the profile HMM, spanning over three positions $(k-1,k,k+1)$ the multiple alignment (where squares M, diamonds I, and circles D represent match, insertion and deletion states).

All possible state transitions are represented by arrows, while the emissions of match and insertion states (and all probability values) are not shown to simplify the graphics.

All other emission and transition probabilities are computed in an analogous way.

We now describe briefly the use of a profile HMM to judge a new sequence with respect to a multiple alignment.

One first builds the profile HMM relative to the given multiple alignment.

Then one computes the probability that the new sequence be generated from the profile HMM using one of the

algorithms designed for the so-called evaluation problem, and described above.

Finally, one suitably judges the probability to decide if the new sequence can be considered as belonging to the family of sequences represented by the multiple alignment.

For a good introduction to profile HMM see Eddy [10] and Durbin *et al.* [3].

Apart from some preliminary approaches, the profile HMMs was first introduced by Krogh *et al.* [17].

Soding [18] performed a generalization of the profile HMM in order to pairwise align two profile HMMs for detecting distant homologous relationships.

Eddy [19] described a number of models and related packages that implement profile HMMs, and in particular HMMER, which is commonly used to produce profile HMMs for protein domain prediction.

Genetic Mapping

One of the earliest applications of HMMs in bioinformatics (or even the first, as far as we know) has been the use of a nonstationary HMM for genetic mapping [8], i.e. the estimation of some kind of distance between loci of known (or at least presumed) order along the chromosome.

Lander and Green [8] initially obtained linkage maps (distances in centiMorgans) providing experimental linkage data based on pedigrees; afterwards, in order to obtain radiation maps (distances in centiRays), Slonim *et al.* [20] used a nonstationary HMM starting from experimental radiation data based on gamma irradiation breaks.

Gene Finding

Strictly speaking the term "gene finding" indicates the action of finding genes within a DNA sequence, but is often used with a more general meaning of labeling DNA tracts, for example labeling them as coding, intergenic, introns, etc.

In this last sense gene finding can be considered a special case (the most important in bioinformatics) of the more general action known as sequence labeling (also for non-DNA sequences).

We note that our two toy examples (see above) are in fact two cases of DNA labeling.

In the early 1990s, Krogh *et al.* [21] introduced the use of HMMs for discriminating coding and intergenic regions in *E. coli* genome.

Many extensions to the original "pure" HMM have been developed for gene finding. For example, Henderson *et al.* [22] designed separate HMM modules, each one appropriate for a specific region of DNA. Kulp *et al.* [23] and Burge *et al.* [24] used a generalized HMM (GHMM or "hidden semi-Markov Model") that allows more than one emission for each internal state.

Durbin *et al.* [3] introduced a model called "pair HMM", which is like a standard HMM except that the emission consists in a pair of aligned sequences. This method provides *per se* only alignments between two sequences but, with suitable enhancements, it is sometimes applied to gene finding. For example, Meyer and Durbin [25] presented a new method that predicts the gene structure starting from two homologous DNA sequences, identifying the conserved subsequences. Pachter *et al.* [26], following a similar idea, proposed a generalized pair HMM (GPHMM) that combines the GHMM and the pair HMM approaches, in order to improve the gene finding comparing orthologous sequences. A recent useful open-source implementation is described in Majoros *et al.* [27].

Lukashin and Borodovsky [28] proposed a new algorithm (GeneMark.hmm) that improves the gene finding performance of the old GeneMark algorithm by means of a suitable coupling with an HMM model.

Pedersen and Hein [29] introduced an evolutionary Hidden Markov Model (EHMM), based on a suitable coupling of an HMM and a set of evolutionary models based on a phylogenetic tree.

Secondary Structure Protein Prediction

HMMs are also employed to predict the secondary structure of a protein (i.e. the type of the local three-dimensional structure, usually alpha-helix, beta-sheet, or coil), an important step for predicting the global three-dimensional structure.

Asai *et al.* [30] first used a simple HMM for the secondary structure prediction, while Goldman *et al.* [31] in the HMM approach exploited some evolutionary information contained in protein sequence alignments.

Signal Peptide Prediction

Signal peptide prediction, i.e., the determination of the protein destination address contained in the peptide first tract is often of paramount importance both for diseases analysis and for drug design.

Juncker *et al.* [32] proposed a successful method, using a standard HMM, to predict lipoprotein signal peptides in Gram-negative eubacteria. The method was tested against a neural network model.

Schneider and Fechner [33] provided a thorough review on the use of HMMs and of three other methods for the signal peptide prediction. A very useful feature is a comprehensive list of prediction tools available on the web.

Zhang and Wood [34] created a profile HMM for signal peptide prediction, by means of a novel approach to the use of the HMMER package, together with a suitable tuning of some critical parameters.

Transmembrane Protein Prediction

It is well known that a direct measurement of the complete 3D structure of a transmembrane protein is now feasible only in very few cases. On the other hand, for many practical purposes (such as drug design), it is already very useful to simply know at least the transmembrane protein topology (i.e., whether a tract is cytoplasmatic, extracellular, or transmembrane); and to this end a number of models are available to predict such topology. The secondary structure of the transmembrane tracts of most proteins (the helical transmembrane proteins) is of alpha helix type; important exceptions are the so-called beta-barrels (bundles of transmembrane beta-sheet structures), restricted to the outer membrane of Gram-negative bacteria and of mitochondria.

Some authors [35, 36, 37, 38] specialised the HMM architecture to predict the topology of helical transmembrane proteins. Kahsay *et al.* [38] used unconventional pseudo-counts that they obtained from a modified Dirichlet formula.

Other authors [39, 40, 41] specialised the HMM architecture to predict the topology of beta-barrel transmembrane proteins. Martelli *et al.* [39] trained the model with the evolutionary information computed from multiple sequence alignment, while Bagos *et al.* [41] adopted the conditional Maximum Likelihood proposed by Krogh [42].

Epitope Prediction

A preliminary step in inducing an immune response is the binding of a peptide to a Major Histocompatibility Complex (MHC) molecule, either of class I (as in viral infections or cancer) or of class II (as in bacterial infections). Since, however, most peptides cannot bind to an MHC molecule, it is important to predict which are the epitopes, i.e., the peptides that can bind to an MHC molecule.

Mamitsuka [43] advocated the use of supervised learning (for both class I and II) to improve the performance of HMMs.

A different approach [44, 45], to improve the performance of HMMs in predicting class I epitopes, combines HMM with a new algorithm, the "successive state splitting" (SSS) algorithm.

Yu *et al.* [46] provided a thorough comparative study of several methods, as binding motifs, binding matrices, hidden Markov models (HMM), or artificial neural networks (ANN).

Udaka *et al.* [47], in order to improve the prediction of the binding ability of a peptide to an MHC Class I molecule, used an iterative strategy for the "Query Learning Algorithm" [48], which trains a set of HMMs by means of the so-called "Qbag" algorithm. More specifically the algorithm, within any iteration, indicates the peptides for which the prevision is more uncertain, so that their binding ability is measured, and then fed back, for learning, to the model.

Phylogenetic Analysis

Phylogenetic analysis aims to find probabilistic models of phylogeny and to obtain evolutionary trees of different organisms from a set of molecular sequences.

Felsenstein and Churchill [49] in order to account for the fact that evolution speed varies among positions along the sequences, allowed in their model for three possible speed values as hidden states of the HMM. The optimisation is performed by minimising a suitable objective function by means of Newton-Raphson method.

Thorne *et al.* [50] proposed an evolutionary phylogeny model that uses an HMM to combine the primary structure with a known or estimated secondary structure.

Siepel and Haussler [51] provided a thorough tutorial paper, and considered also HMMs of higher order.

Husmeier [52] used a generalisation of standard HMMs (the so-called factorial HMM), where emissions are due to the combined effect of two internal states belonging to two different hidden Markov chains, the first state representing the tree topology, and the second state the selective pressure.

Mitchinson [53] treated simultaneously alignment and phylogeny by means of the so-called tree-HMM that combines a profile-HMM with a probabilistic model of phylogeny, enhancing it with a number of heuristic approximate algorithms. An iterative version with further enhancements, particularly successful in identifying distant homologs, is described by Qian and Goldstein [54].

RNA Secondary Structure Prediction

The non-coding RNA builds stable and physiologically relevant secondary structures (typically absent in coding RNA) [55]. Such structures are usually stabilised by palindromic tracts, so that predicting the secondary RNA structures essentially amounts to identifying palindromic sequences.

From the standpoint of Chomsky classification of generational grammars, a standard HMM is a stochastic "regular grammar", i.e., belongs to the lowest complexity type (Type 3), and as such is not suitable to identify and study palindromic tracts. This is due to theoretical reasons that obviously cannot be detailed here, but can be roughly understood if one remembers that in a Markov chain the relevant correlation are between neighbour elements, while searching for palindromic tracts requires considering correlations between distant elements.

Therefore, to identify palindromic sequences suitable extensions to pure HMMs must be used, so that they belong to a more complex Chomsky type.

Eddy and Durbin [56] introduced the Covariance Method, which agrees with the stochastic "context-free grammar", one step more general in the Chomsky hierarchy, i.e. Type 2. For a good recent implementation, see Eddy [57].

Knudsen and Hein [58] proposed a method based on a stochastic context-free grammar [59], incorporating evolutionary history information.

Yoon and Vaidyanathan [55] presented a method that can be described as a stochastic "context-sensitive grammar", (one further more general step in the Chomsky hierarchy, i.e. Type 1) which appears to be computationally advantageous with respect to the above approaches.

CONCLUSIONS

As we have seen, the HMMs can be considered a stochastic version of the model that in the Chomsky classification of generative grammars is of the simplest type (Type-3) and is called a regular grammar, the other types being, in order of growing complexity, Context-free (Type-2), Context-sensitive (Type-1), and Recursively enumerable (Type-0).

We have already seen some examples of upgrading HMMs to higher Chomsky levels (see above, RNA secondary structure prediction); we now quote a few examples of models where the HMM concept either undergoes greater variations or plays a less substantial rôle.

McCallum *et al.* [60] introduce a general (non-bioinformatic) model that they call Maximum Entropy Markov

Model (MEMM), and that is basically a Markov model where the internal state does not output an observable "emitted" state, but is determined both from the preceding internal state and from an input observable state. Such a similarity allows exploiting algorithms very similar to those used in a classical HMM. A special kind of enhancement of MEMMs, are the so-called Conditional Random Fields (CRFs) [61], introduced by Lafferty *et al.* [62].

From another, more cybernetic, point of view the use of HMMs can also be considered as special instances of the so-called, and widely used, Machine Learning Techniques, that are often alternatively used for similar applications.

A somehow arbitrary list of such numerous techniques could include, besides HMMs, also:

- Decision Trees (as c4.5)
- Support Vector Machines (SVM)
- Artificial Neural Networks (ANN)
- Clustering
- Genetic Algorithms
- Association Rules
- Fuzzy Sets

Obviously each one of these techniques has pros and cons, often depending on the problem at hand: putting it in somewhat rough terms we can say that the merits of HMMs in bioinformatics are demonstrated by their wide use. Other techniques popular in bioinformatics are ANNs, SVMs and c4.5 [63]. Certainly a detailed comparison of the main techniques, either at conceptual or at benchmark level is beyond the scope of this paper; and on the other hand most available comparisons are too sharply focussed on very narrow subjects. As an example, we recall the comparison between HMMs and ANN'S for epitope prediction, in the already quoted paper by Yu *et al*. [46].

In general terms we can say that the main advantages of HMMs are often the ease of use, the fact that they typically require much smaller training sets, and that the observation of the inner structure of the model provides often a deeper understanding of the phenomenon. Among the main drawbacks of HMMs is often their greater computational cost.

We note that frequently hybrid models are designed combining some of the above techniques, typically with results better than with stand-alone techniques.

For example, HMMs are also used for bioinformatic predictions together with the so-called Support Vector Machine (SVM) [64], a technique based on the Vapnik-Chervonenkis theory [65] that produces decision surfaces in multidimensional spaces, in order to perform various kinds of predictions.

Other examples are provided by several kind of combinations of HMMs with artificial neural networks (ANN): for example Riis and Krogh [66], and Krogh and Riis [67] introduce a model called Hidden Neural Network (HNN), while, in a bioinformatic context, Baldi and Chauvin [68] used them for protein multiple alignments, Boufounos *et al.* [69] for DNA sequencing (without calling them HNNs), and Lin *et al.* [70] use a somehow different model (still called HNN) for protein secondary structure prediction.

If we look at the present state of the HMM concept inside bioinformatics, both from the standpoint of the time of its introduction and of the wealth of available applications, we can say that the concept has been a very fruitful one and that it has reached a somehow mature state. It is also clear that, almost since the very beginning of the field, novel applications have been fostered by many kinds of different extensions, modifications, and contaminations with different techniques, thus producing models that can still be considered, and in fact are still called, more or less appropriately, Hidden Markov Models, and that have been discussed in the preceding sections. We think that the future of HMMs would go on this trend (i.e. continuing along the lines described above), e.g. using more complex and powerful levels in the Chomsky hierarchy, implementing mixed models or further modifying in other ways the true nature of the HMMs, or possibly introducing simultaneously more than one of these variations.

REFERENCES

[1] Rabiner LR. A tutorial on hidden Markov models and selected applications in speech recognition. *Proceedings of the IEEE* **1989**; 77: 257-86.

[2] Eddy SR. What is a hidden Markov model? *Nat Biotechnol* **2004**; 22: 1315-6.

[3] Durbin R, Eddy S, Krogh A, Mitchison G. Biological sequence analysis: probabilistic models of proteins and nucleic acids. Cambridge University Press, Cambridge, UK 1998.

[4] Wu ZB, Hsu LS, Tan CL. A Survey on Statistical Approaches to Natural Language Processing. DISCS Publication No. TRA4/92. Singapore: National University of Singapore, 1992.

[5] Charniak E. Statistical techniques for natural language parsing. *AI Mag* **1997**; 18: 33-43.

[6] Natarajan P, Elmieh B, Schwartz R, Makhoul J. Videotext OCR using Hidden Markov Models. *Proceedings of the Sixth International Conference on Document Analysis and Recognition* **2001**: 947-51.

[7] Pollastri E, Simoncelli G. Classification of Melodies by Composer with Hidden Markov Models. *Proceedings of the First International Conference on WEB Delivering of Music WEDELMUSIC'01*. IEEE Computer Press, **2001**: 88-95.

[8] Lander ES, Green P. Construction of multilocus genetic linkage maps in humans. *Proc Natl Acad Sci U S A* **1987**; 84: 2363-7.

[9] Churchill GA. Stochastic models for heterogeneous DNA sequences. *Bull Math Biol* **1989**; 51: 79-94.

[10] Eddy SR. Hidden Markov models. *Curr Opin Struct Biol* **1996**; 6: 361-5.

[11] Choo KH, Tong JC, Zhang L. Recent applications of Hidden Markov Models in computational biology. *Genomics Proteomics Bioinformatics* **2004**; 2: 84-96.

[12] Sonnhammer EL, Eddy SR, Birney E, Bateman A, Durbin R. Pfam: multiple sequence alignments and HMM-profiles of protein domains. *Nucleic Acids Res* **1998**; 26: 320-2.

[13] Karplus K, Barrett C, Hughey R. Hidden Markov models for detecting remote protein homologies. *Bioinformatics* **1998**; 14: 846-56.

[14] Gough J. The SUPERFAMILY database in structural genomics. *Acta Crystallogr D Biol Crystallogr* **2002**; 58: 1897-900.

[15] Truong K, Ikura M. Identification and characterization of subfamily-specific signatures in a large protein superfamily by a hidden Markov model approach. *BMC Bioinformatics* **2002**; 3: 1.

[16] Gribskov M, McLachlan AD, Eisenberg D. Profile analysis: Detection of distantly related proteins. *Proc Natl Acad Sci USA* **1987**; 84: 4355-8.

[17] Krogh A, Brown M, Mian IS, Sjolander K, Haussler D. Hidden Markov models in computational biology. Applications to protein modeling. *J Mol Biol* **1994b**; 235: 1501-31.

[18] Soding J. Protein homology detection by HMM-HMM comparison. *Bioinformatics* **2005**; 21: 951-60.

[19] Eddy SR. Profile hidden Markov models. *Bioinformatics* **1998**; 14: 755-63.
[20] Slonim D, Kruglyak L, Stein L, Lander E. Building human genome maps with radiation hybrids. *J Comput Biol* **1997**; 4: 487-504.
[21] Krogh A, Mian IS, Haussler D. A hidden Markov model that finds genes in E. coli DNA. *Nucleic Acids Res* **1994**; 22: 4768-78.
[22] Henderson J, Salzberg S, Fasman KH. Finding genes in DNA with a Hidden Markov Model. *J Comput Biol* **1997**; 4: 127-41.
[23] Kulp D, Haussler D, Reese MG, Eeckman FH. A generalized hidden Markov model for the recognition of human genes in DNA. *Proc Int Conf Intell Syst Mol Biol* **1996**; 4: 134-42.
[24] Burge CB, Karlin S. Finding the genes in genomic DNA. *Curr Opin Struct Biol* **1998**; 8: 346-54.
[25] Meyer IM, Durbin R. Comparative ab initio prediction of gene structures using pair HMMs. *Bioinformatics* **2002**; 18: 1309-18.
[26] Pachter L, Alexandersson M, Cawley S. Applications of generalized pair hidden Markov models to alignment and gene finding problems, In: Lengauer T, Sankoff D, Istrail S, Pevzner P, Waterman M Eds, *Proceedings of the Fifth International Conference on Computational Biology (RECOMB)*. Vol. 1, ACM Press, New York, 2001: 241-8.
[27] Majoros WH, Pertea M, Salzberg SL. Efficient implementation of a generalized pair hidden Markov model for comparative gene finding. *Bioinformatics* **2005**; 21: 1782-8.
[28] Lukashin AV, Borodovsky M. GeneMark.hmm: new solutions for gene finding. *Nucleic Acids Res* **1998**; 26: 1107-15.
[29] Pedersen JS, Hein J. Gene finding with a hidden Markov model of genome structure and evolution. *Bioinformatics* **2003**; 19: 219-27.
[30] Asai K, Hayamizu S, Handa K. Prediction of protein secondary structure by the hidden Markov model. *Comput Appl Biosci* **1993**; 9: 141-46.
[31] Goldman N, Thorne JL, Jones DT. Using evolutionary trees in protein secondary structure prediction and other comparative sequence analyses. *J Mol Biol* **1996**; 263: 196-208.
[32] Juncker AS, Willenbrock H, Von Heijne G, Brunak S, Nielsen H, Krogh A. Prediction of lipoprotein signal peptides in Gram-negative bacteria. *Protein Sci* **2003**; 12: 1652-62.
[33] Schneider G, Fechner U. Advances in the prediction of protein targeting signals. *Proteomics* **2004**; 4: 1571-80.
[34] Zhang Z, Wood WI. A profile hidden Markov model for signal peptides generated by HMMER. *Bioinformatics* **2003**; 19: 307-8.
[35] Sonnhammer EL, von Heijne G, Krogh A. A hidden Markov model for predicting transmembrane helices in protein sequences. *Proc Int Conf Intell Syst Mol Biol* **1998**; 6: 175-82.
[36] Tusnady GE, Simon I. Principles governing amino acid composition of integral membrane proteins: application to topology prediction. *J Mol Biol* **1998**; 283: 489-506.
[37] Krogh A, Larsson B, von Heijne G, Sonnhammer EL. Predicting transmembrane protein topology with a hidden Markov model: application to complete genomes. *J Mol Biol* **2001**; 305: 567-80.
[38] Kahsay RY, Gao G, Liao L. An improved hidden Markov model for transmembrane protein detection and topology prediction and its applications to complete genomes. *Bioinformatics* **2005**; 21: 1853-8.
[39] Martelli PL, Fariselli P, Krogh A, Casadio R. A sequence-profile-based HMM for predicting and discriminating beta barrel membrane proteins. *Bioinformatics* **2002**; 18: S46-53.
[40] Liu Q, Zhu YS, Wang BH, Li YX. A HMM-based method to predict the transmembrane regions of beta-barrel membrane proteins. *Comput Biol Chem* **2003**; 27: 69-76.
[41] Bagos PG, Liakopoulos TD, Spyropoulos IC, Hamodrakas SJ. A Hidden Markov Model method, capable of predicting and discriminating beta-barrel outer membrane proteins. *BMC Bioinformatics* **2004**; 5: 29.
[42] Krogh A. Two methods for improving performance of an HMM and their application for gene finding. *Proc Int Conf Intell Syst Mol Biol* **1997**; 5: 179-86.
[43] Mamitsuka H. Predicting peptides that bind to MHC molecules using supervised learning of hidden Markov models. *Proteins* **1998**; 33: 460-74.
[44] Noguchi H, Kato R, Hanai T, Matsubara Y, Honda H, Brusic V, Kobayashi T. Hidden Markov model-based prediction of antigenic peptides that interact with MHC class II molecules. *J Biosci Bioeng* **2002**; 94: 264-70.
[45] Kato R, Noguchi H, Honda H, Kobayashi T. Hidden Markov model-based approach as the first screening of binding peptides that interact with MHC class II molecules. *Enzyme Microb Technol* **2003**; 33: 472–81.
[46] Yu K, Petrovsky N, Schonbach C, Koh JY, Brusic V. Methods for prediction of peptide binding to MHC molecules: a comparative study. *Mol Med* **2002**; 8: 137-48.
[47] Udaka K, Mamitsuka H, Nakaseko, Abe N. Prediction of MHC Class I Binding Peptides by a Query Learning Algorithm Based on Hidden Markov Models. *J Biol Phys* **2002**; 28: 183-94.
[48] Abe N, Mamitsuka H. Query learning strategies using boosting and bagging. *Proceedings of the Fifteenth International Conference on Machine Learning* **1998**; 1-9.
[49] Felsenstein J, Churchill GA. A Hidden Markov Model approach to variation among sites in rate of evolution. *Mol Biol Evol* **1996**; 13: 93-104.
[50] Thorne JL, Goldman N, Jones DT. Combining protein evolution and secondary structure. *Mol Biol Evol* **1996**; 13: 666-73.
[51] Siepel A, Haussler D. Phylogenetic hidden Markov models. In Nielsen R Ed, Statistical Methods in Molecular Evolution. Springer, New York 2005; 325-351.
[52] Husmeier D. Discriminating between rate heterogeneity and interspecific recombination in DNA sequence alignments with phylogenetic factorial hidden Markov models. *Bioinformatics* **2005**; 21: II166-72.
[53] Mitchison GJ. A probabilistic treatment of phylogeny and sequence alignment. *J Mol Evol* **1999**; 49: 11-22.
[54] Qian B, Goldstein RA. Performance of an iterated T-HMM for homology detection. *Bioinformatics* **2004**; 20: 2175-80.
[55] Yoon B-J, Vaidyanathan PP. HMM with auxiliary memory: a new tool for modeling RNA secondary structures. *Proceedings of the Thirty-Eighth Asilomar Conference on Signals, Systems, and Computers*, Monterey, CA, **2004**; 2: 1651-5.
[56] Eddy SR, Durbin R. RNA sequence analysis using covariance models. *Nucleic Acids Res* **1994**; 22: 2079-88.
[57] Eddy SR. A memory-efficient dynamic programming algorithm for optimal alignment of a sequence to an RNA secondary structure. *BMC Bioinformatics* **2002**; 2; 3-18.
[58] Knudsen B, Hein J. RNA secondary structure prediction using stochastic context-free grammars and evolutionary history. *Bioinformatics* **1999**; 15: 446-54.
[59] Sakakibara Y, Brown M, Hughey R, Mian IS, Sjolander K, Underwood RC, Haussler D. Stochastic context-free grammars for tRNA modeling. *Nucleic Acids Res* **1994**; 25: 5112-20.
[60] McCallum A, Freitag D, Pereira F. Maximum entropy Markov models for information extraction and segmentation. *Proceedings of the Seventeenth International Conference on Machine Learning* **2000**: 591–8.
[61] McCallum A. Efficiently inducing features of conditional random fields. *Proceedings of the Nineteenth Conference on Uncertainty in Artificial Intelligence* **2003**: 403-10.
[62] Lafferty J, McCallum A, Pereira F. Conditional random fields: Probabilistic models for segmenting and labeling sequence data. *Proceedings of the Eighteen International Conference on Machine Learning* **2001**: 282-9.
[63] Liu H, Wong L. Data Mining Tools for Biological Sequences. *J Bioinform Comput Biol* **2003**;1: 139-68.
[64] Tsuda K, Kin T, Asai K. Marginalized kernels for biological sequences. *Bioinformatics* **2002**; 18: 268-75.
[65] Boser B, Guyon I, Vapnik V. A training algorithm for optimal margin classifiers. *Fifth Annual Workshop on Computational Learning Theory*. ACM Press, Pittsburgh, 1992; 144-52.
[66] Riis SK, Krogh A. Hidden neural networks: A framework for HMM/NN hybrids. *Proceedings of International Conference on Acoustics, Speech and Signal Processing* **1997**: 3233-6.
[67] Krogh A, Riis SK. Hidden neural networks. *Neural Comput* **1999**; 11: 541-63.
[68] Baldi P, Chauvin Y. Hybrid modeling, HMM/NN architectures, and protein applications. *Neural Comput* **1996**; 8: 1541-65.
[69] Boufounos P, El-Difrawy S, Ehrlich D. Base Calling Using Hidden Markov Models. *J Franklin Inst* **2004**; 341: 23-6.
[70] Lin K, Simossis VA, Taylor WR, Meringa J. A simple and fast secondary structure prediction method using hidden neural networks. *Bioinformatics* **2005**; 21: 152-9.

Received: June 19, 2006 Revised: November 1, 2006 Accepted: November 1, 2006