Application of Hidden Markov Models for the Identification of Short Protein Repeats

Friedhelm Bongardt[1], Ingrid Vetter[2], Wolfgang Urfer[1]

[1] Fachbereich Statistik, Universität Dortmund, 44221 Dortmund, Germany

[2] Max-Planck-Institut für molekulare Physiologie, Otto-Hahn-Str. 11, 44227 Dortmund, Germany

Abstract

In this paper, **h**idden **M**arkov **m**odels (HMMs) are discussed in the context of molecular biological sequence analysis. The statistics relevant in the HMM approach are described in detail. An HMM based method is used to analyze two proteins that contain **s**hort **p**rotein **r**epeats (SPRs). As a benchmark, a state-of-the-art program for the detection of SPRs is also used for both proteins. Finally, an outlook for combination possibilities of HMMs with phylogenetic approaches is given.
Keywords: Hidden Markov models, short protein repeats

1 Introduction

Molecular biologists have to deal with a rapidly increasing amount of data, as new methods facilitate the decipherment of biological sequences. Along with the new data, exact statistical methods must be considered which effectively differentiate between significant relationships and random similarities in the sequences. Suitable models must be biologically meaningful, i. e. they must account for the evolutionary origin of the data and correctly rate mutational events. In the previous years, *profile hidden Markov models* were proven to be useful tools in this field.

In Section 2, profile hidden Markov models are introduced. The features of an HMM architecture specific to biological needs are shown. The statistical background of HMMs, the major problems to deal with and annotations helpful in the practical application of HMMs are given. In Section 3, two proteins are analyzed. They both contain short repetitive units and therefore pose slightly different demands than most other sequences on analysis tools. An HMM based approach is compared to a method specialized in the analysis of short protein repeats. Section 4 gives a short outlook on promising approaches using HMMs.

2 Profile hidden Markov models

In the field of molecular biology, *profile* methods are excessively used for the classification of proteins into protein families. A profile is a model that defines position-specific residue scores and insertion or deletion penalties (Eddy, 1996). A major drawback of these models is the lack of a probabilistic basis. The theory of (profile) hidden Markov models forms an extension to profiles that approaches the problems of protein analysis in a statistically consistent way. HMMs were originally used in speech recognition applications. Therefore, most literature on HMMs is dedicated to this field. An outstanding tutorial on HMMs is given in Rabiner (1989). Krogh *et al.* (1994) describe the theory specialized on biological needs.

2.1 HMM architecture

The theory of hidden Markov models was first described by Baum & Petrie (1966). A hidden Markov model combines two stochastic processes. One of these produces no observable output and therefore inferences about it are only possible on the basis of its influence on the second process. Formally, the situation can be described with a sequence of hidden states $\mathbf{Q} = \{Q_n : n = 0, \ldots, N+1\}$ and an emission sequence $\mathbf{X} = \{X_l : l = 1, \ldots, L\}$. The *emission probability* distribution $\mathrm{P}_{X|Q}$ is given by a matrix $\mathcal{P}$ with

$$\begin{aligned} \mathcal{P} = \Big(\mathrm{P}(X = x | Q = q)\Big)_{x \in \Sigma,\ q \in \Pi} \quad , \\ \text{with} \quad X \in \{X_1, \ldots, X_L\} \quad \text{and} \quad Q \in \{Q_0, \ldots, Q_{N+1}\}. \end{aligned} \tag{1}$$

The realizations $x_1, \ldots, x_n$ from the random variables $X_1, \ldots, X_L$ come from a discrete alphabet Σ, where $|\Sigma| = 20$ for proteins (the 20 amino acids). The realizations q from the discrete random variables $Q_0, \ldots, Q_{N+1}$ come from a set Π of hidden states. The states are specified later for the HMM in Figure 1.

The state sequence $\mathbf{Q}$ is characterized by the *Markov property.* A stochastic process possesses the Markov property, if the outcome of the random variable at one position

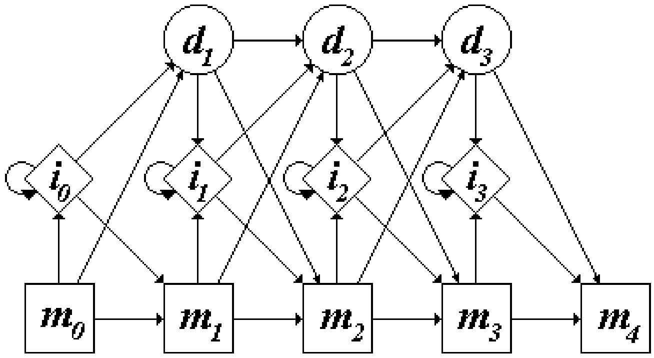

Figure 1: An HMM architecture for biological sequences. The model length is $M = 3$. The figure is taken from Krogh *et al.* (1994).

only depends on the outcome at the previous position. For $\mathbf{Q}$ this means

$$\mathrm{P}(Q_n = q_n | Q_{n-1} = q_{n-1}, \ldots, Q_0 = q_0) = \mathrm{P}(Q_n = q_n | Q_{n-1} = q_{n-1}), \tag{2}$$

for every $n \in \{1, \ldots, N+1\}$. A stochastic process satisfying equation (2) is called a *Markov process* (*of order* 1). Moreover, an initial probability $\mathrm{P}(Q_0 = r)$ must be specified for every state r, because it is the first state of the process and thus depends on no previous states. The basic properties of an HMM as outlined above have given the model its name: a hidden Markov model consists of a sequence of *hidden* states produced by a *Markov* process. Note that the *transition probabilities* $\mathrm{P}(Q_n = r | Q_{n-1} = q)$ for any two states q and r do not depend on the index n, i. e. the Markov process is *stationary* or *homogeneous*. The probability distribution $\mathrm{P}_{Q_n|Q_{n-1}}$ for $\mathbf{Q}$ is given as

$$\mathcal{T} = \Big(\mathrm{P}(Q_n = r | Q_{n-1} = q)\Big)_{q \in \Pi,\ r \in \Pi}, \quad \forall\, n \in \{0, \ldots, N+1\}, \tag{3}$$

where $\mathrm{P}(Q_0 = r | Q_{-1} = q) := \mathrm{P}(Q_0 = r)$, thus including the initial distribution.

A transition from one state q to another state r is called *admissible* when the transition probability $\mathrm{P}(Q_n = r | Q_{n-1} = q)$ is non-zero, and else *nonadmissible*. An HMM is determined by the emission probabilities $\mathrm{P}(X = x | Q = q)$ in equation (1) and the transition probabilities $\mathrm{P}(Q_n = q_n | Q_{n-1} = q_{n-1})$ in equation (2).

Figure 1 shows an architecture typical of HMMs for biological sequence analysis. The states are differentiated into three kinds, namely *match* or *model* states, *insert* states and *delete* states. They are drawn as boxes, diamonds and circles, respectively. The number of match states (without the first and last match state which are only dummy states) is called the length M of the HMM and needs to be fixed *a priori*. Admissible transitions are displayed by an arc between the corresponding states in the figure. Using this architecture, the set of hidden states is $\Pi = \{m_0, \ldots, m_{M+1}, i_0, \ldots, i_M, d_1, \ldots, d_M\}$.

The main line of the HMM can be interpreted as an abstraction of an ancestral protein, where the actual residues are substituted by a probability distribution over the residues in Σ to account for point mutations. It consists of the *model* or *match states* $m_0, \ldots, m_{M+1}$ of the HMM. While m_0 and m_{M+1} only act as dummy begin and end states without output, every state m_k, with $k \in \{1, \ldots, M\}$, produces one of the 20 amino acids according to $\mathrm{P}(X = x|Q = m_k)$ of equation (1), so every match state $m_k, k \in \{1, \ldots, M\}$, is modeled separately and has its own probability distribution over Σ.

Apart from the main line there are states representing deletion and insertion events. Every *insert state* $i_k, k \in \{1, \ldots, M\}$, emits a residue x with probability $\mathrm{P}(X = x|Q = i_k)$ analogous to the distributions over the match states for $X \in \{X_1, \ldots, X_L\}$ and $Q \in \{Q_0, \ldots, Q_{N+1}\}$. Again, every state has its individual probability distribution over Σ. Unlike the match and insert states, the *delete states* $d_k, k \in \{1, \ldots, M\}$, are *silent states*, i. e. they do not emit any residue as output. Obviously, the length M of an HMM specifies the number of non-silent match states of the HMM. All in all, the HMM consists of $3M + 3$ states.

The arrows in Figure 1 indicate the admissible transitions between the states of the HMM. The silent match states m_0 and m_{M+1} play a special role in the modeling process as dummy 'begin' and 'end' states. The initial probability is fixed to

$$\mathrm{P}(Q_0 = m_0) = 1, \qquad \mathrm{P}(Q_0 = q) = 0 \quad \forall\, q \in \Pi \setminus \{m_0\}. \tag{4}$$

In that way, m_0 is predetermined as the starting point for every *path* $\mathbf{q}$ through the HMM, where a path $\mathbf{q}$ denotes a realization of the stochastic process $\mathbf{Q}$ generating the output sequence $\mathbf{X}$. The state m_{M+1} is the only state for which holds

$$\mathrm{P}(Q_n = q|Q_{n-1} = m_{M+1}) = 0 \quad \forall\, q \in \Pi, \tag{5}$$

so m_{M+1} is the terminal state of every path.

Each state m_k, i_k and d_k in Π with $k \leq M-1$ can take three transitions with positive probability. These transitions lead to the match state m_{k+1}, the delete state d_{k+1} and to the insert state i_k. The states m_M, i_M and d_M make an exception in that they only have a transition to i_M and a transition to m_{M+1} because no state d_{M+1} exists. Thus, a transition from a state to the next match state, to the next deletion state or to the insertion state with the same index are modeled in this HMM architecture. This implies the possibility of a self-loop from an insert state. With this method, insertions of any length can be modeled without disturbance of the remaining sequence.

2.2 The three basic problems of an HMM

There are three major tasks that must be solved in an HMM approach. The first one is to determine the probability $\mathrm{P}(\mathbf{X} = \mathbf{x}|\lambda)$ of a given sequence $\mathbf{x} = (x_1, \ldots, x_L)$ of amino acids under model $\lambda = (\mathcal{P}, \mathcal{T})$. Secondly, one has to find the 'optimal' state sequence when a model λ and a residue sequence $\mathbf{x} = (x_1, \ldots, x_L)$ are given. This also demands an optimality criterion. The third problem is the question how to adjust the model parameters of $\lambda = (\mathcal{P}, \mathcal{T})$ so that they maximize the likelihood $\mathrm{P}(\mathbf{X} = \mathbf{x}|\lambda)$.

2.2.1 The evaluation problem

The calculation of the probability of $\mathrm{P}(\mathbf{X} = \mathbf{x}|\lambda)$ is also called the *evaluation problem*. For a fixed length M, model $\lambda = (\mathcal{P}, \mathcal{T})$ is fully specified when all transition probabilities $\mathrm{P}(X = x|Q = q)$ and all emission probabilities $\mathrm{P}(Q_n = r|Q_{n-1} = q)$ are known for all $q, r \in \Pi$. A sequence of L residues is generated by a path of states $q_0, \ldots, q_{N+1}$ with $q_0 = m_0$ and $q_{N+1} = m_{M+1}$ as begin and end state. Obviously, N is larger or equal to L, as no residues arise in delete states.

For a given path $\mathbf{q} = (q_0, \ldots, q_{N+1})$, a new variable $l(n)$ is introduced with $l(0) = 0$ and $l(n) = l(n-1) + 1$ for all insert and match states q_n (including m_{M+1}) and $l(n) = l(n-1)$ for all delete states q_n in path $\mathbf{q}$, $(n \in \{1, \ldots, N+1\})$. This results in a counter $l(n)$ which, denotes the index l in the output sequence $x_1, \ldots, x_L$ of the residue x_l produced in state q_n for match and insert states q_n. For delete states q_n, $l(n)$ means the index of the last observed residue before entering state q_n. If $l(n) = 0$, no residue is already emitted, i. e. the path up to q_n consists of silent states only. Obviously, it holds $l(n) \in \{0, \ldots, L+1\}$ and $n \in \{0, \ldots, N+1\}$.

Using this notation, one can formulate the joint probability of the sequence $\mathbf{x} = (x_1, \ldots, x_L)$ and path $\mathbf{q} = q_0, \ldots, q_{N+1}$ given model λ as

$$
\begin{aligned}
\mathrm{P}(\mathbf{X}=\mathbf{x},\mathbf{Q}=\mathbf{q}|\lambda) &= \mathrm{P}(X_1=x_1,\ldots,X_{l(N+1)}=x_{l(N+1)},\\
&\qquad Q_0=q_0,\ldots,Q_{N+1}=q_{N+1}|\lambda)\\
&\overset{\substack{\text{Markov}\\\text{property}}}{=} \prod_{n=0}^{N+1}\mathrm{P}(Q_n=q_n|Q_{n-1}=q_{n-1})\mathrm{P}(X_{l(n)}=x_{l(n)}|Q_n=q_n),
\end{aligned}
\tag{6}
$$

where $\mathrm{P}(X_{l(n)} = x_{l(n)}|Q_n = q_n) = 1$ by definition if q_n is a silent state. To get the probability of a certain output sequence $\mathbf{x}$ one has to sum over every possible path $\mathbf{q}$ that produces that sequence:

$$
\mathrm{P}(\mathbf{X}=\mathbf{x}|\lambda) = \sum_{\text{path }\mathbf{q}} \mathrm{P}(\mathbf{X}=\mathbf{x},\mathbf{Q}=\mathbf{q}|\lambda). \tag{7}
$$

Although equation (7) gives the probability of the output sequence $\mathbf{x}$, the summation over every possible path needs too many calculations. A more efficient way to get the result is the *forward algorithm*, which is described in Rabiner [1989].This is a *dynamic programming* procedure, which uses the probabilities

$$
\alpha_{l(n)}(q) = \mathrm{P}(X_1=x_1,\ldots,X_{l(n)}=x_{l(n)},Q_n=q|\lambda) \tag{8}
$$

of the partial sequence up to residue $x_{l(n)}$, assuming that $x_{l(n)}$ is generated in state q (or immediately before if q is a delete state), given the model. Modifying Rabiner's calculations for the situation of profile HMMs leads to the following expressions.

1. Initialization

$$
\begin{gathered}
\alpha_0(m_0)=1, \quad \alpha_{l(n)}(m_0)=0 \qquad \forall\, l(n)\in\{1,\ldots,L\},\\
\alpha_0(m_k)=0 \qquad \forall\, k\in\{1,\ldots,M\}, \qquad \alpha_0(i_k)=0 \qquad \forall\, k\in\{0,\ldots,M\}.
\end{gathered}
$$

2. Recursion

$$
\alpha_{l(n)}(m_k) \;=\; \mathrm{P}(X_{l(n)}=x_{l(n)}|Q_n=m_k,\lambda)
$$

$$
\begin{aligned}
& * \quad \sum_{q_{k-1}} \mathrm{P}(Q_n = m_k | Q_{n-1} = q_{k-1}, \lambda) \alpha_{l(n-1)}(q_{k-1}) \\
& \qquad \forall\, l(n) \in \{1, \dots, L\}, \forall\, k \in \{1, \dots, M\}, \\
& \qquad q_{k-1} \in \{m_{k-1}, i_{k-1}, d_{k-1}\} \cap \Pi, \\
\alpha_{l(n)}(i_k) &= \mathrm{P}(X_{l(n)} = x_{l(n)} | Q_n = i_k, \lambda) \sum_{q_k} \mathrm{P}(Q_n = i_k | Q_{n-1} = q_k, \lambda) \alpha_{l(n-1)}(q_k) \\
& \qquad \forall\, l(n) \in \{1, \dots, L\}, \forall\, k \in \{1, \dots, M\}, \\
& \qquad q_k \in \{m_k, i_k, d_k\} \cap \Pi, \\
\alpha_{l(n)}(d_k) &= \sum_{q_{k-1}} \mathrm{P}(Q_n = d_k | Q_{n-1} = q_{k-1}, \lambda) \alpha_{l(n)}(q_{k-1}) \\
& \qquad \forall\, l(n) \in \{0, \dots, L\}, \forall\, k \in \{1, \dots, M\}, \\
& \qquad q_{k-1} \in \{m_{k-1}, i_{k-1}, d_{k-1}\} \cap \Pi.
\end{aligned}
$$

3. Termination

$$
\begin{aligned}
\alpha_{l(N+1)}(m_{M+1}) &= \alpha_{L+1}(m_{M+1}) \\
&= \sum_{\substack{q_M \in \\ \{m_M, i_M, d_M\}}} \mathrm{P}(Q_{N+1} = m_{M+1} | Q_n = q_M, \lambda) \alpha_L(q_M) \\
&= \mathrm{P}(\mathbf{X} = \mathbf{x} | \lambda).
\end{aligned}
$$

The recursion terms $\alpha_{l(n)}(q_n)$ consist of the product of the probability of observing residue $x_{l(n)}$ in state q_n (which is defined as one for a delete state as in equation (6) and can therefore be omitted), the probability of getting from any state q_{n-1} to q_n and the probability to arrive at the state q_{n-1} before the emission of $x_{l(n)}$, given model λ. The termination step yields the probability of getting to the end state m_{M+1} after observing the residues $x_1, \dots, x_L$ under model λ.

2.2.2 Finding the 'optimal' state sequence

The second task is to find the path through the model which best explains a given output sequence $\mathbf{x} = (x_1, \dots, x_L)$ under model λ. Unfortunately, there is no always valid criterion to confirm which path is 'best'. Usually, one tries to find the single best state sequence, the path $\mathbf{q} = (q_1, \dots, q_N)$ that maximizes $\mathrm{P}(\mathbf{Q} = \mathbf{q} | \mathbf{X} = \mathbf{x}, \lambda)$. Equivalently, the probability $\mathrm{P}(\mathbf{Q} = \mathbf{q}, \mathbf{X} = \mathbf{x} | \lambda)$ can be maximized, for the two terms only differ by a factor of $\mathrm{P}(\mathbf{X} = \mathbf{x}|\lambda)$, which is a constant for a given model λ, if the sequence $\mathbf{x}$ is known. Different criteria can be useful in other contexts.

The problem of getting the most probable path is very similar to the evaluation problem cited in Subsection 2.2.1 and it exists an analogous dynamic programming algorithm which solves this task. The summations are replaced by maximization, so the variable $\delta_{l(n)}(q)$ defines the value

$$\delta_{l(n)}(q) = \max_{q_0,\ldots,q_{n-1}} \{\mathrm{P}(Q_0 = q_0, \ldots, Q_{n-1} = q_{n-1}, X_1 = x_1, \ldots, X_{l(n)} = x_{l(n)}|\lambda)\}. \quad (9)$$

A second difference is the need of a backtracking variable $\psi_{l(n)}(q)$, because not the probability of a path but the path itself is searched. The corresponding algorithm is provided in Viterbi (1967) and also named *Viterbi algorithm* after the author. In Bongardt (2001), the computations modified for the current situation are given.

2.2.3 Estimation of the HMM parameters

The estimation of the transition probabilities and amino acid distributions is the most difficult problem of the three mentioned above. No analytical solution to the problem is known, such that the probability $\mathrm{P}(\mathbf{X} = \mathbf{x}|\lambda)$ of the residue sequence $\mathbf{x}$ is maximized with respect to model λ. One approach is an iterative procedure called the Baum-Welch algorithm which finds local optima without any prior knowledge (although prior knowledge can be used to get better results). Although it only obtains local optima, there are methods to increase performance (see Section 2.3).

The Baum-Welch algorithm needs the forward variables $\alpha_{l(n)}(q)$ that have already been introduced in Section 2.2.1. For the parameter estimation a second set of variables $\beta_{l(n)}(q)$ is defined which computes the probability of observing the residues of sequence $\mathbf{x}$ from position $x_{l(n+1)}$ till the end, given that the underlying path $\mathbf{Q}$ is in state q and given model λ. Formally, the β's satisfy the equation

$$\beta_{l(n)}(q) = \mathrm{P}(X_{l(n+1)} = x_{l(n+1)}, \ldots, X_L = x_L|Q_n = q, \lambda), \quad \forall\, q \in \Pi, l(n) \in \{0, \ldots, L\}. \quad (10)$$

Consequently, the values are called backward variables. They are defined by:

1. Initialization

$$\begin{aligned} \beta_{l(n+1)}(m_{M+1}) = \beta_{L+1}(m_{M+1}) &= 1, \\ \beta_{L+1}(q) &= 0 \quad q \in \Pi \setminus \{m_{M+1}\}. \end{aligned}$$

2. Recursion

$$\begin{aligned}
\beta_{l(n)}(q_k) &= \mathrm{P}(X_{l(n+1)} = x_{l(n+1)}|Q_{n+1} = m_{k+1})\mathrm{P}(Q_{n+1} = m_{k+1}|Q_n = q_k)\beta_{l(n+1)}(m_{k+1}) \\
&+ \mathrm{P}(X_{l+1} = x_{l+1}|Q_{n+1} = i_k)\mathrm{P}(Q_{n+1} = i_k|Q_n = q_k)\beta_{l(n+1)}(i_k) \\
&+ \mathrm{P}(Q_{n+1} = d_{k+1}|Q_n = q_k)\beta_{l(n)}(d_{k+1}), \\
& \quad q_k \in \{m_k, i_k, d_k\} \cap \Pi, \quad \forall\, l(n) \in \{0, \ldots, N\}.
\end{aligned}$$

3. Termination

$$\beta_0(m_0) = \mathrm{P}(\mathbf{X} = \mathbf{x}|\lambda).$$

With the aid of the forward variables $\alpha_{l(n)}(q)$ and the backward variables $\beta_{l(n)}(q)$, it is possible to obtain estimations $\hat{\mathcal{P}}$ and $\hat{\mathcal{T}}$ for the emission probability matrix $\mathcal{P}$ and the transition probability matrix $\mathcal{T}$. Details are given in Rabiner (1989) and in Bongardt (2001).

The complete set of probabilities results in the estimation matrix $\hat{\mathcal{T}}$. The following Baum-Welch algorithm is an iterative procedure to obtain locally optimal estimates $\hat{\mathcal{T}}$ and $\hat{\mathcal{P}}$.

The Baum-Welch algorithm

1. Create an initial model $\lambda^{(0)} = (\hat{\mathcal{P}}^{(0)}, \hat{\mathcal{T}}^{(0)})$ by assigning values to the transition probabilities in $\mathcal{T}$ and the emission probabilities in $\mathcal{P}$ for each residue $x \in \Sigma$ and each state $q \in \Pi$. The current model $\lambda^{(t)} = (\hat{\mathcal{P}}^{(t)}, \hat{\mathcal{T}}^{(t)})$ is set to the initial model $\lambda^{(0)}$.

2. Calculate new estimates $\hat{\mathcal{P}}^{(t+1)}$ and $\hat{\mathcal{T}}^{(t+1)}$ of $\mathcal{P}$ and $\mathcal{T}$. Therefore, the estimates for $\mathcal{T}$ and $\mathcal{P}$ have to be calculated for each residue x and all states q of the HMM using the old estimates $\hat{\mathcal{P}}^{(t)}$ and $\hat{\mathcal{T}}^{(t)}$ for the emission and transition probabilities $\mathcal{P}$ and $\mathcal{T}$.

3. Replace $(\hat{\mathcal{P}}^{(t)}, \hat{\mathcal{T}}^{(t)})$ by $(\hat{\mathcal{P}}^{(t+1)}, \hat{\mathcal{T}}^{(t+1)})$ in the current model.

4. Repeat steps 2 and 3 until a previously determined convergence criterion is fulfilled. For example, the procedure can be iterated a fixed number of times (e. g. ten times), or until the model parameters only change insignificantly (Krogh *et al.*, 1994).

The Baum-Welch algorithm can be interpreted as a variant of the **e**xpectation-**m**aximization algorithm (EM algorithm), which is widely used in statistical applications (Selinski *et al.*, 2001). The EM algorithm is a procedure for maximum likelihood estimation with missing data. In profile HMMs the missing data are the unobservable state paths that generate biological sequences.

2.3 Modifications and enhancements

Section 2.2 gives an introduction to the theory of hidden Markov models in the context of biological sequence analysis. For a more complex consideration of HMMs in biology some problems need to be addressed, e. g. the evasion of numerical problems or the avoidance of bad local maxima. This sort of problems is addressed in this section.

An HMM of fixed length and given model architecture is fully specified when all emission and transition probabilities are given, i. e. when $\mathcal{T}$ and $\mathcal{P}$ are known. The Baum-Welch algorithm as introduced in Subsection 2.2.3 searches the model $\lambda = (\mathcal{P}, \mathcal{T})$ that maximizes the likelihood $\mathrm{P}(\mathbf{X} = \mathbf{x}|\lambda)$ of the sequence $\mathbf{x} = (x_1, \ldots, x_n)$. Consequently, this variant of Baum-Welch is also called the **m**aximum **l**ikelihood (ML) approach.

However, in most cases the Baum-Welch algorithm is used in a somewhat altered version, a Bayesian approach called **m**aximum **a** **p**osteriori (MAP). The idea behind the MAP approach is that the real value of interest is the probability $\mathrm{P}(\lambda|\mathbf{X} = \mathbf{x})$ of a model given the observed sequence which shall be maximized. With Bayes' rule, this probability can be obtained with the expression

$$\mathrm{P}(\lambda|\mathbf{X} = \mathbf{x}) = \frac{\mathrm{P}(\mathbf{X} = \mathbf{x}|\lambda) \cdot \mathrm{P}(\lambda)}{\mathrm{P}(\mathbf{X} = \mathbf{x})}. \tag{11}$$

The denominator $\mathrm{P}(\mathbf{X} = \mathbf{x})$ is just a normalizing constant and can therefore be ignored in a maximization problem. The term to be optimized then reduces to

$$\mathrm{P}(\mathbf{X} = \mathbf{x}|\lambda) \cdot \mathrm{P}(\lambda). \tag{12}$$

The MAP approach is basically similar to the ML approach. The only difference is the incorporation of an *a priori* probability $\mathrm{P}(\lambda)$ that must be obtained by prior knowledge about the data. Assuming an appropriate prior over the space of all models helps punishing models that are known to be bad and rewarding good models. The procedure to estimate the model parameters is the same as outlined in Subsection 2.2.3, except that in step 3 the new estimators $\hat{\mathcal{T}}^{(t+1)}$ and $\hat{\mathcal{P}}^{(t+1)}$ are modified with the prior probability of the model. Often used priors for model estimation are simple pseudocounts, where a constant is added to all observed counts, or Dirichlet distributions (For more information see Durbin *et al.* (1998)). Especially for small datasets using priors can be helpful to prevent over-fitting of the model to the training data.

All calculations considered so far are based on one sequence $\mathbf{x}$ only. Usually, a set of training sequences $\mathbf{x}^1, \ldots, \mathbf{x}^m$ is given, from which a model shall be estimated. In that case, the above likelihood of a single sequence is substituted with the joint likelihood of the training sequences. For the sake of computation, the sequences are commonly assumed to be independent of each other. Then the probability of $\mathbf{x}^1, \ldots, \mathbf{x}^m$ is simply the product of the joint probabilities of the single sequences:

$$\mathrm{P}(\mathbf{X}^1 = \mathbf{x}^1, \ldots, \mathbf{X}^m = \mathbf{x}^m | \lambda) = \prod_{j=1}^{m} \mathrm{P}(\mathbf{X}^j = \mathbf{x}^j | \lambda). \tag{13}$$

To maximize this quantity, small changes in the estimation of $\mathcal{P}$ and $\mathcal{T}$ are necessary, namely the sequences have to be weighted according to their probability under the current model λ. For a more detailed description see Bongardt (2001).

An important assumption of hidden Markov models is the independence of the sequences. Obviously, this does not hold in reality, because all biological sequences have developed in an evolutionary process from a single ancestor and are therefore phylogenetically related to each other (Durbin *et al.*, 1998). This is made worse by biases in sampling, as not all proteins are of the same interest for researchers. Thus, some precautions must be taken to account for the violation of the independence assumption. Normally, such over-fitting is circumvented by down-weighting related sequences in the training set or by eliminating sequences from the set by hand (Durbin *et al.*, 1998).

One problem of the variables defined by now is that they tend to get very small when the sequences are long. For instance, the probability of the most probable path, which is given by the Viterbi variable $\delta_{l(n+1)}(m_{M+1})$, is a product of about $2L$ terms lesser than one, namely the emission and transition probabilities at each position. If even whole databases of proteins are analyzed, computers run into numerical problems. This can lead to an underflow error in the course of a program. The errors can be escaped by working in log-space or by scaling of the variables (Durbin *et al.*, 1998).

The MAP approach shows one possibility to use prior information in the estimation process. Another way is to use prior knowledge in the choice of a starting model $\lambda^{(0)}$ in the Baum-Welch algorithm. The algorithm is guaranteed to find a local optimum, but there is no way of knowing how near it is to the global optimum. Finding a good starting point is known to be a powerful heuristic.

Even when no information is available before, there are ways to avoid bad local maxima. The simplest possibility is starting the procedure many times from different random models and keeping the best scoring one, i. e. the one with the highest likelihood. Another approach is to bump the Baum-Welch algorithm of minor local optima by adding noise to the model before each re-estimation step and decreasing the noise gradually to zero. The idea to use a stochastic process for the evasion of local extrema is burrowed from physics, where it is used in a procedure called *simulated annealing*. It is common to combine both methods, so that several runs of the Baum-Welch algorithm with simulated annealing are performed and the best scoring model is kept.

The length M of an HMM must be chosen *a priori*. This is normally done using biological knowledge of the sequences or by taking the average number of residues per sequence. However, sometimes better models can be found by changing the model length. With a heuristic procedure called *model surgery*, match states can be added to or removed from an HMM. After Baum-Welch training, the path of the training sequences are analyzed. If more than a predefined fraction γ_{del} of the sequences uses the delete state d_k, position k is removed from the model. On the other hand, if more than a fraction γ_{ins} makes use of the insert state i_k, position k is split into a number of new positions according to the average number of insertions made there.

The HMM architecture of Figure 1 is only capable of modeling whole proteins. To deal with protein subunits (*domains*), the architecture must be modified as in Figure 2. Central to the new model is the old model from Figure 1 being responsible for the domain itself. The surrounding regions of the main model consist of new dummy begin and end states B and E of the whole protein. Before and after the main model, new insert states I_B and I_E take care of the regions outside the domain. The new transitions are usually specified with only one new parameter p. The transition probabilities are then given as p for B to I_B, from I_B to I_B, from m_{M+1} to I_E and from I_E to I_E, and as $1 - p$ for B to m_0, from I_B to m_0, from m_{M+1} to E and from I_E to E. The usage of the same parameter p before and after the main model inhibits biases to put the domain towards the beginning or the end of the protein. Figure 2 also incorporates the

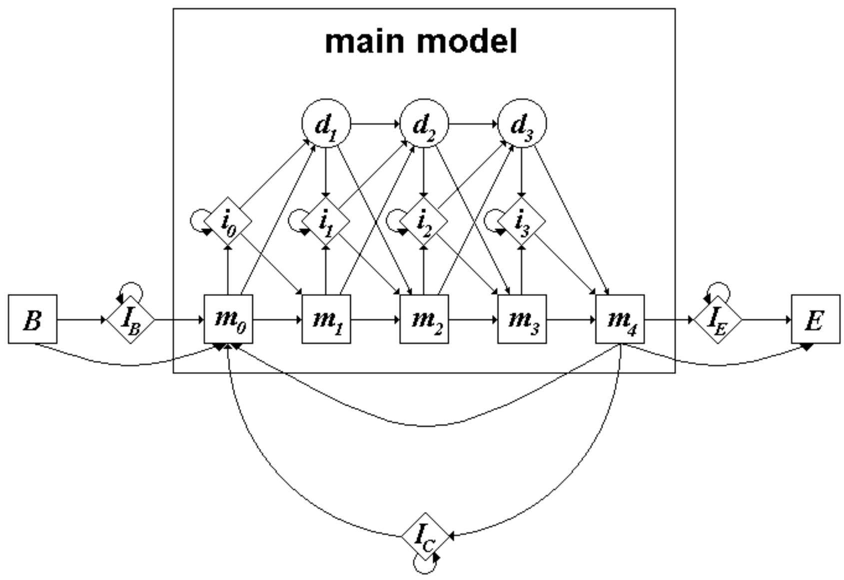

Figure 2: An HMM architecture for modeling multiple domains.

possibility to model several domains in a protein. This is enabled through transitions from state m_{M+1} to a new state I_C, a self-loop over I_C and from I_C to m_0.

New technologies increase the speed of biosequence decipherment so that huge amounts of data are available. Therefore, reliable methods are needed which discriminate between related proteins and random matches in database searches in a statistically meaningful way. Using the scores mentioned in the sections above, *expectation values* (*E-values*) can be calculated for a sequence with respect to a protein family. The E-value of a sequence $\mathbf{x}$ then specifies the number of sequences in the database that yield a score at least as good as $\mathbf{x}$ expected per chance alone for the family of interest. Therefore, a low E-value suggests that the resemblance between sequence $\mathbf{x}$ and the protein family is not based purely on chance but results from a biological relationship instead. The statistics involved in the calculation of the E-value rely on the **e***xtreme* **v***alue* **d***istribution* (*EVD*) and can be looked up in Dembo *et al.*(1994a) and Dembo *et al.*(1994b) for ungapped comparisons. Altschul & Gish (1996) performed computational experiments which suggested that the theory remains valid for gapped comparisons.

3 Results

In this paper, the HMM-based program HMMER 2 (Eddy, 2001) was used to analyze two proteins, namely the human variant of the importin β-1 subunit and the subunit A of the human variant of serine/threonine protein phosphatase 2A in the α isoform. Their entry names in the protein database SWISS-PROT are used here, which are IMB1_HUMAN for the importin β and 2AAA_HUMAN for the protein phosphatase. As a benchmark for the performance of HMMER 2, a second program REP (Andrade *et al.*, 2000) is run with the same proteins. The program REP is especially designed for the analysis of short protein repeats.

Both proteins are known to contain short protein repeats from the repeat family HEAT (an acronym of proteins and domains containing this sort of repeats, namely human **H**untingtin, elongation factor III (**E**F3), subunit **A** of PP2A and the lipid kinase **T**OR1 (Andrade *et al.*, 2001)). These usually comprise 37 to 43 amino acids. They occur in blocks of three to 22 tandem repeats consisting of two anti-parallel helices. Neighboring repeats stack together into a single domain with a hydrophobic core, forming an elongated super-helical (*solenoid*) structure which plays a crucial role in protein-protein interactions (Andrade *et al.*, 2001). The protein 2AAA_HUMAN contains 15 and IMB1_HUMAN 19 HEAT repeats (Groves *et al.* (1999) and Andrade *et al.* (2001)). Vetter *et al.* (1999) describe the structure of HEAT repeats in Importin β.

Closely related to the HEAT motif is a repeat motif called ARM repeats. Its name is deduced from the armadillo protein found in *Drosophila melanogaster*. Kobe *et al.*(1999) introduce it as an acronym of **a**nother **r**epetitive **m**otif. The ARM motif strongly resembles HEAT in its structure. The most striking difference between HEAT and ARM repeats is a third helix present in ARM. Due to their relatedness, the differentiation between HEAT and ARM repeats is a difficult task.

The REP program is specialized on the detection of short protein repeats. It adopts a special scoring scheme in which a repeat motif is only significant in connection with further motifs to account for the fact that such repeats are usually propagated in several copies. To this end, REP uses two values. The first value n_{min} is simply the number of significant motifs REP finds in a given protein. The second value is an E-value threshold P_θ common to all repeats that the motifs must reach to be considered significant. A heuristic approach is taken to increase the sensitivity of the program which is described in Andrade *et al.* (2000). Therefore, a motif in a protein does

Repeat	P_θ	$n_{\min}$
ARM	10^{-8}	3
HEAT	10^{-6}	4
HEAT_AAA	10^{-5}	5
HEAT_ADB	10^{-8}	4
HEAT_IMB	10^{-6}	5

Table 1: Thresholds used in the REP program. The three lower repeats are the sub-families of the HEAT repeat.

not have to reach the threshold exactly, if more high-scoring motifs of the same kind are present. A more detailed description of the application of the threshold values is available in Andrade *et al.* (2000).

The program HMMER 2 is based on the theory of HMMs. However, it uses a slight deviation of the HMM-architecture of Figure 1 called the *plan 7* architecture. The effect on scores is negligible and the theory stays basically the same. The exact differences are described in Eddy (2001). For the analysis of the two sequences, only the E-values computed by the programs are used, not the raw scores that cannot be interpreted statistically like the E-values.

Both programs need family alignments for protein analysis. In this application, 14 alignments for 14 repeat families are used. These are available via internet under the address *http://www.embl-heidelberg.de/˜andrade/papers/rep/search.html*. Information about the repeat families are available in Andrade *et al.* (2000). The quality of the alignments is crucial for the correct identification of repeat motifs as shown in Bongardt (2001). Among the 14 families under investigation are the formerly mentioned ARM repeat family and the HEAT family, and additionally three sub-families of the more divergent HEAT family, namely HEAT IMB, HEAT AAA and HEAT ADB. The protein IMB1_HUMAN belongs to the HEAT IMB sub-family, the protein 2AAA_HUMAN belongs to HEAT AAA. For the HEAT sub-families, REP must specify new thresholds n_{min} and P_θ. All thresholds are given in Table 1. The thresholds for ARM, HEAT and its sub-families are extremely low to prevent misclassifications.

The package HMMER 2 gives overall scores for families, which are calculated as the sum of the scores of all domains found. Table 2 shows the results for the two proteins. Only families with an E-value lesser than 10^{-4} are displayed. With this restraint, the best E-value is obtained with the model of the correct HEAT sub-family, HEAT IMB for IMB1_HUMAN (E-value: 6.6×10^{-72}) and HEAT AAA for 2AAA_HUMAN (E-value: 8.6×10^{-90}). For both proteins, the second best scoring model is the common HEAT model, which makes sense because the HEAT family contains both sub-families. Relations to further HEAT sub-families and even to the ARM family are also detectable.

IMB1_HUMAN		
Model	E-value	Domains
HEAT_IMB	6.6×10^{-72}	13
HEAT	4.9×10^{-37}	11
HEAT_AAA	1.8×10^{-31}	10
ARM	2.2×10^{-16}	6

2AAA_HUMAN		
Model	E-value	Domains
HEAT_AAA	8.6×10^{-90}	14
HEAT	5.8×10^{-62}	14
HEAT_IMB	3.0×10^{-30}	12
ARM	3.0×10^{-7}	6
HEAT_ADB	1.5×10^{-6}	4

Table 2: The HMMER 2 family scores for the proteins IMB1_HUMAN (*left*) and 2AAA_HUMAN (*right*).

Besides the family scores, HMMER 2 also gives E-values for every single domain found. Table 3 shows the best E-values computed by HMMER 2 below 10^{-4} for every domain, i. e. the value for the HEAT model, for instance, is not shown even if it is less than 10^{-4} if the value for HEAT IMB is lower. Under these conditions, HMMER 2 finds 6 occurrences of the HEAT IMB sub-family in the protein IMB1_HUMAN (Table 3, *left*), and 10 occurrences of the HEAT AAA sub-family in the protein 2AAA_HUMAN (Table 3, *right*).

IMB1_HUMAN				
Family	Domain	from	to	E-value
HEAT IMB	4/6	404	441	2.8×10^{-10}
HEAT IMB	3/6	362	399	1.1×10^{-9}
HEAT IMB	5/6	447	483	3.4×10^{-8}
HEAT IMB	1/6	124	163	8.6×10^{-7}
HEAT IMB	6/6	687	726	4.2×10^{-6}
HEAT IMB	2/6	213	250	1.5×10^{-5}

2AAA_HUMAN				
Family	Domain	from	to	E-value
HEAT AAA	5/10	276	314	4.9×10^{-9}
HEAT AAA	3/10	198	236	2.3×10^{-8}
HEAT AAA	4/10	237	275	3.0×10^{-8}
HEAT AAA	7/10	358	396	3.3×10^{-8}
HEAT AAA	10/10	514	552	3.9×10^{-8}
HEAT AAA	6/10	319	357	5.1×10^{-8}
HEAT AAA	8/10	397	435	9.0×10^{-7}
HEAT AAA	2/10	159	197	2.0×10^{-6}
HEAT AAA	1/10	82	120	9.7×10^{-6}
HEAT AAA	9/10	475	513	1.4×10^{-5}

Table 3: Repeat motifs found by HMMER 2 in the proteins IMB1_HUMAN (*left*) and 2AAA_HUMAN (*right*).

With the scoring system described in Andrade *et al.* (2001), the REP program finds 9 repeats of the HEAT IMB sub-family in the protein IMB1_HUMAN and 13 repeats of the HEAT AAA sub-family (Table 4). In both cases, REP finds more repeats than HMMER 2, and their E-values tend to be better, too. The REP values for 2AAA_HUMAN are especially good in comparison to the HMMER 2 scores. For a more elaborate analysis of the results of both programs for IMB1_HUMAN and 2AAA_HUMAN see also Bongardt (2001).

IMB1_HUMAN				
Family	Domain	from	to	E-value
HEAT IMB	5/9	361	399	1.0×10^{-9}
HEAT IMB	7/9	446	483	3.2×10^{-9}
HEAT IMB	9/9	686	726	5.9×10^{-9}
HEAT IMB	3/9	212	250	7.3×10^{-8}
HEAT IMB	1/9	123	163	7.4×10^{-8}
HEAT IMB	4/9	316	361	8.2×10^{-8}
HEAT IMB	2/9	167	207	1.2×10^{-7}
HEAT IMB	8/9	601	641	1.6×10^{-7}
HEAT IMB	6/9	403	441	3.4×10^{-7}

2AAA_HUMAN				
Family	Domain	from	to	E-value
HEAT AAA	5/13	198	236	2.5×10^{-14}
HEAT AAA	10/13	398	435	6.7×10^{-14}
HEAT AAA	7/13	276	314	8.2×10^{-14}
HEAT AAA	12/13	514	552	8.9×10^{-14}
HEAT AAA	9/13	359	396	1.3×10^{-13}
HEAT AAA	6/13	238	275	9.3×10^{-13}
HEAT AAA	2/13	45	81	3.0×10^{-11}
HEAT AAA	1/13	5	43	4.8×10^{-11}
HEAT AAA	3/13	83	120	2.0×10^{-10}
HEAT AAA	13/13	554	588	1.5×10^{-9}
HEAT AAA	4/13	159	197	1.0×10^{-7}
HEAT AAA	8/13	319	357	1.5×10^{-7}
HEAT AAA	11/13	475	513	3.7×10^{-6}

Table 4: Repeats found by the REP program in both proteins IMB1_HUMAN (*left*) and 2AAA_HUMAN (*right*).

4 Outlook

Although the results of HMMER 2 are worse than these of REP, its approach is a promising alternative in the detection of short protein repeats. The theory of hidden Markov models is highly developed and well understood in the context of protein analysis. It is founded on a proper statistical model, whereas the REP approach is more heuristic in its nature. For the future, modifications of the HMM architecture to account for the special situation of short protein repeats could lead to an increased performance of HMM-based methods. In particular, HMMs for HEAT repeats have to consider their tendency to occur tandemly, to be propagated partially and similar features that can be modeled by using a different HMM architecture with additional transitions.

The program REP has no feature allowing it to find repeats in unaligned data. In contrast, the Baum-Welch procedure of Section 2.2.3 offers the possibility to work with unaligned sequences. Attempts to incorporate the procedure into the HMMER 2 package have already started (Eddy, S. R., 2001). In general, HMM methods that use the Baum-Welch algorithm are more appropriate than REP if no reliable alignment is at hand.

As mentioned in Section 2.3, the assumption underlying an HMM that the sequences have developed along independent lines is not quite appropriate. In Section 2.3 measures to minimize the effect of the faulty assumption have been proposed. Another promising approach is the linkage of HMMs to methods that operate on the basis of phylogenetic trees. Rehmsmeier & Vingron (2001) propose a procedure that is able to build an alignment and a phylogenetic tree simultaneously. In their paper, they hold an optimistic view that phylogenetic and hidden Markov methods can be conjoined to exploit the advantages of each method, namely, the evolutionary perspective of phylogenetic trees and the machine learning view of HMMs.

Acknowledgement

The financial support of the Deutsche Forschungsgemeinschaft (SFB 475, 'Reduction of complexity in multivariate data structures') is gratefully acknowledged.

References

Altschul, S. F., Gish, W. (1996), Local alignment statistics, *Methods in Enzymology*, **266**, 460–488.

Andrade, M. A., Ponting, C. P., Gibson, T. J., Bork, P. (2000), Homology-based method for identification of protein repeats using statistical significance estimates, *Journal of Molecular Biology*, **298**, 521–537.

Andrade, M. A., Petosa, C., O'Donoghue, S. I., Müller, C. W., Bork, P. (2001), Comparison of ARM and HEAT protein repeats, *Journal of Molecular Biology*, **309**, 1–18.

Baum, L. E., Petrie, T. (1966), Statistical inference for probabilistic functions of finite state Markov chains, *Annals of Mathematical Statistics*, **1966**, 1554–1563

Bongardt, F. (2001), Identification of Short Protein Repeats in Two Amino Acid Sequences, *Diploma Thesis*, Department of Statistics, University of Dortmund.

Durbin, R., Eddy, S., Krogh, A., Mitchison, G. (1998), Biological Sequence Analysis, Cambridge University Press, Cambridge.

Eddy, S. R. (1996), Hidden Markov models, *Current Opinion in Structural Biology*, **6**, 361–365.

Eddy, S. R. (2001), HMMER: Profile hidden Markov models for biological sequence analysis, *http://hmmer.wustl.edu/*

Groves, M. R., Hanlon, N., Throwski, P., Hemmings, B. A., Bartford, D. (1999), The structure of the protein phosphatase 2A PR65/A subunit reveals the conformation of its 15 tandemly repeated HEAT motifs, *Cell*, **96**, 99–110.

Kobe, B., Gleichmann, T., Horne, J., Jennings, I. G., Scotney, P. D., Teh, T. (1999), Turn up the HEAT, *Structure*, **7**, R91–R97.

Krogh, A., Brown, M., Mian, S., Sjölander, K., Haussler, D. (1994), Hidden Markov models in computational biology, *Journal of Molecular Biology*, **235**, 1501–1531.

Rehmsmeier, M., Vingron, M. (2001), Phylogenetic information improves homology detection, *Proteins: Structure, Function & Genetics* **45**, 360–371.

Selinski, S., Golka, K., Bolt, H. M., Urfer, W. (2001), Estimation of toxicokinetic parameters in population models for inhalation studies with ethylene, *Environmetrics*, **11**, 479–495.

Vetter, I. R., Arndt, A., Kutay, U., Görlich, D., Wittinghofer, A. (1999), Structural view of the Ran-importin β interaction at 2.3 Å resolution, *Cell*, **97**, 635–646.

Viterbi, A. J. (1967), Error bounds for convolutional codes and an asymptotically optimal decoding algorithm, *IEEE Transactions on Informational Theory*, **13**, 260–269.