

Chapter 8

Hidden Markov Chains

8.1 Definition and basic theory

Let X_1, X_2, X_3, \dots be the nucleotide bases observed along a randomly chosen DNA strand. The Markov chain models introduced so far for modelling $\{X_n\}$ have made the assumption of *homogeneity*, that is, that the transition probabilities

$$a_{ij} = \mathbb{P}(X_{n+1} = j \mid x_n = i),$$

are the same for all sites n . (Since n is generally thought of as time in Markov chain theory, this property is often called *time-homogeneity*.) As different parts of a DNA strand may have different functions or chemical properties, the assumption of homogeneity may be very inaccurate. A case in point is the phenomenon of *CpG*-islands. In some regions of DNA, the two base sequence $5' - GC - 3'$ appears with markedly greater frequency than in other regions. The culprit is a process called DNA methylation, whereby methyl groups ($-CH^3$) attach themselves to nucleotide bases. The methyl group binds to base C in the pair and promotes its decay, or rather, transformation, to T . This process operates commonly over most of the DNA sequence. (I read that in human DNA it reduces the occurrence of GC to about one-fifth of what one would expect; I guess this means that the frequency is one-fifth of what one would expect from an i.i.d. site model, but biologists don't seem to be given to such accuracy of statement (grumble,grumble).) However, there are regions, called *CpG*-islands, which are often close to genes and are associated with regulating gene expression, where this decay is prevented and a higher frequency of GC is maintained. Suppose we built a Markov chain to model

the succession of bases inside *CpG* islands alone and a separate Markov chain to model the bases outside *CpG* islands. The transition probability a_{GC} would differ substantially in the two models. In models estimated from biological data in the book, *Biological Sequence Analysis*, by Durbin, Eddy, Krogh, and Mitchison, the values for a_{GC} are 0.078 and 0.274 in the respective cases. A homogeneous Markov chain model for entire DNA sequences would miss the *CpG* islands entirely.

One obvious way to handle *CpG* islands using Markov chains would be to relax homogeneity by allowing the transition probabilities to depend on n :

$$a_{ij}^{(n)} = \mathbb{P}(X_{n+1} = j \mid x_n = i).$$

There are at least two problems with this approach. It is first of all impractical. If we want to model sequences of even moderate length, say 10^3 base pairs, the model will have 12,000 parameters, because we will need to specify 1000 4×4 transition matrices, each of which has 12 free parameters. Second, the inhomogeneous model would not allow random variation in the location of *CpG* islands; their locations would be fixed around those sites n for which the transition probability $a_{GC}^{(n)}$ exceeds some suitably large threshold value.

In this section we introduce so-called *hidden Markov models* (HMM). They require many fewer parameters than general, inhomogeneous Markov chain models, yet can model randomly changing transition probabilities. They are popular and flexible models for many other biological sequence problems as well. Finally, they retain much of the simplicity of Markov models, and so are still relatively easy to compute with.

Let $\mathcal{A} := \{A_1, \dots, A_K\}$ be a finite set, which we refer to as an alphabet. For biological sequence analysis, \mathcal{A} is either the DNA alphabet $\{A, G, C, T\}$, or the protein alphabet, a set of 20 capital letters used as code letters for the 20 amino acids appearing in proteins. A hidden Markov model for a random sequence $\{X_1, X_2, X_3, \dots\}$ of letters from \mathcal{A} is constructed as follows. We imagine a finite set of states, labelled from 1 through N , each of which has a device that emits letters from \mathcal{A} . These emitters will be responsible for producing $\{X_n\}$, which is called the *observed sequence* or *observed process*. It is important for the flexibility of the model that different states emit the letters with different probabilities. Now we add a start state 0, and introduce an auxiliary, homogeneous Markov chain Z_0, Z_1, Z_2, \dots evolving in $S = \{0, 1, \dots, N\}$. We shall always assume $Z_0 = 0$, since 0 is the start state; in some models we may also want to add to S a stop state that does not

emit letters. A visit to the stop state terminates the chain. $\{Z_n\}$ is the *hidden chain*. In the model, it determines the sequence of states that emit the letters of the observed process. That is, state Z_1 emits letter X_1 , state Z_2 emits letter X_2 , and so on. To complete the definition of the model, it is necessary to specify the joint probabilities of sequences of emissions and paths of the hidden chain. The crucial assumption is that *given the sequence of visited states* the successive emissions are mutually independent. The precise mathematical meaning of this assumption is expressed in (8.1) in the formal definition.

Definition. A hidden Markov chain model is a pair of process $(\{Z_n\}, \{X_n\})$, where $\{Z_n\}$ is a Markov chain and $\{X_n\}$ is the sequence of letters in alphabet \mathcal{A} emitted by the states visited by $\{Z_n\}$, such that the joint emission and path probabilities are calculated as follows. Let A denote the transition probability matrix for $\{Z_n\}$. For each state k , $1 \leq k$, let

$$e_k(x) \triangleq \mathbb{P}(\text{state } k \text{ emits letter } x)$$

Then

$$\begin{aligned} & \mathbb{P}(X_1 = x_1, \dots, X_n = x_n; Z_1 = s_1, \dots, Z_n = s_n) \\ &= [\mathbb{P}(Z_1 = s_1) e_{s_1}(x_1)] [a_{s_1 s_2} e_{s_2}(x_2)] \dots [a_{s_{n-1} s_n} e_{s_n}(x_n)] \end{aligned} \quad (8.1)$$

Our first goal is to understand the formula (8.1) for the joint probabilities. To simplify notation, we shall adopt the notation

$$p(x_1 \cdots x_n; s_1 \cdots s_n) = \mathbb{P}(X_1 = x_1, \dots, X_n = x_n; Z_1 = s_1, \dots, Z_n = s_n).$$

Since it is assumed that $Z_0 = 0$, where 0 is the start state, $\mathbb{P}(Z_1 = s_1) = a_{0s_1}$. Thus (8.1) can be rewritten:

$$\begin{aligned} p(x_1 \cdots x_n; s_1 \cdots s_n) &= [a_{0s_1} e_{s_1}(x_1)] [a_{s_1 s_2} e_{s_2}(x_2)] \dots [a_{s_{n-1} s_n} e_{s_n}(x_n)] \\ &= \prod_{i=1}^n a_{s_{i-1} s_i} e_{s_i}(x_i) \end{aligned} \quad (8.2)$$

The last expression makes the formula easy to remember; the factor $a_{s_{i-1} s_i} e_{s_i}(x_i)$ is the probability for the chain to move from state s_{i-1} to s_i and then to emit letter x_i from s_i . The probability of the entire path is just the product of these successive transition-with-emission probabilities along

the path. Notice that if we took away the emission probabilities from the formula, we would be left just with the product $a_{0s_1} \cdots a_{s_{n-1}s_n}$, which is the probability of the path $Z_1 = s_1, \dots, Z_n = s_n$ of the hidden chain. Figure 1 illustrates the operation of the hidden Markov model schematically.

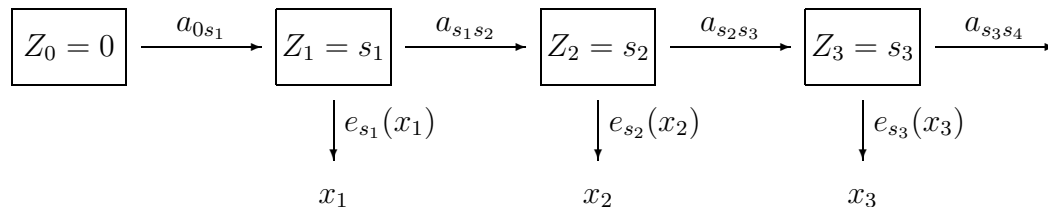


Figure 1.

Example. This example is adapted from the text, *Biological Sequence Analysis* by Durbin, Eddy, Krogh, and Mitchison, wherein it is called the occasionally dishonest casino. Imagine a betting game which depends on independent coin tosses. Let us call the coin tosser the casino. The casino has two coins. Coin 1 is fair and coin 2 is loaded with probability 0.3 of heads. At each play, the casino will change the coin being flipped with a small probability, say 0.01, but the player does not know what coin is being tossed. All the player sees is the sequence of heads and tails that result. Suppose that the casino chooses which coin to start with at random, with each coin having the same probability 0.5 of being picked.

This coin tossing process is a hidden Markov chain. The alphabet \mathcal{A} is $\{H, T\}$. The state space S contains three states; the start state 0, the state corresponding to tossing coin 1, which we call state 1, and the state corresponding to tossing coin 2, which we call state 2. The hidden Markov chain $\{Z_n\}$ evolving in S determines which coin, fair or foul, is tossed in each play. According to the given prescription, the transition probability matrix for $\{Z_n\}$ is

$$\begin{array}{c}
 \\
 0 \quad 1 \quad 2 \\
 0 \quad \left[\begin{array}{ccc} 0 & 0.5 & 0.5 \\ 0 & 0.9 & 0.1 \\ 0 & 0.1 & 0.9 \end{array} \right] \\
 1 \\
 2
 \end{array}$$

The emission probabilities are

$$e_1(H) = 0.5 \quad e_1(T) = 0.5$$

$$e_2(H) = 0.3 \quad e_2(T) = 0.7$$

As an example we compute

$$\begin{aligned} & p(TTTHHTTHTT; 2222221111) \\ &= [(.5)(.7)][(.9)(.7)]^2[(.9)(.3)]^2[(.9)(.7)][(.1)(.5)][(.9)(.5)]^3 \\ &= (.5)^5[(.9)^8][(.7)^4][(.3)^2](.1) \end{aligned}$$

Example 2. A hidden Markov model for CpG islands. In this example we shall build a HMM for DNA with CpG islands. The model we discuss here won't be the most general one, because, in the interests of simplicity, we shall try to keep the number of parameters in the model to the minimum necessary. We will assume that DNA strictly inside a CpG island can be modelled well by a homogeneous Markov chain, and that we have built such a model. Denote the transition probability matrix of this model by

$$B_1 \triangleq \begin{bmatrix} a_{AA} & a_{AC} & a_{AG} & a_{AT} \\ a_{CA} & a_{CC} & a_{CG} & a_{CT} \\ a_{GA} & a_{GC} & a_{GG} & a_{GT} \\ a_{TA} & a_{TC} & a_{TG} & a_{TT} \end{bmatrix} \quad (8.3)$$

Likewise, assume we have a homogeneous Markov model for DNA outside of CpG-islands, specified by a transition probability matrix

$$B_2 \triangleq \begin{bmatrix} b_{AA} & b_{AC} & b_{AG} & b_{AT} \\ b_{CA} & b_{CC} & b_{CG} & b_{CT} \\ b_{GA} & b_{GC} & b_{GG} & b_{GT} \\ b_{TA} & b_{TC} & b_{TG} & b_{TT} \end{bmatrix} \quad (8.4)$$

Numerical values for these matrices may be estimated by using the maximum likelihood estimates discussed in section 6 of the lecture on Markov chains. Using human DNA data, Durbin, Eddy, Krogh and Mitchinson calculate the following (which appear to be for DNA read in the 3' – 5' direction).

$$B_1 = \begin{bmatrix} .180 & .274 & .426 & .120 \\ .171 & .368 & .274 & .188 \\ .161 & .339 & .375 & .125 \\ .079 & .355 & .384 & .182 \end{bmatrix} \quad B_2 = \begin{bmatrix} .300 & .250 & .285 & .210 \\ .322 & .298 & .078 & .302 \\ .248 & .246 & .298 & .208 \\ .177 & .239 & .292 & .292 \end{bmatrix}$$

The object is to create a HMM that occasionally and randomly switches between the two different Markov chain models. To do this, we shall construct a state space S for the hidden chain consisting of nine states—the start state 0, four states $A1, C1, G1, T1$ and another four states $A2, C2, G2, T2$. (The start state is included only for consistency with the general framework developed above.) The states $A1$ and $A2$ emit only A ; that is $e_{A1}(A) = 1$ and $e_{A2}(A) = 1$. Likewise $C1$ and $C2$ emit only C , and similarly for the other states; emissions are not random in this model. However, whether it is, for example, $A1$ or $A2$ that emits A is not observed. The hidden chain, will determine whether sites are emitted in a CpG island or not. CpG islands correspond to sequences of sites for which the hidden chain is moving about in the set of states $\{A1, C1, G1, T1\}$; the site is not in a CpG island if the hidden chain is instead in the set $\{A2, C2, G2, T2\}$. In short, the states $\{A1, C1, G1, T1\}$ emit letters in CpG islands, while the states $\{A2, C2, G2, T2\}$ emit letters outside of the islands. We want the chain to spend relatively long stretches in either set of states, with transitions according to the respective matrices $B1$ and $B2$, with occasional transitions from one set to the other. Specifically, assume there are two small parameters δ and τ such that

- (i) If Z_i is in $\{A1, C1, G1, T1\}$ (site i is in a CpG island), then with probability δ , the island will end at site i , and Z_{i+1} take on one of the values $\{A2, C2, G2, T2\}$, each state being equally likely. With probability $1 - \delta$, the island does not end and the move from Z_i to Z_{i+1} takes place according to the transition probability matrix B_1 .
- (ii) If Z_i is in $\{A2, C2, G2, T2\}$ (site i is not in a CpG island), then with probability τ , an island will begin at site $i + 1$, and Z_{i+1} take on one of the values $\{A1, C1, G1, T1\}$, each state being equally likely. With probability $1 - \tau$, an island does not begin and the move from Z_i to Z_{i+1} takes place according to the transition probability matrix B_2 .

These assumptions translate into the transition probabilities for the hidden chain. For example, according to assumption (i),

$$\mathbb{P}(Z_{i+1} = A_2 \mid Z_i = A_1) = \delta \frac{1}{4} \quad \mathbb{P}(Z_{i+1} = A_1 \mid Z_i = A_1) = (1 - \delta)a_{AA} \quad (8.5)$$

By following the pattern in these equalities, can write the full state transition matrix. Arrange the states in the order, $A1, C1, G1, T1, A2, C2, G2, T2$. Let

8.2. GRAPHICAL REPRESENTATION OF HIDDEN CHAIN PATHS 7

J denote the 4 matrix whose every entry is 1. (The start state 0 is ignored in this matrix—to complete the model, one should also specify the transition probabilities from 0 to the other eight states.) The state transition matrix is:

	A1	C1	G1	T1	A2	G2	C2	T2
A1	$(1-\delta)a_{AA}$	$(1-\delta)a_{AC}$	$(1-\delta)a_{AG}$	$(1-\delta)a_{AT}$	$\delta/4$	$\delta/4$	$\delta/4$	$\delta/4$
C1	$(1-\delta)a_{CA}$	$(1-\delta)a_{CC}$	$(1-\delta)a_{CG}$	$(1-\delta)a_{CT}$	$\delta/4$	$\delta/4$	$\delta/4$	$\delta/4$
G1	$(1-\delta)a_{GA}$	$(1-\delta)a_{GC}$	$(1-\delta)a_{GG}$	$(1-\delta)a_{GT}$	$\delta/4$	$\delta/4$	$\delta/4$	$\delta/4$
T1	$(1-\delta)a_{TA}$	$(1-\delta)a_{TC}$	$(1-\delta)a_{TG}$	$(1-\delta)a_{TT}$	$\delta/4$	$\delta/4$	$\delta/4$	$\delta/4$
A2	$\tau/4$	$\tau/4$	$\tau/4$	$\tau/4$	$(1-\tau)b_{AA}$	$(1-\tau)b_{AC}$	$(1-\tau)b_{AG}$	$(1-\tau)b_{AT}$
C2	$\tau/4$	$\tau/4$	$\tau/4$	$\tau/4$	$(1-\tau)b_{CA}$	$(1-\tau)b_{CC}$	$(1-\tau)b_{CG}$	$(1-\tau)b_{CT}$
G2	$\tau/4$	$\tau/4$	$\tau/4$	$\tau/4$	$(1-\tau)b_{GA}$	$(1-\tau)b_{GC}$	$(1-\tau)b_{GG}$	$(1-\tau)b_{GT}$
T2	$\tau/4$	$\tau/4$	$\tau/4$	$\tau/4$	$(1-\tau)b_{TA}$	$(1-\tau)b_{TC}$	$(1-\tau)b_{TG}$	$(1-\tau)b_{TT}$

This completes the description of the CpG island hidden chain. In practice, it would be necessary to estimate δ and τ . It is not hard to see that with this model, the length of a CpG island has a geometric distribution with parameter $p = \delta$. Indeed, suppose we start a CpG island. At each site i imagine flipping a coin and ending the island at i if the coin is heads. If the probability of heads is δ , this will correspond exactly to the model we have given. The length of the CpG island is then precisely the number of trials until heads appears. Thus, the average length of a CpG island will be the expected value of a geometric random variable with parameter $p = \delta$, and this expected value is $1/\delta$. One could then estimate δ by setting $1/\delta$ to the average length of CpG islands in the data set. Similarly we can estimate τ using the average length of segments between CpG islands in the data.

8.2 Graphical representation of hidden chain paths

It will be helpful to have another graphical representation for *emission-path sequences* $\{X_1 = x_1, \dots, X_n = x_n; Z_1 = s_1, \dots, Z_n = s_n\}$. Our shorthand notation for such a sequence is $\{x_1x_2 \dots x_n; s_1s_2 \dots s_n\}$. We will assume the sequence of emitted letters is fixed x_1, x_2, \dots, x_n , and give a graphical representation for paths $s_1 \dots s_n$ in the hidden chain generating emitting this sequence. The idea is to use a lattice, as in the next figure. It has a row for

each state of the hidden chain, and it has a start column, labelled “o” and then a column for each letter x_i . An example is drawn in Figure 2 below the observed sequence $(x_1, \dots, x_7) = (C, C, G, T, A, A, C)$ from the DNA alphabet and for the state space $S = \{0, 1, 2, 3, 4, 5\}$. The vertex corresponding to state j and to the i^{th} letter in the sequence will be labelled by the pair (j, i) . As part of this convention, the first row is labelled by 0 and also the first column is labelled by 0. Thus the vertex in the upper left corner of the lattice bears the label $(0, 0)$.

We shall associate to each path in the hidden chain a path of directed edges in the lattice graph. Figure 2 illustrates with an example. We want to represent the event that the path 2413352 in the hidden chain emits the sequence $CCGT AAC$. The starting position for the path is the $(0, 0)$ vertex corresponding to the start state before any letters are emitted. In the first step, the hidden chain moves from the start state to state 2 ($Z_1 = 2$) and emits a C . This is represented by an arrow from vertex $(0, 0)$ to vertex $(1, 2)$, the vertex in row 2 (corresponding to state 2) and column 1 (corresponding to the first emission). Next, the hidden process moves to state $Z_2 = 4$ and emits another C . This transition with emission is represented by an arrow from vertex $(1, 2)$ to vertex $(2, 4)$. We continue in this way moving from column to column, so that, the vertices of the path show in each column the state that emits that column’s letter.

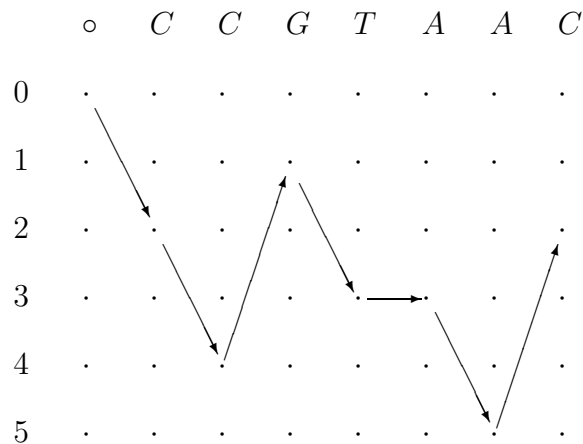


Figure 2.

Let us make some formal definitions that express the path representations idea here. All the edges in Figure 3 connect points between adjacent columns only. Thus, define an *admissible edge* to be any directed edge in the graph that points from a vertex (j, i) in one column, to a vertex $(k, i + 1)$ in the next column. Let us define a path through the lattice to be an *admissible path* if it starts at vertex $(0, 0)$ and is composed of admissible edges. In summary, admissible paths start at $(0, 0)$ and move successively through the columns of the lattice. Every combined path through the hidden chain emitting sequence $x_1 \dots x_n$ may be represented in the manner of the example by an admissible path, and, conversely, every admissible path that does not return to the zero row represents a possible path through the hidden chain.

Representing path-emission sequences using lattices is very handy for visualizing various computational algorithms that we will develop for hidden Markov models. These algorithms are based on the device of scoring each admissible directed edge with the probability of the event that edge represents. Consider the directed edge from vertex $(j, i - 1)$ to (k, i) and suppose that column i corresponds to the emission of letter x_i . Then the edge represents a transition from state j to state k in the hidden chain and an emission of letter x_i from state k ; the probability of this happening is $a_{jk}e_k(x_i)$, and hence we use this value as the score of the edge.

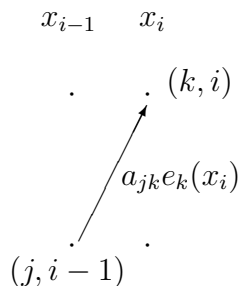


Figure 3.

Now define the *score of an admissible path* through the lattice as the product of the scores of the directed edges in the path. For example, the score of the path shown in Figure 2 is

$$a_{02}e_2(C)a_{24}e_4(C)a_{41}e_1(G)a_{13}e_3(T)a_{33}e_3(A)a_{35}e_5(A)a_{52}e_2(C).$$

By formula (8.2) this score is just the probability $p(CCGTAAC; 2413353)$ of the emission-path sequence $\{CCGTAAC; 2413353\}$. This is true in general;

the score of path will equal the probability of the emission-path sequence it represents.

(If the emission-path sequence is even moderately long, its probability will in general be very small. For this reason it is preferable numerically to work with logarithms of probabilities, that is, with $\ln p(x_1 \dots x_n; s_1 \dots s_n)$. In this notice that from (8.2),

$$\ln p(x_1 \dots x_n; s_1 \dots s_n) = \sum_{k=1}^n \ln [a_{s_{i-1}s_i} e_{s_i}(x_i)].$$

Therefore when working with logarithms one should score the admissible edge in Figure 3 by $\ln[a_{jk}e_k(x_i)]$, and define the score of an admissible path in the lattice to be the *sum* of the scores of its edges.)

8.3 The forward algorithm

The definition of a hidden Markov model specifies how to compute an emission-path probability. But since we only observe emissions, we want to compute joint probabilities of the emission process alone. That is, we want to calculate $p(x_1 \dots x_n) \triangleq \mathbb{P}(X_1 = x_1, \dots, X_n = x_n)$. In theory this is easily enough done; we merely list every possible path in the hidden chain $s_1 \dots s_n$ that can give rise to $x_1 \dots x_n$, calculate $p(x_1 \dots x_n; s_1 \dots s_n)$ for each such path and sum these probabilities over all possible paths:

$$p(x_1 \dots x_n) = \sum_{s_1 \dots s_n} p(x_1 \dots x_n; s_1 \dots s_n) \quad (8.6)$$

(Here the summation is taken over all hidden chain sequences $\{s_1 \dots s_n\}$.) However, this is not the approach one would want to take numerically. Even for moderate n , the number of sequences one would have to list and compute is forbiddingly large. Instead, we develop a computationally efficient algorithm called the forward algorithm.

The forward algorithm will compute the probability of a sequence of emissions $x_1 \dots x_n$ by recursively computing an auxiliary function indexed by the vertices in our lattice graph. Consider the sequence of letters $x_1 \dots x_n$ to be given and fixed. Assume that the state space of the underlying hidden chain is $S = \{0, 1, \dots, N\}$. For each state k in S and each letter x_i , where $1 \leq i \leq N$, define

$$f_k(i) = \mathbb{P}(X_1 = x_1, \dots, X_i = x_i; Z_i = k). \quad (8.7)$$

In words, $f_k(i)$ is the probability that $x_1 \dots x_i$ are the first i letters emitted and that, at the same time, the hidden chain is at state k at the time of the i^{th} emission, that is, that $Z_i = k$. By convention, we set $f_0(0) = 1$, but $f_k(0) = 0$ for all states $k \geq 1$. The reason for this will be clear in a moment.

The forward algorithm will recursively compute $f_k(i)$ for all values of k in S and for each i , $1 \leq n$. This will solve the problem. Because the hidden chain must be one of the states in S for the final emission of x_n ,

$$p(x_1 \dots x_n) = \sum_{k=1}^N \mathbb{P}(X_1 = x_1, \dots, X_n = x_n; Z_n = k) = \sum_{k=1}^N f_k(n). \quad (8.8)$$

Before stating the algorithm, we point out that the event

$$\{X_1 = x_1, \dots, X_i = x_i; Z_i = k\}$$

is the union of all emission-path sequences, $\{x_1 \dots x_i; s_1 \dots s_i\}$ such that $s_i = k$. Therefore, in the lattice graph representation of emission-path sequence, $\{X_1 = x_1, \dots, X_i = x_i; Z_i = k\}$ corresponds to the set of all admissible paths from vertex $(0, 0)$ to vertex (i, k) . It follows that $f_k(i)$ is the *sum of the scores of all admissible paths from $(0, 0)$ to (i, k)* . Figure 4 illustrates the set of admissible paths corresponding to $f_3(2)$.

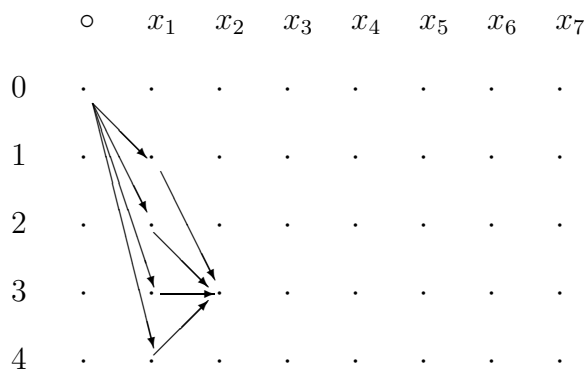


Figure 4.

The advantage of defining $f_k(i)$ is that it can be recursively and efficiently computed.

Theorem 1 *The values $f_k(i)$, $0 \leq k \leq N$, $0 \leq i \leq n$ satisfy the equations:*

$$f_0(0) = 1 \text{ and for } 1 \leq k \leq N, \quad f_k(0) = 0; \quad (8.9)$$

$$\text{for } 1 \leq i \leq n \quad f_k(i) = \sum_{j=0}^N f_j(i-1) a_{jk} e_k(x_i). \quad (8.10)$$

When $i \geq 2$ the first term in this sum is zero and we can write

$$f_k(i) = \sum_{j=1}^N f_j(i-1) a_{jk} e_k(x_i). \quad (8.11)$$

Equations (8.9) and (8.10) constitute the forward equations for $\{f_k(i)\}$. To visualize how the equations work it is useful to imagine the algorithm entering each value $f_k(i)$ at vertex (i, k) in the path representation lattice, because, as we saw, $f_k(i)$ is the total score of all admissible paths ending at (i, k) . Equation (8.9) initializes the algorithm by specifying the values of $f_k(0)$ to enter in the first column (with index $i = 0$) of the lattice. Equation (8.10) says that to compute $f_k(i)$, look at all the vertices in the lattice graph from which one can reach vertex (i, k) by an admissible edge; these are precisely the vertices in the previous column, the column with index $i-1$. Take the value of $f_j(i-1)$ at each of these vertices, multiply it by the score of the move from $(i-1, j)$ to (i, k) and sum. Figure 5 illustrates (for $N = 4$).

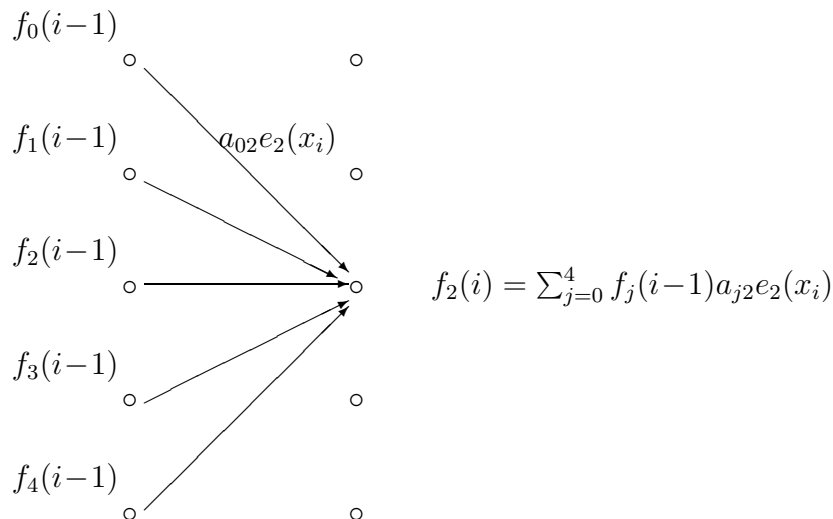


Figure 5.

From this picture we see that we can compute the values of $f_k(i)$ column by column, moving from left to right in the graph. Consider for example, computing the entries in column 1 (that is, the column with index 1). We know by (8.9) that in column 0, the entry at vertex $(0, 0)$ is 1, and all the entries below it are 0. By (8.10) with $i = 1$,

$$f_k(1) = \sum_{j=0}^N f_j(0)a_{jk}e_k(x_1) = a_{0k}e_k(x_1), \quad 0 \leq k \leq N. \quad (8.12)$$

(Since $a_{00} = 0$, because 0 is a stop state, we find of course that $f_0(1) = 0$.) Enter these values down column $i = 1$ of the lattice. Now we can continue compute the values in column 1 to compute those in column 2 and so on. At the end we get the values $f_k(n)$, $0 \leq k \leq n$, and obtain the final answer using (8.8). Notice that in all columns with index 1 or greater, there is no admissible edge that leads to vertex $(i, 0)$ since this vertex corresponds the the start state. Hence $f_0(i) = 0$ for all $i \geq 1$, as we stated in Theorem 1. Because of this we can suppress the arrow from $(i-1, 0)$ to $(i, 2)$ in Figure 5, when $i-1 \geq 1$.

Derivation of the forward equations This is not too hard if we use the lattice visualization. First we can check directly that the equations give the

correct answer for the column with index $i = 1$ corresponding to emission of x_1 . Indeed, there is only one admissible path to vertex $(k, 1)$, namely the path consisting of one directed edge from $(0, 0)$ to $(k, 1)$, because this edge represent the event that $X_1 = x_1$ and $Z_1 = k$. We know the probability of this event is $a_{0k}e_k(x_1)$ and this is precisely what we derived in (8.12) from the forward equation.

Now suppose that $i - 1 \geq 1$ and consider $f_j(i - 1)$. This is the total probability of all admissible paths that end at vertex $(i - 1, j)$:

$$f_j(i - 1) = \sum_{s_1, \dots, s_{i-2}} p(x_1 \dots x_{i-1}; s_1 \dots s_{i-2}j)$$

Thus

$$f_j(i - 1)a_{jk}e_k(x_i) = \sum_{s_1, \dots, s_{i-2}} p(x_1 \dots x_{i-1}; s_1 \dots s_{i-2}j)a_{jk}e_k(x_i) \quad (8.13)$$

But $p(x_1 \dots x_{i-1}; s_1 \dots s_{i-2}j)a_{jk}e_k(x_i)$ is the product of the score of an admissible path to vertex $(i - 1, j)$ times the score of the edge from $(i - 1, j)$ to k ; thus it is the score of an admissible path that reaches (i, k) by passing through $(i - 1, j)$. It follows from (8.13) that $f_j(i - 1)a_{jk}e_k(x_i)$ is the sum of probabilities of all admissible paths that reach (i, k) by passing through $(i - 1, j)$. Since any admissible path to (i, k) must pass through a vertex in column $i - 1$,

$$f_k(i) = \sum_{j=0}^N f_j(i - 1)a_{jk}e_k(x_i),$$

as we claim in (8.10).

8.4 Maximum probability paths

Here is another calculational issue that arises with hidden Markov models. Suppose we observe the sequence of emitted letters $x_1 \dots x_n$ from a hidden Markov model. In some applications, we would like to estimate the path in the hidden chain that gave rise to the observed sequence. How should this be done? One idea is to choose as an estimate that path $s_1^* \dots s_n^*$ which makes the observed sequence $x_1 \dots x_n$ most likely. This means that we want to find $s_1^* \dots s_n^*$ to maximize the probability of the emission-path sequence $p(x_1 \dots x_n; s_1 \dots s_n)$ over all possible paths $s_1 \dots s_n$. This could

be done by simply listing all the paths in the hidden chain, calculating $p(x_1 \dots x_n; s_1 \dots s_n)$ for each of them and identifying the largest one. However, such a procedure is very inefficient and the computational effort it requires grows exponentially with n . In this section, we will describe the Viterbi algorithm for solving this maximization problem with a procedure of polynomial complexity.

The idea is to introduce an auxiliary function, called the *value function*, which can be computed recursively. For each state k and each x_i , let $v_k(i)$ be the maximum probability of the emission-path sequence $\{x_1 \dots x_i; s_1 \dots s_i\}$ such that the last state of the hidden chain is $s_i = k$. In lattice-path language, it is the maximum score over all admissible paths to the vertex (i, k) . In symbols,

$$v_k(i) = \max\{p(x_1 \dots x_i; s_1 \dots s_{i-1}k); s_1, s_2, \dots, s_{i-1}\} \quad (8.14)$$

(In our algorithm, we will not need to worry about the values in column 0; likewise, along row 0 of the lattice $v_0(i) = 0$ for $i \geq 1$, and we will not concern ourselves further with row 0.)

It is easy to compute the values $v_k(1)$, $k \in S$ in the the column indexed by x_1 . In this case, $v_k(1) = p(x_1; k)$, which is the probability that the hidden chain moves from the start state to state k and emits letter x_1 . Thus,

$$v_k(1) = a_{0k}e_k(x_1) \quad 1 \leq k \leq N \quad (8.15)$$

Now suppose $i \geq 2$. Consider taking $v_j(i-1)$ and multiplying it by the score of the directed edge from $(i-1, j)$ to (i, k) :

$$v_j(i-1)a_{jk}e_k(x_i) = \max\{p(x_1 \dots x_{i-1}; s_1 \dots s_{i-2}j)a_{jk}e_k(x_i); s_1, s_2, \dots, s_{i-2}\} \quad (8.16)$$

But $p(x_1 \dots x_{i-1}; s_1 \dots s_{i-1}j)a_{jk}e_k(x_i)$ is the probability $p(x_1 \dots x_i; s_1 \dots s_{i-2}jk)$, which is the score of an admissible path to (i, k) which passes through vertex $(i-1, j)$. Hence $v_j(i-1)a_{jk}e_k(x_i)$ is the maximum score over all admissible paths to the vertex (i, k) passing through vertex $(i-1, j)$. Since each path to (i, k) must pass through a vertex of the preceding column, we see that

$$v_k(i) = \max\{v_j(i-1)a_{jk}e_k(x_i); 1 \leq j \leq N\}. \quad (8.17)$$

Notice that this equation may be written

$$v_k(i) = e_k(x_i) \max\{v_j(i-1)a_{jk}; 1 \leq j \leq N\}. \quad (8.18)$$

This equation, which is an example of a *dynamic programming* equation, allows the calculation of the values of $v_k(i)$ column by column, starting with the values in column 1, which were specified in equation (8.15). We have derived the following.

Theorem 2 $\{v_k(i)\}$ is the unique solution of the equations (8.17) and (8.15).

There is one more step—determining the optimal path from the calculation of $\{v_k(i)\}$. This is done as follows. When calculating $v_k(i)$ using (8.17) record the value j^* such that $v_k(i) = v_{j^*}(i-1)a_{j^*k}e_k(x_i)$; that is j^* is the index of the term on the right hand side of (8.17) that achieves the maximum. If there is more than one j at which the maximum is achieved, choose any one of them. Now draw an arrow back from vertex (i, k) to $(i-1, j^*)$; this is called a trace back. Do this for all vertices as $\{v_k(i)\}$ is computed. In column 1, the traceback from every vertex points back to $(0, 0)$, which represents starting in the start state. After the values of $\{v_k(i)\}$ are found for all vertices, find the vertex in the last column with the highest value. This is the largest emission-path probability that can be achieved for the sequence of letters $x_1 \dots x_n$. Follow the trace backs from this vertex back to vertex $(0, 0)$. This path, in the forward direction, gives the maximum probability path.

Remark. For reasonably long sequences $x_1 \dots x_n$, the path probabilities rapidly become very small. To correct for this problem numerically, it is often preferable in practice to compute $g_k(i) \triangleq \ln v_k(i)$ instead. (Here, we assign $\ln 0$ the value $-\infty$). The dynamic programming equation (8.17) translates to

$$g_k(i) = \max \{g_j(i-1) + \ln a_{jk} + \ln e_k(x_i); 0 \leq j \leq N\}. \quad (8.19)$$

Example. Let the hidden chain have state space $S = \{0, 1, 2, 3\}$ with state transition matrix

$$A \triangleq \begin{bmatrix} 0 & .2 & .3 & .5 \\ 0 & .3 & .3 & .4 \\ 0 & .4 & .4 & .2 \\ 0 & .1 & .6 & .3 \end{bmatrix}$$

Assume that the states output one of three letters $\{A, B, C\}$ with probabilities,

$$\begin{bmatrix} e_1(A) & e_1(B) & e_1(C) \\ e_2(A) & e_2(B) & e_2(C) \\ e_3(A) & e_3(B) & e_3(C) \end{bmatrix} = \begin{bmatrix} .25 & .35 & .4 \\ .1 & .25 & .65 \\ .5 & .45 & .05 \end{bmatrix}.$$

Find the path $s_1s_2s_3$ in S maximizing the probability of $p(ABC; s_1s_2s_3)$.

We shall apply the equations, (8.17) and (8.15), to compute $\{v_k(i); 0 \leq k \leq 3, 0 \leq i \leq 3\}$ while drawing tracebacks in the lattice graph. The tracebacks are all entered in Figure 6 below.

The values of $v_k(1)$ are specified by the initialization condition (8.5): $v_k(1) = a_{0k}e_k(x_1)$. Thus

$$v_1(1) = (.2)(.25) = .05, \quad v_2(1) = (.3)(.1) = .03, \quad v_3(1) = (.5)(.5) = .25.$$

Next, we calculate the values of $v_k(2)$, using (8.18); in each calculation, the value in the argument of $\max\{\dots\}$ that gives the maximum is in bold face and the traceback in Figure 6 is drawn to the vertex in column 1 achieving the maximum:

$$\begin{aligned} v_1(2) &= e_1(B) \max \{v_j(1)a_{j1}; 1 \leq j \leq 3\} \\ &= (.35) \max\{(.05)(.3), (.03)(.4), (.25)(.1)\} = .00875 \\ v_2(2) &= e_2(B) \max \{v_j(1)a_{j2}; 1 \leq j \leq 3\} \\ &= (.25) \max\{(.05)(.3), (.03)(.4), (.25)(.6)\} = .0375 \\ v_3(2) &= e_3(B) \max \{v_j(1)a_{j3}; 1 \leq j \leq 3\} \\ &= (.45) \max\{(.05)(.4), (.03)(.2), (.25)(.3)\} = .03375 \end{aligned}$$

Finally, we apply the algorithm to column 3:

$$\begin{aligned} v_1(3) &= e_1(C) \max \{v_j(2)a_{j1}; 1 \leq j \leq 3\} \\ &= (.4) \max\{(.00875)(.3), (.0375)(.4), (.03375)(.1)\} = .006 \\ v_2(3) &= e_2(C) \max \{v_j(2)a_{j2}; 1 \leq j \leq 3\} \\ &= (.65) \max\{(.00875)(.3), (.0375)(.4), (.03375)(.6)\} = .013625 \\ v_3(3) &= e_3(C) \max \{v_j(2)a_{j3}; 1 \leq j \leq 3\} \\ &= (.05) \max\{(.00875)(.4), (.0375)(.2), (.03375)(.3)\} = .0050625 \end{aligned}$$

The tracebacks are shown in Figure 6. We have also indicated the values of $v_k(3)$, $1 \leq k \leq 3$ in the last column. Note that the highest value of $v_k(3)$ is achieved at $k = 2$. The path of tracebacks from the vertex $(3, 2)$ to $(0, 0)$ is indicated by bold face arrows. Tracing this path forward shows that the maximum probability path is $z_1 = 3, z_2 = 3, z_3 = 2$.

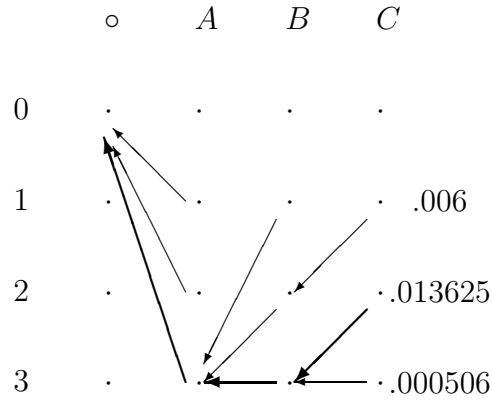


Figure 6.

8.5 Profile HMM for protein families

In current technical lingo, two DNA sequences sharing a common ancestor are called *homologous*. Likewise two proteins or protein domains determined by genes with a common ancestor are called homologous. Through the course of time, a given DNA segment, S , in the genome gives rise through reproduction to many progeny segments in different individuals. After a long enough time the descendants of S may even be distributed across species or appear in different parts of the genome. The collection of the descendants of S forms a class of homologous DNA segments.

Consider the collection of all descendants of S after a time T has elapsed. Now DNA and the proteins coded for in genes evolve by random mutation and selection. Thus mutations to S occur at a slow rate down the lines of descent. These mutations can be substitutions of one amino acid for another, deletions of existing amino acids, or insertions of new amino acids between existing ones. Let us illustrate with some examples. Suppose that $WHKKKCP$ is the amino acid sequence in the ancestral protein. Imagine we can follow one branch of descendants down to the present time, and along that branch no insertions or deletions occur, but H mutates to R and the third K to N . We present the two sequence aligned so that each amino acid of the descendant

matches the amino acid of the ancestor from which it derives:

$$WHKKKCP \quad (8.20)$$

$$WRKKNCP \quad (8.21)$$

Let us next suppose that we follow a different line of descent in which H is deleted, a W gets inserted between C and P and W suffers a substitution by F . Again, we align the descendant with the ancestor, matching amino acids to show the evolutionary connection:

$$WHKKKC - P \quad (8.22)$$

$$F - KKKCWP \quad (8.23)$$

This time gaps are inserted to indicate where insertion or deletion events occur, because there is no amino acid that originates from H in the descendant, nor an amino acid in the ancestor from which W in the descendant originates. From the alignments (8.21) and (8.23), we can align the two descendants to show their relationship to each other as homologous protein sequences:

$$WRKKNC - P \quad (8.24)$$

$$F - KKKCWP \quad (8.25)$$

This alignment matches amino acids deriving from common amino acids in the ancestral sequence, and otherwise inserts gaps. Note that if we were just handed this alignment, we could not tell for example whether the gap in the lower sequence resulted from an insertion or deletion. For this reason, insertions and deletions are often referred to together as *indels*.

In practice, given sequences from a homologous family, we know neither the ancestral sequence, nor the history of mutations giving rise to the family. The alignment problem is the problem of reconstructing the alignments of the family members due to their common evolutionary origin. One way to approach alignment and to understand the variation of sequences in a homologous family is to construct a probability model for the family. A very popular model is the profile HMM, which we explain in the next section.

8.5.1 Profile HMMs

The profile HMM is a hidden Markov model with one new twist. Some of the states in the model, other than the start state, will be silent. The existence of

silent states will require modifications to the forward and Viterbi algorithms. In the profile HMM, there will be three sorts of states: match states, insertion states, and deletion states. Their names describe what they do. Let $b_1 b_2 \dots b_\ell$ denote the ancestral sequence of the family we are trying to model. The match state m_j will be responsible for emitting the amino acid in the descendant that substitutes for b_j in the ancestor. (If m_j emits b_j itself, then no substitution has been made.) The insertion state i_j emits amino acids that are inserted between amino acids b_i and b_{i+1} of the ancestor. The delete state d_i is silent and is responsible for causing the deletion of amino acid i ; the hidden chain will be so constructed that a visit to d_i precludes a visit to m_i . Finally, we add a start state, labelled m_0 and a stop state, labelled $m_{\ell+1}$. Figure 6 shows a state diagram of a profile HMM for $\ell = 3$; only transitions with non-zero probability are drawn.

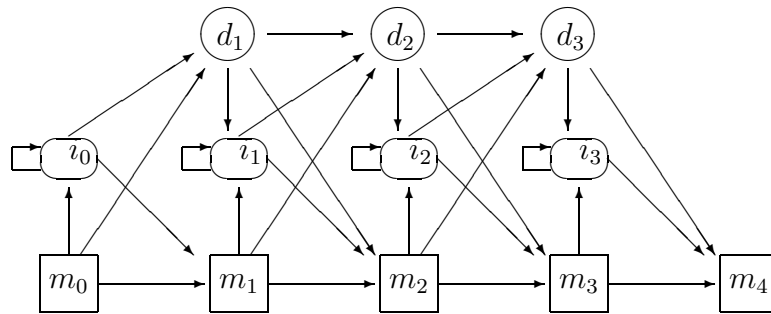


Figure 6.

Since the model of Figure 6 is defined as an example only, with no actual application in mind, we shall not try to set particular transition and emission probabilities. Typically these would be estimated from data assembled from a set of sequences already identified or surmised to be in the homologous family, subject to constraints on the parameters in the interests of simplicity. Nevertheless, as an exercise in understanding the model, it is worthwhile to consider some likely characteristics of these probabilities. There are only 3 emitting match states in Figure 6, which means that the ancestral protein sequence has length 3. Imagine that this sequence is AKW . The match state m_1 emits the amino acid, if any, in the descendant sequence corresponding to A , m_2 plays the same role for K , and m_3 for W . In an actual model the state m_1 would likely emit A (no substitution) with overwhelming high

probability and emit other letters with lesser probability; likewise m_2 and m_3 would be apt to emit, respectively, K and W . The states i_k are responsible for producing inserted amino acids. It is reasonable to suppose that the emission probabilities are the same for all insertion states, and to assume that the insertion states emit letters according to their overall background frequency in proteins. That is, if p_A is the frequency of amino acid A in proteins, use p_A as the probability that an insertion state emits A . We may also suppose that the transition probabilities between match states and to match states are relatively higher than other transition probabilities; otherwise, paths in the hidden chain that don't hit many match states will be relatively more common, and the sequences they emit will not look very much at all like the ancestral sequence.

Now imagine some emission-path sequences in the chain of Figure 6, assuming ancestral sequence AKW . Suppose for example that AGR is emitted by the path $m_0m_1m_2d_3i_3m_4$. We trace out in detail what this means. The hidden chain starts out, as always, in the start state m_0 . It moves from there to the match state m_1 from where it emits the A ; from m_1 it moves to m_2 , from where it emits G ; and it then moves to d_3 —the third amino acid in the ancestor sequence AKW suffers a deletion. Finally, an insertion occurs after the third site; i_3 emits the state R .

Consider now the emission-path sequence: $APHKW; m_0m_1i_1i_1m_2m_3m_4$. For this case, m_1 emits A , then hidden chain moves to i_1 , and stays there at the next step; this corresponds to an insertion of two amino acids between the first and second amino acids of the ancestral sequence, P and H being the inserted amino acids. Then the chain moves to m_2 and next m_3 from which it emits K and W (corresponding to no substitutions).

Finally consider $KWR; m_0d_1m_2m_3i_3m_4$. The original A has been deleted, but then the hidden chain moves to m_2 and m_3 from which it emits K and then W ; finally R is inserted.

We can use the joint emission-paths of these three examples to align the emitted sequences, without direct reference to the ancestral sequence. The alignment is given below; the principle is that all states produced by a given match state are aligned, with a minimal number of gaps inserted to compensate for insertions and deletions.

$$\begin{array}{c} \mathbf{A} \text{ -- } \mathbf{-G} \text{ -- } \mathbf{R} \\ \mathbf{A} \mathbf{P} \mathbf{H} \mathbf{K} \mathbf{W} \text{ --} \\ \text{-- --} \mathbf{-K} \mathbf{W} \mathbf{R} \end{array}$$

In representing this alignment, I use boldface letters to distinguish those columns deriving from a common match state. Although the final R of the first and third sequence are aligned, they do not come from a common match state. Unambiguous alignments may not always be possible. Consider, as an example, a fourth sequence, $AGFK; m_0m_1i_1m_3m_4$. Staying consistent to the principle of aligning letters from a common match state with minimal gaps, this sequence can be aligned with the others according either to

$$\begin{array}{ccc}
 \mathbf{A} - - \mathbf{G} - R & & \mathbf{A} - - \mathbf{G} - R \\
 \mathbf{A} P H \mathbf{K} \mathbf{W} - & \text{or} & \mathbf{A} P H \mathbf{K} \mathbf{W} - \\
 - - - \mathbf{K} \mathbf{W} R & & - - - \mathbf{K} \mathbf{W} R \\
 \mathbf{A} G - \mathbf{F} \mathbf{K} - & & \mathbf{A} - G \mathbf{F} \mathbf{K} -
 \end{array}$$

The problem is that the second sequence opened up a gap of size 2 between the letters emitted by match states m_1 and m_2 ; the insertion of the fourth sequence can then be aligned with either of the inserted letters of sequence 2. One must introduce further protocols to handle such cases, but at least the hidden Markov chain model gives a narrow set of choices.

We have seen so far that the profile HMM not only is a sensible probabilistic model for a homologous family, but also that it provides information on alignment. It can be used in practice to help solve the *multiple alignment problem*: given a collection of proteins that are presumed homologous, align them to reflect their evolutionary similarities. Assume that a profile HMM for the family from which the proteins are supposed to come has been built. This means that a model length has been selected, and values for the transition and emission probabilities have been set. For each sequence, find the maximum probability path through the states that generates it. Now use these estimated paths to align the proteins in the manner discussed in the previous paragraph. In this procedure, we see a concrete use for the Viterbi algorithm for maximum probability paths. However, the Viterbi algorithm we saw in the previous section was developed for hidden Markov chains all of whose states were emitting. It is an interesting problem to extend the algorithm to profile hidden Markov models.

8.5.2 Viterbi algorithm for profile HMM

Before presenting the algorithm, it is important to emphasize the important feature of profile HMMs that allow it to work: the chain can only visit each silent state once, and must visit them in the order d_1, d_2, d_3, \dots . This may be easily verified by studying Figure 6.

The Viterbi algorithm for profile HMM requires a more careful definition of the value function $v_k(i)$, when k represents one of the silent states. It is simplest to express the definition in terms of paths in the lattice graph for the chain. In displaying the lattice graph of a profile HMM, the final rows shall be used for the silent states, and it is helpful to list the silent states in the order in which they are visited. Figure 7 illustrates for the chain of Figure 6. The silent states are isolated at the bottom because the admissible directed edges into a vertex corresponding to a silent state will be different than for emitting states.

The basic idea is that an edge can point into a new column if and only if an emission of the letter indexing that column takes place. Thus, if row k corresponds to an emitting state, the admissible edges **pointing to** (i, k) and their interpretations and scores, **shall remain as previously defined in section 4.1**: the only admissible edges into (i, k) are those from vertices in the previous column, as in Figure 3, and the an edge from $(i-1, j)$ to (i, k) means a transition from state j to state k and emission of letter x_i from state k .

However, if index k corresponds to a silent state, an edge **pointing to** (i, k) is admissible, if and only if it originates from a vertex (i, j) in the same column, and there is an arrow from state j to k in the state transition diagram—that is, the hidden chain can move directly from j to k . As the edge originates from column i , x_i has been emitted already, and the fact that it points to a hidden state in the same column signifies that the hidden chain moves to this silent state. The probability of this transition is a_{ij} , which value is then used as the score of the edge a_{ij} . For example, in Figure 6, the admissible edges pointing to the vertex $(d_2, 2)$ corresponding to the silent state d_2 in column 2, are $(m_1, 2)$, $(i_1, 2)$ and (d_1, m_2) . These edges will all be directed downward. In fact, all admissible edges to silent states in Figure 6 will point downward, because the silent states are arranged in order at the bottom of the lattice.

Figure 7 on the next page illustrates the path $m_0 d_1 d_2 i_2 m_3 i_3 i_3 m_4$ emit-

ting $x_1x_2x_3x_4x_5$. The probability of this emission-path combinations is

$$a_{m_0d_1} a_{d_1d_2} [a_{d_2i_2} e_{i_2}(x_1)] [a_{i_2m_3} e_{m_3}(x_2)] [a_{m_3i_3} e_{i_3}(x_3)] [a_{i_3i_3} e_{i_3}(x_4)] [a_{i_3i_3}(x_5)] a_{i_3m_4}$$

Notice in this setup that since m_4 is the stop state, a path must end when it hits a vertex in the row corresponding to m_4 , and, of course, there is no emission from m_4 .

The value function $v_k(i)$ may now be defined in terms of admissible paths in the lattice:

$$v_k(i) \triangleq \text{maximum score over all admissible paths ending at } (i, k). \quad (8.26)$$

In words, $v_k(i)$ is the maximum probability of all emission-path sequences, $x_1 \dots x_i; m_0s_1 \dots s_{\ell-1}k$; notice now that the emitted sequence and the hidden path may have different lengths because of the presence of silent states.

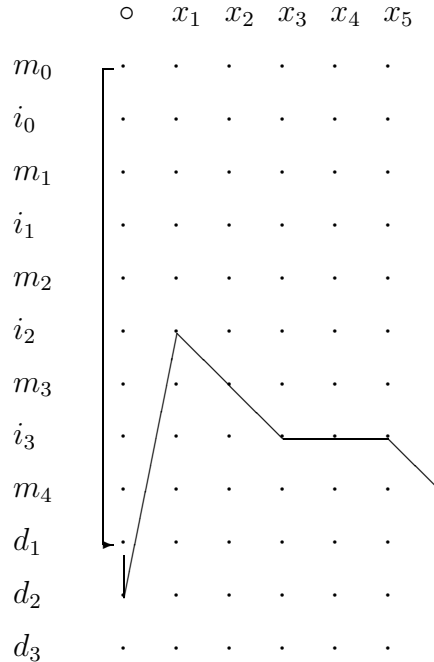


Figure 7.

We shall now repeat the reasoning that leads to a dynamical programming equation for $\{v_k(i)\}$, with the aid of some new notation. If ϵ denotes an

admissible edge, let $s(\epsilon)$ denote its score. For example the first edge in the path of Figure 7 has score $a_{m_0d_1}$. Let $\epsilon_{(\ell,j)\rightarrow(i,k)}$ denote the score of the directed edge from (ℓ, j) to (i, k) .

Consider $v_k(i)$ at the vertex (i, k) , and let (ℓ, j) be vertex from which there is an admissible edge pointing to $\mathcal{L}(i, k)$. The score of an admissible path to (i, k) that passes through (ℓ, j) is the score of the path to (j, ℓ) times the score $s(\epsilon_{(\ell,j)\rightarrow(i,k)})$ of the edge from (ℓ, j) to (i, k) . Therefore the maximum emission-path probability of any path from the hidden chain that ends at (i, k) and passes through (ℓ, j) is

$$v_j(\ell)s(\epsilon_{(\ell,j)\rightarrow(i,k)}), \quad (8.27)$$

because $v_j(\ell)$, is, by definition, that maximum emission-path probability over paths ending at vertex (ℓ, j) .

Clearly, $v_k(i)$ must be given by the maximum value of (8.27) over all vertices (ℓ, j) which lead to (i, k) by an admissible edge:

$$v_k(i) = \max \left\{ v_j(\ell)s(\epsilon_{(\ell,j)\rightarrow(i,k)}); \text{ the edge from } (\ell, j) \text{ to } (i, k) \text{ is admissible} \right\} \quad (8.28)$$

To start the algorithm, the values of $v_k(0)$ must be specified for only the rows k that correspond to the start state, the match states, or the insert state. These values are,

$$v_{m_0}(0) = 1, \quad v_{i_0}(0) = v_{i_1} = \dots = 0, \quad v_{m_1}(0) = v_{m_2} = \dots = 0. \quad (8.29)$$

Equations (8.28) and (8.29) give the general form of the dynamic programming equation for the profile hidden Markov model. Using it, the state transition diagram and the edge scores, we can derive the dynamic programming equation at each specific vertex, and, starting from the initial values, calculate $v_k(i)$ by recursive application of (8.28) at all other vertices. The optimal path is found as before from the tracebacks. We shall illustrate for the model defined in Figure 6, whose lattice is given in Figure 7.

Consider the vertex $(0, d_1)$. The state d_1 can be reached only from m_0 or i_0 , and so the only vertices from which $(0, d_1)$ is accessible by an admissible edge are $(0, m_0)$ and $(0, i_0)$. Since $s(\epsilon_{(0,m_0)\rightarrow(0,d_1)}) = a_{m_0d_1}$ and $s(\epsilon_{(0,i_0)\rightarrow(0,d_1)}) = a_{i_0d_1}$, (8.28) implies,

$$v_{d_1}(0) = \max \{ v_{m_0}(0)a_{m_0d_1}, v_{i_0}(0)a_{i_0d_1} \}.$$

But $v_{m_0}(0) = 1$ and $v_{i_0}(0) = 0$, and so $v_{d_1}(0) = a_{m_0d_1}$.

Similar reasoning shows that since the hidden chain can reach the silent state d_2 only from d_1 , i_1 , or m_1 :

$$v_{d_2}(0) = \max \{v_{m_1}(0)a_{m_1d_2}, v_{i_1}(0)a_{i_1d_2}, v_{d_1}(0)a_{d_1d_2}\} = v_{d_1}(0)a_{d_1d_2}.$$

Likewise,

$$v_{d_3}(0) = \max \{v_{m_2}(0)a_{m_2d_3}, v_{i_3}(0)a_{i_3d_3}, v_{d_2}(0)a_{d_2d_3}\} = v_{d_2}(0)a_{d_2d_3}.$$

The pattern established here for the silent states of column 0 repeats in all later columns, with minor modifications. As we know, the silent state d_1 can be reached only through transitions from i_0 or m_0 . However, if the column index i is greater than or equal to 1, the hidden chain will not be in state m_0 . Hence

$$v_{d_1}(i) = v_{i_0}(i)a_{i_0d_1}. \quad (8.30)$$

Since d_2 is reached by transitions from d_1 , i_1 , or m_1 only, we have, as for $i = 0$,

$$v_{d_2}(i) = \max \{v_{m_1}(i)a_{m_1d_2}, v_{i_1}(i)a_{i_1d_2}, v_{d_1}(i)a_{d_1d_2}\} \quad (8.31)$$

Likewise,

$$v_{d_3}(i) = \max \{v_{m_2}(i)a_{m_2d_3}, v_{i_3}(i)a_{i_3d_3}, v_{d_2}(i)a_{d_2d_3}\} \quad (8.32)$$

For the other vertices, those corresponding to match or insert states, the edges leading to (i, k) originate in the previous column. Take, for instance, $v_{m_2}(i)$ for $i \geq 1$. State m_2 can be reached only from states m_1 , 1_1 , and d_1 . Thus,

$$v_{m_2}(i) = \max \{v_{m_1}(i-1)a_{m_1m_2}e_{m_2}(x_i), v_{i_1}(i-1)a_{i_1m_2}e_{m_2}(x_i), v_{d_1}(i-1)a_{d_1m_2}e_{m_2}(x_i)\}. \quad (8.33)$$

The equations for the other vertices may be derived similarly.