Brazilian Journal of Biometrics, **42**, 307–328 (2024)
doi:10.28951/bjb.v42i4.704

BRAZILIAN JOURNAL OF BIOMΣTRICS
ISSN:2764-5290

ARTICLE

Mathematics behind the identifying CpG islands

Bijan Sarkar*

Department of Mathematics, Neotia Institute of Technology, Management and Science; and The Neotia University, Diamond Harbour Road, 24 Parganas (South), West Bengal - 743368, India.
*Corresponding author. Email: bijan0317@yahoo.com

(Received: November 13, 2023; Revised: January 23, 2024; Accepted: February 27, 2024; Published: October 15, 2024)

Abstract

The major objective of the paper is to review the theory for an hidden Markov model, a very general type of probabilistic model for sequences of symbols. In order for the hidden Markov model to be applicable to real-world applications, three key problems about the model must be addressed, and to do this, first we go over how to choose the best state sequence to explain an observation sequence, then we go over how to calculate the probability of an observation sequence, and finally we go over how to compute the maximization of the probability of the observation sequence. From these three angles, we review the mathematical concept behind the identification of CpG islands. The entire process and study of the outcomes have been tackled by examining both hypothetical and real DNA sequences side by side. We use well-known biological sequence analysis servers to carry out the experiment. Analytical and algorithmic approaches are compared while taking the hypothetical DNA sequence example into consideration.

Keywords: Viterbi algorithm; Forward algorithm; Backward algorithm; Posterior decoding; Baum-Welch algorithm

1. Introduction

A statistical model known as an hidden Markov model (HMM) can be used to explain how observable events evolve as a result of internal, indirectly visible causes. The seen event is referred to as a symbol, while the unobservable element underlying the observation is referred to as a state. Two stochastic processes — one invisible with hidden states and the other visible with observable symbols — combine to form an HMM. The observed symbol's probability distribution is determined by the underlying state, which is made up of hidden states that form a Markov chain (Compeau & Pevzner, 2015).

Since many practical problems involve categorising raw observation into a variety of categories or class labels that are more understandable to us, modelling observation in these two layers — one visible and the other unseen — is quite helpful. For illustration, let us look at the speech recognition issue, for which HMMs have been widely employed for many years. Predicting the spoken word from a recorded audio signal is what speech recognition is all about. In order to identify the phonemes (states) that led to the actual sound (observations) produced, the speech recognizer searches for them. The original phonemes must be predicted because there can be a significant difference in how a word is actually pronounced.

The cell is the fundamental unit of life, comprising various organelles that perform specialized functions within a membrane-bound structure. Cells can be categorized as prokaryotic or eukaryotic, differing in their complexity and presence of a nucleus. Genetic information is stored in nucleic acids, including DNA (deoxyribonucleic acid) and RNA (ribonucleic acid). DNA consists of double-stranded helical molecules, encoding genetic instructions. RNA, often single-stranded, aids in protein synthesis. The basic structure of nucleic acids consists of nucleotides, each composed of a phosphate group, a sugar molecule (deoxyribose in DNA, ribose in RNA), and a nitrogenous base. The four types of nitrogenous bases in DNA are adenine (A), thymine (T), cytosine (C), and guanine (G); in RNA, thymine is replaced by uracil (U). Genes within DNA encode details for protein synthesis, a process known as gene expression. The flow of genetic information involves DNA, RNA, and proteins, occurring through transcription and translation in the central dogma of molecular and cell biology (Alberts *et al.,* 2017).

Transcription is the process where RNA is synthesized from a DNA template. RNA polymerase reads the DNA sequence and produces a complementary RNA strand, allowing genetic information transfer. Translation is the process in protein synthesis where mRNA (messenger RNA) is decoded by ribosomes to assemble a corresponding sequence of amino acids, forming a functional protein in cells. The general gene structure in eukaryotic species includes a start codon (a triplet of nucleotides), typically AUG, signaling the initiation of protein synthesis, and a promoter region upstream that regulates transcription. The $5'$ UTR precedes the start codon, serving a regulatory role, while exons, the coding regions, contain information for protein synthesis. Introns, non-coding regions interspersed between exons, are removed during RNA splicing. The $3'$ UTR follows the stop codon, with regulatory functions. Signals in the intermediate scheme define exon/intron boundaries. After splicing, a mature mRNA comprises only exonic regions, primed for translation (Rocha & Ferreira, 2018).

In biological sequence analysis, HMMs are frequently employed. Typically, a biological sequence is made up of smaller substructures that each serve a different purpose, and various functional areas frequently exhibit different statistical characteristics. Proteins, for instance, in most cases have a number of domains, as is widely known. Predicting a new protein's composition domains (corresponding to one or more HMM states) and where they are located in the amino acid sequence (observations) would be intriguing. We could also want to determine in which protein family, this novel protein sequence belongs to. HMMs have been used to build a variety of sophisticated sequence analysis techniques that effectively describe biological sequences (Birney, 2001; Durbin *et al.,* 1998; Yoon, 2009).

CpG islands are specific DNA sequences or regions in a genome characterized by a high frequency of a particular dinucleotide called CpG (cytosine followed by guanine) and they play a significant role in gene regulation and epigenetics. The process by which a cell manages the expression of its genes to generate particular proteins or functional RNA molecules at the proper time and in the proper quantity is known as gene regulation, whereas epigenetics is the study of heritable changes in gene expression, which frequently involve chemical changes like DNA methylation and histone modifications, that change gene expression without changing the underlying DNA sequence. The identification of potential CpG islands has advanced our knowledge of the epigenetic origins of cancer in addition to assisting in the prediction of the promoters of numerous tissue-specific and housekeeping genes - a class of genes in all cells of an organism that are essential for basic cellular functions and maintenance. We know that the regions of genes responsible for initiating their expression are known as promoters. Genes tend to be "turned off" or silenced when these CpG islands in their promoter regions are methylated, meaning that methyl groups are added to them. This prevents the gene from being transcribed and from producing the appropriate protein or RNA, i.e., increased methylation within promoter regions can indicate potential cancer-associated changes (Robeva *et al.,* 2013).

In the context of evolutionary biology, one might wonder why DNA exhibits CpG islands. From an evolutionary standpoint, CpG islands are present in DNA because they serve a crucial role in governing gene activity. These islands often mark the initiation points of genes and contribute to the regulation of gene expression, which is essential for the adaptability and survival of organisms throughout the course of evolution. On a different note, when methylated C undergoes spontaneous deamination, a chemical transformation occurs, converting cytosine to thymine. Consequently, this process can lead to the transformation of a CpG pair into a TpG pair. This prompts the question of whether a high frequency of TpG is a distinctive feature of non-CpG islands. We know that methylation-associated deamination produces a significant increase in TpG prevalence outside of CpG islands, which provides the explanation. This finding clarifies the complex chemical processes behind DNA alterations and their consequences for evolution (David *et al.,* 2007).

A key tool for locating CpG islands in a DNA sequence is the concept of an HMM. Hidden Markov models usually include two states: one for the CpG island state and another for the non-CpG island state when it comes to CpG island identification. These states are connected by transitions that evaluate the probability of moving in or out of a CpG island. Each state's emission probabilities represent the possibility of finding particular nucleotide inside that state. The hidden Markov process can scan DNA sequences and estimate the most likely locations of CpG islands by learning the underlying sequence patterns through training on predetermined regions of CpG islands. The hidden Markov process is a useful method to advance our knowledge of genomic annotation and gene regulation (Fuentes-Beals *et al.,* 2022; Lan *et al.,* 2009).

In the present article, we review the theory for HMM by following Isaev's approach (Isaev, 2006). We concentrate on the three core issues with HMM design, namely: the identification of the optimal sequence for model states; the assessment of an observational sequence's probability (or likelihood) given a certain HMM; and the adjustment of model parameters to best explain the observed sequence (Coelho *et al.,* 2019; Rabiner, 1989; Rocha & Ferreira, 2018). Based on these three angles of views, we review how the mathematical concept works behind the identifying CpG islands. Both hypothetical and realistic DNA sequences have side by side been considered to address the whole procedure and analyze the results. We employ established biological sequence analysis servers for experimentation and discuss the comparison between analytical and algorithmic approaches, with a focus on a hypothetical DNA sequence.

2. Foundation pillars of the mathematical structure

2.1 The subject of Markov models

In this short article, to understand the HMM, we have to introduce the basic structure of Markov chains, or Markov models. To do that, first we take into account a finite collection X of all potential states $G_1, G_2, ..., G_N$, where one of these states is occupied by a Markov chain at each of the time points $t = 1, 2, 3, ...$ In each of the time steps t to $t + 1$, the process either remains in the same state or switches to another state in X. If the process is in state G_i at time t, then with certain probability at time $t + 1$ the process goes through every potential state G_j – specifically, including itself and any other states. Such probabilities $p_{G_iG_j}$ or simply p_{ij} depend only on the states that the process occupied before to the state G_i. The probabilities $p_{ij}, i, j = 1, 2, ..., N$ are referred to as the Markov chain transition probabilities. We select specific transition probability estimates based on the available biological data where data are used to estimate transition probabilities. In such application, the connectivity of the chain is fixed in advance.

Basically, a Markov chain is a method for creating every feasible sequence with a predetermined, finite length $L \geq 2$ of the form $x_1x_2...x_L$, where $x_j \in X$. To initiate the process, we need the initialization probabilities $P(G_j)$ for all $j = 1, 2, ..., N$, and the process shifts with probability p_{ij} from state G_i to state G_j. Thus, the probability of a particular sequence $x = x_1x_2...x_L$ is determined by

$$P(x) = P(x_1)p_{x_1x_2}p_{x_2x_3} \times ... \times p_{x_{L-1}x_L}. \tag{1}$$

Since a general representation of a sequence entails L symbols with state nature, the total number of sequences is $L!$. Each sequence's existence is associated with a probability, and the sum of these probabilities equals 1, because the values of $P(x_1)$ and p_{ij} are derived from the $L!$ possible sequences. A Markov chain can therefore be defined by initialization probabilities in the form of a vector and transition probabilities in the form of a matrix.

In most situations involving symbol representation, a sequence of letters is denoted by x. However, in Markov models, there is no distinction between a state sequence and a letter sequence in the sense that both are given in terms of a Markov chain. In this context, a state sequence can directly be characterized by a sequence of DNA alphabets. The reason for using x for both letter sequences and state sequences is that they share the same representation. On the other hand, in an HMM, a letter sequence and a state sequence are not equivalent. Only the state sequence is given by a Markov chain. Therefore, we need to use distinct symbols for letter sequences and state sequences. To represent the sequences, we consider two mathematical variables: x for the letter sequence and π for the state sequence comprising state elements such as G_is. Both variables can take any specific sequence of numeric as well as symbolic elements.

In our theoretical groundwork, we confine ourselves only to biological DNA sequences in prokaryotic organisms – simple cells without a nucleus, like bacteria. A prokaryotic gene is made up of a coding region that is flanked by a start codon at the beginning and a stop codon at the end. Each triplet of nucleotides (codon) between the start and stop codons codes for an amino acid. Such an arrangement is called open reading frames (ORFs). Here, to use real DNA data, we consider a set of n prokaryotic DNA sequences, and for such a collection, we can set the transition probabilities by the following formulation

$$p_{ab} = \frac{H_{ab}}{\sum\limits_{c \in \mathcal{Q}} H_{ac}}, \tag{2}$$

where the number of times in the data that nucleotide b follows nucleotide a is expressed as H_{ab}, and $\mathcal{Q}$ is the DNA alphabet. Applying such a concept, we can also calculate the initialization probability $P(a)$ as nucleotide a's frequency at the start of the sequences divided by n. The Markov chain that is created as a result is referred to as a model for the n sequences that are used to calculate the parameters. The collection of n sequences is also called training data.

Now, if we consider the unannotated DNA sequence $x = x_1...x_L$ generated by Markov chain, then the probability $P(x)$ of x is found from our known formula. At this position, if the value of $P(x)$ is large, then we can infer that x is derived from a prokaryotic gene; otherwise, the training data and x probability have no relationship. To calculate $P(x)$ we consider a Markov chain with begin state; we denote it by $\mathcal{B}$. Similarly, we add an end state $\mathcal{E}$ as an absorbing state. $\mathcal{B}$ and $\mathcal{E}$ do not transform into each other. With these two states, the Markov chain takes the form $\mathcal{B}x_1x_2...x_{L-1}x_L\mathcal{E}$ where the calculation formula for $P(x)$ changes to

$$P(x) = p_{\mathcal{B}x_1}p_{x_1x_2}p_{x_2x_3} \times ... \times p_{x_{L-1}x_L}p_{x_Lx_{\mathcal{E}}}. \tag{3}$$

It is important to mention that for simplification, $\mathcal{B}$ and $\mathcal{E}$ are often denoted by 0. We keep in mind that non-trivially connected model means if a path from $\mathcal{B}$ to G_j exists, then a path from G_j to $\mathcal{E}$ also exits in the graphical presentation of Markov chain connectivity, where states are represented by circles and transitions are denoted by arrows.

Rolling two six-sided dice, one of them fair and one unfair, is a classic example of a simple Markov chain. In the Markov process π of switching between two consecutive events, the state space of two elements is described by the $\mathcal{Q} = \{f, u\}$ with one specific value of transition probabilities $p_{ff} = 0.95, p_{fu} = 0.05, p_{uf} = 0.10$ and $p_{uu} = 0.90$, where the initial distribution is $p_{0f} = 0.5$ and $p_{0u} = 0.5$. Based on this information, we can compute any path of this process. As, for example, $P(\pi) = P(fufu) = p_{0f} \times p_{fu} \times p_{uf} \times p_{fu} = 0.5 \times 0.05 \times 0.10 \times 0.05 = 0.000125$. Generally, Markov chains can be applied for modeling situations with inherent randomness and sequential dependencies.

2.2 The subject of hidden Markov models

According to the definition of hidden Markov model (HMM), an HMM is a typical discrete time finite Markov chain of transition probabilities $p_{0j}, p_{ij}, p_{j0}, i, j = 1, 2, ..., N$ of states $G_1, ..., G_N$, which at each state emits a symbol of the alphabet $\mathcal{Q}$ in the sense that an emission probability $q_{G_k}(a) = q_k(a)$ is defined for each state and each symbol $a \in \mathcal{Q}$. In this context, the Markov chain is referred to as the HMM's underlying Markov chain. Basically, an HMM is a process of sequence generation. On the one way, we can think of an HMM as a method for producing a pair of sequences (x, π) where a sequence of letters from $\mathcal{Q}$ is x and π represents a sequence of Markov chain non-zero state path. Both x and π have equal length. We take $x = x_1 x_2 ... x_L$ and $\pi = \mathcal{B}\pi_1\pi_2...\pi_L\mathcal{E} = 0\pi_1\pi_2...\pi_L 0 = \pi_1\pi_2...\pi_L$, where element x_j is emitted at the state π_j for $j = 1, 2, ..., L$. For increasing the number of such model, it is justified to mention that state G_k emits an element a with probability $q_k(a)$. On the other way, HMM can also be viewed as a method that creates letter sequences from $\mathcal{Q}$ where we disregard pathways along which sequences are formed. Clearly, this way is useful when unknown pathways exist in the underlying Markov chain. Similar to Markov chains, HMMs are also derived from given training data in which estimates can be made of the transition and emission probabilities.

Now, if we let $x = x_1 ... x_L$ be a sequence of letters from $\mathcal{Q}$ and $\pi = \pi_1 ... \pi_L$ be a path that is the same length, then the probability $P(x, \pi)$ of the pair (x, π) is given by the following expression

$$P(x, \pi) = p_{0\pi_1} q_{\pi_1}(x_1) p_{\pi_1\pi_2} q_{\pi_2}(x_2) \times ... \times p_{\pi_{L-1}\pi_L} q_{\pi_L}(x_L) p_{\pi_L 0}. \tag{4}$$

The expression can be interpreted as this is the probability of the sequence x being formed along the path π. However, the above formula is useless in practice because we generally do not know the path for biological sequences. In such a situation, we find all paths that maximize $P(x, \pi)$ across all paths π of length equal to x's length, where we are also able to define the total probability $P(x)$ of x as

$$P(x) = \sum_{\text{all } \pi \text{ of length } L} P(x, \pi). \tag{5}$$

Thus, we define an HMM by four different collections of sets:

1. Set of observations $\mathcal{O} = \{O_1, O_2, ..., O_M\}$.
2. Set of states $\mathcal{S} = \{\mathcal{B}, G_1, G_2, ..., G_N, \mathcal{E}\}$.
3. Set of transition probabilities, $\mathcal{T}$, such that changing from state i to state j including itself is p_{ij} where clearly $\sum_{j \in \{G_1, G_2, ..., G_N, \mathcal{E}\}} p_{ij} = 1$, for every $i \in \{\mathcal{B}, G_1, G_2, ..., G_N\}$.
4. Set of emission probabilities, $\mathcal{R}$, such that the probability of observation O_j in state k is $q_k(O_j)$ where clearly $\sum_{O_j \in \mathcal{O}} q_k(O_j) = 1$.

And in the field of hidden Markov models we have three basic intentions:

First intention: Based on the observation sequence $x = x_1x_2...x_L$ and a model $\mathcal{M} = (\mathcal{T}, \mathcal{R})$, we determine a corresponding optimal state sequence $\pi = \pi_1\pi_2...\pi_L$ in the sense it best explains the observation.

Second intention: We compute $P(x)$, the probability of the observation sequence, where the observation sequence $x = x_1x_2...x_L$ and a model $\mathcal{M} = (\mathcal{T}, \mathcal{R})$ are given.

Third intention: From the maximized $P(x)$, we adjust the model parameter $\mathcal{M} = (\mathcal{T}, \mathcal{R})$.

In the occasionally dishonest casino example (Durbin *et al.,* 1998), the loss of a game is decided by the outcome 5 or 6 of a rolling die, where same as previous, one fair six-sided die and one unfair six-sided die are used to conduct the game. The unfair die has a higher probability, $P(6) = 0.5$, of landing on number 6, whereas $P(i) = 0.1$ for $i = 1, 2, 3, 4, 5$. Here, switching between the dice is an hidden Markov process, and the observations are a sequence of wins ($\mathcal{W}$s) and losses ($\mathcal{L}$s) that result from the dice rolls. The determined values of emission probabilities are $q_f(\mathcal{W}) = 0.67, q_f(\mathcal{L}) = 0.33, q_u(\mathcal{W}) = 0.4$ and $q_u(\mathcal{L}) = 0.6$. Based on this information, we can accurately compute any observed sequence of wins and losses if the exact path of a Markov chain of the same length is known. As, for example, if $x = 6563 = \mathcal{LLLW}$ and $\pi = fufu$, we obtain $P(x, \pi) = p_{0f} \times q_f(\mathcal{L}) \times p_{fu} \times q_u(\mathcal{L}) \times p_{uf} \times q_f(\mathcal{L}) \times p_{fu} \times q_u(\mathcal{W}) = 0.00000327$. Whenever the exact Markov path is unknown, either we find out the path π^* for which the probability $P(x, \pi)$ is maximized or we compute $P(x)$ for all possible hidden sequences π.

2.2.1 Elaboration of first intention

We are aware that decoding is a specific type of HMM-based search. In most of the situations, we wish to know a likely path for a sequence of letters x through the underlying Markov chain. Under the term "likely", one such intention is to search for a path π^* that maximizes the probability $P(x, \pi)$ among all pathways π with length equal to the length of x. It is important to note that the most probable obtained path may not be the only one. Also, we determine the path through recursive method because application of the direct approach is not practical where the number of possible paths increases exponentially with the length of the considering sequence. Here, the first intention is explained by Viterbi algorithm.

Assume we are given an HMM with an underlined Markov chain that has a state set of $\mathcal{S}' = \{G_1, G_2, ..., G_N\}$, begin and end states, and transition probabilities of $p_{0j}, p_{ij}, p_{j0}, i, j = 1, 2, ..., N$. For the given sequence $x = x_1...x_L$, we introduce the following quantity

$$v_k(i) = \max_{\pi_1,...,\pi_{i-1} \in \mathcal{S}'} p_{0\pi_1} q_{\pi_1}(x_1) p_{\pi_1\pi_2} q_{\pi_2}(x_2) \times ... \\ \times p_{\pi_{i-2}\pi_{i-1}} q_{\pi_{i-1}}(x_{i-1}) p_{\pi_{i-1}G_k} q_k(x_i),$$

where $i = 2, ..., L$ and $k = 1, ..., N$. Thus, we have

$$v_k(i+1) = q_k(x_{i+1}) \max_{l=1,...,N} (v_l(i) p_{lk}) \tag{6}$$

for all $i = 1, ..., L-1$ and $k = 1, ..., N$. Additionally, we define a set $\mathcal{V}_k(i)$ of all integers m for which $v_m(i)p_{mk} = \max_{l=1,...,N}(v_l(i)p_{lk}), i = 1, ..., L-1, k = 1, ..., N$, and the set $\mathcal{V}(L)$ consists of all integers m for which $v_m(L)p_{m0} = \max_{l=1,...,N}(v_l(L)p_{l0})$. Through the traceback procedure, we can determine the most probable Viterbi path. If we choose the states such that $m_L \in \mathcal{V}(L), m_{L-1} \in \mathcal{V}_{m_L}(L-1), m_{L-2} \in \mathcal{V}_{m_{L-1}}(L-2), ..., m_1 \in \mathcal{V}_{m_2}(1)$, then for the resulting path $\pi^* = 0G_{m_1}G_{m_2}...G_{m_L}0 = G_{m_1}G_{m_2}...G_{m_L}$ we get the maximum value of the probability term $P(x, \pi)$. The complete procedure steps are the following:

- Initialization step:

$$v_k(1) = p_{0k} q_k(x_1), \qquad k = 1, ..., N. \tag{7}$$

- Iteration step:

$$\begin{aligned} v_k(i) &= q_k(x_i) \max_{l=1,...,N} (v_l(i-1)p_{lk}), \qquad i = 2, ..., L, k = 1, ..., N, & (8)\\ \mathcal{V}_k(i) &= \max_{l=1,...,N} (v_l(i)p_{lk}), \qquad i = 1, 2, ..., L, k = 1, ..., N. & (9)\end{aligned}$$

- Termination step:

$$\begin{aligned} P &= \max_{k=1,...,N} v_k(L), & (10)\\ \mathcal{V}(L) &= \max_{l=1,...,N} (v_l(L)p_{l0}). & (11)\end{aligned}$$

To understand the procedure more deeply, as the base example we take into account the HMM whose underlying Markov chain is depicted in Figure 1 in which A and B are the two letters of the alphabet $\mathcal{Q}$ and whose probabilities of emission are as follows

$$\begin{gathered} q_1(A) = 0.5, q_1(B) = 0.5,\\ q_2(A) = 0.1, q_2(B) = 0.9,\\ q_3(A) = 0.9, q_3(B) = 0.1. \end{gathered}$$

Here, for $x = ABA$ we determine the unique Viterbi path π^*. We have

$$\begin{aligned} &v_1(1) = 0.2 \times 0.5 = 0.1, &&\\ &v_2(1) = 0.3 \times 0.1 = 0.03, &&\\ &v_3(1) = 0.5 \times 0.9 = 0.45, &&\\ &v_1(2) = 0.5 \times 0.1 \times 0.3 = 0.015, && \mathcal{V}_1(1) = \{1\},\\ &v_2(2) = 0.9 \times 0.45 \times 0.3 = 0.1215, && \mathcal{V}_2(1) = \{3\},\\ &v_3(2) = 0.1 \times 0.45 \times 0.3 = 0.0135, && \mathcal{V}_3(1) = \{3\},\\ &v_1(3) = 0.5 \times 0.015 \times 0.3 = 0.00225, && \mathcal{V}_1(2) = \{1\},\\ &v_2(3) = 0.1 \times 0.1215 \times 0.4 = 0.00486, && \mathcal{V}_2(2) = \{2\},\\ &v_3(3) = 0.9 \times 0.1215 \times 0.4 = 0.04374, && \mathcal{V}_3(2) = \{2\}. \end{aligned}$$

As a result, we have

$$\max_{\text{all } \pi \text{ of length 3}} P(x, \pi) = 0.04374 \times 0.4 = 0.017496,$$

and

$$\mathcal{V}(3) = \{3\}.$$

Following the trackback procedure, we get $m_3 = 3, m_2 = 2, m_1 = 3$, that is, the unique Viterbi path for x is $\pi = G_3G_2G_3$.

Now, by using Viterbi algorithm, we can calculate the most likely path for the same sequence on the MATLAB Online - MathWorks. Here, the transition matrix and emission matrix respectively

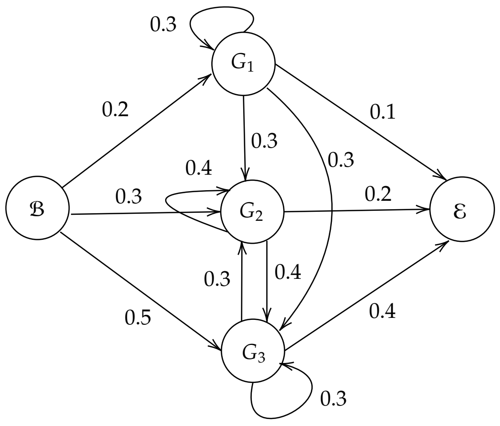

Figure 1. Graphical presentation of underlying Markov chain.

are given by: `trans` = [0, 0.2, 0.3, 0.5, 0; 0, 0.3, 0.3, 0.3, 0.1; 0, 0, 0.4, 0.4, 0.2; 0, 0, 0.3, 0.3, 0.4; 0, 0, 0, 0, 1] and `emis` = [0, 0; 0.5, 0.5; 0.1, 0.9; 0.9, 0.1; 0, 0]. The probability of transition from state i to state j is given by `trans`(i, j) and the probability of symbol A is emitted from state k is given by `emis`(k, A). To generate the sequence, $x = ABA$, we use the function `hmmgenerate`. Given the sequence ABA, the function `hmmviterbi` calculates the most likely path as 4 3 4, which is same as above obtained unique Viterbi path $G_3G_2G_3$.

2.2.2 Elaboration of second intention

To reach our intention, we will go over an algorithm for figuring out the total probability $P(x)$ of a sequence $x = x_1 ... x_L$ of letters from the $\mathcal{Q}$ alphabet. Just like the Viterbi algorithm, we use the recursive method to determine the total probability $P(x)$ because in the direct approach, enumerating all possible paths is not practical. The algorithm is referred to as the forward algorithm since it reads the sequence x forward. We introduce the following quantity

$$f_k(i) = \sum_{\pi_1,...,\pi_{i-1}\in\mathcal{S}'} p_{0\pi_1} q_{\pi_1}(x_1) p_{\pi_1\pi_2} q_{\pi_2}(x_2) \times ... \times p_{\pi_{i-2}\pi_{i-1}} q_{\pi_{i-1}}(x_{i-1}) p_{\pi_{i-1}G_k} q_k(x_i),$$

where $i = 2, ..., L$ and $k = 1, ..., N$. Thus, we have

$$f_k(i+1) = q_k(x_{i+1}) \sum_{l=1}^{N} f_l(i) p_{lk} \tag{12}$$

for all $i = 1, ..., L-1$ and $k = 1, ..., N$. The complete procedure steps are the following:

- Initialization step:

$$f_k(1) = p_{0k} q_k(x_1), \qquad k = 1, ..., N. \tag{13}$$

- Iteration step:

$$f_k(i) = q_k(x_i) \sum_{l=1}^{N} f_l(i-1) p_{lk}, \qquad i = 2, ..., L, k = 1, ..., N. \tag{14}$$

- Termination step:

$$P(x) = \sum_{k=1}^{N} f_k(L) p_{k0}. \tag{15}$$

To understand the procedure more deeply, we take into account the base example of HMM and find $P(x)$, where $x = BBA$. We have

$$\begin{aligned}
&f_1(1) = 0.2 \times 0.5 = 0.1,\\
&f_2(1) = 0.3 \times 0.9 = 0.27,\\
&f_3(1) = 0.5 \times 0.1 = 0.05,\\
&f_1(2) = 0.5 \times 0.1 \times 0.3 = 0.015,\\
&f_2(2) = 0.9(0.1 \times 0.3 + 0.27 \times 0.4 + 0.05 \times 0.3) = 0.1377,\\
&f_3(2) = 0.1(0.1 \times 0.3 + 0.27 \times 0.4 + 0.05 \times 0.3) = 0.0153,\\
&f_1(3) = 0.5 \times 0.015 \times 0.3 = 0.00225,\\
&f_2(3) = 0.1(0.015 \times 0.3 + 0.1377 \times 0.4 + 0.0153 \times 0.3) = 0.006417,\\
&f_3(3) = 0.9(0.015 \times 0.3 + 0.1377 \times 0.4 + 0.0153 \times 0.3) = 0.057753.
\end{aligned}$$

As a result, we have

$$P(x) = 0.00225 \times 0.1 + 0.006417 \times 0.2 + 0.057753 \times 0.4 = 0.0246096.$$

Another approach that can be used to calculate $P(x)$ is the backward algorithm. The reason this algorithm has a different name from the forward algorithm is because the sequence x is read in reverse order, starting from the end and proceeding towards the beginning. The forward algorithm calculates the probability of observing a sequence, given a model, by summing over all possible paths. In contrast, the backward algorithm calculates the probability of observing the remaining part of the sequence, given a state and model, by summing over possible future paths. We introduce the following quantity

$$b_k(i) = \sum_{\pi_{i+1},\dots,\pi_L \in \mathcal{S}'} p_{G_k \pi_{i+1}} q_{\pi_{i+1}}(x_{i+1}) p_{\pi_{i+1}\pi_{i+2}} q_{\pi_{i+2}}(x_{i+2}) \times \dots \\ \times p_{\pi_{L-1}\pi_L} q_{\pi_L}(x_L) p_{\pi_L 0},$$

where $i = 1, \dots, L-1$ and $k = 1, \dots, N$. Based on this quantity, the full procedure's steps are as follows:

- Initialization step:

$$b_k(L) = p_{k0}, \qquad k = 1, \dots, N. \tag{16}$$

- Iteration step:

$$b_k(i) = \sum_{l=1}^{N} p_{kl} q_l(x_{i+1}) b_l(x_{i+1}), \qquad i = 1, \dots, L-1, k = 1, \dots, N. \tag{17}$$

- Termination step:

$$P(x) = \sum_{k=1}^{N} p_{0k} q_k(x_1) b_k(1). \tag{18}$$

To understand the procedure more deeply, we take into account the base example of HMM and find $P(x)$, where $x = BBA$. We have

$$
\begin{aligned}
&b_1(3) = 0.1,\\
&b_2(3) = 0.2,\\
&b_3(3) = 0.4,\\
&b_1(2) = 0.3 \times 0.5 \times 0.1 + 0.3 \times 0.1 \times 0.2 + 0.3 \times 0.9 \times 0.4 = 0.129,\\
&b_2(2) = 0.4 \times 0.1 \times 0.2 + 0.4 \times 0.9 \times 0.4 = 0.152,\\
&b_3(2) = 0.3 \times 0.1 \times 0.2 + 0.3 \times 0.9 \times 0.4 = 0.114,\\
&b_1(1) = 0.3 \times 0.5 \times 0.129 + 0.3 \times 0.9 \times 0.152 + 0.3 \times 0.1 \times 0.114 = 0.06381,\\
&b_2(1) = 0.4 \times 0.9 \times 0.152 + 0.4 \times 0.1 \times 0.114 = 0.05928,\\
&b_3(1) = 0.3 \times 0.9 \times 0.152 + 0.3 \times 0.1 \times 0.114 = 0.04446.
\end{aligned}
$$

As a result, we have

$$P(x) = 0.2 \times 0.5 \times 0.06381 + 0.3 \times 0.9 \times 0.05928 + 0.5 \times 0.1 \times 0.04446 = 0.0246096.$$

We obtain the same result as before.

Now, for the given HMM, we define the sample space

$$
\begin{aligned}
\Omega = \Big\{(y, \pi) : y \text{ is sequence of letters and } \pi \text{ has same path length}\\
\text{through the underlying Markov chain}\Big\}.
\end{aligned}
$$

Next, we define two events $E(x)$ and $E_{i,k}$ such that for a fixed sequence $x = x_1 \ldots x_L$ the event

$$E(x) = \Big\{(y, \pi) \in \Omega : y = x\Big\},$$

and the event

$$
\begin{aligned}
E_{i,k} = \Big\{(y, \pi) \in \Omega : 1 \leq i \leq L \text{ with length of sequences} \geq i\\
\text{and } 1 \leq k \leq N \text{ with } \pi_i = G_k\Big\}.
\end{aligned}
$$

Thus, the conditional event $E_{i,k}|E(x)$ is defined as the state G_k emitting the i^{th} element x_i of x ,i.e., event $E_{i,k}$ occurring given that event $E(x)$ has occurred. Therefore, through a simple calculation, we can show that

$$
\begin{aligned}
P(E_{i,k}|E(x)) &= \frac{P(E_{i,k} \cap E(x))}{P(E(x))}\\
&= \frac{P(E_{i,k} \cap E(x))}{P(x)}\\
&= \frac{f_k(i) b_k(i)}{P(x)} \qquad (19)
\end{aligned}
$$

for $i = 1, \ldots, L$ and $k = 1, \ldots, N$. Here, the probability $P(E_{i,k}|E(x))$ is called the posterior probability of state G_k at observation i given x.

For decoding the posterior probabilities, we introduce the set $\mathcal{B}(i)$ which is the collection of the most probable states for x_i, $i = 1, ..., L$, i.e., the set $\mathcal{B}(i)$ contains all states $G_m \in \mathcal{S}'$ for which $P(E_{i,m}|E(x)) = max_{k=1,...,N} P(E_{i,k}|E(x))$.

To understand the procedure more deeply, we take into account the base example of HMM and find posterior decoding, where $x = BBA$. We have

$$\begin{aligned} &P(E_{1,1}|E(x)) = 0.25928906, && P(E_{1,2}|E(x)) = 0.65038034,\\ &P(E_{1,3}|E(x)) = 0.0903306, && P(E_{2,1}|E(x)) = 0.07862785,\\ &P(E_{2,2}|E(x)) = 0.85049737, && P(E_{2,3}|E(x)) = 0.07087478,\\ &P(E_{3,1}|E(x)) = 0.00914277, && P(E_{3,2}|E(x)) = 0.05215038,\\ &P(E_{3,3}|E(x)) = 0.93870685, \end{aligned}$$

and the posterior decoding is

$$\mathcal{B}(1) = \{G_2\}, \ \ \mathcal{B}(2) = \{G_2\}, \ \ \mathcal{B}(3) = \{G_3\}.$$

Consequently, decoding yields a single path

$$\pi = G_2 G_2 G_3,$$

where we obtain

$$\begin{aligned} P(x, \pi) &= p_{02} q_2(B) p_{22} q_2(B) p_{23} q_3(A) p_{30}\\ &= 0.3 \times 0.9 \times 0.4 \times 0.9 \times 0.4 \times 0.9 \times 0.4\\ &= 0.0139968. \end{aligned}$$

Here, Viterbi algorithm and posterior decoding procedure give the same result.

Unless there is a problem with the HMM parameters or the observation sequence is quite unclear, Viterbi and posterior decoding rarely yield significantly different outcomes in practice. To understand the difference in the outcomes of Viterbi algorithm and posterior decoding, we take into account the HMM with state set $\{G_1, G_2\}$, whose transition probabilities of underlying Markov chain are given by $p_{11} = 0.6$, $p_{12} = 0.4$, $p_{21} = 0.3$, $p_{22} = 0.7$ where the initial distribution is $p_{01} = 0.45$, $p_{02} = 0.55$. A and B are the two letters of the alphabet $\mathcal{Q}$ and whose probabilities of emission are as follows

$$\begin{aligned} q_1(A) = 0.3, q_1(B) = 0.7,\\ q_2(A) = 0.6, q_2(B) = 0.4. \end{aligned}$$

Here, for $x = BA$ we determine the unique Viterbi path $\pi^* = G_2 G_2$ with probability $P(x, \pi^*) = 0.0924$, whereas decoding yields a single path $\pi = G_1 G_2$ with probability $P(x, \pi) = p_{01} q_1(B) p_{12} q_2(A) = 0.0756$. In this case, the emission probability for observation B, $q_1(B) = 0.7$, is higher for state G_1 than $q_2(B) = 0.4$ for state G_2 while initial distribution of G_2 is greater than G_1. This creates ambiguity because depending on the observations at each step, the Viterbi algorithm may select a sequence that maximizes the joint probability, but posterior decoding may select a sequence with greater probabilities for individual states. This is why the Viterbi algorithm is commonly favored when our focus lies in deducing the optimal state sequence for the entire observation x. Conversely, the posterior-decoding approach is favored when our priority is to predict the optimal state at a particular position (Yoon, 2009).

Now, by using `hmmdecode` algorithm, we can calculate the posterior state probabilities, logarithm of the probability, forward and backward probabilities of the same sequence BBA on the

MATLAB Online - MathWorks. To consider the effect of p_{k0}, we have to take the sequence $BBAZ$ where at the end we include an additional symbol Z having $q_0(Z) = 1$. However, all calculation are performed under the assumption of p_{k0} being equal to 1, i.e., the transition probability vector of end states $[0.1; 0.2; 0.4]$ is replaced by $[1; 1; 1]$. Regarding such replacement, the backward probability matrix $[b_k(i) : (k, i)^{th}\text{element}]_{3\times 3}$ takes the matrix form of $[0.184500, 0.450000, 1.000000; 0.156000, 0.400000, 1.000000; 0.117000, 0.300000, 1.000000]$ and the probability of the sequence $P(x)$ is equal to 0.06642 instead of 0.0246096. We note that $P(BBA) = p_{A0}p_{BA}p_{BB}p_{0B}$, and get scaling factors as $S = [p_{A0} : 1.000, p_{0B} : 0.4200, p_{BB} : 0.4000, p_{BA} : 0.3954]$, i.e., $P(BBA) = 0.06642$ (Gundersen, 2022). Here, the posterior state probabilities `PSTATES` are the array presentation of conditional probabilities in which $(i, j)^{th}$ element gives the probability of j^{th} symbol of observed sequence being at state i. Similarly, 5×4 forward probability array, `FORWARD`, at $(k, i)^{th}$ position gives the scaled probability value of $f_k(i)$ while 5×4 backward probability array, `BACKWARD`, at $(k, i)^{th}$ position gives the scaled probability value of $b_k(i)$. Set the sequence as [2 2 1]. On that position, the function `hmmdecode` calculates the following quantities: `PSTATES` = $[0, 0, 0; 0.2778, 0.1016, 0.0339; 0.6341, 0.8293, 0.0966; 0.0881, 0.0691, 0.8695; 0, 0, 0]$, `logpseq` = -2.7118, `FORWARD` = $[1.0000, 0, 0, 0; 0, 0.1 \div (0.1 + 0.27 + 0.05) = 0.2381, 0.015 \div (0.015 + 0.1377 + 0.0153) = 0.0893, 0.00225 \div (0.00225 + 0.006417 + 0.057753) = 0.0339; 0, 0.27 \div (0.1 + 0.27 + 0.05) = 0.6429, 0.1377 \div (0.015 + 0.1377 + 0.0153) = 0.8196, 0.006417 \div (0.00225 + 0.006417 + 0.057753) = 0.0966; 0, 0.05 \div (0.1 + 0.27 + 0.05) = 0.1190, 0.0153 \div (0.015 + 0.1377 + 0.0153) = 0.0911, 0.057753 \div (0.00225 + 0.006417 + 0.057753) = 0.8695; 0, 0, 0, 0]$, `BACKWARD` = $[1.0000, 1.0623, 1.4670, 1.0000; 1.1037, 0.184500 \div (0.4000 \times 0.3954) \approx 1.1667, 0.450000 \div 0.3954 \approx 1.1382, 1.0000; 0.9160, 0.156000 \div (0.4000 \times 0.3954) \approx 0.9864, 0.400000 \div 0.3954 \approx 1.0117, 1.0000; 0.6870, 0.117000 \div (0.4000 \times 0.3954) \approx 0.7398, 0.300000 \div 0.3954 \approx 0.7588, 1.0000; 0, 0, 0, 1.0000]$. The `PSTATES` data clearly determine the most likely path $G_2G_2G_3$, and all other generated results are identical to those already obtained.

2.2.3 Elaboration of third intention

We will describe how one might select parameter values for a given training dataset in a logical manner in order to clarify the third intention. The components of a training dataset are either a collection of pairs of sequences $(x^1, \pi^1), \ldots, (x^n, \pi^n)$ or a collection of sequences $x^1, \ldots, x^n$, where a finite sequence of letters is x^j and a path of the same length is π^j. We will try to choose the parameter values so that $P(x^1, \pi^1) \times \ldots \times P(x^1, \pi^1)$ is maximally possible for datasets of the first type and that $P(x^1) \times \ldots \times P(x^n)$ is maximally possible for datasets of the second type. We can easily determine how many times each specific transition or emission is utilized in the collection of training sequences for the first type. If these be indicated by $H_{\alpha\beta}$ and $J_l(a)$, respectively, with $\alpha = \mathcal{B}, 1, \ldots, N$, $\beta = 1, \ldots, N, \mathcal{E}$, then we get

$$p_{\alpha\beta} = \frac{H_{\alpha\beta}}{\sum\limits_{\gamma=1,\ldots,N,\mathcal{E}} H_{\alpha\gamma}}, \qquad q_l(a) = \frac{J_l(a)}{\sum\limits_{b\in Q} J_l(b)}. \tag{20}$$

The estimating processes for the second type of datasets are more difficult than those for the first, where paths are unknown. In this case, we want to select parameter values that maximize the likelihood $P(x^1) \times \ldots \times P(x^n)$. The Baum-Welch training algorithm is a specific iteration technique that is frequently employed for the special situation of maximization of the function $P(x^1) \times \ldots \times P(x^n)$.

The algorithm will now be thoroughly explained. Similar to previous, we define the event

$$E_{i,(k,l)} = \Big\{(y, \pi) \in \Omega : 1 \le i \le L \text{ with length of sequences} \ge i + 1 \\ \text{and } 1 \le k, l \le N \text{ with } \pi_i = G_k,\ \pi_{i+1} = G_l \Big\}.$$

Therefore, through a simple calculation, we can show that

$$\begin{aligned} P(E_{i,(k,l)}|E(x)) &= \frac{P(E_{i,(k,l)} \cap E(x))}{P(E(x))} \\ &= \frac{P(E_{i,(k,l)} \cap E(x))}{P(x)} \\ &= \frac{f_k(i)p_{kl}q_l(x_{i+1})b_l(i+1)}{P(x)} \end{aligned} \tag{21}$$

for $i = 1, \dots, L-1$ and $k, l = 1, \dots, N$. The probability that x_i and x_{i+1} are released at states G_k and G_l, respectively, is given by the expression $P(E_{i,(k,l)}|E(x))$. Next, on the assumption that states $\mathcal{B}$ and $\mathcal{E}$ emit symbols $\mathcal{B}$ and $\mathcal{E}$, respectively with a probability of 1, one can get the following relation

$$\begin{aligned} P(E_{0,(\mathcal{B},l)}|E(x)) &= \frac{f_{\mathcal{B}}(0)p_{\mathcal{B}l}q_l(x_1)b_l(1)}{P(x)} \\ &= \frac{p_{0l}q_l(x_1)b_l(1)}{P(x)} \\ &= P(E_{1,l}|E(x)), \\ P(E_{L,(k,\mathcal{E})}|E(x)) &= \frac{f_k(L)p_{k\mathcal{E}}q_{\mathcal{E}}(L+1)b_{\mathcal{E}}(L+1)}{P(x)} \\ &= \frac{f_k(L)p_{k0}}{P(x)} \\ &= P(E_{L,k}|E(x)). \end{aligned} \tag{22}$$

The Baum-Welch algorithm starts with initial parameter values $p^{(0)}_{\alpha\beta}, q^{(0)}_k(a)$, $\alpha = \mathcal{B}, 1, \dots, N$, $\beta = 1, \dots, N, \mathcal{E}$, where to set the initial values we could take previous information about the training data into account. Let the training data be represented as $x^r = x^r_1 \dots x^r_{m_r}$, $r = 1, \dots, n$, and let $p^{(s)}_{\alpha\beta}, q^{(s)}_k$ be the parameter values at the algorithm's sth step. Also, $P^{(s)}(x^r), f^{r(s)}_k(i)$ and $b^{r(s)}_l(i)$ denote usual parameter values for the sequence x^r, $i = 1, \dots, m_r$. Now using formulas (21) and (22), we can determine the recursion step for the following transition probabilities

$$\begin{aligned} H^{(s)}_{kl} &= \sum_{r=1}^{n} \frac{1}{P^{(s)}(x^r)} \sum_{i=1}^{m_r-1} f^{r(s)}_k(i)p^{(s)}_{kl}q^{(s)}_l(x^r_{i+1})b^{r(s)}_l(i+1), \\ H^{(s)}_{\mathcal{B}l} &= \sum_{r=1}^{n} \frac{p^{(s)}_{0l}q^{(s)}_l(x^r_1)b^{r(s)}_l(1)}{P^{(s)}(x^r)}, \\ H^{(s)}_{k\mathcal{E}} &= \sum_{r=1}^{n} \frac{f^{r(s)}_k(m_r)p^{(s)}_{k0}}{P^{(s)}(x^r)}, \end{aligned} \tag{23}$$

and using the first formula in (20), the transition probabilities $p^{(s+1)}_{\alpha\beta}$ are calculated from $H^{(s)}_{\alpha\beta}$. Finally, based on the formula (19), we can determine the recursion step for the following emission probabilities

$$J_l^{(s)}(a) = \sum_{r=1}^{n} \frac{1}{P^{(s)}(x^r)} \sum_{\{i=1,\ldots,m_r : x_i^r = a\}} f_l^{r(s)}(i) b_l^{r(s)}(i), \tag{24}$$

and using the second formula in (20), the emission probabilities $q_l^{(s+1)}(a)$ are calculated from $J_l^{(s)}(a)$.

Now, to understand the procedure more deeply, we take into account the connectivity in Figure 2 and suppose the following training data are given to us

$$\begin{aligned} x^1 &: BAB \\ x^2 &: BAA \\ x^3 &: BA. \end{aligned}$$

Here, we apply the Baum-Welch algorithm to adjust the model parameter $\mathcal{M} = (\mathcal{T}, \mathcal{R})$ under the maximized $P(x)$. Basically, using the Baum-Welch procedure, we provide an estimate and update for emission and transition probabilities. Set the initial parameters to the following values

$$\begin{aligned} &p_{01}^{(0)} = 1, \quad p_{02}^{(0)} = 0, \\ &p_{11}^{(0)} = \frac{1}{2}, \quad p_{12}^{(0)} = \frac{1}{2}, \quad p_{10}^{(0)} = 0, \\ &p_{21}^{(0)} = 0, \quad p_{22}^{(0)} = 0, \quad p_{20}^{(0)} = 1, \\ &q_1^{(0)}(A) = \frac{1}{4}, \quad q_1^{(0)}(B) = \frac{3}{4}, \\ &q_2^{(0)}(A) = \frac{1}{2}, \quad q_2^{(0)}(B) = \frac{1}{2}. \end{aligned}$$

To each of the three training sequences, we apply the forward and backward algorithms, and we get

$$\begin{aligned} &f_1^{1(0)}(1) = \frac{3}{4}, f_2^{1(0)}(1) = 0, f_1^{1(0)}(2) = \frac{3}{32}, f_2^{1(0)}(2) = \frac{3}{16}, f_1^{1(0)}(3) = \frac{9}{256}, f_2^{1(0)}(3) = \frac{3}{128}, f_1^{2(0)}(1) = \frac{3}{4}, \\ &f_2^{2(0)}(1) = 0, f_1^{2(0)}(2) = \frac{3}{32}, f_2^{2(0)}(2) = \frac{3}{16}, f_1^{2(0)}(3) = \frac{3}{256}, f_2^{2(0)}(3) = \frac{3}{128}, f_1^{3(0)}(1) = \frac{3}{4}, f_2^{3(0)}(1) = 0, \\ &f_1^{3(0)}(2) = \frac{3}{32}, f_2^{3(0)}(2) = \frac{3}{16}, b_1^{1(0)}(1) = \frac{1}{32}, b_2^{1(0)}(1) = 0, b_1^{1(0)}(2) = \frac{1}{4}, b_2^{1(0)}(2) = 0, b_1^{1(0)}(3) = 0, \\ &b_2^{1(0)}(3) = 1, b_1^{2(0)}(1) = \frac{1}{32}, b_2^{2(0)}(1) = 0, b_1^{2(0)}(2) = \frac{1}{4}, b_2^{2(0)}(2) = 0, b_1^{2(0)}(3) = 0, b_2^{2(0)}(3) = 1, \\ &b_1^{3(0)}(1) = \frac{1}{4}, b_2^{3(0)}(1) = 0, b_1^{3(0)}(2) = 0, b_2^{3(0)}(2) = 1, P^{(0)}(x^1) = \frac{3}{128}, P^{(0)}(x^2) = \frac{3}{128}, P^{(0)}(x^3) = \frac{3}{16}. \end{aligned}$$

Consequently, the initial value of the likelihood is

$$P^{(0)}(x^1)P^{(0)}(x^2)P^{(0)}(x^3) = \frac{27}{262144}.$$

Next, by using formulas (23) and (24) we calculate the following quantities

$$\begin{aligned} &H_{11}^{(0)} = 2, \quad H_{12}^{(0)} = 3, \quad H_{21}^{(0)} = 0, \quad H_{22}^{(0)} = 0, \\ &H_{\mathcal{B}1}^{(0)} = 3, \quad H_{\mathcal{B}2}^{(0)} = 0, \quad H_{1\mathcal{E}}^{(0)} = 0, \quad H_{2\mathcal{E}}^{(0)} = 3, \end{aligned}$$

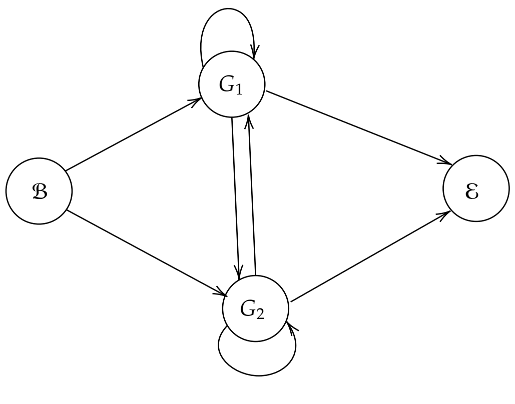

Figure 2. Graphical presentation of connectivity.

$$J_1^{(0)}(A) = 2, \quad J_1^{(0)}(B) = 3, \quad J_2^{(0)}(A) = 2, \quad J_2^{(0)}(B) = 1.$$

Thus, new parameter values are

$$\begin{array}{lll} p_{01}^{(1)} = 1, & p_{02}^{(1)} = 0, & \\ p_{11}^{(1)} = \frac{2}{5}, & p_{12}^{(1)} = \frac{3}{5}, & p_{10}^{(1)} = 0, \\ p_{21}^{(1)} = 0, & p_{22}^{(1)} = 0, & p_{20}^{(1)} = 1, \\ q_1^{(1)}(A) = \frac{2}{5}, & q_1^{(1)}(B) = \frac{3}{5}, & \\ q_2^{(1)}(A) = \frac{2}{3}, & q_2^{(1)}(B) = \frac{1}{3}. & \end{array}$$

Applying the forward algorithm further to each of the three training sequences with the updated parameter values, we obtain

$$f_1^{1(1)}(1) = \frac{3}{5}, f_2^{1(1)}(1) = 0, f_1^{1(1)}(2) = \frac{12}{125}, f_2^{1(1)}(2) = \frac{6}{25}, f_1^{1(1)}(3) = \frac{72}{3125}, f_2^{1(1)}(3) = \frac{12}{625}, f_1^{2(1)}(1) = \frac{3}{5}, f_2^{2(1)}(1) = 0, f_1^{2(1)}(2) = \frac{12}{125}, f_2^{2(1)}(2) = \frac{6}{25}, f_1^{2(1)}(3) = \frac{48}{3125}, f_2^{2(1)}(3) = \frac{24}{625}, f_1^{3(1)}(1) = \frac{3}{5}, f_2^{3(1)}(1) = 0, f_1^{3(1)}(2) = \frac{12}{125}, f_2^{3(1)}(2) = \frac{6}{25}, P^{(1)}(x^1) = \frac{12}{625}, P^{(1)}(x^2) = \frac{24}{625}, P^{(1)}(x^3) = \frac{6}{25}.$$

Consequently, the likelihood value following a single iteration of the Baum-Welch algorithm is

$$P^{(1)}(x^1)P^{(1)}(x^2)P^{(1)}(x^3) = \frac{1728}{9765625},$$

which is in fact higher than the initial figure that has been previously determined. For a specific observation, this result can be compare with the Baum-Welch algorithm result for the three sequences of $x^1 : ABA, x^2 : ABB, x^3 : AB$, under the same transition matrix and emission matrix (Isaev, 2006).

Now, by using the Baum-Welch algorithm, we can estimate the HMM parameters for the same problem on the MATLAB Online - MathWorks. Here, the transition matrix and emission matrix respectively are given by: `trans` = $[0,\ 1,\ 0,\ 0; 0,\ \frac{1}{2},\ \frac{1}{2},\ 0; 0,\ 0,\ 0,\ 1; 0,\ 0,\ 0,\ 1]$ and `emis` = $[0,\ 0; \frac{1}{4},\ \frac{3}{4}; \frac{1}{2},\ \frac{1}{2}; 0,\ 0]$. The probability of transition from state i to state j is given by `trans`(i,j) and the probability of symbol A is emitted from state k is given by `emis`(k, A). To generate the three considering sequences of lengths $3, 3, 2$, respectively, we use the function `hmmgenerate`. After, combining these three sequences in a cell array, the function `hmmtrain` estimates the transition probabilities as `estTR` = $[0,\ 1,\ 0,\ 0; 0,\ 0,\ 1,\ 0; 0,\ 0,\ 1,\ 0; 0,\ 0,\ 0,\ 1]$ and the emission probabilities as `estE` = $[0,\ 0; 0,\ 1; \frac{4}{5},\ \frac{1}{5}; 0,\ 0]$. The input argument `tolerance` takes the default value 10^{-6} which controls the number of execution steps of the `hmmtrain` algorithm. The difference between two consecutive steps of the algorithm defines the value of the `tolerance`.

3. Study of identifying CpG islands

In order to apply an HMM for detecting CpG islands in a DNA sequence, we consider two possible hidden states $\{+, -\}$ analogous to the fair and unfair dice, where the CpG island in a DNA sequence is labeled by + region and the non-CpG island in a DNA sequence is labeled by – region. Since each of these hidden states has the ability to emit a nucleotide, the configuration has an eight-element state space $\mathcal{S} = \{\ \text{A}_+, \text{A}_-, \text{C}_+, \text{C}_-, \text{G}_+, \text{G}_-, \text{T}_+, \text{T}_-\}$ corresponding to the constructed element set $\mathcal{Q} = \{\ \text{A, C, G, T}\ \}$ of a DNA sequence x. By using formula 2, from the given training data (a set of sequences of some genes), we can compute the transition probabilities for CpG islands and non-CpG islands. Here, we define two probability terms, p of the consecutive transition staying in a CpG island and q of the consecutive transition staying in a non-CpG island. Two procedures are used to identify whether x is coming from a CpG island or not.

In the first procedure of a short stretch of genome sequence, we compute the probability of a sequence x in a CpG island, $P(x|+)$, and the probability of a sequence x in a non-CpG island, $P(x|-)$, by assigning $p = 1$ and $q = 1$ in Table 1, respectively. Therefore, the odd ratio or the log-odds ratio, $log\frac{P(x|+)}{P(x|-)} = log\frac{\prod_{i=1}^{L} P^+_{x_{i-1}x_i}}{\prod_{i=1}^{L} P^-_{x_{i-1}x_i}} = \sum_{i=1}^{L} log\frac{P^+_{x_{i-1}x_i}}{P^-_{x_{i-1}x_i}}$ is computed to determine CpG islands by verifying whether the value of this ratio is greater than the threshold value 0. If we consider the sequence x = GCGCA, then the value of log-odds ratio is equal to $log\frac{0.34}{0.25} + log\frac{0.27}{0.08} + log\frac{0.34}{0.25} + log\frac{0.17}{0.32} > 0$. Thus, the sequence, GCGCA, is a part of a CpG island. However, for a long sequence this procedure does not work (Franzese & Iuliano, 2018).

In the second procedure of a long sequence, we have to incorporate both CpG islands and non-CpG islands into a single model, where each nucleotide can be emitted by either of the two states and thus the character of a one-to-one correspondence between the states and the symbols does not preserve. That is why we can not uniquely determine the path of a nucleotide sequence x in such

Table 1. The transition probabilities in CpG and non-CpG island

p_{ab}	A₊	C₊	G₊	T₊	A₋	C₋	G₋	T₋
A_+	$0.18p$	$0.27p$	$0.43p$	$0.12p$	$\frac{1-p}{4}$	$\frac{1-p}{4}$	$\frac{1-p}{4}$	$\frac{1-p}{4}$
C_+	$0.17p$	$0.37p$	$0.27p$	$0.19p$	$\frac{1-p}{4}$	$\frac{1-p}{4}$	$\frac{1-p}{4}$	$\frac{1-p}{4}$
G_+	$0.16p$	$0.34p$	$0.37p$	$0.13p$	$\frac{1-p}{4}$	$\frac{1-p}{4}$	$\frac{1-p}{4}$	$\frac{1-p}{4}$
T_+	$0.08p$	$0.36p$	$0.38p$	$0.18p$	$\frac{1-p}{4}$	$\frac{1-p}{4}$	$\frac{1-p}{4}$	$\frac{1-p}{4}$
A_-	$\frac{1-q}{4}$	$\frac{1-q}{4}$	$\frac{1-q}{4}$	$\frac{1-q}{4}$	$0.30q$	$0.20q$	$0.29q$	$0.21q$
C_-	$\frac{1-q}{4}$	$\frac{1-q}{4}$	$\frac{1-q}{4}$	$\frac{1-q}{4}$	$0.32q$	$0.30q$	$0.08q$	$0.30q$
G_-	$\frac{1-q}{4}$	$\frac{1-q}{4}$	$\frac{1-q}{4}$	$\frac{1-q}{4}$	$0.25q$	$0.25q$	$0.29q$	$0.21q$
T_-	$\frac{1-q}{4}$	$\frac{1-q}{4}$	$\frac{1-q}{4}$	$\frac{1-q}{4}$	$0.18q$	$0.24q$	$0.29q$	$0.29q$

configuration. In that situation, we can intuitively detect that this problem is similar to HMM of occasionally dishonest casino dice. However, the emission probability, $q_{n_*}(n)$, of each of the eight states, n_*, is exactly 1 for the particular nucleotide n and 0 for any other nucleotide.

Now, we consider a hypothetical example of a 100-nucleotide DNA sequence with a CpG island: ATGCATGCATGCATGCGCAGCTACGATGCATGCGCAGCTACGATGCATGCG CAGCTACGATGCATGCGCAGCTACGATGCATGCGCAGCTACG, and before go to the HMM procedure, we use a computational technique known as sliding window algorithm to identify potential CpG islands in DNA sequences. In this technique, to characterize the genomic regions with higher frequency of CG dinucleotides than expected based on the overall nucleotide composition, we measure two quantities: % C + G (percentage of Cytosine + Guanine) and O/E CpG (observed/expected CpG ratio). The % C + G of a sequence is calculated by the following formula: $\% \text{ C + G} = \frac{\text{(Number of Cs)+(Number of Gs)}}{\text{Total number of nucleotides in the sequence}} \times 100$, and the formula to determine the O/E CpG ratio is as follows: $\text{O/E CpG} = \frac{\text{Observed CpG frequency}}{\text{Expected CpG frequency}}$, where the expected number of CpG dinucleotides in a sequence of a specific length would be $\frac{\text{Number of Cs}\times\text{Number of Gs}}{\text{Length of the sequence}}$. The default threshold values, % C + G $\geq$ 50% and O/E CpG $\geq$ 0.6 or 0.65, indicate a potential CpG island. The sliding windows algorithm starts by defining a fixed-size window (the size defined by a number of base pairs) that moves one base pair at a time, from the $5'$ (start) end to the $3'$ (end) end of the sequence. The starting and terminating positions of a DNA strand are determined by the pentose carbon atoms at $5'$ and $3'$, respectively. DNA is read in the $5'$ to $3'$ direction, with the complementary strand running in the opposite direction. At each position of the window, the algorithm calculates the % C + G and the O/E CpG. The sensitivity and specificity of CG island detection are impacted by the size of the sliding window. Smaller windows can give more false positives but are more sensitive to smaller islands. After the entire sequence has been scanned, in the post-processing steps merging of identified islands that are very close to each other can help to refine the result and reduce redundancy (Takai & Jones, 2002). For the identification of CpG islands in our DNA sequence, we set window size 20, minimum length of island 30, minimum O/E CpG ratio 0.6, minimum percentage 50% on EMBOSS Cpgplot. We observe that in the final output a CpG island of length 68 starts at position 9 and extends to position 76. Instead of 20 if we use window size 5, then the sequence is characterized by a CpG island of length 89 with genomic coordinate (6...92). To improve accuracy and reduce false positives in CpG island prediction, we are often using sliding window algorithm as a preprocessing step in more complex methods, such as HMMs.

We observed that CpG island starts at position either 6 or 9 and extends to position either 92 or 76 which is identified by a high density of Cytosine-Guanine pairs. At this position, taking p = 0.9, q = 0.95 and assuming the uniform initial distribution of eight hidden states equal to 0.125, we re-identify this CpG island region by using algorithms related to HMM. The predicted path through the hidden states produced by Viterbi algorithm is used to identify CpG islands. Here, to generate the predicted path of the 100-nucleotide DNA sequence with HMM parameters, we use a user-friendly web server HMMTeacher. We assume that in CpG islands, the hidden states, A_+ and T_+, are suppressed with probability 0.3, i.e., emission probability, $q_{A_+}(A) = 0.7$ and $q_{T_+}(T) = 0.7$ instead of 1. Similarly, in non-CpG islands, the hidden states, C_- and G_-, are suppressed with probability 0.3, i.e., emission probability, $q_{C_-}(C) = 0.7$ and $q_{G_-}(G) = 0.7$ instead of 1.

The Viterbi algorithm is used to find the most likely sequence of hidden states that generates the observed DNA sequence. Applying the algorithm, we get the following Viterbi path of eight hidden states: A-T-G-C-A-T-G-C-A-T-G-C-A-T+G+C+G+C+A+G+C+T+A+C+G+A+T+G+ C+A+T+G+C+G+C+A+G+C+T+A+C+G+A+T+G+C+A+T+G+C+G+C+A+G+C+T+A+C+G+A+ T+G+C+A+T+G+C+G+C+A+G+C+T+A+C+G+A+T+G+C+A+T+G+C+G+C+A+G+C+T+A+C+ G+. The probability of the path corresponding to the most probable sequence of hidden states

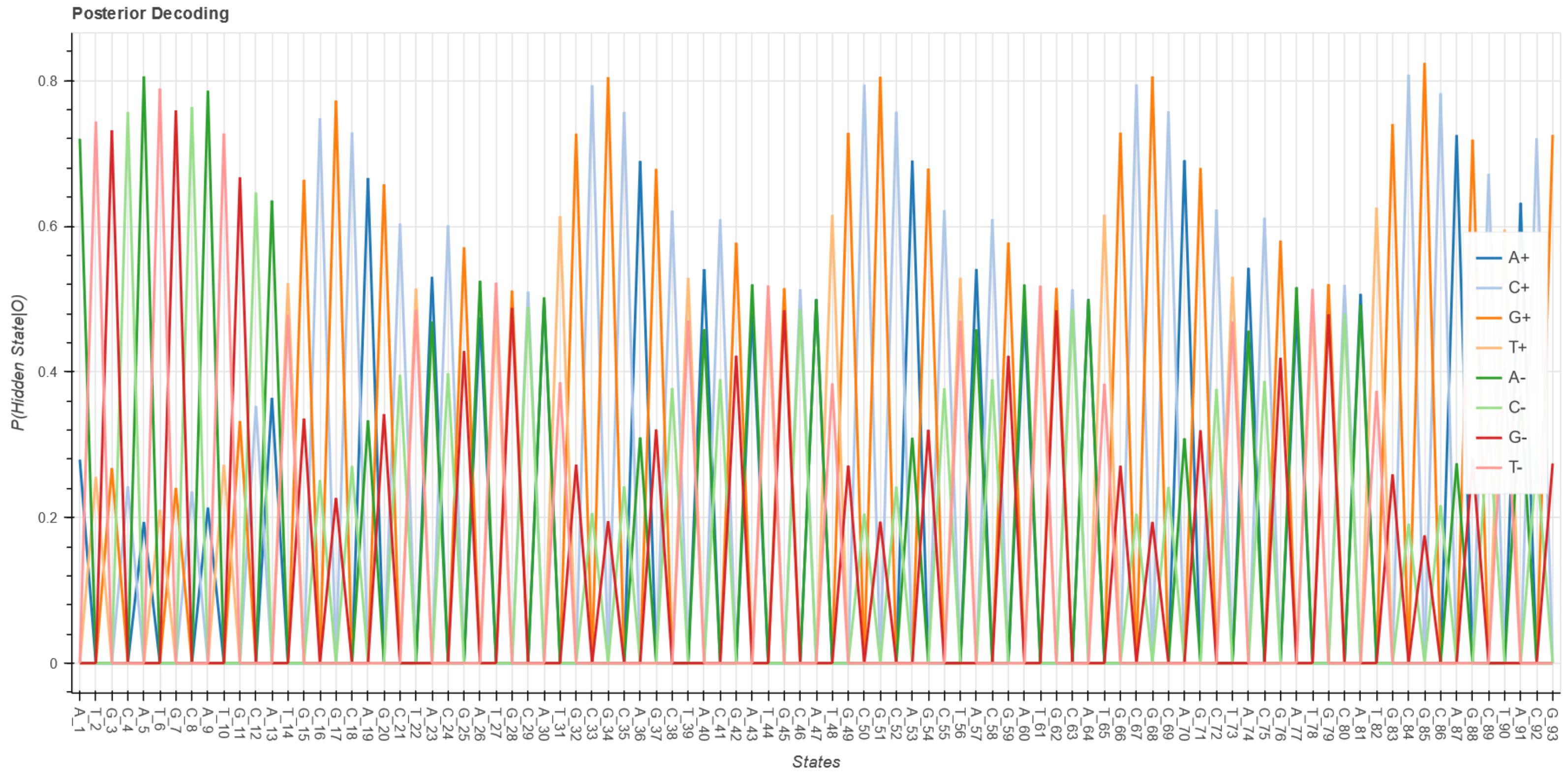

Figure 3. Graphical presentation of probabilities of each nucleotide from position 1 to position 93 in eight hidden states. The observed sequence is denoted by O.

is $7.51013181035043 \times 10^{-70}$. We observe that the CpG island starts at position 14 and extends to position 93. Both the forward and backward algorithms are used to compute the probability of observing the DNA sequence which is equal to $4.495059133983972 \times 10^{-66}$. By using the backward algorithm, we see that the probability that the position 9 (A) is emitted by hidden state C_+ is 0 . In case of state G_+, it is also equal to 0, whereas the probability that the position 15 (G) is emitted by hidden state G_+ is equal to 0.6639258973546764. Thus, the total probability that the position 15 (G) is emitted by other remaining seven hidden states is 0.33607410264. At each position of the DNA sequence, Figure 3 illustrates this scenario graphically using posterior decoding presentation. We know that by using Bayes' theorem posterior probability can be calculated by the following formula: $P(\text{Hidden state } k \text{ at position } i|O) = \frac{f_k(i) \times b_k(i)}{P(O)}$. It aims to find the probabilities of states at each position having specific nucleotide. As a result, at each position, we can select the hidden state with highest posterior probability, and like Viterbi algorithm, this is a probabilistic approach to finding the most likely state sequence in an HMM.

For the simple version of the same problem of finding CpG islands in the same DNA sequence, we consider only two hidden states: ' + ' for CpG island and ' – ' for non-CpG island. The initial probabilities for both hidden states are 0.5, while the transition probabilities between these two hidden states are the following: $p_{++} = 0.7, p_{+-} = 0.3, p_{-+} = 0.4$ and $p_{--} = 0.6$. The emission probabilities, which represent the probability of observing a particular nucleotide, are also taken into account as: $q_+(A) = 0.1$, $q_+(C) = 0.4$, $q_+(G) = 0.4$, $q_+(T) = 0.1$ and $q_-(A) = 0.4$, $q_-(C) = 0.1$, $q_-(G) = 0.1$, $q_-(T) = 0.4$. For these hidden Markov parameters, we determine the following Viterbi path of two hidden states: --++--++--++--+++++++--++--++--+++++++--++--++--+++++++--++--++--+++++++ --++--++--+++++++--++. The probability of the path corresponding to the most probable sequence of hidden states is $1.8702432948975872 \times 10^{-66}$. Merging of identified islands, the CpG island starts at position 3 and extends to position 61. Here, the calculated probability of finding the DNA sequence is $7.638069058758216 \times 10^{-57}$. By calculating the posterior probability, we see that the probability that the position 6 (T) is emitted by CpG island is 0.2673005494858582. As a result, that positioned T is emitted with probability 0.73269945051 by non-CpG island.

We know that through computational and bioinformatic methods without experimental confirmation, the presence of genes can be predicted based on various features and patterns found in the DNA. Among the various features, the presence of CpG islands which is associated with gene promoter regions is a valuable clue in gene prediction where the promoter region which is a critical section of DNA towards the $5'$ end of a gene controls when and to what extent a gene is transcribed into messenger RNA (mRNA) and subsequently translated into a protein. Therefore, investigating predicted genes by identifying CpG islands is an important research approach to conclude the predicted gene to be an actual gene.

Now, we collect few predicted genes from NCBI site, and by identifying CpG islands, we examine whether they are actual genes. To determine the overlapping genomic coordinate, we use BLAST - a search tool for DNA/protein sequence similarity.

- **Predicted sequence_1:** Onychostoma macrolepis major histocompatibility complex class I-related gene protein-like (LOC131534978), mRNA, NCBI Reference Sequence (Version No.): XM_058768130.1. This record is derived from a genomic sequence (NC_081179), NCBI Reference Sequence (Version No.): NC_081179.1.
  Investigating predicted gene by identifying CpG islands: Examine the 16.69kb region from base 1027548 to 1044240. Under the default settings, both DBCAT and EMBOSS Cpgplot are used to identify CpG islands. Here, the predicted gene does not appear to be a real gene from Onychostoma macrolepis, the bony fishes. There is one possible CpG island found from position 15461 to position 15681 with a length of 221 nucleotides and GC content of 52%. Clearly, it is not associated with promoter region. However, on the EMBOSS Cpgplot, we identify one

CpG island of genomic coordinate (1539...1748) with a length of 210 nucleotides and it may be part of promoter region. Another supporting evidence includes similarity to 56 other proteins.

- **Predicted sequence_2:** Canis lupus familiaris 28S ribosomal protein S22, mitochondrial-like (LOC119878138), mRNA, NCBI Reference Sequence (Version No.): XM_038588866.1. This record is derived from a genomic sequence (NC_049260.1), NCBI Reference Sequence (Version No.): NC_049260.1.
  Investigating predicted gene by identifying CpG islands: Examine the 9.17kb region from base 12056432 to 12065599. The predicted gene does appear to be a real gene from Canis lupus familiaris, the dog. There are four possible CpG islands found. The first one is found from position 1929 to position 2417 with a length of 489 nucleotides and GC content of 62%. Clearly, it is associated with promoter region. Other three identified CpG islands respectively are: (4170....4601, length of 432, GS content of 50%), (5820....6045, length of 226, GS content of 50%), (7876....8570, length of 695, GS content of 57%). However, on the EMBOSS Cpgplot, we identify only one CpG island of genomic coordinate (7968...8469) with a length of 502 nucleotides. Similarity to five proteins and RNAseq alignments that encompass 88% of the specified genomic feature provide additional supportive data.
- **Predicted sequence_3:** Pan troglodytes colony stimulating factor 2 receptor subunit alpha (CSF2RA), transcript variant X1, mRNA, NCBI Reference Sequence (Version No.): XM_05 5078641.1. This record is derived from a genomic sequence (NC_072422), NCBI Reference Sequence (Version No.): NC_072422.1.
  Investigating predicted gene by identifying CpG islands: Examine the 35.54 kb region from base 1575812 to 1611349. The predicted gene does appear to be a real gene from Pan troglodytes, the chimpanzee. There are twenty three possible CpG islands found. The first one is found from position 1 to position 230 with a length of 230 nucleotides and GC content of 60%. Clearly, it is associated with promoter region. Other identified CpG islands respectively are: (2593....2866, length of 274, GS content of 53%), (3092....3788, length of 697, GS content of 52%),......., (35148....35402, length of 255, GS content of 56%). On the EMBOSS Cpgplot, we identify eleven possible CpG islands. Another supporting evidence includes similarity to 2 other proteins.
- **Predicted sequence_4:** Oreochromis niloticus immunoglobulin lambda-1 light chain (LOC1 06097410), transcript variant X1, mRNA, NCBI Reference Sequence (Version No.): XM_019 357348.2. This record is derived from a genomic sequence (NC_031986.2), NCBI Reference Sequence (Version No.): NC_031986.2.
  Investigating predicted gene by identifying CpG islands: Examine the 13.15 kb region from base 27423744 to 27436896. The predicted gene does not appear to be a real gene from Oreochromis niloticus, the Nile tilapia. There are two possible CpG islands found. The first one is found from position 2048 to position 2448 with a length of 401 nucleotides and GC content of 54%. Clearly, it is not associated with promoter region. The second one, which has a GC content of 51% and a length of 229 nucleotides, is located between positions 11325 and 115553. On the EMBOSS Cpgplot, there is no CpG island. Similarity to two proteins and RNAseq alignments that encompass 97% of the specified genomic feature provide additional supportive data.

4. Conclusion

The theory of the HMM is reviewed in this paper. The use of HMMs has grown in popularity during the past few decades. Fundamentally, because the HMMs are so rich in mathematical structure, they can serve as the theoretical foundation for a variety of applications. Additionally, the models perform well in practice for many applications when used correctly. Because of this, we make an effort to concisely and methodically study the theoretical background of this kind of statistical modelling. Hidden Markov models have been used to model voice signals with success, and they are now one of the most used tools for analyzing biological sequences. Indeed, for modelling and analyzing biological sequences, HMMs offer a solid mathematical framework.

In genomic annotation, HMMs are essential, especially when it comes to predicting functional components in DNA sequences. In this context, the inherent complexity and unpredictability of genomic sequences are modelled using an HMM. Each state in the HMM represents a specific biological feature, such as exons, introns, promoters, or intergenic regions. An HMM can be trained to predict the position and boundaries of genes, splice sites, and other functional elements inside genomic sequences by using a dataset of known gene structures. This predictive power is crucial for genome annotation because it allows researchers to detect and characterise genes and regulatory elements within a DNA sequence without the need for time-consuming and expensive experimental procedures. Complex tasks such as modelling non-coding RNAs, epigenetic changes, or alternative splicing can be accomplished with advanced HMMs. Typically, genomic annotation is not a simple, yes-or-no, black-and-white task, i.e., not a binary operation; elements may overlap or be nested. It could be necessary to do post-processing procedures to improve the annotation and clear up any ambiguities (Ejigu & Jung, 2020; Mathé *et al.,* 2002).

Practically, HMMs are commonly constructed on large molecular sequence databases, where bioinformatics systems use the computed HMM parameters to predict the structure or function of new molecular sequences. In fact, the majority of applications of HMMs may need a lot of data to accurately estimate the model parameters. On the other hand, the number of hidden states, which is necessary for HMMs but can be tricky to estimate, must also be specified. In addition, since HMMs are statistical learning techniques, the accuracy of obtained outcomes may vary across different scenarios. The current study has been carried out within this context, and we observed that the results obtained from both the pure/manual algorithm of sliding window procedure and the mathematics-based algorithm of the HMM procedure are nearly identical for specific values of model parameters. Therefore, in order to avoid the time-consuming critical procedure, the examination of the appropriateness of such comparison method between pure and mathematical algorithms to adjust the model parameters must be included in our next research endeavor.

Acknowledgment

I wish to extend my appreciation to the insightful reviewer who highlighted several overlooked and significant biological and mathematical perspectives in the initial draft of my work. The quality of the article has unquestionably improved as a result of correcting and addressing these issues.

Conflicts of interest

The author declares no conflict of interest.

References

1. Alberts, B., Johnson, A., Lewis, J., Raff, M., Roberts, K. & Walter, P. *Molecular Biology of the Cell* (Garland science, 2017).
2. Bioinformatics and Biostatistics Core Research Center for Medical Excellence, National Taiwan University. DBCAT. Accessed: 2023-10-21. http://dbcat.cgm.ntu.edu.tw/.
3. Birney, E. Hidden Markov models in biological sequence analysis. *IBM Journal of Research and Development* **45,** 449 –454 (2001).
4. Coelho, J. P., Pinho, T. M. & Boaventura-Cunha, J. *Hidden Markov Models: Theory and Implementation using MATLAB* (CRC Press, 2019).
5. Compeau, P. & Pevzner, P. A. *Bioinformatics Algorithms: An Active Learning Approach* (Active Learning Publishers, 2015).

6. David, A. C., Thomas, J., Danny, R. & Marie-Laure, C. *Epigenetics* (Cold Spring Harbor Press, New York, NY, USA, 2007).

7. Durbin, R., Eddy, S. R., Krogh, A. & Mitchison, G. *Biological Sequence Analysis: Probabilistic Models of Proteins and Nucleic Acids* (Cambridge University Press, 1998).

8. Ejigu, G. F. & Jung, J. Review on the computational genome annotation of sequences obtained by next-generation sequencing. *Biology* **9,** 295 (2020).

9. EMBL's European Bioinformatics Institute. EMBOSS Cpgplot. Accessed: 2023-10-21. https://www.ebi.ac.uk/jdispatcher/seqstats/emboss_cpgplot.

10. Franzese, M. & Iuliano, A. Hidden Markov models. *Encyclopedia of Bioinformatics and Computational Biology: ABC of Bioinformatics,* 672 (2018).

11. Fuentes-Beals, C., Valdés-Jiménez, A. & Riadi, G. Hidden Markov modeling with HMMTeacher. *PLOS Computational Biology* **18,** e1009703 (2022).

12. Fuentes-Beals, C., Valdés-Jiménez, A. & Riadi, G. HMMTeacher. Accessed: 2023-10-21. https://hmmteacher.mobilomics.org/.

13. Gundersen, G. W. *Scaling factors for hidden Markov models* https://gregorygundersen.com/blog/2022/08/13/hmm-scaling-factors/. Accessed: 2023-10-21. Aug. 2022.

14. Isaev, A. *Introduction to Mathematical Methods in Bioinformatics* (Springer, 2006).

15. Lan, M., Xu, Y., Li, L., Wang, F., Zuo, Y., Chen, Y., Tan, C. L. & Su, J. *CpG-discover: A machine learning approach for CpG islands identification from human DNA sequence* in *2009 International Joint Conference on Neural Networks* (2009), 1702–1707.

16. Mathé, C., Sagot, M. F., Schiex, T. & Rouzé, P. Current methods of gene prediction, their strengths and weaknesses. *Nucleic Acids Research* **30,** 4103–4117 (2002).

17. National Library of Medicine. NCBI. Accessed: 2023-10-21. https://www.ncbi.nlm.nih.gov/.

18. Rabiner, L. R. A tutorial on hidden Markov models and selected applications in speech recognition. *Proceedings of the IEEE* **77,** 257 –286 (1989).

19. Robeva, R., Garrett, A., Kirkwood, J. & Davies, R. Identifying CpG islands: Sliding window and hidden Markov model approaches. *Mathematical Concepts and Methods in Modern Biology: Using Modern Discrete Models,* 267 (2013).

20. Rocha, M. & Ferreira, P. G. *Bioinformatics Algorithms: Design and Implementation in Python* (Academic Press, 2018).

21. Takai, D. & Jones, P. A. Comprehensive analysis of CpG islands in human chromosomes 21 and 22. *Proceedings of the National Academy of Sciences* **99,** 3740–3745 (2002).

22. The MathWorks Inc. MATLAB Online - MathWorks. Accessed: 2023-10-21. https://in.mathworks.com/products/matlab-online.html.

23. Yoon, B. J. Hidden Markov models and their applications in biological sequence analysis. *Current Genomics* **10,** 402 – 415 (2009).