Methods in Molecular Biology 1552

Springer Protocols

David R. Westhead
M. S. Vijayabaskar *Editors*

Hidden Markov Models

Methods and Protocols

Humana Press

METHODS IN MOLECULAR BIOLOGY

Series Editor
John M. Walker
School of Life and Medical Sciences
University of Hertfordshire
Hatfield, Hertfordshire, UK

For further volumes:
http://www.springer.com/series/7651

Hidden Markov Models

Methods and Protocols

Edited by

David R. Westhead

University of Leeds School
of Molecular and Cellular Biology
Leeds, UK

M.S. Vijayabaskar

University of Leeds School
of Cellular and Molecular Biology
Leeds, UK

Humana Press

Editors
David R. Westhead
University of Leeds School
of Molecular and Cellular Biology
Leeds, UK

M.S. Vijayabaskar
University of Leeds School
of Cellular and Molecular Biology
Leeds, UK

ISSN 1064-3745 ISSN 1940-6029 (electronic)
Methods in Molecular Biology
ISBN 978-1-4939-6751-3 ISBN 978-1-4939-6753-7 (eBook)
DOI 10.1007/978-1-4939-6753-7

Library of Congress Control Number: 2016956251

© Springer Science+Business Media LLC 2017
This work is subject to copyright. All rights are reserved by the Publisher, whether the whole or part of the material is concerned, specifically the rights of translation, reprinting, reuse of illustrations, recitation, broadcasting, reproduction on microfilms or in any other physical way, and transmission or information storage and retrieval, electronic adaptation, computer software, or by similar or dissimilar methodology now known or hereafter developed.
The use of general descriptive names, registered names, trademarks, service marks, etc. in this publication does not imply, even in the absence of a specific statement, that such names are exempt from the relevant protective laws and regulations and therefore free for general use.
The publisher, the authors and the editors are safe to assume that the advice and information in this book are believed to be true and accurate at the date of publication. Neither the publisher nor the authors or the editors give a warranty, express or implied, with respect to the material contained herein or for any errors or omissions that may have been made. The publisher remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.

Printed on acid-free paper

This Humana Press imprint is published by Springer Nature
The registered company is Springer Science+Business Media LLC
The registered company address is: 233 Spring Street, New York, NY 10013, U.S.A.

Preface

The biology of life is highly complex and dynamic, making it an attractive system for us to study and understand. The life-forms on earth range from single-celled organisms to large multicellular organisms and can sustain themselves in a variety of environmental conditions such as hot thermal vents and cold polar ice caps. Our fascination for this diversity has led us to study them in much detail, and we have come to understand that there is a plethora of patterns occurring in nature. For example the migratory patterns of different species of birds, and the presence of specific codons in DNA that allows a gene to be transcribed and translated into a protein of a specific amino acid sequence. Nature being too complex, these patterns are often not apparent and there is almost always an element of stochasticity. With the increasing influence of computer-based algorithms and statistics in biology, we have successfully come up with methods for modeling such complex and noisy biological systems. Hidden Markov Model (HMM) is one such statistical model widely used in modeling complex systems and in the identification of "hidden" patterns. It was initially used in the field of signal processing and speech recognition but because of its assumptions, its ability to process sequential data, and its tolerance to intrinsic noise it soon became an ideal tool to address many biological questions. By far the most extensively studied field in biology that uses HMM is the sequence analysis of protein and DNA/RNA biomolecules. There are numerous books that substantially cover HMM's seminal applications in studying biological sequences. But our understanding of biology has changed immensely since the start of this century, and with it the methods used to study them. These changes have enabled HMM to find its way into various new aspects of biological research. This book aims to provide the readers with a perspective of the newer and a broader usage of HMMs in biology. It is roughly structured in the order of the physical size of the biological system under study, ranging from single biomolecule (Chapters 2–7), cellular level (Chapters 8–13) to organism level (Chapters 14 and 15). All the authors in this book are experts in their respective fields with publications related to HMMs in peer-reviewed journals, and they have attempted to give you a good overview of what system they are modeling and how they have adopted HMM to attain their objectives. Therefore each chapter is autonomous and expert readers can get up to speed by reading only the chapters of their interest. For readers with no prior experience in HMM, we have provided you with an easy to read introduction to HMM (Chapter 1) that will aid you in understanding the rest of the chapters in this book. The beauty of HMM is that it is simple to understand and easy to apply to real-world scenarios, and hence the reader need not necessarily rely on equations given in the chapters in order to understand them. We hope that this book helps you in appreciating the impact of HMM in biology and inspire you in your own research.

Leeds, UK

David R. Westhead
M.S. Vijayabaskar

Contents

Contributors

Toshiya Arakawa • *Department of Mechanical Systems Engineering, Aichi University of Technology, Aichi, Gamagori, Japan*

Pantelis G. Bagos • *Department of Computer Science and Biomedical Informatics, University of Thessaly, Lamia, Greece*

William-Scott Beaufort • *Institute of Marine Biology and Genetics, Center for Marine Research, East Attica, Greece*

Jiawen Bian • *School of Mathematics and Physics, China University of Geosciences, Wuhan, China*

Hyungwon Choi • *Saw Swee Hock School of Public Health, National University of Singapore, Singapore, Singapore*

Julien Y. Dutheil • *Department of Evolutionary Genetics, Max Planck Institute for Evolutionary Biology, Molecular Systems Evolution, Plön, Germany*

Themis Exarchos • *Unit of Medical Technology and Intelligent Information Systems, Department of Materials Science and Engineering, University of Ioannina, Ioannina, Greece*

Vanessa Ferle • *Faculty of Sciences, School of Biology, University of Athens, Panepistimiopolis, Athens, Greece*

Christos Ferles • *Scientific Computing Research Unit and Department of Chemistry, University of Cape Town, Rondebosch, Cape Town, South Africa*

Dimitrios I. Fotiadis • *Unit of Medical Technology and Intelligent Information Systems, Department of Materials Science and Engineering, University of Ioannina, Ioannina, Greece*

Debashis Ghosh • *Department of Biostatstics & Informatics, University of Colorado Anschutz Medical Campus, University Park, PA, USA*

Stavros J. Hamodrakas • *Faculty of Biology, Department of Cell Biology and Biophysics, University of Athens, Panepistimiopolis, Athens, Greece*

Satoshi Kakihara • *Formerly National Graduate Institute for Policy Studies, Minato-ku, Tokyo, Japan*

Michelle Karg • *Electrical and Computer Engineering, University of Waterloo, Lindau, Germany*

Tsuyoshi Koide • *Transdisciplinary Research Integration Center, Research Organization of Information and Systems, Minato-ku, Tokyo, Japan; Mouse Genomics Resource Laboratory, National Institute of Genetics (NIG), Mishima, Shizuoka, Japan; Department of Genetics, SOKENDAI, Mishima, Shizuoka, Japan*

Dana Kulić • *Electrical and Computer Engineering, University of Waterloo, Waterloo, Ontario, Canada*

Christos Lampros • *Unit of Medical Technology and Intelligent Information Systems, Department of Materials Science and Engineering, University of Ioannina, Ioannina, Greece*

Alexander S. Moffett • *Center for Biophysics and Quantitative Biology, University of Illinois at Urbana-Champaign, Urbana, IL, USA*

Sunghee Oh • *Department of Computer Science and Statistics, Jeju National University, Jeju City, South Korea*

Kenji Okamoto • *Cellular Informatics Laboratory, RIKEN, Wako, Saitama, Japan*
Costas Papaloukas • *Department of Biological Applications and Technology, University of Ioannina, Ioannina, Greece*
Zhaohui Qin • *Department of Biostatistics and Bioinformatics, Rollins School of Public Health, Emory University, Atlanta, GA, USA; Department of Biomedical Informatics, Emory University School of Medicine, Atlanta, GA, USA*
Balaji Selvam • *Department of Chemical & Biomolecular Engineering, University of Illinois at Urbana-Champaign, Urbana, IL, USA*
Zahra Shamsi • *Department of Chemical & Biomolecular Engineering, University of Illinois at Urbana-Champaign, Urbana, IL, USA*
Saurabh Shukla • *Department of Chemical & Biomolecular Engineering, University of Illinois at Urbana-Champaign, Urbana, IL, USA*
Diwakar Shukla • *Department of Chemical & Biomolecular Engineering, Center for Biophysics and Quantitative Biology, National Center for Supercomputing Applications, Plant Biology, University of Illinois at Urbana-Champaign, Urbana, IL, USA*
Seongho Song • *Department of Mathematical Science, University of Cincinnati, Cincinnati, OH, USA*
Aki Takahashi • *Mouse Genomics Resource Laboratory, National Institute of Genetics (NIG), Mishima, Shizuoka, Japan; Laboratory of Behavioral Neuroendocrinology, University of Tsukuba, Tsukuba, Ibaraki, Japan*
Akira Tanave • *Transdisciplinary Research Integration Center, Research Organization of Information and Systems, Minato-ku, Tokyo, Japan; Mouse Genomics Resource Laboratory, National Institute of Genetics (NIG), Mishima, Shizuoka, Japan*
Margarita C. Theodoropoulou • *Faculty of Biology, Department of Cell Biology and Biophysics, University of Athens, Panepistimiopolis, Athens, Greece; Department of Computer Science and Biomedical Informatics, University of Thessaly, Papasiopoulou, Lamia, Greece*
Georgios N. Tsaousis • *Faculty of Biology, Department of Cell Biology and Biophysics, University of Athens, Panepistimiopolis, Athens, Greece*
Takashi Tsuchiya • *National Graduate Institute for Policy Studies, Minato-ku, Tokyo, Japan; The Institute of Statistical Mathematics, Tachikawa, Tokyo, Japan*
M.S. Vijayabaskar • *Faculty of Biological Sciences, School of Molecular and Cellular Biology, University of Leeds, Leeds, UK*
Veronica Vinciotti • *Department of Mathematics, Brunel University London, Uxbridge, UK*
Tao Wang • *Quantitative Biomedical Research Center, Department of Clinical Science, University of Texas Southwestern Medical Center, Dallas, TX, USA*
Guanghua Xiao • *Quantitative Biomedical Research Center, Department of Clinical Science, University of Texas Southwestern Medical Center, Dallas, TX, USA*
Yang Xie • *Quantitative Biomedical Research Center, Department of Clinical Science, University of Texas Southwestern Medical Center, Dallas, TX, USA*
Jonghyun Yun • *Department of Mathematics, University of Texas at Arlington, Arlington, TX, USA*
Xiaobo Zhou • *Center for Bioinformatics & Systems Biology, Department of Diagnostic Radiology, Wake Forest University—School of Medicine, Winston-Salem, NC, USA*

Chapter 1

Introduction to Hidden Markov Models and Its Applications in Biology

M.S. Vijayabaskar

Abstract

A number of real-world systems have common underlying patterns among them and deducing these patterns is important for us in order to understand the environment around us. These patterns in some instances are apparent upon observation while in many others especially those found in nature are well hidden. Moreover, the inherent stochasticity in these systems introduces sufficient noise that we need models capable to handling it in order to decipher the underlying pattern. Hidden Markov model (HMM) is a probabilistic model that is frequently used for studying the hidden patterns in an observed sequence or sets of observed sequences. Since its conception in the late 1960s it has been extensively applied in biology to capture patterns in various disciplines ranging from small DNA and protein molecules, their structure and architecture that forms the basis of life to multicellular levels such as movement analysis in humans. This chapter aims at a gentle introduction to the theory of HMM, the statistical problems usually associated with HMMs and their uses in biology.

Key words Hidden Markov model, Pattern recognition, Transition probability, Emission probability, Viterbi algorithm, Forward–backward procedure, Expectation maximization, Baum–Welch algorithm

1 Introduction

Nature is an impressive ever evolving system comprising a multitude of interacting microcosms. As a result of this evolution we are presented with a variety of life forms and their relationship with each other, as the environment we live in. These life forms range from single-celled microbes to multicellular organisms and we as humans have long been interested in studying these organisms in order to understand the way nature and evolution work. We have found that it is easier to understand and interpret if we base our studies on an assumption that there exist definitive patterns, a blueprint of sorts that drives nature. Some of these patterns are macroscopic in nature and has helped us to better understand the environment. For example Charles Darwin compiled his most famous work, "On the Origin of Species" that jump-started the

David R. Westhead and M.S. Vijayabaskar (eds.), *Hidden Markov Models: Methods and Protocols*, Methods in Molecular Biology, vol. 1552, DOI 10.1007/978-1-4939-6753-7_1, © Springer Science+Business Media LLC 2017

theory of evolution, by observing patterns in the physical features of animals and their changes that leads to transmutation of species. Another example, is Mendel's laws of classical genetics which are laws proposed on genetic level, based on the observations of phenotypic patterns during his experiments on *Pisum sativum*. Many patterns exists at a microscopic level, for example the pattern in base pairing of the double helical DNA that is eventually responsible for all life forms. We have numerous such studies based on observing patterns that occur in nature and if we compile all the works, we can definitively conclude that (a) while some patterns are evident, many of them are hidden and only become evident upon close observation, (b) the observations vary in infinite number of ways resulting in observables that contain inherent noise, and (c) leading to patterns that are more often stochastic than deterministic.

2 Hidden Markov Model

A hidden Markov model (HMM) is a probabilistic model that can be used for representing a sequence of observations [1] and these observations can be either discrete for example a sequence of DNA or continuous such as the aligned tag density profile of a ChIP-seq experiment, and can either be time dependent or independent. HMMs have existed since the 1960s, when the initial theories were published by Baum and his coworkers in 1967 [2] and have since gained traction in modeling various systems both natural and man-made [3]. Although initial applications were mostly in the field of engineering to model real-world processes like speech recognition, signals, and temperature variations [3–5] HMMs have found wide-ranging use in biology. One of the seminal works of HMMs in biology involves DNA and protein sequence analysis [6], and later they have been applied extensively in various areas of research such as protein topology prediction, especially transmembrane proteins [7], protein molecular dynamics [8], protein secondary structure prediction [9], gene identification in prokaryotes and eukaryotes [10], study of SNPs [11] and copy number variations in genomes [12], identification of transcription factor binding sites [13] and motifs, study of signals from spectroscopic measurements [14], study of behavioral patterns [15], and animal movements [16]. It can be said that this is one of the most extensively applied statistical models in biology with a long history of success. This chapter is aimed to give you a gentle introduction to hidden Markov models and introduce the possibilities of using HMM to model both real-world and man-made systems.

2.1 A Real-Time Example of HMM

Let us assume that you are confined to a closed room with no information on how the weather is outside. You receive postal mails through your mail slot in the front door every day of the

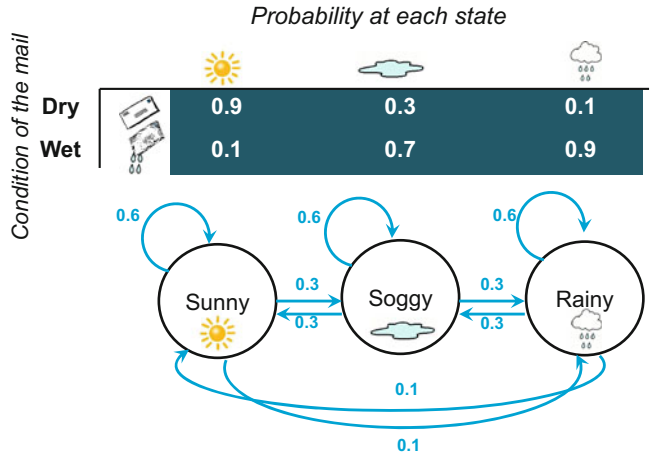


Fig. 1 Illustration of a three state HMM. The weather of a given day is represented as three states $S = \{\text{sunny, soggy, rainy}\}$. The states are represented as *circles*. The transitions from one state to another are given as *blue arrows* and they represent the transition probability matrix *A*. The transition probabilities are given alongside the arrows and for any given state the sum of probabilities arising from it or into it must be 1. The dampness of the mails that are used as the observation symbols $V = \{\text{dry, wet}\}$ form the rows of the matrix provided in *blue shaded matrix* above the state diagram. The matrix denotes the emission probability matrix *B*

week. The only way by which you can guess the weather outside is by looking at the state of the mails you get. For example it is natural to expect your mails to be dry on a dry and sunny day, while you can expect your mails to be dripping wet on a rainy day. Suppose, your aim is to model the weather as being sunny, soggy, or rainy based on the condition of your mail: you can use a three-state HMM model. Here the weather forms the states "sunny," "soggy," or "rainy," or $S = \{\text{sunny, soggy, rainy}\}$ is a set of "hidden" discrete random variables that can take $N = 3$ distinct values (Fig. 1, circles). The reason why the states are described as hidden is because the observer is unaware of the nature of the states, which in this example arises from the fact that you are not allowed to leave the room.

2.2 Markov Models

A stochastic system is said to be a Markov process if the next future state is dependant solely on the present state. Applying this to the above example it translates to the next day's weather dependant only on today's weather. With the states hidden, the above model is therefore a hidden Markov model of the first order. A second order HMM assumes the present state to be dependent on the two previous states, and the third order HMM considers three previous states and so on. However, most of the time when HMMs are mentioned it is the first order HMM. The assumption of a Markovian process is a very important one because of which we achieve a great degree of simplicity, but also lose important information that

may otherwise be relevant to building a good fitting model. In the example, we can easily argue that the next day's weather may be better predicted from the previous one week's weather pattern. However because of the assumption of Markov process we can reduce the complexity of the model into a manageable set of model parameters.

2.3 Transition Probabilities (A)

One of the three primary parameters of any given HMM is the transition probability matrix. The probability of the next state given the present state, i.e., the Markov process is given as $P(X_{t+1}|X_t)$ where t is the present time point and $t + 1$ is the next time point. For example in our weather HMM it can denote the probability of tomorrow being "rainy" if today was "sunny" which from Fig. 1 is 0.1, or 10 % chance of rain the next day (Fig. 1, blue arrows). These probabilities can be obtained by, for instance from the data of weather patterns for the past two years. Since it is a first order Markov process we can conveniently represent the probability of all possible transitions from one state to another as a matrix A, which is termed as the "transition matrix." For the above example,

$$A = \begin{pmatrix} 0.6 & 0.3 & 0.1 \\ 0.3 & 0.6 & 0.3 \\ 0.1 & 0.3 & 0.6 \end{pmatrix}$$

where each of the row or column represents the states of the HMM $S = \{\text{sunny, soggy, rainy}\}$.

2.4 Emission Probabilities (B)

Since the model is a HMM, the actual states are hidden from the observer. In our example weather model, the observer is confined to a room and is unaware of the actual weather outside and the only observable data is the dampness of the mails received each day. The observer is allowed to classify the mail as either being "Dry" or "Wet" ($V = \{\text{dry, wet}\}$ where $M = 2$) and can further proceed on to model the possible state of weather for that day. In this example it is intuitive to derive a conclusion that if the mails are dry the weather is sunny, if they are all wet then it is raining outside and if they are damp the weather may be soggy. In other words one can generate a probability of each type of observation (V) in each state and these probabilities are termed as "emission probabilities" and the resultant matrix is B (also see Fig. 1, blue shaded matrix). In the emission probability matrix given below, the rows correspond to condition of the mail (V) and the column corresponds to the three states (S) corresponding to the weather outside.

$$B = \begin{pmatrix} 0.9 & 0.3 & 0.1 \\ 0.1 & 0.7 & 0.9 \end{pmatrix}$$

3 Parameters Generally Defined in a HMM

Apart from a HMM defined by parameters A and B, there is also another parameter π which is the set of probabilities of starting at different states, i.e., when $t = 1$. π can be either uniform, random or any vector generated from prior knowledge. Therefore, any HMM (λ) can be defined as $\lambda = (A, B, \pi)$. There are also other related variables often defined in a HMM: $N =$ total number of states $S = \{s_1, s_2, s_3 \ldots s_N\}$, $M =$ total number of observation symbols $V = \{v_1, v_2, v_3 \ldots v_M\}$. When provided with an observation sequence, T is the total length of the sequence of observations $O = \{O_1, O_2, O_3 \ldots O_T\}$ and there exists a corresponding state sequence $X = \{X_1, X_2, X_3 \ldots X_T\}$. Therefore, $a_{ij} = P(X_{t+1} = j | X_t = i)$ represents the transition probability from state i to j where $i, j \in S$ and $b_j(k) = P(v_k|j)$ represents the emission probability of v_k in state j.

4 Statistical Theory of HMM

HMMs have a strong statistical basis and while using it in modeling most of the time we are interested in three statistical aspects of HMMs broadly termed as the "Three problems of HMM."

4.1 Evaluation: Probability of Observing a Sequence Given a Model

Given a model λ and an observation sequence O, our aim is to calculate the probability of the observing the sequence, i.e., $P(O|\lambda)$. This is a problem of evaluating the observed sequence when we know the parameters of the HMM. This helps us in finding the most optimal HMM if we have more than one HMM and also the observation sequence which is closest to a HMM if we are provided with a set of observation sequences. In the simplest form we can break down the evaluation of $P(O|\lambda)$ as follows. Given a state sequence X, the probability of jointly observing the observed sequence and the state sequence is given by $P(O, X|\lambda) = P(O|X, \lambda)P(X|\lambda)$, which is the product of the probability of the observation sequence given X, and the probability of the state sequence X given the model. The first term can be obtained from the emission matrix B, as $P(O|X, \lambda) = \prod_{t=1}^{T} b_j(O_t)$ where $X_t = S_j$ and the second term can be obtained from the transition matrix A as $P(X|\lambda) = \pi_{X_1} \prod_{t=2}^{T-1} a_{ij}$ where $X_t = S_i$ and $X_{t+1} = S_j$. We can then derive $P(O|\lambda) = \sum_{\forall X} P(O, X|\lambda)$, which is the summation of $P(O, X|\lambda)$ over all possible state paths X. The total number state paths increases quickly with the length (T) and therefore can become computationally expensive and not feasible depending on N and T. Fortunately there exists an algorithm called

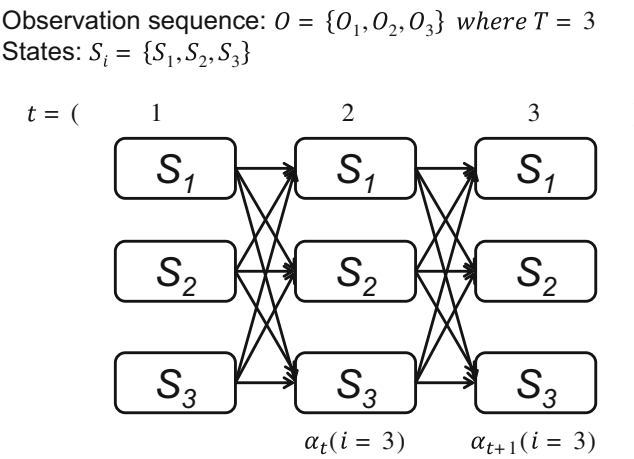


Fig. 2 Trellis diagram for a HMM model of three states, for $T = 3$. Trellis diagram for a three state HMM with an observed sequence O of length 3. *Arrows* are drawn between a state from *t* and all the states in $t + 1$. The distinct states are provided at each stage and $\alpha_t(i = 3)$ represent the partial probability $P(O_1 \ldots O_t, X_t = S_i|\lambda)$

forward–backward algorithm that can be used to obtain $P(O|\lambda)$ which reduces the computational burden.

4.1.1 Forward–Backward Procedure

The forward–backward procedure is better explained using a trellis diagram as illustrated in Fig. 2 which shows a sample trellis for a three-state HMM with $T = 3$. The procedure consists of two steps, the forward step and the backward step. In practice only the forward procedure will suffice for identification of $P(O|\lambda)$ but the procedure has its uses in addressing the other two problems of HMM as well. In the forward procedure, the aim is to obtain the partial probabilities for each state S_i, $1 \leq i \leq N$, for each time point t, $1 \leq t \leq T$. It should be noted that when $t = 1$, the transition probability is the starting probability π. The partial probability at t is given by,

$$\alpha_t(i) = P(O_1 O_2 \ldots O_t, X_t = S_i|\lambda)$$

which is the probability of observing the partial observed sequence until time t, when the state at t is S_i. $\alpha_t(i)a_{ij}$ is therefore the probability of observing $O_1, O_2 \ldots O_t$ and $X_t = S_i$ and $X_{t+1} = S_j$. Therefore $\alpha_{t+1}(i)$ can be obtained inductively as

$$\alpha_{t+1}(j) = \sum_{i=1}^{N} \alpha_t(i)a_{ij} \times b_j(O_{t+1})$$

$$P(O|\lambda) = \sum_{i=1}^{N} \alpha_T(i)$$

This method of calculation allows us to reduce the computation time as the probabilities are calculated from the partial probabilities from each state at each time point (Fig. 2). In a similar way, we can design the backward procedure as

$$\beta_t(i) = P(O_{t+1}O_{t+2}\ldots O_T|X_t = S_i, \lambda)$$

$$\beta_t(i) = \sum_{j=1}^{N} \beta_{t+1}(j)a_{ij} \times b_j(O_{t+1})$$

4.2 Estimation: Probability of a State Sequence Given an Observed Sequence

The most probable state sequence X for an observed sequence O can be obtained exhaustively by deriving all the possible state sequences, generating $P(X|O, \lambda)$ and finally obtaining the X for which this probability is maximum. This brute-force method may be viable for shorter O, but can quickly become computationally expensive as the length of O increases. Furthermore we can define most optimal X in various different ways. For example by using the emission probabilities for each state (B), we can find the state sequence whose individual states have the highest emission probability for the corresponding observed symbol. However, we may end with a highly unlikely state sequence because we are not considering the transition probabilities (A) in our calculation [3]. Most of the HMMs use the Viterbi algorithm to obtain the most probable state path for a given observation sequence [17]. This algorithm based on dynamic programming was first proposed by Andrew Viterbi in 1967 [18], is the most commonly used method for obtaining the maximum-likelihood state sequence that would result in a given observation sequence and the most likely path is called the "Viterbi path." This algorithm uses two variables δ, φ and involves three major steps. The first step is similar to the forward procedure mentioned above, except here we define $\delta_t(i)$ as the maximum partial probability instead of the summation of partial probabilities that we compute for the forward procedure. Therefore

$$\delta_{t+1}(i) = \left(\max_i \delta_t(i)a_{ij}\right) \times b_j(O_{t+1})$$

Also we define $\varphi_t(i)$ that allows us to keep track of the state path that gives the highest partial probability at time $t+1$, $\delta_{t+1}(i)$. The second step is to obtain the $\max_i \delta_T(i)$ and the corresponding $\varphi_T(i) = \text{argmax}_i\, \delta_T(i)$ by iterative calculation similar to the forward procedure. The third step employs backtracking using φ to obtain the best state sequence similar to dynamic programming.

4.3 Optimization: Maximize a Model for a Given Observed Sequence

More often we may not have a priori knowledge of the HMM parameters and may have to optimize them for a given observation sequence. The aim therefore is to optimize the model parameters such that $P(\lambda|O)$ is maximized and it can be obtained by using Expectation Maximization (EM) algorithms. Baum–Welch algorithm that is discussed below is the most commonly used EM for optimizing $\lambda = (A, B, \pi)$ [19]. Let us

consider $\xi_{ij}(t) = P\left(X_t = S_i, X_{t+1} = S_j \middle| O, \lambda\right)$ is the probability of observing states S_j at time $t+1$ and S_i at time t given the observed sequence, then using the forward–backward procedure

$$\xi_{ij}(t) = \frac{\alpha_t(i)a_{ij}b_j(O_{t+1})\beta_{t+1}(j)}{\sum_{i=1}^{N}\sum_{j=1}^{N}\alpha_t(i)a_{ij}b_j(O_{t+1})\beta_{t+1}(j)}$$

If $\gamma_t(i) = \frac{\alpha_t(i)\beta_t(j)}{P(O|\lambda)}$ is the probability of being in S_i at time t given the observed sequence, then

$$\gamma_t(i) = \sum_{i=1}^{N}\xi_{ij}(t)$$

For a HMM with new updated parameters $\widehat{\lambda} = \left(\widehat{A}, \widehat{B}, \widehat{\pi}\right)$

$$\widehat{\pi} = \gamma_{t=1}(i)$$

$$E_i = \sum_{t=1}^{T-1}\gamma_t(i)$$

$$E_{ij} = \sum_{t=1}^{T-1}\xi_{ij}(t)$$

$$\widehat{a}_{ij} = \frac{E_{ij}}{E_i}$$

$$\widehat{b}_j(k) = \frac{O_t = v_k \sum_{t=1}^{T}\gamma_t(j)}{\sum_{t=1}^{T}\gamma_t(j)}$$

Here E_i is the expected number of transitions from S_i and E_{ij} is the expected number of transitions from S_i to S_j. It can be tested if the new likelihood $P\left(O|\widehat{\lambda}\right) > P(O|\lambda)$ and therefore by iteratively replacing λ with $\widehat{\lambda}$ if the above condition is true, we can converge at a model with the maximum likelihood estimate of the parameters. We should be aware that we may end up with an overfitted model and that EM most of the time provide us with local maximum and not the global maximum [2, 19].

4.4 Sequence Analysis Using HMM

In this section we see how HMMs can be used in sequence analysis. One of the classical applications of HMMs in biology is to study homology of protein sequences [20, 21]. Proteins are composed of amino acids and are products of genes that control cellular functions. Due to the chemical properties of amino acids proteins are shown to have specific structures and their structural integrity is essential for them to perform their functions. It is now established that proteins with similar amino acid sequences have similar structures and functions. Newly identified proteins can be compared with existing proteins in order to infer their structural and

functional properties by aligning them to known families of proteins. Here we try to build a HMM model to one such alignment and we will be using a toy example of only seven types of amino acids in the alignment for clarity, however in reality all the 20 amino acids are used. Figure 3a shows a toy alignment with nine alignment positions of five sequences. The alignment has conserved hydrophobic blocks of residues comprising positions 1, 2, 3 and positions 7, 8, 9. There is a less conserved region in the middle with inserts and hydrophilic residues at positions 4, 5, 6. Our aim is to construct a HMM, that can capture this alignment profile. The straight forward HMM would be to use a nine-state model where each alignment position is represented as a state. While this HMM is sufficient, highly conserved positions are more important than variable regions. Therefore to create a more meaningful model, we can represent each conserved position (positions 1, 2, 3, 7, 8, 9) as a state and integrate all the least conserved positions into a single state. It should be noted that there are insertions at positions 4, 5, and 6 and our model should reflect this [6]. For each state in S_1, S_2, S_3, S_5, S_6 and S_7 we can calculate the emission probabilities that populate matrix B as the frequency of occurrence of each amino acid in the corresponding alignment position. For state S_4, we can calculate the emission probabilities in a way similar to other positions as the fractional occurrence of each amino acid in the inserted positions. For example, the total number of insert positions in S_4 is 10 and the total number of times Arginine (R) has occurred is 3. Therefore $b_{j=S_4}(k = "R") = 3/10$. All the transition probabilities except those involving S_4 is 1 because each state has to transition to the next state. In case of transitions to S_4, there are three sequences that has insertions, therefore $a_{i=S_3, j=S_4} = 3/5$, and since all these sequences has to go to S_5 at some point, $a_{i=S_4, j=S_5} = 3/5$. Because of the property of A that the total incident probability should be one, $a_{i=S_4, j=S_4} = 1 - 3/5$. π can be either a random probability matrix, a uniform probability matrix or generated by computing the total number of times a given amino acid is observed in the alignment. Given a query sequence say sequence 5 "VAAGEEVAL," find the probability of a sequence given the model as

$$P(\text{seq5}|\lambda) = \pi_1 b_1(O_1 = "V")a_{12} b_2(O_2 = "A")a_{23} b_3(O_3 = "A") \ldots a_{67} b_7(O_9 = "L")$$

You can also represent such probabilities as log-odds score as $-\log\left(\frac{P(\text{seq5}|\lambda)}{0.05^T}\right)$ which is the same as dividing each emission probability by the probability of observing that amino acids by chance. Here a uniform probability of observing each amino acid by chance $(1/20)$ is used. You can substitute it with values that can be computed from a consensus set of proteins available in a non-redundant database. HMMs that are similar in approach to the

(A)

Alignment positions

Sequences	1	2	3	4	5	6	7	8	9
1	V	A	L	G	R	D	V	A	L
2	V	V	L	V	D	R	V	A	A
3	V	A	L	.	.	R	L	A	A
4	A	A	V	.	.	.	V	V	L
5	V	A	A	G	E	E	V	A	L

(B)

States (S)

Emission probabilities	1	2	3	4	5	6	7	8	9
G	0	0	0	0.4	0	0	0	0	0
A	0.2	0.8	0.2	0	0	0	0	0.8	0.4
V	0.8	0.2	0.2	0.2	0	0	0.8	0.2	0
L	0	0	0.6	0	0	0	0.2	0	0.6
E	0	0	0	0	0.2	0.2	0	0	0
D	0	0	0	0	0.2	0.2	0	0	0
R	0	0	0	0	0.2	0.4	0	0	0

(C)

G	0	0	0	0.2	0	0	0
A	0.2	0.8	0.2	0	0	0.8	0.4
V	0.8	0.2	0.2	0.1	0.8	0.2	0
L	0	0	0.6	0	0.2	0	0.6
E	0	0	0	0.2	0	0	0
D	0	0	0	0.2	0	0	0
R	0	0	0	0.3	0	0	0

(D)

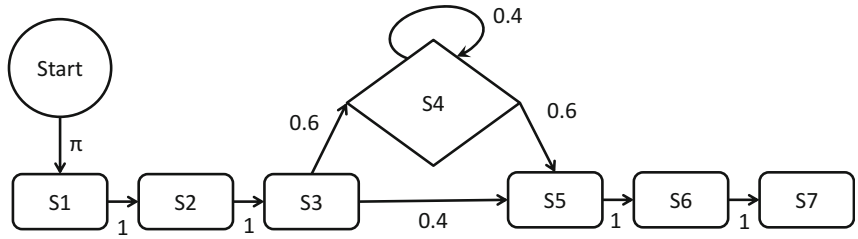


Fig. 3 Sequence analysis using HMM. (**a**) A toy multiple sequence alignment, where each row represents an amino acid sequence and each column represents alignment positions. The shaded columns correspond to conserved hydrophobic regions. (**b**) Emission probability matrix that represent the probability of observing an aminoacid at each alignment position. (**c**) Emission probability matrix for the HMM given in (**d**). (**d**) HMM model for the alignment given in (**a**). Here the each conserved alignment position is considered as a state and the less conserved positions are grouped together (diamond shaped state, S4). The illustration of the seven-state HMM is similar to Fig. 1

above mentioned toy example are used extensively in identifying sequences or family of sequences in existing databases such as Pfam, that are homologous to our sequence of interest. HMMPfam is one

such example, which contains a database of HMMs for each family of homologous proteins [22, 23]. When queried with a sequence of our interest, it is designed to generate scores (similar to the log-odds scores) that quantifies the similarity of the query with each of HMMs in the database, thereby allowing us to identify which family our query sequence is mostly likely to belong to. The power of such HMM based sequence profiling is that it can identify remote homologs compared to conventional methods like BLAST.

It is a common practice to use logarithms of the probabilities because larger O can generate very small probabilities and many computer languages have a limit on floating point integers. Furthermore the use of logarithms converts multiplications of probabilities to additions simplifying the computational burden. Also, many HMMs use a pseudo-count, a small value to add to the probabilities since they use logarithms for computational efficiency and $\log(0)$ which is undefined, causes errors.

As with any statistical model HMMs have their disadvantages despite a broad range of usability. As mentioned earlier, the most important assumption that the next state is dependant only on the present state makes HMMs statistically manageable but also imposes restrictions on the power of the model. We may often end up with a system where the state may be dependent on multiple past states not just one. Also, the emission probabilities of a state remain constant irrespective of the previous or the future states which may not necessarily be true for many systems. Finally, HMMs can be used to model only sequences of observations.

In this chapter we provide a basic introduction of HMMs and the theory behind it. It is evident that HMMs are flexible enough that researchers have adjusted the underlying statistics and its parameters to best describe the system of their study. In this book instead of focussing on HMM's classical works we aim at exposing you to the wider scope of HMMs in biology starting from nano-scale biomolecular dynamics to large-scale animal behavioral dynamics.

References

1. Ghahramani Z, (2001) An introduction to hidden Markov models and Bayesian networks. Int J Pattern Recognit Artif Intell 15(1): 9–42. doi: 10.1142/S0218001401000836
2. Baum LE, Petrie T (1966) Statistical inference for probabilistic functions of finite state Markov chains. Ann Math Statist 37 (6):1554–1563. doi:10.1214/aoms/1177699147
3. Rabiner LR (1989) A tutorial on hidden Markov models and selected applications in speech recognition. Proc IEEE 77(2):257–286. doi:10.1109/5.18626
4. Schuster-Bockler B, Bateman A (2007) An introduction to hidden Markov models. Current protocols in bioinformatics/editoral board, Andreas D. Baxevanis [et al] Appendix 3:Appendix 3A. doi:10.1002/0471250953.bia03as18
5. Yoon BJ (2009) Hidden Markov models and their applications in biological sequence analysis. Curr Genomics 10(6):402–415

6. Durbin R, Eddy S, Krogh A, Mitchison G (2006) Biological sequence analysis. doi:citeulike-article-id:3346650
7. Krogh A, Larsson B, von Heijne G, Sonnhammer EL (2001) Predicting transmembrane protein topology with a hidden Markov model: application to complete genomes. J Mol Biol 305(3):567–580. doi:10.1006/jmbi.2000.4315
8. Shukla D, Hernandez CX, Weber JK, Pande VS (2015) Markov state models provide insights into dynamic modulation of protein function. Acc Chem Res 48(2):414–422. doi:10.1021/ar5002999
9. Won KJ, Hamelryck T, Pruegel-Bennett A, Krogh A (2007) An evolutionary method for learning HMM structure: prediction of protein secondary structure. BMC Bioinformatics 8:357. doi:10.1186/1471-2105-8-357
10. Stanke M, Waack S (2003) Gene prediction with a hidden Markov model and a new intron submodel. Bioinformatics 19:Ii215–Ii225. doi:10.1093/bioinformatics/btg1080
11. Bian JW, Liu CL, Wang HY, Xing J, Kachroo P, Zhou XB (2013) SNVHMM: predicting single nucleotide variants from next generation sequencing. BMC Bioinformatics 14:225. doi:10.1186/1471-2105-14-225
12. Seiser EL, Innocenti F (2014) Hidden Markov model-Based CNV detection algorithms for illumina genotyping microarrays. Cancer Inform 13(Suppl 7):77–83. doi:10.4137/CIN.S16345
13. Bao Y, Vinciotti V, Wit E, t Hoen PA (2014) Joint modeling of ChIP-seq data via a Markov random field model. Biostatistics 15 (2):296–310. doi:10.1093/biostatistics/kxt047
14. Okamoto K, Sako Y (2012) Variational Bayes analysis of a photon-based hidden Markov model for single-molecule FRET trajectories. Biophys J 103(6):1315–1324. doi:10.1016/j.bpj.2012.07.047
15. Arakawa T, Tanave A, Ikeuchi S, Takahashi A, Kakihara S, Kimura S, Sugimoto H, Asada N, Shiroishi T, Tomihara K, Tsuchiya T, Koide T (2014) A male-specific QTL for social interaction behavior in mice mapped with automated pattern detection by a hidden Markov model incorporated into newly developed freeware. J Neurosci Methods 234:127–134. doi:10.1016/j.jneumeth.2014.04.012
16. Karg M, Venture G, Hoey J, Kulic D (2014) Human movement analysis as a measure for fatigue: a hidden Markov-based approach. IEEE Trans Neural Syst Rehabil Eng 22 (3):470–481. doi:10.1109/TNSRE.2013.2291327
17. Jr. GDF (2005) The Viterbi algorithm: a personal history. CoRR abs/cs/0504020
18. Viterbi A (1967) Error bounds for convolutional codes and an asymptotically optimum decoding algorithm. IEEE Trans Inform Theory 13(2):260–269. doi:10.1109/tit.1967.1054010
19. Welch LR (2003) Hidden Markov models and the Baum-Welch algorithm. IEEE Inform Theory Soc Newsl 53(4):10–13
20. Eddy SR (1996) Hidden markov models. Curr Opin Struct Biol 6(3):361–365
21. Eddy SR (1998) Profile hidden Markov models. Bioinformatics 14(9):755–763
22. Finn RD, Clements J, Arndt W, Miller BL, Wheeler TJ, Schreiber F, Bateman A, Eddy SR (2015) HMMER web server: 2015 update. Nucleic Acids Res 43(W1):W30–W38. doi:10.1093/nar/gkv397
23. Finn RD, Clements J, Eddy SR (2011) HMMER web server: interactive sequence similarity searching. Nucleic Acids Res 39(Web Server issue):W29–W37. doi:10.1093/nar/gkr367

Chapter 2

HMMs in Protein Fold Classification

Christos Lampros, Costas Papaloukas, Themis Exarchos, and Dimitrios I. Fotiadis

Abstract

The limitation of most HMMs is their inherent high dimensionality. Therefore we developed several variations of low complexity models that can be applied even to protein families with a few members. In this chapter we present these variations. All of them include the use of a hidden Markov model (HMM), with a small number of states (called reduced state-space HMM), which is trained with both amino acid sequence and secondary structure of proteins whose 3D structure is known and it is used for protein fold classification. We used data from Protein Data Bank and annotation from SCOP database for training and evaluation of the proposed HMM variations for a number of protein folds that belong to major structural classes. Results indicate that the variations have similar performance, or even better in some cases, on classifying proteins than SAM, which is a widely used HMM-based method for protein classification. The major advantage of the proposed variations is that we employed a small number of states and the algorithms used for training and scoring are of low complexity and thus relatively fast. The main variations examined include a version of the reduced state-space HMM with seven states (7-HMM), a version of the reduced state-space HMM with three states (3-HMM) and an optimized version of the reduced state-space HMM with three states, where an optimization process is applied to its scores (optimized 3-HMM).

Keywords Fold classification, Hidden Markov model, Optimization

1 Introduction

Today we possess a huge amount of sequence information concerning proteins. The problem of determining the function of these proteins requires the knowledge of their structures [1–3]. Many experimental methods for structure determination exist, such as nuclear magnetic resonance (NMR) spectroscopy [4] and X-ray crystallography. Nevertheless, the exact structure determination remains expensive and time consuming. On the other hand, there is an indirect way to make predictions for both structural and functional attributes of a protein with unknown structure. Its amino acid sequence can be compared with proteins in annotated databases for which the 3D structure is known [1], since similarity

David R. Westhead and M.S. Vijayabaskar (eds.), *Hidden Markov Models: Methods and Protocols*, Methods in Molecular Biology, vol. 1552, DOI 10.1007/978-1-4939-6753-7_2,

© Springer Science+Business Media LLC 2017

in the structure of proteins typically results in similarity of their function.

Moreover, proteins have similar structure if they share a common fold. Specifically, if proteins belong to the same fold category they have the same major secondary structures in the same arrangement and with the same topological connections. The fold of two common proteins can be the same even when there is a very low sequence similarity among them [5, 6]. The task of classifying the proteins into the appropriate fold category is called fold classification [7].

Fold classification can directly lead to the determination of the tertiary (3D) structure of a protein. There are two main methodological approaches in tertiary structure prediction, the template free methods [8–12] and the template-based methods [6, 13–29]. Template free methods try to directly approach the specific 3D structure of a protein using energy minimization based on physical principles rather than on previously identified structures. Template-based methods use already known structures (templates) as candidate structures of the protein. They are further divided into two categories. The first category consists of the 3D-1D fold recognition methods (otherwise called threading methods), which use a scoring function that assesses the compatibility of the amino acid sequence of unknown structure to already known structures [13–18]. The second category consists of the comparative modeling methods, where sequence alignment is used to discern the relationship between target sequence and template [6, 19–29].

Comparative modeling approaches are also known as sequence-based approaches.

There are many different machine learning methods that have been used for that purpose, such as HMMs [6, 19–21], genetic algorithms [22], artificial neural networks [23], support vector machinesSupport Vector Machines (SVMs) [24, 25], data mining [26, 27], similarity networks [28], and FRAN and RBFRadial basis function (RBF) networks [29].

Among these, HMMs have been widely used and also achieve high performance. HMMs have initially been applied for protein fold classification in the context of HMMER [19] and of the sequence alignment and modeling (SAM) system [20]. Moreover, secondary structure information was incorporated into HMMs [6] to increase their fold recognition performance. A similar approach was also used together with the evaluation of several backbone geometry alphabets [21].

Nevertheless, the most important disadvantage of HMMs was the use of models with many states, which demand much data and significant computational effort for training [30]. Thus, it is necessary to reduce the parameters of HMMs and also employ HMMs that can be trained even with a small amount of data, while their performance in fold classification is maintained. For this purpose

we introduced different variations related to a HMM with a small number of states (reduced state-space HMM or simply reduced HMM), which is trained with both amino acid sequence and secondary structure for proteins whose 3D structure is known and it is used for protein fold classification. In this chapter we discuss these HMM models and a comparative analysis of these models in detail.

2 7-HMM

In the seven-state HMM model we propose the use of a simple architecture with a small number of states and a training algorithm with low complexity. We also incorporate secondary structure information into the model, which is employed in such a manner that enables us to use the low complexity algorithm and also improves its performance. The emission states are equal in number to the total of possible formations of secondary structure like helix (H), β strands (B), and loops (S) (Fig. 1), and there is also a begin state. The model is trained for each candidate fold. Its major advantage is that the computational complexity of the training procedure of the model is significantly smaller than the complexity of other current methods based on larger HMMs [30].

The topology of the model employs the mathematical background of a typical HMM. It models a sequence of amino acids through a hypothesized (hidden) procedure. The model contains a number of states S (nodes) and a number of possible transitions T (arrows) between them (Fig. 1). Every emission state emits an amino acid based upon a set of emission probabilities. Then transition to some other emission state takes place with a probability

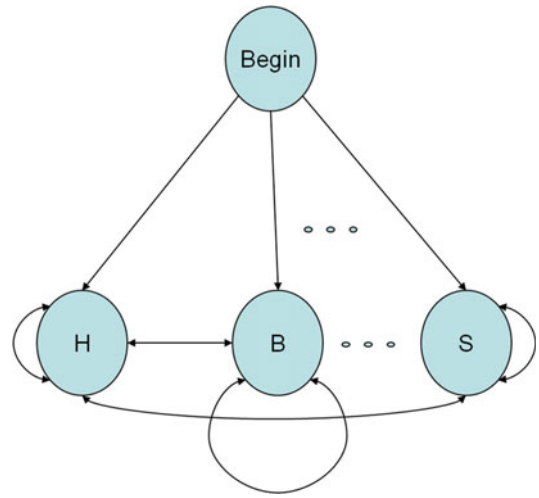


Fig. 1 Topology of the reduced state-space HMM. H, B, . . ., S are the letters of DSSP secondary structure alphabet

depending on the previous state. This process continues until all of the amino acids of the protein sequence are emitted. There are also transition probabilities from the begin state, where the process starts, to each emission state. The sum of these probabilities is equal to one and so does the sum of emission probabilities of all possible amino acids in each state and the sum of transition probabilities from each state to each next possible state. Another basic feature of the model is the Markov property, which assumes that the current state S_t and all future states are independent of all the states prior to the previous state S_{t-1} [31].

The most important characteristic of the reduced state–space HMM, which makes it different from other HMMs, is that it incorporates the secondary structure information in such a manner that each state of the model corresponds to a different type of secondary structure. The correct assignment of the protein to its structural group will be done not only with its primary structure information but also with the use of its secondary structure information. We use secondary structure sequences which are incorporated in the context of our HMM as hidden state sequences. This allows the benefit of employing a HMM with a number of states equal to the number of the different letters ({H, B, E, G, I, T, S}, see Table 1) in the definition of secondary structure of proteins (DSSP) alphabet [32], which means a small number of states. These letters represent the possible secondary structure formations of each amino acid. Moreover, the sequence of states for each amino acid sequence produced by the model is known and that fact enables us to employ a low complexity training algorithm based on likelihood maximization [31]. So we can avoid the complicated Baum–Welch algorithm [33], which is an iterative learning process commonly used in other HMMs.

Table 1
Correspondence between letters of the DSSP alphabet and the letters of the DSSP-EHL alphabet

DSSP	Type	Corresponding letter of the DSSP-EHL alphabet
H	Alpha-helix	H
G	3_{10}-helix	H
I	Π-helix	H
E	Extended(β-strand)	E
B	Residue in isolated β-bridge	E
T	Turn	L
S	Bend	L

All possible transitions between the states of the model are permitted. There is also one to one correspondence between the amino acids and the secondary structure formations in the training set, which means that for each state we should estimate the distribution over all possible amino acids. There are 21 possible amino acids which are the variables in each distribution. Among them, 20 correspond to the standard amino acids that exist and one more symbolizes amino acids of unknown origin. The total number of the model parameters is 203, which is the sum of 7×21 parameters for the possible emissions, 7×7 for the possible transitions between emission states and 1×7 for the transitions from the begin state [30].

The likelihood maximization algorithm is used for the learning procedure of the 7-HMM. By implementing this algorithm, the emission and transition parameters are estimated in one step with the use of the maximum likelihood estimators. The estimators are calculated by the following equations, which are the estimation equations in the HMMs when the state sequences are known [31]:

$$t_{kl} = \frac{T_{kl}}{\sum_{l'} T_{kl'}} \tag{1}$$

and

$$e_k(a) = \frac{E_k(a)}{\sum_{a'} E_k\left(a'\right)}, \tag{2}$$

where t_{kl} is the transition probability from state k to another state l and $e_k(a)$ the emission probability of the amino acid a in the state k. Also T_{kl} is the number of times that the transition from k to l takes place and $E_k(a)$ is the number of times the emission of α from k takes place in the training set of sequences. Whenever there is a state k that has never been used in the training set, then the estimation equations are undefined for that state (numerator and denominator will be equal to zero). In order to avoid this problem it is better to add predetermined pseudocounts to the T_{kl} and $E_k(\alpha)$ before using Eqs. (1) and (2), thus we have:

$$T_{kl} = (\text{number of transitions } k \text{ to } l \text{ in training data}) + p_{kl}, \tag{3}$$

$$E_k(a) = (\text{number of emissions of } \alpha \text{ from } k) + p_k(a). \tag{4}$$

The pseudocounts p_{kl} and $p_k(a)$ should reflect our prior beliefs about the probability values. In our case we do not actually use any prior knowledge, which means that $p_{kl} = p_k(a) = 1$. This practically means that both the prior distribution of amino acids in each state and the prior distribution of transitions from each state are considered to be

the uniform distributions. Thus, the use of pseudocounts here simply aims to avoid definition problems and does not include the incorporation of some specific prior knowledge [30].

The posterior probability scores are used for assessing 7-HMM and they are otherwise called log-likelihood scores. These scores are logarithmic probabilities of a test sequence, given the model of each candidate fold. The test sequence is assigned to that fold whose model gives the maximum posterior probability score compared to the scores produced from all the other models of the candidate folds. The sequences used for the assessment are divided into training and test sets. The test set contains only amino acid sequences, as all other forms of data related to the structure of a protein are considered unknown. Moreover, the likelihood score for a sequence against a model is divided with the score of that sequence against the so-called *null* model. The *null* model is based on the assumption that the amino acids are statistically independent at each position. It also gives fixed emission probabilities based on the uniform distribution over the possible amino acids. Therefore, the log-likelihood score of a sequence against the null model is given as:

$$score(x_i) = \log_z \frac{P_m(x_i)}{P_\emptyset(x_i)}, \tag{5}$$

where $P_m(x_i)$ corresponds to the probability that a sequence x_i has been produced by model m and $P_\emptyset(x_i)$ to the probability that the same sequence has been produced by the *null* model. We have chosen this type of scoring in order to limit the effect of length variations among sequences. The criterion for selecting the fold category, where a particular protein most possibly belongs, is to select the model that gives the highest posterior probability score for that protein's amino acid sequence. Thus, the fold selected as the best classification for protein x_i would be model m_i for which $P_{m_i}(x_i) > P_{m_j}(x_i)$ for all $i \neq j$, or equivalently $score_{m_i}(x_i) > score_{m_j}(x_i)$. The posterior probability scores correspond to the log-likelihood scores against the *null* model and are calculated with the use of the forward algorithm [31].

The forward algorithm is necessary for calculating the probability of a protein sequence x given the model, when we do not know the exact path π that produced the sequence. That probability is given by the following equation:

$$P(x) = \Sigma_\pi P(x, \pi). \tag{6}$$

If the number of states is K and the length of the test sequence is N, the time complexity of the forward algorithm is $O(K^2 N)$. It is also necessary to use logarithms in order to avoid underflow problems, because such problems appear when there is the need to calculate a long product of probabilities.

3 3-HMM

In the 3-HMM version of the model we introduce specific improvements in the reduced state-space model which lead to a substantial increase in its ability to classify proteins in the correct fold category. In the first improvement we change the topology of the model while in the second the test proteins are scored against the 3-HMM in a different manner [34].

More specifically, in the first improvement we use a different alphabet in order to encode the secondary structure of the proteins. The different possible secondary structure formations of each amino acid are represented by the three-letter alphabet DSSP-EHL instead of the seven-letter DSSP alphabet. As it is shown in [21], the DSSP-EHL alphabet is a reduced representation of the DSSP alphabet. Specifically, H, G and I correspond to H, E and B to E, while T and S to L. Those amino acids which were considered of unknown structure in the DSSP representation are considered as loops (L) in the DSSP-EHL representation. The correspondence of letters between the two alphabets is shown in Table 1. So, now we employ three states in the model that correspond to the three different possible formations of the underlying secondary structure that each amino acid may have according to the DSSP-EHL alphabet (Fig. 2).

In the 3-HMM model there are 3×21 emission parameters, 3×3 transition parameters between the states and three parameters for the transitions from the starting state. Therefore, the total number of parameters is 75, which is much less than the 203 parameters which were used in the 7-HMM. In the first

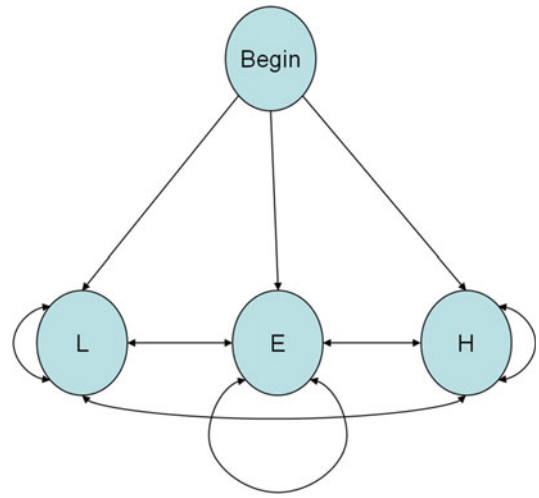


Fig. 2 Topology of the HMM with three states. Each state corresponds to one of the letters of the DSSP-EHL alphabet

improvement the training and scoring procedures of the 3-HMM model are the same with those of the 7-HMM.

In the second improvement, we maintain the model with the three states for training, but we use a different way of scoring the test proteins against the model. Our goal is to use also the secondary structure information of the test set proteins, which is not possible when we employ the forward algorithm. When the forward algorithm is used for scoring, the probabilities of all possible paths of the amino acid sequence through the model are added. Alternatively, we can compute the probability across the sequence of states (path) that corresponds to the secondary sequence of each protein. The use of the forward algorithm for scoring is necessary only when the secondary structure of the test protein is unknown. But if the secondary sequence is known, the path that produces the amino acid sequence through the model is also known, so the calculation of the score of each test protein becomes computationally less expensive. So this time we use the secondary sequence in scoring in addition to the primary sequence of test proteins and at the same time scoring becomes simpler [34].

Specifically, we avoid the iterative procedure of the forward algorithm in scoring each amino acid sequence. Here the path π^* of the amino acid sequence of the model is known, so the probability of the protein sequence x is:

$$P(x) = P(x, \pi^*). \tag{7}$$

Therefore, one step is needed for the calculations and the complexity depends only on the length of the sequence. Thus the time complexity is $O(N)$.

4 Optimized 3-HMM

We apply the following optimization approach to the scores of the 3-HMM. First, the scores of the model are normalized between 0 and 1 for each protein classification. Then, a set of parameters are added to the model, each one as a multiplier to the scores of each category (fold). These parameters are initially set to 1, which corresponds to the actual scores calculated using the 3-HMM. The parameters are subsequently introduced in the following optimization function:

$$F(D, p) = size(D) - trace(CM), \tag{8}$$

where D is the training dataset of proteins, p are the parameters, $size(D)$ is the number of training proteins and $trace(CM)$ is the correct predictions of the model, measured as the sum of the diagonal elements of the produced confusion matrix CM created for the training set D. Therefore Eq. (8) can be formulated as an

optimization problem. Different values for the parameters have an impact to the value of the optimization function, since they affect the confusion matrix. The result of the optimization procedure is the optimal set of parameters p^* [7].

The minimum value of F is 0 and this value is obtained if all proteins are correctly classified. Thus, the minimization of the above function increases the accuracy of the model by optimizing the parameters for the folds. The local optimization strategy was chosen, since an initial point is available (values of all parameters are equal to 1, $p = 1$) while the analytical calculation of the derivatives of the objective function is computationally expensive. Based on the above, we employed the Nelder–Mead simplex search method [35] to solve the optimization problem. This is a direct search method for multidimensional unconstrained minimization which does not use numerical or analytic gradients.

The whole procedure is depicted in Fig. 3. The scores for each fold and for each protein of the training set were entered into the optimization procedure, along with the initial parameters (which are all set equal to 1). The optimization procedure identifies the optimal parameters, which are multiplied with the scores of the test set proteins against the 3-HMM. The final scores were used for the fold classification of the test proteins. It should be noted that the training proteins correspond to the primary and the true secondary structure of the proteins, while the test proteins correspond to the primary and the predicted secondary structure of the test proteins.

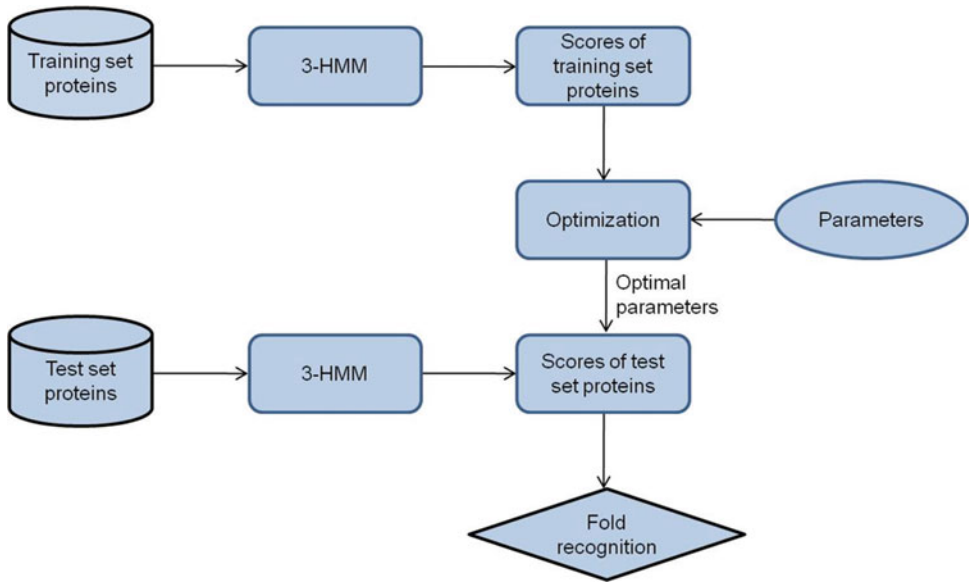


Fig. 3 The optimization procedure applied to the 3-HMM for the identification of the optimal parameters for fold classification

5 Results and Discussion

5.1 7-HMM

The 7-HMM was assessed with the ASTRAL40 [36] 1.69 dataset, where only proteins with less than 40 % similarity are included. The primary and secondary structure information was extracted from the PDB database [37]. Moreover, the 34 most populated SCOP [38] folds (those with at least 30 proteins in this case) of the four major structural classes were used for the fold recognition experiment [30]. In total, 1513 proteins were used for training and 758 for testing.

Two experiments took place. In the first experiment, 34 reduced HMMs were trained for each one of the 34 most populated folds. The test sequences were scored against all folds of every class and the prediction accuracy was calculated for all folds. The prediction accuracy is the number of test proteins uniquely recognized as belonging to a specific fold divided by the total number of test proteins belonging to that fold. In the second experiment, 1513 reduced HMMs were trained for each one of the training sequences of all folds. In that case the prediction accuracy is calculated in the same way, but the test sequences are first scored against the models of all proteins of every fold. Then each protein was classified to the fold whose member was the protein the model of which gave the maximum probability score among all 1513 models of sequences used for training. The results of the experiments are shown in Table 2.

The 7-HMM performance deteriorates slightly when different models are trained for every sequence of the training set (it falls from 17.9 % to 16.1 %), indicating the ability of 7-HMM to learn families with fewer members. These performances were also compared with SAM, which is considered as the most effective method

Table 2
Comparison of 7-HMM for the 34 SCOP folds when different models were trained for each fold and when different models were trained for each sequence in the training set

Folds[a]	7-HMM accuracy (fold models)		7-HMM accuracy (sequence models)	
Overall class A	31/131	23.7 %	18/131	13.7 %
Overall class B	61/203	30.1 %	33/203	16.3 %
Overall class C	34/329	10.3 %	61/329	18.5 %
Overall class D	10/95	10.5 %	10/95	10.5 %
Overall	136/758	17.9 %	122/758	16.1 %

[a]A = all alpha proteins; B = all beta proteins; C = alpha and beta (a/b) proteins (mainly parallel beta sheets); D = alpha and beta ($a + b$) proteins (mainly antiparallel beta sheets)

that employs HMM for protein classification [20, 39]. In the first case the performance of SAM was 23.5 % while the second case it dropped to 11.9 % [30]. These results indicate that models like SAM, cannot perform adequately with insufficient data, in contrast to the proposed 7-HMM architecture.

5.2 3-HMM

The sequences used in the assessment of the improvements introduced with the 3-HMM come from the newer ASTRAL40 SCOP 1.71 dataset, where again no proteins with more than 40 % similarity are included [34]. Moreover, the SCOP folds of classes A, B, C, and D, and specifically those which have at least 30 members, were included. There were 38 such folds. In total, the training set contained 1727 proteins and the test set 864.

Predicted secondary structure information was also needed for our dataset. More specifically, it was necessary in the test set so that we would be able to score the test proteins against the model not only with their amino acid sequence but also using their secondary structure sequence. The predictions were obtained from PSIPRED [40], which provides reliability indices for all three secondary states {E,H,L} for each residue in the query sequence.

The 3-HMM model was compared with the 7-HMM model in order to assess the novelties introduced either in its structure (first improvement) or in the way of scoring (second improvement). Four experiments took place for this purpose using the same dataset which includes data from the 38 folds. The comparison among the results of those four experiments is shown in Table 3.

Table 3
Comparison among the 7-HMM, the 3-HMM when only primary structure is used in scoring, the 3-HMM when predicted secondary structure is also used in scoring and the 3-HMM when true secondary structure is used for the 38 SCOP folds

Folds	7-HMM	3-HMM (primary only) (o primary) recognition accuracy	3-HMM (predicted secondary)	3-HMM (true secondary)
Overall class A	39/140(27.9 %)	39/140(27.9 %)	54/140(38.6 %)	50/140(35.7 %)
Overall class B	70/243(28.8 %)	74/243(30.5 %)	94/243(38.7 %)	115/243 (47.3 %)
Overall class C	38/363(10.5 %)	53/363(14.6 %)	80/363(22 %)	97/363(26.7 %)
Overall class D	18/118(15.3 %)	24/118(20.3 %)	32/118(28 %)	38/118(32.2 %)
Overall	165/864 (19.1 %)	190/864(22 %)	260/864(30.1 %)	300/864 (34.7 %)

In the first experiment, we trained the 7-HMM for the 38 candidate folds. Then, the test set proteins were scored against all models of candidate folds and were assigned to that fold whose model gave the maximum posterior probability score. In the second experiment, we trained the 3-HMM for the 38 candidate folds, while the scoring and classification procedure remains the same. In both cases, forward algorithm was used for scoring only the primary sequences against the model.

In the third experiment, we trained again the 3-HMM for all candidate folds but we scored each test set protein against the model by taking into account its predicted secondary sequence. In the fourth experiment we replaced the predicted secondary sequences of the test set proteins with the correct ones given by PDB [37] and everything else remained the same as in the third one. In that way we could theoretically assess the maximum performance of the 3-HMM, given that the secondary structure predictor is 100 % accurate.

We can see that the 3-HMM outperforms 7-HMM in the same dataset as the performance increased by 2.9 % (from 19.1 % to 22 %). When we used also the predicted secondary structure in scoring, the fold classification performance was further improved, from 22 % to 30 %. Finally, when the correct secondary structure was used in scoring, the fold classification accuracy in the fourth experiment reached its maximum of 34.7 %.

5.3 Optimized 3-HMM

The performance of the optimized 3-HMM was compared against the performance of the most effective version of 3-HMM, as well as against the latest version of SAM (3.5). As a result two experiments took place for this purpose.

The sequences employed in the experiments come from the newest ASTRAL40 SCOP 2.03 dataset, where we did not include proteins with more than 40 % similarity [7]. Moreover, the SCOP folds of classes A, B, C, D, and G, and specifically those which have at least 50 members, were included. We employed only the folds with more than 50 members so that there would be enough data for proper training and testing (especially in the case of SAM). This yielded a set of 42 such folds with 5024 sequences in total.

We also used tenfold cross validation for the evaluation procedure, so the sequences were separated ten times in training and test sets. Each time 90 % of the sequences of each fold were used as the training set of the fold and the rest 10 % were used as the test set. Furthermore, secondary structure predictions were needed for the whole dataset, so predictions were obtained from SymPsiPred [41]. SymPsiPred is considered as a more effective method for secondary structure prediction compared to PSIPRED which was used earlier for assessing the 3-HMM [7].

In the first experiment, we trained the improved version of the 3-HMM for the 42 candidate folds, using both primary and true

secondary structures of the proteins in the training set. Then the test set proteins were scored against all models of candidate folds and were assigned to that fold for which the model gives the maximum posterior probability score. The test set proteins were scored against the 3-HMM by employing not only their primary structure but also their predicted secondary structure. Overall, the 3-HMM achieves a 37.8 % accuracy.

In the second experiment, we employed the optimization method described above (Subheading 4) for the identification of the optimal parameters for the 3-HMM. When optimization was performed to the 3-HMM, the test set accuracy of the model increased by almost 10 % (from 37.8 % to 41.4 %). Moreover, we compared the above results with the classification results of SAM, which was applied in the same dataset. The classification accuracy of SAM reached 38 % which was approximately 8 % lower than the accuracy obtained by the optimized 3-HMM. Results for the above procedure for both experiments in the dataset are shown in Table 4.

Finally, in order to evaluate the robustness of the optimized 3-HMM, the receiver operating characteristics (ROCReceiver operating characteristics (ROC)) analysis was performed following the class reference formulation, where each class (fold) is considered separately against all others [7]. The attained area under the curve (AUC) was 0.88, indicating the robustness of our method. It is worth mentioning that when ROC analysis was performed for the 7-HMM, the AUC was found to be 0.76 when different models were trained for each fold and 0.73 when different models were trained for each sequence [30].

Table 4
Classification results for the two experiments

Folds	3-HMM (%)	3-HMM optimized (%)	SAM 3.5 (%)
Overall class A	39.3	44.9	31.5
Overall class B	48.3	50.0	33.0
Overall class C	32.7	37.2	42.9
Overall class D	28.4	31.1	41.8
Overall class G	75.6	76.2	50.0
Overall	37.8	41.4	38.0

In the first column the classes that correspond to the folds used are shown. The results from the 3-HMM when it is used alone and when its scores are optimized are shown in the second and third column, respectively. The classification results of SAM 3.5 are shown in the fourth column

6 Conclusions

In this chapter, we discuss methods for building HMMs with reduced number of states for predicting the fold of a protein given its sequence and assess the performances of different variations of these HMMs including 7-HMM, 3-HMM, and optimized 3-HMM. 7-HMM and optimized 3-HMM were also compared with the latest version of SAM, a widely used HMM-based method. 3-HMM was more effective than 7-HMM and optimized 3-HMM achieved the maximum performance. The results indicate that the variations of the model were equally or more effective than SAM while they use a small architecture and require less data for training. Moreover, all of them use a low complexity training algorithm while the 3-HMM also uses a testing algorithm less complex than the forward algorithm which is used in the 7-HMM. Other possible improvements that could be incorporated to the context of the proposed reduced state-space HMM architecture may exploit the use of more structural features, apart from the secondary structure. The aim would be to further enhance the classification efficiency of the model and at the same time maintain its low size and complexity.

References

1. Whitford D (2005) Proteins: structure and function. John Wiley & Sons, NJ, USA
2. Lee SY, Lee JY, Jung KS, Ryu KH (2009) A 9-state hidden Markov model using protein secondary structure information for protein fold recognition. Comp Biol Med 39(6):527–534
3. Camproux A, Guyon F, Gautier R, Laffray J, Tuffery P (2005) A hidden Markov model applied to the analysis of protein 3D-structures. in: Proc. int. symp. applied stochastic models and data analysis
4. Orengo CA, Jones DT, Thornton JM (2003) Bioinformatics: genes, proteins and computers. Bios Scientific Pub. Ltd, Oxford
5. Zhang Y, Skolnick J (2005) The protein structure prediction problem could be solved using the current PDB library. Proc Natl Acad Sci U S A 102(4):1029–1034
6. Hargbo J, Elofsson A (1999) Hidden Markov models that use predicted secondary structures for fold recognition. Proteins 36(1):68–76
7. Lampros C, Simos T, Exarchos TP, Exarchos KP, Papaloukas C, Fotiadis DI (2014) Assessment of optimized Markov models in protein fold classification. J Bioinform Comput Biol 12 (4):1450016
8. Murzin AG (1999) Structure classification based assessment of CASP3 predictions for the fold recognition targets. Proteins (Suppl 3):88–103
9. Orengo CA, Bray JE, Hubbard T, LoConte L, Sillitoe I (1999) Analysis and assessment of ab initio three-dimensional prediction, secondary structure, and contacts prediction. Proteins 37:149–170
10. Zhang Y (2008) Progress and challenges in protein structure prediction. Curr Opin Struct Biol 18(3):342–348
11. Zhou Y, Duan Y, Yang Y, Farragi E, Lei H (2011) Trends in template/fragment-free protein structure prediction. Theor Chem Acc 128 (1):3–16
12. Maurice KJ et al (2014) SSThread: template-free protein structure prediction by threading pairs of contacting secondary structures followed by assembly of overlapping pairs. J Comput Chem 35(8):644–656
13. Bowie JU, Luthy R, Eisenberg D (1991) A method to identify protein sequence that fold into a known three-dimensional structure. Science 253:164–170

14. Flockner H, Domingues F, Sippl MJ (1997) Proteins folds from pair interactions: a blind test into fold recognition. Proteins 1:129–133
15. Xu J (2005) Fold recognition by predicted alignment accuracy. IEEE/ACM Trans Comput Biol Bioinform 2(2):157–165
16. Sander O, Sommer I, Lengauer T (2006) Local protein structure prediction using discriminative models. BMC Bioinformatics (7):14
17. Hu Y, Dong X, Wu A, Cao Y, Tian L, Jiang T (2011) Incorporation of local structural preference potential improves fold recognition. PLoS One 6(2):e17215
18. Mahajan S, De Brevern AG, Sanejouand YH, Srinivasan N, Offmann B (2015) Use of a structural alphabet to find compatible folds for amino acid sequences. Protein Sci 24 (1):145–153
19. Finn RD, Clements J, Eddy SR (2011) HMMER web server: interactive sequence similarity searching. Nucleic Acids Res 39(Suppl 2):W29–W37
20. Karplus K, Karchin R, Shackelford G, Hughey R (2005) Calibrating E-values for hidden Markov models using reverse-sequence null models. Bioinformatics 21:4107–4115
21. Karchin R, Cline M, Mandel-Gutfreund Y, Karplus K (2003) Hidden Markov models that use predicted local structure for fold recognition: alphabets of backbone geometry. Proteins 51:504–514
22. Dandekar T, Argos P (1996) Identifying the tertiary fold of small proteins with diferent topologies from sequence and secondary structure using the genetic algorithm and extended criteria specific for strand regions. J Mol Biol 256:645–660
23. Zangoei MH, Jalili S (2013) Protein fold recognition with a two-layer method based on SVM–SA, WP–NN and C4. 5 (TLM–SNC). Int J Data Mining Bioinform 8(2):203–223
24. Deschavanne P, Tuffery P (2009) Enhanced protein fold recognition using a structural alphabet. Proteins 76:129–137
25. Chmielnicki W, Stapor K (2012) A hybrid discriminative/generative approach to protein fold recognition. Neurocomputing 75 (1):194–198
26. Exarchos TP, Papaloukas C, Lampros C, Fotiadis DI (2008) Mining sequential patterns for protein fold recognition. J Biomed Inform 41 (1):165–179
27. Tsai CY, Chen CJ, (2015) A PSOAB classifier for solving sequence classification problems. Appl Soft Comput 27(C):11–27
28. Valavanis I, Spyrou G, Nikita K (2010) A similarity network approach for the analysis and comparison of protein sequence/structure sets. J Biomed Inform 43(2):257–267
29. Abbasi E, Mehdi G, Shiri ME (2013) FRAN and RBF-PSO as two components of a hyper framework to recognize protein folds. Comput Biol Med 43(9):1182–1191
30. Lampros C, Papaloukas C, Exarchos TP, Goletsis Y, Fotiadis DI (2007) Sequence-based protein structure prediction using a reduced state-space hidden Markov model. Comput Biol Med 37:1211–1224
31. Durbin R, Eddy S, Krogh A, Mitchison G (1998) Biological sequence analysis: probabilistic models of proteins and nucleic acids. Cambridge University Press, New York
32. Kabsch W, Sander C (1983) Dictionary of protein secondary structure: pattern recognition of hydrogen-bonded and geometrical features. Biopolymers 22:2577–2637
33. Baum LE (1972) An inequality and associated maximization technique in statistical estimation for probabilistic functions of Markov processes. Inequalities 3:1–8
34. Lampros C, Papaloukas C, Exarchos K, Fotiadis DI, Tsalikakis D (2009) Improving the protein fold recognition accuracy of a reduced state-space hidden Markov model. Comput Biol Med 39:907–914
35. Lagarias JC, Reeds JA, Wright MH, Wright PE (1998) Convergence properties of the Nelder-Mead simplex method in low dimensions. SIAM J Optim 9(1):112–147
36. Chandonia JM, Hon G, Walker NS, Lo Conte L, Koehl P, Levitt M, Brenner SE (2004) The ASTRAL compendium in 2004. Nucleic Acids Res 32(Database issue):D189–D192
37. Berman HM, Westbrook J, Feng Z, Gilliland G, Bhat TN, Weissig H, Shindyalov IN, Bourne PE (2000) The protein data bank. Nucleic Acids Res 28:235–242
38. Andreeva A, Howorth D, Brenner SE, Hubbard TJ, Chothia C, Murzin AG (2004) SCOP database in 2004: refinements integrate structure and sequence family data. Nucleic Acids Res 32(Database issue):D226–D229
39. Machado-Lima A, Kashiwabara AY, Durham AM (2010) Decreasing the number of false positives in sequence classification. BMC Genomics 22(11 Suppl 5):S10
40. Jones DT (1999) Protein secondary structure prediction based on position specific scoring matrices. J Mol Biol 292:195–202
41. Lin HN, Sung TY, Ho SY, Hsu WL (2010) Improving protein secondary structure prediction based on short subsequences with local structure similarity. BMC Genomics 2 (Suppl 4):S4

Chapter 3

Application of Hidden Markov Models in Biomolecular Simulations

Saurabh Shukla, Zahra Shamsi, Alexander S. Moffett, Balaji Selvam, and Diwakar Shukla

Abstract

Hidden Markov models (HMMs) provide a framework to analyze large trajectories of biomolecular simulation datasets. HMMs decompose the conformational space of a biological molecule into finite number of states that interconvert among each other with certain rates. HMMs simplify long timescale trajectories for human comprehension, and allow comparison of simulations with experimental data. In this chapter, we provide an overview of building HMMs for analyzing bimolecular simulation datasets. We demonstrate the procedure for building a Hidden Markov model for Met-enkephalin peptide simulation dataset and compare the timescales of the process.

Keywords Hidden Markov models, Molecular dynamics, Protein dynamics, Markov state models, Protein conformational change

1 Introduction

Conformational changes in proteins are essential for their biological function. Experimental techniques like X-ray crystallography and NMR techniques yield static protein structures but fail to provide detailed information concerning conformational dynamics. Molecular dynamics (MD) simulations provide atomistic details of such microscopic processes [1] and have emerged as a successful tool for obtaining protein conformational ensembles [2, 3], studying protein stabilization [4–6], and elucidating drug binding events [7, 8], activation and signaling mechanisms [9]. These biological events occur on microsecond to millisecond timescales, corresponding to oftentimes intractably long simulations. However, with recent progress in hardware and software systems, it has become more practical to run simulations on biologically relevant timescales, allowing for better characterization and understanding of protein functional behavior. Such simulations generate "big" simulation datasets that are infeasible to analyze through

David R. Westhead and M.S. Vijayabaskar (eds.), *Hidden Markov Models: Methods and Protocols*, Methods in Molecular Biology, vol. 1552, DOI 10.1007/978-1-4939-6753-7_3, © Springer Science+Business Media LLC 2017

visualization. Therefore, efficient statistical methods are needed to analyze these massive datasets without any loss of critical information [10]. Use of geometric clustering techniques has simplified analysis and interpretation of molecular dynamics trajectories, allowing for definition of states containing structurally similar conformations from which free energies of discretized conformational space can be estimated provided adequate sampling. However, this process does not yield any kinetic information about the protein, a critical limitation given the previously emphasized importance of dynamic information in understanding of protein function. Markov state models (MSMs) are kinetic models that provide this missing information by assuming that protein dynamics in discrete space evolve as a Markov process. By clustering structures from trajectories and constructing MSMs, it is possible to identify metastable states of the protein along with dominant pathways and timescales of conformational changes [11, 12]. MSMs typically require the definition of thousands of states in order for the dynamics to be Markovian, which hinders interpretability of the model [13].

The application of hidden Markov models overcomes the challenges described above. HMMs are probabilistic models that serve as a tool to identify unobservable states in complex systems by considering the frequency of corresponding observable attributes [14], that is, HMMs assume that states of the system are not directly visible but are defined probabilistically by the set of observations. This method efficiently and accurately computes easily interpretable hidden states from the large simulation datasets and predicts the rate of interconversion between states, allowing for more accurate calculation of kinetic properties than with MSMs. The probabilities of HMM states are assumed to evolve as a Markov process, that is, the probability of transitioning to any other state depends only on the current state of the system [15].

HMMs have been used extensively in pattern recognition problems such as speech recognition [16], image processing [17], and robotics [18]. In speech recognition, HMMs are applied to identify a particular word spoken in recorded audio after training on audio datasets of a set of words spoken by different people [16]. More recently, HMMs have been used to analyze complex biological problems such as sequencing [19], protein folding [20], protein functional mechanisms [21], and binding site detection [22]. Furthermore, close analogs of protein and DNA sequences have been successfully predicted using HMM method by inserting or deleting gaps in the sequence alignment [23]. In this chapter, we discuss the methodology and advantages of using HMMs in analyzing large MD datasets. We also discuss examples of applications of HMMs and present a tutorial using the met-enkephalin peptide to identify the hidden states and compare the times scales of conformational changes using the software package MSMbuilder 3.2 [24–26].

2 Hidden Markov Model

A hidden Markov model (HMM) is a statistical model on Markov processes, which has both unknown "hidden" states and associated observables. The observation at time k is assumed to occur according to a certain probability distribution depending on states that are hidden from the observer. It is also assumed that transitions between hidden states follow a Markov process [27]. Figure 1 depicts the basic configuration of HMM. In application of HMMs to biomolecular simulations, metastable wells in the conformational free energy landscape of the molecule correspond to hidden states, where the probability of transition from the current state at a certain time to others depends only on the current state [28]. Accordingly, exact conformations of the molecule found in the output of MD are the observables of the model. Fluctuations inside each metastable state are rapid, while interstate transitions take place on longer timescales and are of more biological importance; consequently, a goal of HMM (or MSM) analysis is to identify the dominant pathways between metastable states.

In HMMs, x_k is the hidden state of system at time k. If system has N states, then system can reside in any of the N states [14]. One can define a transition matrix $T(i,j)$, elements of which represent the transition probability from state i to j:

$$T(i,j) = P\left(x_{k+1} = j \middle| x_k = i\right) \tag{1}$$

Transition matrix is an N × N matrix, where N is the number of states of system. **Note 1** in notes section describes briefly the process of construction of transition probability matrix. HMM is also composed of some observations y_k, such as 3D positions of atoms in protein or magnitude of dihedrals in structures or any other defined feature in molecular dynamics simulations, which can be observed. By definition, y can be continuous or a discrete variable. Emission probability is the probability of observing a particular observation y given a state x. For discrete y, the emission probability is described as a matrix E:

$$E_i(x) = P\left(y_k = i \middle| X_k = x\right) \tag{2}$$

Initial state probabilities define the first state of the system in the time series:

$$\pi_{x_1} = P(x_1) \tag{3}$$

By building the HMM, we intend to find out the transition probabilities of a sequence of states and emission probabilities of the sequence of observations, $P\left(x_1, x_2, \ldots, x_n, y_1, y_2, \ldots, y_n\right)$. We know that using basic probability theory, we can factor the joint

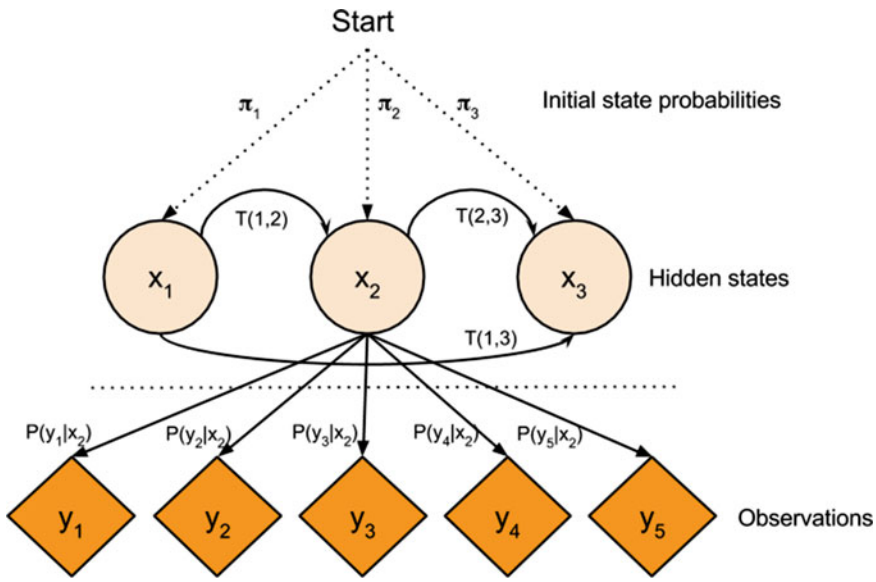


Fig. 1 Structure of hidden Markov model. In HMM formulation, π_i is the probability that system is in state x_i in the beginning. System makes transition from state x_i to x_j with the transition probability $T(i,j)$ and each state produces an observation (y_k) with probability $P(y_k|x_i)$

probability as a product of conditional probabilities. So, we can calculate the probability as follows [29].

$$P(x_1, x_2, \ldots, x_n, y_1, y_2, \ldots, y_n) = P(x_1)P(y_1|x_1)\prod_{k=2}^{n} P(x_k|x_{k-1})P(y_k|x_k) \quad (4)$$

There are different methods suggested in the literature for the estimation of these parameters. In the molecular dynamics simulations, the observation y is usually a continuous-time series. Therefore, it is not possible to define the E matrix. Emission probabilities are multivariate Gaussian distributions. Therefore, L1-regularized reversible HMMs with Gaussian emissions are used yielding generative probabilistic models over multivariate discrete-time continuous-space time series. L1-regularization reduces the effect of less important degrees of freedom on the output of the model. Detailed balance is also enforced over the system so that system does not violate the second law of thermodynamics [28].

Suppose $x_k = l$, the emission distribution is a multivariate normal distribution, which is parameterized by mean μ_l and diagonal covariance matrix $\mathrm{diag}(\sigma_l^2)$.

$$P(x_1, x_2, \ldots, x_n, y_1, y_2, \ldots, y_n) = \pi_{x_1}\prod_{k=2}^{n} T_{x_{k-1},x_k}\prod_{k=1}^{n} N\left(y_k; \mu_{x_k}, \sigma_{x_k}^2\right) \quad (5)$$

Learning in this method is finding appropriate parameters,

$T, \mu, and \sigma$ for the model. McGibbon et al. used a two-step method for learning the parameters of these kinds of HMMs. The first step is E-step, in which some probabilities are computed using forward–backward algorithm. In the following step, M-step, the normal distribution mean is iteratively updated [28].

3 Application of HMMs in Biomolecular Simulations

HMMs in various forms have been successfully applied to numerous problems in biology, from describing conformational dynamics in single molecule experiments [30] to identification of homologous DNA sequences [31]. HMMs have also been shown to better approximate the continuous dynamics of real molecules in modeling of long-timescale protein conformational changes from molecular simulation over discrete state MSMs [27] and thus are of great interest to the field. In this section, we highlight several recent applications of HMMs to bimolecular simulation analysis.

Theyer et al. [22], used HMMs to predict DNA binding sites of the catabolite activator protein (CAP) using sequence as well as the molecular dynamics simulation of the DNA structure. The accuracy of the results was improved by incorporating the structural ensemble of the DNA generated from MD, compared to only sequence information. By using HMMs, the authors showed a clear difference between the DNA binding and nonbinding sites in the CAP protein.

Noe et al. [27] show that one can approximate the projection of continuous dynamics in phase space onto discrete states using HMMs. The use of HMMs in the place of MSMs allows one to discard the assumption that molecular systems evolve in time through a Markov process on discretized conformational space, an approximation required for construction of MSMs. This method of HMM construction leads to overlapping emission probabilities of observables, allowing for transition states of low probability where the system has some probability of being in more than one hidden state given the observable. Noe et al. applied this method to a toy two-well potential, yielding more accurate relaxation times of slowly evolving degrees of freedom and more resistance to poor clustering than with the direct MSM method. In addition, they used the HMM to successfully analyze a molecular dynamics simulation of BPTI and estimate relaxation timescales.

Similarly, McGibbon et al. [28] created an HMM with Gaussian emission probability distributions for each hidden state, giving observable states with soft boundaries. They applied the HMM to Brownian dynamics simulations in a double well potential, where it was able to accurately learn the longest relaxation timescale of the system while a direct MSM could not. Additionally, McGibbon et al.

simulated the molecular dynamics of ubiquitin and used the HMM to identify two important metastable states in agreement with experimental findings [32], providing a clearly interpretable two-state model, whereas the interpretation of the result of MSM analysis was not as clear. Finally, they applied the HMM to simulation trajectories of src kinase, which was able to learn the critical states for the kinase activation pathway with a three state model, previously identified by MSM analysis using 2000 microstates [2].

It is clear from these examples that the use of HMMs can not only improve quantitative accuracy of molecular dynamics analysis but also facilitates the creation of physically interpretable models of biological systems. In next section, we present a tutorial on application of HMMs on simulation dataset of a peptide.

4 Tutorial on Met-enkephalin

In this section, we demonstrate the process of making a hidden Markov model from ten ~50 ns MD simulation trajectories of the five residue Met-enkephalin peptide. We will cluster the trajectories and examine the scaling of the timescales implied by the model with the lag time (τ). We use MSMBuilder (version 3.2), a Python package built for analyzing biomolecular simulation data. The MSMBuilder package contains the Met-enkephalin dataset used in this tutorial, and can be downloaded from MSMbuilder website (http://msmbuilder.org/, also includes installation instructions). Basic Python knowledge is required for understanding this tutorial, as well as a working installation of the Matplotlib Python library [33].

1. After installing the MSMBuilder package, we run the iPython notebook [34]. First, we import the required Matplotlib and MSMbuilder packages. Comments (green colored and followed by #) are used to explain the code.

```
# Module for plotting
from matplotlib.pyplot import *
# Module for featurization
from msmbuilder.featurizer import SuperposeFeaturizer
# Importing Met-enkephalin simulation dataset
from msmbuilder.example_datasets import MetEnkephalin
# MSMbuilder module for building HMM
from msmbuilder.hmm import GaussianFusionHMM
```

2. We are now ready to import the Met-enkephalin data set that consists of ten ~50 ns MD simulation trajectories. The details of the dataset can be obtained by using the following command.

```
print(MetEnkephalin.description())
```

Refer to **Note 2** for the output of the command. The following commands fetch the dataset and assign it to a variable called 'trajectories'.

```
dataset = MetEnkephalin().get()
trajectories = dataset.trajectories
```

3. We assign first frame of the trajectory as the topology (reference conformation) of the molecule.

```
topology = trajectories[0].topology
```

4. Each clustering algorithm classifies the data based on some structural features, which are functions of the atomic coordinates. In molecular dynamics simulations, various features such as root mean-squared deviation (RMSD), dihedral angles, and inter-residue distances are used for featurization purposes. The featurization process converts each snapshot in the trajectory from a vector in $\mathbb{R}^{3N}$ into a vector that represents the specified features describing the current conformation [13]. In this case, we superimpose each snapshot in the trajectory over the first snapshot (reference structure) and compute the distance between corresponding atoms in the trajectories and reference structure as the feature. Before we run the featurization, we assign heavy atoms (carbon, oxygen and nitrogen) in the topology file to an array called 'indices' that will be used in featurization.

```
# Specifying heavy atoms (O, N & C) and assigning them to a
variable
Indices = [atom.index for atom in topology.atoms if atom.ele-
ment.symbol in ['C', 'O', 'N']]
Featurization is achieved by the following commands:
Featurizer = SuperposeFeaturizer(indices, trajectories[0][0])
sequences = featurizer.transform(trajectories)
```

5. We now compare the relaxation timescales of the Hidden Markov model with different lag-times and numbers of states. We define the arrays for the different lag times and numbers of states (*see* **Note 3**).

```
# Initial specification of lag times and number of states
lag_times = [1, 10, 20, 30, 40]
hmm_ts0 = {}
hmm_ts1 = {}
n_states = [3, 5]
```

6. Now we are ready to estimate the timescales of the models under consideration.

```
# for loop for different number of states
for n in n_states:
   hmm_ts0[n] = [ ]
   hmm_ts1[n] = [ ]
# for loop for differrnt lag times
   for lag_time in lag_times:
# We skip i(same as lag time) frames in trajectory while building
HMM...
# and save every ith frame in variable named strided data.
```

```
    strided_data = [s[i::lag_time] for s in sequences for i in range
(lag_time)]
# Function for building hidden markov model
# Function takes number of states and features as inputs and
acts...
# on strided_data. Detailed explanation of inputs and outputs of
the...
# function can be found on MSMbuilder website (msmbuilder.
org).
    hmm = GaussianFusionHMM(n_states = n,    n_features =
sequences[0].shape[1], n_init = 1).fit(strided_data)
# Calculation of timescales
    timescales = hmm.timescales_ * lag_time
# first timescale of model
    hmm_ts0[n].append(timescales[0])
# second timescale for the process
    hmm_ts1[n].append(timescales[1])
    print('n_states = %d\tlag_time = %d\ttimescales = %s' % (n,
lag_time, timescales))
print()
```

We get the following output:

n_states = 3	lag_time = 1	timescales = [14.25163078 9.09222412]
n_states = 3	lag_time = 10	timescales = [45.15606308 25.75660896]
n_states = 3	lag_time = 20	timescales = [37.86439514 26.81169319]
n_states = 3	lag_time = 30	timescales = [76.23511505 43.87307739]
n_states = 3	lag_time = 40	timescales = [55.33748245 38.56738281]

n_states = 5	lag_time = 1	timescales = [15.0960 9.6521 8.088 7.3205]
n_states = 5	lag_time = 10	timescales = [48.5430 31.1624 20.9082 17.7139]
n_states = 5	lag_time = 20	timescales = [72.0424 47.1938 28.6126 24.1619]
n_states = 5	lag_time = 30	timescales = [92.4098 59.5764 33.9523 29.559]
n_states = 5	lag_time = 40	timescales = [107.5127 70.249 38.729 33.3036]

7. We plot the results and compare the relaxation time in Fig. 2:

```
# Plotting the first and second timescale of process with N = 3
& N = 5
figure(figsize = (14,3))
for i, n in enumerate(n_states):
```

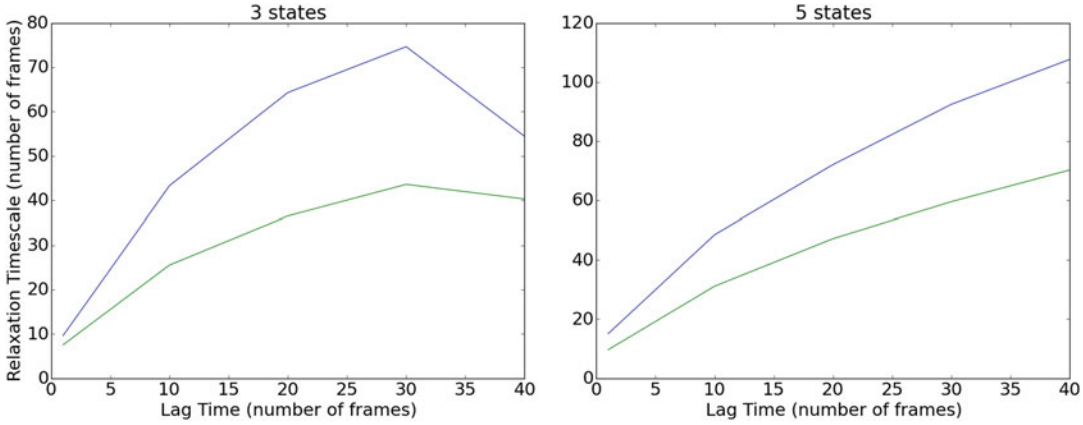


Fig. 2 Comparison between the relaxation timescales with lag-time of the process (with $N = 3$ and $N = 5$). *Blue* and *green lines* correspond first and second slowest relaxation timescales respectively

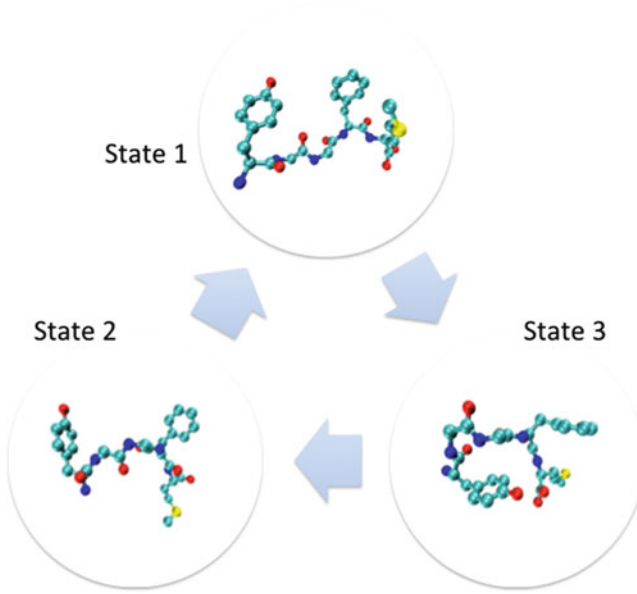


Fig. 3 Centroids of three states predicted by hidden Markov model interconverting among each other. Timescale of the process is 14.24 frames for lag time of 1 frame

```
    subplot(1,len(n_states),1 + i)
    plot(lag_times, hmm_ts0[n])
    plot(lag_times, hmm_ts1[n])
    if i == 0:
        ylabel('Relaxation Timescale')
    xlabel('Lag Time')
    title('%d states' % n)
show()
```

Timescales of the process should converge as the lag time is increased. If we have N states, we get N − 1 timescales. Please *see* **Note 4** for discussion of relaxation timescale and lag time.

8. In a separate tutorial, we estimate the cluster centers of three states with a lag time of one. Explanation and code for estimation of cluster center is provided in **Note 5**. Cluster centers of three states are depicted in Fig. 3.

5 Notes

1. Estimation of HMM parameters (transition matrix, emission probability distribution, and initial state probabilities) is done by expectation-maximization (EM). Expectation maximization is done in two steps. The forward–backward algorithm is applied in the E-step. In the M-step, a modified log likelihood function, which is penalized for L1 regularization, is maximized to estimate HMM parameters. Detailed balance is also enforced during the M-step in order to satisfy the second law of thermodynamics. Mathematical details of the expectation-maximization algorithm implementation can be found in McGibbon et al. [28].
2. Details of simulation dataset used in the tutorial is given below
   The dataset consists of ten ~50 ns molecular dynamics (MD) simulation trajectories of the five residue Met-enkephalin peptide. The aggregate sampling is 499.58 ns. Simulations were performed starting from the first model in the 1PLX PDB file, solvated with 832 TIP3P water molecules using OpenMM 6.0. The coordinates (protein only—the water was stripped) are saved every 5 ps. Each of the ten trajectories is roughly 50 ns long and contains about 10,000 snapshots.
   Forcefield: amber99sb-ildn; water: tip3p; nonbonded method: PME; cutoffs: 1nm; bonds to hydrogen were constrained; integrator: langevin dynamics; temperature: 300 K; friction coefficient: 1.0/ps; pressure control: Monte Carlo barostat (interval of 25 steps); timestep 2 fs.
   The dataset is available on figshare at:
   http://dx.doi.org/10.6084/m9.figshare.1026324.
3. The number of states (N) is a critical parameter, which needs to be defined before constructing an HMM. A model with fewer states may not represent the true kinetics of the process whereas a model with a large number of states may be very difficult to interpret. AIC [35] and BIC [36] criterions are used for determining the states analytically. Keller et al. made a fine-grained HMM with many states with subsequent coarse graining based on kinetics of the process [37]. Coarse graining can be done by eigenvector/eigenvalue analysis of transition probability matrix.

Prior knowledge of the process at hand greatly helps in determining the optimal number of states.

4. The transition probability matrix is estimated by calculating the pairwise probabilities of transition between states in a time interval τ, called the lag time of the Markov model. In order for the model to have an acceptable level of "Markovianity," τ should be chosen to be appropriately larger than the interconversion time within a state to approach homogeneity within states in terms of kinetic behavior. The relaxation timescales are computed from the eigenvalues of the transition matrix **T**:

$$t_i = -\frac{\tau}{\ln\lambda_i(\tau)}$$

where λ_i is the ith eigenvalue of the transition matrix. The eigenvectors and eigenvalues of **T** are defined as:

$$\boldsymbol{T}\boldsymbol{\psi}_i = \lambda_i\boldsymbol{\psi}_i$$

where $\boldsymbol{\psi}_i$ is the ith right eigenvector of **T**, corresponding to eigenvalue λ_i.

5. Tutorial on Met-enkephalin peptide (three states, lagtime = 1): We provide the python commands (in bold) and the corresponding output (in italics) for building the HMM. Explanation of steps can be found in the tutorial discussed in paper.

```
from matplotlib.pyplot import *
from msmbuilder.featurizer import SuperposeFeaturizer
from msmbuilder.example_datasets import MetEnkephalin
from msmbuilder.hmm import GaussianFusionHMM
from msmbuilder.cluster import KCenters
from msmbuilder.msm import MarkovStateModel

print(MetEnkephalin.description())

dataset = MetEnkephalin().get()
trajectories = dataset.trajectories
topology = trajectories[0].topology
indices = [atom.index for atom in topology.atoms if atom.
element.symbol in ['C', 'O', 'N']]
featurizer = SuperposeFeaturizer(indices, trajectories[0]
[0])
sequences = featurizer.transform(trajectories)

lag_time = 1
hmm_ts0 = {}
hmm_ts1 = {}
n_states = 3
```

```
hmm_ts0[n_states] = []
hmm_ts1[n_states] = []

strided_data = [s[i::lag_time] for s in sequences for i in
range(lag_time)]

hmm = GaussianFusionHMM(n_states = n_states, n_features =
sequences[0].shape[1], n_init = 1).fit(strided_data)

timescales = hmm.timescales_ * lag_time

centroids = hmm.draw_centroids(strided_data)

print('n_states = %d\tlag_time = %d\ttimescales = %s' %
(n_states, lag_time, timescales))

n_states = 3  lag_time = 1  timescales = [14.24336433
9.09043217]

centroids

(array([[[9, 2418]],

   [[6, 8237]],

   [[8, 9948]]]),
array([[[0.60136747, 0.38697317, ......0.91163236,
0.33621228]],

   [[0.35191458, 0.21339083, ......0.35373399, 0.2641741]],

   [[0.423269, 0.2762385, ......, 0.60179132,
0.32357678]]], dtype = float32))
```

References

1. Karplus M, McCammon JA (2002) Molecular dynamics simulations of biomolecules. Nat Struct Biol 9:646–652
2. Shukla D, Meng Y, Roux B, Pande VS (2014) Activation pathway of Src kinase reveals intermediate states as targets for drug design. Nat Commun 5:1–11
3. Lapidus LJ, Acharya S, Schwantes CR, Wu L, Shukla D, King M, DeCamp SJ, Pande VS (2014) Complex pathways in folding of protein G explored by simulation and experiment. Biophys J 107:947–955
4. Shukla D, Trout BL (2010) Interaction of arginine with proteins and the mechanism by which it inhibits aggregation. J Phys Chem B 114:13426–13438
5. Shukla D, Shinde C, Trout BL (2009) Molecular computations of preferential interaction coefficients of proteins. J Phys Chem B 113:12546–12554
6. Shukla D, Schneider CP, Trout BL (2011) Molecular level insight into intra-solvent interaction effects on protein stability and aggregation. Adv Drug Deliv Rev 63:1074–1085

7. Shan Y, Kim ET, Eastwood MP, Dror RO, Seeliger MA, Shaw DE (2011) How does a drug molecule find its target binding site? J Am Chem Soc 133:9181–9183
8. Lawrenz M, Shukla D, Pande VS (2015) Cloud computing approaches for prediction of ligand binding poses and pathways. Sci Rep 5:1–5
9. Kohlhoff KJ, Shukla D, Lawrenz M, Bowman GR, Konerding DE, Belov D, Altman RB, Pande VS (2014) Cloud-based simulations on Google Exacycle reveal ligand modulation of GPCR activation pathways. Nat Chem 6:15–21
10. Lane TJ, Shukla D, Beauchamp KA, Pande VS (2013) To milliseconds and beyond: challenges in the simulation of protein folding. Curr Opin Struct Biol 23:58–65
11. Shukla D, Hernández CX, Weber JK, Pande VS (2015) Markov state models provide insights into dynamic modulation of protein function. Acc Chem Res 48:414–422
12. Sultan MM, Kiss G, Shukla D, Pande VS (2014) Automatic selection of order parameters in the analysis of large scale molecular dynamics simulations. J Chem Theory Comput 10:5217–5223
13. Pande VS, Beauchamp KA, Bowman GR (2010) Everything you wanted to know about Markov State Models but were afraid to ask. Methods 52:99–105
14. Rabiner LR, Juang BH (1986) An introduction to hidden Markov models. IEEE ASSP Mag 3:4–16
15. Eddy SR (1996) Hidden Markov models. Curr Opin Struct Biol 6:361–365
16. Rabiner LR (1989) A tutorial on hidden Markov models and selected applications in speech recognition. Proc IEEE 77:257–286
17. de Trazegnies C, Urdiales C, Bandera A, Sandoval F (2003) A Hidden Markov Model object recognition technique for incomplete and distorted corner sequences. Image Vis Comput 21:879–889
18. Fox M, Ghallab M, Infantes G, Long D (2006) Robot introspection through learned hidden Markov models. Artif Intell 170:59–113
19. Hughey R, Krogh A (1996) Hidden Markov models for sequence analysis: extension and analysis of the basic method. Bioinformatics 12:95–107
20. Bouchaffra D, Tan J (2006) Protein fold recognition using a structural hidden Markov model. In: Proceedings of the 18th international conference on pattern recognition, vol 3, pp 186–189
21. Chiang TH, Hsu D, Latombe JC (2010) Markov dynamic models for long-timescale protein motion. Bioinformatics 26:269–277
22. Thayer KM, Beveridge DL, Thayer KM, Beveridget DL (2011) Markov Hidden on simulations models DNA from molecular dynamics. Proc Natl Acad Sci U S A 99:8642–8647
23. Durbin R, Eddy SR, Krogh A, Mitchison G (1998) *Biological sequence analysis*: probabilistic models of proteins and nucleic acids. Cambridge University Press, Cambridge
24. Bowman GR, Huang X, Pande VS (2009) Using generalized ensemble simulations and Markov state models to identify conformational states. Methods 49:197–201
25. Haque IS, Beauchamp KA, Pande VS (2014) A fast $3 \times N$ matrix multiply routine for calculation of protein RMSD. bioRxiv 008631:1–13
26. Beauchamp KA, Bowman GR, Lane TJ, Maibaum L, Haque IS, Pande VS (2011) MSMBuilder2: modeling conformational dynamics at the picosecond to millisecond scale. J Chem Theory Comput 7:3412–3419
27. Noé F, Wu H, Prinz JH, Plattner N (2013) Projected and hidden Markov models for calculating kinetics and metastable states of complex molecules. J Chem Phys 139:1–17
28. Mcgibbon RT, Ramsundar B, Sultan MM, Kiss G, Pande VS (2014) Understanding protein dynamics with L1-regularized reversible hidden Markov models. In: Proceedings of the 31st international conference on machine learning, vol 32, pp 1197–1205
29. Ghahramani Z (2001) An introduction to hidden Markov models and Bayesian networks. Int J Pattern Recognit Artif Intell 15:9–42
30. Talaga D (2007) Markov processes in single molecule fluorescence. Curr Opin Colloid Interface Sci 12:285–296
31. Eddy SR (1998) Profile hidden Markov models. Bioinformatics 14:755–763
32. Hunter JD (2007) Matplotlib: a 2D graphics environment. Comput Sci Eng 9:90–95
33. Zhang Y, Zhou L, Rouge L, Phillips AH, Lam C, Liu P, Sandoval W, Helgason E, Murray JM, Wertz IE (2013) Conformational stabilization of ubiquitin yields potent and selective inhibitors of USP7. Nat Chem Biol 9:51–58
34. Perez F, Granger BE (2007) IPython: a system for interactive scientific computing. Comput Sci Eng 9:21–29
35. Akaike H, Company NP (1981) Likelihood of a model and information criteria. J Econom 16:3–14
36. Schwarz G (1978) Estimating the dimension of a model. Ann Math Stat 6:461–464
37. Keller BG, Kobitski AY, Jaeschke A, Nienhaus GU, Noe F (2014) Complex RNA folding kinetics revealed by single molecule FRET and hidden Markov models. J Am Chem Soc 136:4534–4543

Chapter 4

Predicting Beta Barrel Transmembrane Proteins Using HMMs

Georgios N. Tsaousis, Stavros J. Hamodrakas, and Pantelis G. Bagos

Abstract

Transmembrane beta-barrels (TMBBs) constitute an important structural class of membrane proteins located in the outer membrane of gram-negative bacteria, and in the outer membrane of chloroplasts and mitochondria. They are involved in a wide variety of cellular functions and the prediction of their transmembrane topology, as well as their discrimination in newly sequenced genomes is of great importance as they are promising targets for antimicrobial drugs and vaccines. Several methods have been applied for the prediction of the transmembrane segments and the topology of beta barrel transmembrane proteins utilizing different algorithmic techniques. Hidden Markov Models (HMMs) have been efficiently used in the development of several computational methods used for this task. In this chapter we give a brief review of different available prediction methods for beta barrel transmembrane proteins pointing out sequence and structural features that should be incorporated in a prediction method. We then describe the procedure of the design and development of a Hidden Markov Model capable of predicting the transmembrane beta strands of TMBBs and discriminating them from globular proteins.

Keywords Hidden Markov model, Algorithms, Prediction, Membrane, Transmembrane, Beta barrel, Protein

1 Introduction

Transmembrane beta-barrels (TMBBs) constitute one of the two major structural classes of transmembrane proteins. They are located in the outer membrane of gram-negative bacteria, and in the outer membrane of chloroplasts and mitochondria. Their membrane-spanning segments are formed by antiparallel beta strands, creating a channel in the form of a barrel that spans the outer membrane [1]. The TMBBs perform a wide variety of functions such as active ion transport, passive nutrient uptake, membrane anchoring, adhesion, and catalytic activity [2–4]. Interestingly, a large number of pathogens belong to the gram-negative bacteria class, and the virulence activity in a lot of cases has been proven to depend on specific outer membrane proteins. Thus,

David R. Westhead and M.S. Vijayabaskar (eds.), *Hidden Markov Models: Methods and Protocols*, Methods in Molecular Biology, vol. 1552, DOI 10.1007/978-1-4939-6753-7_4, © Springer Science+Business Media LLC 2017

besides the obvious theoretical interest concerning the research on protein structure and function in general, this is an additional reason for attracting an increased medical interest.

Recent years have seen the development of several methods for predicting the transmembrane strands of outer membrane proteins and/or identifying these proteins in completely sequenced genomes. There is a large variation on the algorithmic techniques used for this purpose as they vary from hydrophobicity analysis [5–8], pattern recognition [9], statistical analysis using special empirical rules of amino-acid propensities and prior knowledge of the structural nature of the proteins [10], to other more refined methods including Neural Networks (NNs) [11–13], Hidden Markov Models [14–16], Support Vector Machines (SVMs) [17], Nearest Neighbors [18, 19], quadratic discriminant analysis [20], Radial Basis Function (RBF) Networks [21], Grammatical-Restrained Hidden Conditional Random Fields (GRHCRFs) [22], hybrid techniques [23, 24], and consensus approaches [25]. From a historical perspective, the prediction algorithms are divided in three categories; approaches based in hydrophobicity analysis, approaches that use statistical properties of the residues found in beta barrels and more advanced machine learning approaches. Moreover, there are two major classes of prediction methods, the methods aiming at predicting the location of the transmembrane beta strands and the methods aiming at discriminating TMBBs from other classes of proteins such as globular and alpha-helical membrane ones.

2 Structural Features of Beta Barrel Transmembrane Proteins

Transmembrane protein topology prediction has been pursued for many years in bioinformatics, mostly focusing on alpha helical membrane proteins. Alpha helical transmembrane segments are more easily predicted by computational methods, due to the easily detectable pattern of highly hydrophobic consecutive residues, and the application of simple rules as the "positive-inside rule" [26]. Another reason is the relative abundance of alpha helical membrane proteins compared to that of the beta barrel transmembrane proteins. Currently, the number of structures of beta barrel transmembrane proteins known at atomic resolution raises rapidly, due to improvements in the cloning and crystallization techniques [27], but still their numbers are significantly lower compared to their alpha-helical counterparts. Therefore, the investigation of the amino acid sequence of beta barrel transmembrane proteins and detailed examination of the available three-dimensional structures (Fig. 1) can provide distinct characteristics and general rules that may be used for the construction of a predictive method [1, 31]. Below we describe some simple rules and observations that should

be considered for the design of an HMM-based method for the prediction of beta-barrel transmembrane proteins [1, 31, 32]:

- The transmembrane strands are mainly amphipathic, as they exhibit alternating patterns of hydrophobic–hydrophilic residues. The hydrophobic residues interact with the nonpolar chains of membrane lipids whereas the polar residues face toward the interior of the barrel and interact with the aqueous environment of the formed pore.
- The aromatic residues have higher abundance in the interfaces with the polar heads of the lipids. This way an "aromatic belt" is formed at the lipid bilayer interface around the perimeter of the barrel (Fig. 1).
- Both the N- and C-terminal regions are located in the periplasmic space, restricting the strand number to even values. The known three-dimensional structures contain 8–24 TM strands and in some cases, the N- and C-terminal tails of the protein may be formed by more than 100 residue-long stretches.
- The segments that connect the transmembrane strands (loops) and are located in the periplasmic space ("inside") are generally shorter than those of the extracellular space ("outside").

Fig. 1 A ribbon diagram of the structure of Outer membrane protein A (ompA) from *Escherichia coli* (PDB ID: 1BXW [28]). Aromatic side chains are represented as rods to illustrate the aromatic belts. The *horizontal lines* indicate the approximate position of the lipid bilayer boundaries using the annotation of PDBTM [29] for the location of the transmembrane strands. The diagram was drawn using the PyMol molecular graphics package [30]

Specifically, as observed in most three-dimensional structures, loops of the periplasmic space have approximately a maximum length of 12 amino acid residues, while extracellular loops with more than 30 residues have been observed.

- The length of the transmembrane strands varies according to the inclination angle of the strand with respect to the barrel axis (or the lipid bilayer), and ranges between 6 and 22 residues. However, in some cases, only a small portion of the strand is embedded in the lipid bilayer, while the rest of it protrudes away from the membrane towards the extracellular space, forming flexible hairpins.
- Beta barrel transmembrane proteins show high sequence variability compared to globular proteins that adopt the structure of a beta barrel. Extracellular loops exhibit the highest sequence variability in beta barrel transmembrane proteins and often function as antigenic epitopes.
- All beta strands are antiparallel and locally connected to their neighbors through a network of hydrogen bonds which stabilizes the structure of the barrel.

Below, we will provide a short review of most available methods for the prediction of beta barrel transmembrane proteins in order to describe their special characteristics in their sequence and structure that can be exploited for the design and the development of a predictive algorithm.

3 Prediction of Beta Barrel Transmembrane Proteins

3.1 Prediction Methods Based on Hydrophobicity Analysis

The alternating patterns of hydrophobic–hydrophilic residues in transmembrane strands were initially used for the prediction of transmembrane beta barrels. Vogel and Jahnig [5] calculated the average amphipathicity of each residue using a sliding window across the amino acid sequence. Jeanteur and colleagues [6], combined amphipathicity with sequence alignments of members of the protein family of porins to determine the beta strands in the porin barrel. Their approach was extended to include specific locations of aromatic residues [8] and differences between the amphipathicity of each residue and the average hydrophobicity [7]. This way, the aromatic belt of the transmembrane stands was more accurately identified. During the same period, Gromiha and Ponnuswamy [33], derived the concept of the surrounding hydrophobicity which is independent of the amphipathic features of beta strands. Years later, the Beta Barrel Finder (BBF) method [34] combined secondary structure prediction, hydrophobicity, amphipathicity, and signal peptide prediction, in order to predict transmembrane beta barrels in bacterial genomes. Based on the analysis of known

three-dimensional structures they proposed that transmembrane segments could be identified as protein regions, which are predicted to form beta strands and are characterized by peaks of both hydrophobicity and amphipathicity. However, methods based on hydrophobicity analysis and/or secondary structure predictions have inherent limitations as beta sheet forming propensities and hydrophobicity scales are different between globular and transmembrane proteins [35]. Such issues should be taken into consideration when developing and using empirical predictive methods.

3.2 Statistical Approaches

Sequence features other than hydrophobicity profiles were incorporated in the prediction of transmembrane beta strands with the use of statistical approaches. Gromiha and colleagues [10], developed a set of conformational parameters for membrane spanning beta strands and described special empirical rules using amino acid propensities and prior knowledge of the structural nature of the proteins. Neuwald and colleagues [9], employed a Gibbs sampling algorithm to detect motif-encoding regions and similar repetitive motifs that characterize bacterial porins. Other approaches used multiple structural alignments [36], in order to describe certain patterns in transmembrane beta barrels that could be used for their discrimination. Wimley developed a scale-based statistical method [32] by aligning the structures with respect to the lipid bilayer and calculating amino acid propensities for residues to belong to a transmembrane beta strand or a non-transmembrane region. The original algorithm was later modified (the "Freeman–-Wimley algorithm") to improve the genome scale discrimination of beta barrel transmembrane proteins [37]. On the other hand, Liu and colleagues [38] calculated differences in the amino acid composition between the two classes beta barrel membrane proteins and globular proteins for discrimination purposes. The BOMP method [39] combined regular expression patterns (in particular, a C-terminal pattern characteristic of most TMBBs) with a post processing step to filter false positives based on the overall amino acid composition, whereas the method of Wimley [32] used an additional step for the filtering of false positive predictions based on the overall amino acid composition of the protein. As the number of available three-dimensional structures continued to grow, it became obvious that the problem of the prediction of beta barrel transmembrane proteins is more complicated from the simple identification of the alternating patterns of hydrophobic–hydrophilic residues.

3.3 Machine Learning Methods

Machine Learning methods are in general more capable of capturing the nonlinear correlations of amino acids in protein sequences and perform better than statistical analyses and heuristic methods. Furthermore, the mathematical foundation of these methods is sounder and elegant predictors can be developed. Some of these

methods are based solely on the amino acid sequence and others use evolutionary information derived from multiple alignments.

The first application of Machine Learning techniques for the prediction of the topology of beta barrel transmembrane proteins used a Feed-Forward Neural Network [11], to predict the relative position of C-alpha atoms with respect to the lipid bilayer. Jacoboni and colleagues [13] described the use of a similar Neural Network (B2TMPRED) which introduced the use of evolutionary information in the form of multiple sequence alignments generated by PSI-BLAST [40]. Moreover, the method included an algorithm for model optimization, based on dynamic programming in order to define more efficiently the ends of transmembrane segments and filter inconsistent predictions. A Neural Network with similar architecture, using only single sequence information as input, was presented with the TMBETA-NET method [12], which used a set of empirical rules to remove spurious predictions (e.g., strands with 3–4 residues and so on). Later, the TBBPred method [24] combined Neural Networks and Support Vector Machines (SVMs) for the prediction of transmembrane regions of beta-barrel proteins using a similar NN with B2TMPRED.

The first method based on Hidden Markov Models for the prediction of the topology of beta barrel transmembrane proteins was the HMMB2TMR method [16]. The method was trained on a dataset of 12 nonredundant outer membrane proteins, using as input multiple sequence alignments generated by PSI-BLAST. It used a HMM with different states to describe the alternating patterns of hydrophobic–hydrophilic residues in transmembrane strands, the aromatic belt located at the lipid bilayer interface and the different periplasmic and extracellular loops. The ProfTMB method [15], also included evolutionary information in the form of multiple alignments and model training and scoring procedures similar to HMMB2TMR. However, the method introduced a novel HMM architecture, different structure-based labeling, a new definition of beta-hairpin motifs, explicit state modeling of transmembrane strands, and a log-odds whole-protein discrimination score. The PRED-TMBB [41] method was the first publicly available through a web-server method for predicting the topology of TMBBs using HMMs, introducing novel training and decoding procedures. A consensus prediction method has also been described (ConBBPRED) [25], that combines the results of several methods and optimizes the predicted topology with a dynamic programming algorithm. Recently, BOCTOPUS [23] employed a hybrid method based on SVMs and HMMs and combined local per-residue predictions with global preferences. The method also incorporated a discrimination filter for more reliable identification of beta barrel transmembrane proteins.

Other methods such as TMBpro [42] and transFold [43] also predict potential interactions between the residues of beta strands

that form the barrel using HMMs and multi-tape S-attribute grammars respectively. TMB-Hunt [18] employs a modified k-nearest neighbor (k-NN) algorithms to classify protein sequences as transmembrane beta-barrel. The HHomp method [44] uses a database of profile Hidden Markov Models (pHMMs) to perform sensitive sequence similarity searches through profile-profile alignments, based on the observation that almost all beta barrel outer membrane proteins have a common ancestry.

Below we describe the basic concepts for the design of a Hidden Markov Model capable of predicting the topology of beta barrel transmembrane proteins and discriminating them from other classes of proteins.

4 A Hidden Markov Model for Beta Barrel Transmembrane Proteins

4.1 Basic Concepts of Hidden Markov Models

Hidden Markov Models have been extensively used for pattern recognition problems, with the most known example found in the speech recognition methodology [45]. Hidden Markov Models have been used in bioinformatics during the last few years for generating probabilistic profiles for protein families [46], the prediction of transmembrane helices in proteins [47], the prediction of signal peptides and their cleavage sites [48], the prediction of genes [49] and for the prediction of transmembrane beta strands [14, 16, 38].

The Hidden Markov Model is a probabilistic model, which consists of several states, connected by means of the transition probabilities, thus forming a Markov process. If we consider an amino acid sequence of a protein with length L, denoted by:

$$\mathrm{x} = x_1, x_2, \ldots, x_{L-1}, x_L$$

with a labeling (in this case corresponding to transmembrane, intracellular, and extracellular regions):

$$\mathrm{y} = y_1, y_2, \ldots, y_{L-1}, y_L$$

then the transition probability for jumping from a state k to a state l is defined as:

$$\alpha_{kl} = P(\pi_i = l | \pi_{i-1} = k)$$

Where π is the "path" in the particular position of the amino acid sequence (i.e., the sequence of states, as opposed to the sequence of symbols). Each state k is associated with a distribution of emission probabilities, meaning the probabilities that any particular symbol could be emitted by the current state. Assuming an alphabet Σ, consisting of the 20 amino acids, the probability that a particular aminoacid b is emitted from state k is defined as:

$$e_k(b) = P(x_i = b | \pi_i = k)$$

The term "hidden" is justified by the fact that when one observes the emitted symbols he cannot see the underlying states, thus the true state process is hidden from the observer. The total probability of the observation sequence given the model, $P(\mathbf{x}|\theta)$, is computed using the efficient forward algorithm [45], whereas the joint probability of the sequence and the labeling denoted by $P(\mathbf{x},\mathbf{y}|\theta)$, is computed by its trivial modification proposed by Krogh [50].

4.2 The Hidden Markov Model Architecture

The architecture of the HMM is chosen so that it could fit as much as possible to the limitations imposed by the known structures. HMM-B2TMR uses a HMM with different types of states to describe the beta strand transmembrane core (two types), the beta strand cap on either side of the membrane, the inner loops, the outer loops, and the globular domain in the middle of each loop [16]. On the other hand, the ProfTMB method [15] uses different states to model explicitly different types of periplasmic loops (turns, hairpins etc). PRED-TMBB, used a Hidden Markov Model of similar architecture, which is described in Fig. 2. The model is cyclic, consisting of 61 states divided in three "sub-models" corresponding to the three desired labels to predict the TM (transmembrane) strand sub-model and the inner (periplasmic) and outer (extracellular) loops sub-models respectively [14].

The TM strand model incorporates states to model the special architecture of the transmembrane strands. Thus, there are states that correspond to the core of the strand and the aromatic belt located at the lipid bilayer interface. Furthermore, other states correspond to the amino acid residues facing the bilayer (the external side of the barrel) and the residues facing the barrel

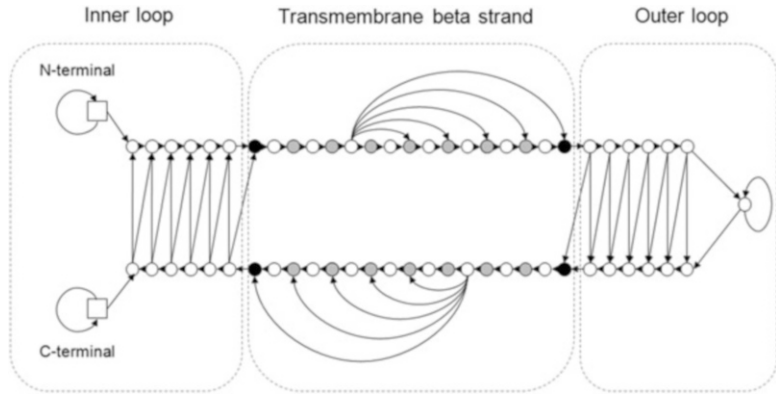


Fig. 2 A schematic representation of the model's architecture. The model consists of three sub-models corresponding to the three labels which shown separately. In the transmembrane sub-model different colors correspond to the tied states. *Black circles* correspond to the aromatic belt, gray to exterior of the strand's core, and white to the interior side. In the inner and outer loop sub-models, the states forming the ladder are tied together respectively, whereas the N-terminal tail is tied with the C-terminal and the globular outer loop state is not tied with another state. The allowed transitions are shown with *arrows*

interior. All states are connected with the appropriate transition probabilities in order to be consistent with the known structures (i.e., to ensure appropriate length distributions and to model the alternating pattern of hydrophobic–non-hydrophobic residues, corresponding to the external-internal residues of the barrel). The minimum allowed length for a transmembrane strand is seven residues, whereas the maximum is 17. However, more variable lengths can and have been used in other published methods such as HMM-B2TMR and ProfTMB.

The inner and outer loops are modeled with a "ladder" architecture of 12 states, whereas at the top of the outer loop there is a self transitioning state corresponding to residues too distant from the membrane; these cannot be modeled as loops, hence that state is named "globular." The "inner" loop sub-model has no corresponding "globular" state, reflecting the fact that inner loops are significantly shorter than the outer ones, since none of the known structures at that time, possessed an inner loop longer than 12 residues. In the future, a globular state can be added so as to cover special cases where longer loops have been observed. In order to capture the fact that all known structures are having their N-terminal region towards the periplasmic space (the "inside" with respect to the outer membrane) we allow the begin state of the model to be followed only by states belonging to the inner loop or to TM strands directing to the external side of the outer membrane. Additionally an end state can be included to fix the location of the C-terminal region at the same side. Finally, we allow a self-transitioning absorbing state to follow the inner loop states, in order to correctly model sequences that have a long C-terminus falling in the periplasmic space. States expected to have the same emission probabilities are tied together. However, different emission probabilities can be used [15] to model the strands with direction from the periplasmic to the extracellular space and strands with the opposite direction. This way the asymmetric amino acid compositions of transmembrane beta-strands [51, 52] are incorporated in the model. Finally, the emission probabilities of some of the states belonging to the extracellular loops can be modified to include the "positive outside rule" which describes higher preference of charged residues in the extracellular loops [53].

4.3 Creation of Training and Testing Datasets

For the creation of a training dataset for the HMM, a set of beta barrel transmembrane proteins with known three dimensional structure and topology must be compiled. In the original publication of PRED-TMBB [14], protein structures from the Protein Data Bank (PDB) [54] were collected after consideration of the SCOP classification [55] and in particular using the fold "Transmembrane beta-barrels." However, with the creation of specialized databases for TMBBs, such a dataset can now be compiled using information deposited in PDBTM [29], OPM [56] or TOPDB

[57] where structural and experimental data are combined. For variants of the same protein, only the structure solved at the highest resolution is collected, and multiple identical chains are removed, keeping only one chain for each structure. The sequences of the remaining structures are submitted to a redundancy check using BLAST [40], removing chains with a sequence identity above some threshold (typically 30 %).

The total number of freely estimated parameters in the model in Fig. 2 is 175. These numbers are adequate for training a prediction method using some dozens of proteins (i.e., using thousands of amino acids as observations) and in any case are significantly lower compared to the number of freely estimated parameters (weights) needed by a NN method. Most available HMM-based methods for TMBBs have used approximately 8 to 36 proteins in their training sets. Larger sets are not expected to increase the prediction performance for a given algorithmic technique. It has been shown [58] that the relationship between the sizes of the training set with the performance of the prediction algorithms is nonlinear and reaches an upper limit, even in the case of transmembrane beta barrel prediction. Following this rationale and due to the fact that the number of available three dimensional structures for transmembrane beta barrels continues to rise, a more refined training set can be compiled by collecting a structural representative from each known family of transmembrane beta barrels [59]. A complete and comprehensive collection and classification of protein families for TMBBs can be obtained from databases as Pfam [60] or OMPdb [4] where each protein family is described using pHMMs.

It is important to point out that even in structures known at atomic resolution, the exact boundaries of the transmembrane strands are not obvious, and in some situations the PDB annotations for the strands are clearly extending far beyond the membrane. Since the primary objective is to predict the TM segments of the strands rather than the entire beta strands, the model ideally must be trained to identify these particular segments. In most of the earlier works [13, 15, 16, 61], the PDB annotations for the whole strand were used but this may cause problems in the performance of the predictor. Alternatively, a manual approach described in [14] can be applied, where the labels for the TM segments are set manually, by precisely locating the aromatic belts of the barrel [31] and the residues facing the barrel interior and exterior, after inspection of the three-dimensional structures of the proteins in the training set, using molecular graphics software (Fig. 1). It is well known that discriminative training algorithms (see below) are very sensitive to data mislabeling, thus this approach was at least in part responsible for the increased performance of PRED-TMBB. More recent works however, rely on the TM assignments deposited in public databases such as PDBTM [29], which are generated by algorithms that identify the precise location of the TM part of the barrel, and thus take these considerations into account. Finally, we

have to note that in [47] an automated method for relabeling the data was proposed, that consists of removing the labels at the boundaries of the TM strands, and then performing a constrained prediction that reassigns the boundaries.. This method has been used previously only in the case of alpha-helical TM proteins, and we expect that in the case of TMBBs, it will also perform well.

For the evaluation of the prediction performance of the HMM a self-consistency test and a cross-validation procedure is commonly performed. In the cross-validation procedure the training set is divided in equal subsets (folds). The model is trained with one subset of proteins and the prediction performance is evaluated against the remaining subsets. This way, prediction results for each protein in the training dataset are derived from models where the protein was not included in the training process. For small training sets measures of accuracy can be obtained using a jackknife test (leave one out cross-validation test). In addition, several test sets are compiled for the evaluation of the performance of the predictive model. An independent test set of beta barrel transmembrane proteins having experimentally verified topologies and no sequence similarity with the proteins used in the training set can be used to evaluate the topology prediction performance.

To assess the accuracy of the predictions, several measures can be used. For the transmembrane strand predictions the number of correctly predicted strands (True Positives, TP), the number of missed strands (False Negatives, FN), and the number of the over-predicted strands (False Positives, FP) are calculated. However, the strands' prediction accuracy is more efficiently measured by the segments overlap measure (SOV)Segments Overlap Measure (SOV) [62]. As measures of the accuracy per residue, the total fraction of the correctly predicted residues (Q_β or Q_3), in a two-state model (transmembrane versus non-transmembrane) or a three-state model (transmembrane versus inner loop versus outer loop), and the Matthews Correlation Coefficient (C_β) can be used [63]. In addition, the number of the correctly predicted transmembrane segments and topologies (i.e., when both strands' localization and orientation of the loops are correctly predicted) can be calculated to evaluate the performance per protein.

The discriminative power of the model can also be evaluated against a nonredundant set of globular proteins with known three-dimensional structure similar to the one used in [37] and against a negative set of alpha helical transmembrane proteins as used in [14].

4.4 Training and Decoding Algorithms

Traditionally, the parameters of a Hidden Markov Model are optimized according to the Maximum Likelihood criterion [45],

$$\hat{\theta}^{\mathrm{ML}} = \underset{\theta}{\operatorname{argmax}}\, P(\mathrm{x}|\theta)$$

A widely used algorithm for this task is the efficient Baum–Welch algorithm (also known as Forward–Backward) [45, 64], which is a special case of the Expectation-Maximization (EM) algorithm, proposed for Maximum Likelihood (ML) estimation for incomplete data [65] and has been widely used for the training of several HMM methods for TMBBs [15, 16]. The algorithm, updates iteratively the model parameters (emission and transition probabilities), with the use of their expectations, computed with the use of the Forward and Backward algorithms. Convergence to at least a local maximum of the likelihood is guaranteed. The main disadvantage of ML training is that it is not discriminative. Alternatively, the Conditional Maximum Likelihood (CML) training for labeled data can be used, as proposed by Krogh [66]. Although CML is computationally more intensive than the ML approach, the predictive ability is better when data with good quality of labeling are used. The Conditional Maximum Likelihood criterion is:

$$\hat{\theta}^{\mathrm{CML}} = \underset{\theta}{\operatorname{argmax}}\, P(\mathrm{y}|\mathrm{x},\theta) = \underset{\theta}{\operatorname{argmax}}\, \frac{P(\mathrm{x},\mathrm{y}|\theta)}{P(\mathrm{x}|\theta)}$$

This kind of training, often referred to as discriminating training, seeks to maximize the probability of the correct prediction, i.e., the probability of the labeling **y** for a given sequence **x** and a model θ. Another major problem with CML is that the Baum–Welch algorithm cannot be applied and variants of the gradient descent method are needed. These methods require parameter fine-tuning (for the so-called "learning rate") and in many cases do not provide stable convergence. However, a variant that uses individual learning rates that are adapted during the process, has been presented, that offers significant advantages [67]. The CML training with the above-mentioned algorithms has been efficiently used in [14]. The parameters of the model (transition and emission probabilities) are updated simultaneously, using the gradients of the likelihood function as described in [68], and the training process terminates when the likelihood does not increase beyond a prespecified threshold. In addition, a genetic algorithm (GA) has been used [69] to train the model, demonstrating better results compared to Baum–-Welch. To reduce the number of the free parameters of the model, and thus improve the generalization capability, states expecting to have the same emission probabilities, can be tied together (Fig. 2). Furthermore, to avoid overfitting, the iterations started from emission probabilities corresponding to the initial amino acid frequencies observed in the known protein structures and small pseudocounts can be added in each step.

The decoding of an HMM can performed using the standard Viterbi algorithm [45] or alternatively the N-best algorithm [70], as formulated in [66]. This algorithm is a heuristic that attempts to find the most probable labeling of a given sequence, as opposed to

the well-known Viterbi algorithm [45], which guarantees to find the most probable path of states. Since there are several states contributing to the same labeling of a given sequence, the N-best algorithm will always produce a labeling with a probability at least as high as that computed by the Viterbi algorithm, in other words it always returns equal if not better results. Its main drawback is the memory requirements and computational complexity, resulting in a slowdown of the decoding process. The original PRED-TMBB method, offered three choices: the standard Viterbi algorithm, the N-best algorith and a variant of the posterior decoder coupled with a post-processing step using dynamic programming. This algorithm has been used by Jacoboni and coworkers [13] and later presented as a general purpose algorithm [61]. However, novel decoding algorithms have been presented later, that combine the advantages of both Viterbi and posterior decoding, in a more mathematical sound way. These are the optimal accuracy posterior decoder (OAPD) [71] and the Posterior-Viterbi algorithm [72], already deployed in the topology prediction of transmembrane proteins. From our experience, these algorithms perform better compared to Viterbi, N-best, and posterior decoding, and should be preferred in future applications.

For the purpose of discrimination, the information included in the prediction of the putative transmembrane segments is not sufficient, since a prediction for a transmembrane strand could occur even in globular proteins. Thus, there is need for a global score reflecting the overall fit of the query sequence to the model. This can be derived from the negative log-likelihood of the sequence given the model, as computed by the Forward algorithm. For models trained with the ML criterion, the score is usually normalized by dividing with the likelihood of a null model, that is, a model of independence with amino acid frequencies derived from Uniprot. This is usually named "log-odds score" and is given by:

$$S(\mathrm{x}|\theta) = \frac{\log P(\mathrm{x}|\theta)}{\log P(\mathrm{x}|\mathrm{null})}$$

The log-odds score however, cannot be used for models trained with the CML criterion. In this case the likelihood is usually normalized for the length of the sequence. Thus, the statistical score used for discrimination by the PRED-TMBB method is:

$$S(\mathrm{x}|\theta) = \frac{\log P(\mathrm{x}|\theta)}{L}$$

where L is the length of the sequence. The proportion of correctly classified proteins as a function of the discrimination score used as the threshold should then be studied in order to define the optimal

threshold as the value that maximizes that function. Proteins with score values below the threshold should be declared as beta-barrel transmembrane proteins. In general, the discrimination performance of such scores range at the vicinity of 90 %, and thus we expect in future applications that these scores be combined with other relative metrics (such as the number of predicted strands) in order to increase the performance.

5 Further Considerations and Improvements

As we already discussed in the respective section, improvements in prediction performance can be achieved by designing a more plausible model architecture. Currently, the potential improvements in this respect may include different emission probabilities to model the strands with direction from the periplasmic to the extracellular space and strands with the opposite direction; this way the asymmetric amino acid compositions of transmembrane beta-strands [51, 52] can be incorporated in the model. Similarly, the emission probabilities of some of the states belonging to the extracellular loops can be modified to include the "positive outside rule" which describes higher preference of charged residues in the extracellular loops [53]. Finally, the whole structure of the model may be optimized. Methods that learn the structure of the HMM using genetic algorithms, have been proposed [73, 74], and applications concerning the prediction of TMBBs may needed.

An extensive comparison and evaluation of methods for predicting the topology of beta barrel transmembrane proteins indicated that HMM-based methods outperform methods based on other types of machine learning techniques such as NNs and SVMs [25]. The regular grammar of the HMMs can capture more effectively the temporal variability of the protein sequence and map successfully the proteins modular nature to a mathematical sound model. HMM-based methods are not influenced significantly whether full-length sequences or just the beta barrel domains are submitted as input for prediction. Interestingly, the NN- and SVM-based methods, often falsely predict the signal peptide sequences as transmembrane strands in the precursors whereas HMMs do not. However, this can be solved by using a specialized signal peptide prediction method such as SignalP [75] or with the inclusion of an additional submodel in the HMM as described in [69]. Moreover, NN methods are more capable of capturing long-range correlations along the sequence. This results to the correct identification of an isolated strand, but since the beta barrel proteins follow strict structural rules, the modular nature of the barrels is captured more effectively by HMMs. NNs may often falsely predict isolated transmembrane strands in non-barrel domains or predict strands

with a non-plausible number of residues or even barrels with an odd number of strands. A potential improvement in future applications can be reached by exploiting hybrid methods that combine the advantages of HMMs and those of the NNs. BOCTOPUS used such a method since it combined an SVM with a HMM-like model. However, this grammar is not a proper HMM since the transitions were not optimized, and hence there is plenty room for improvement. Such hybrid methods, are usually termed Hidden Neural Networks (HNNs) [68], but up to date, there are only a handful of applications in bioinformatics [76, 77]. Clearly, such approaches should be attractive alternatives for future applications.

Other improvements can be applied to existing HMM based prediction methods for beta barrel transmembrane proteins by utilizing algorithmic techniques and decoding algorithms that were successfully applied in the prediction of the topology of alpha helical TM proteins. Using the modifications to the standard Forward and Backward algorithms, as well as on all the above mentioned decoding algorithms for HMMs, that were extensively described in [78], prior topological information derived from experiments can be incorporated in the prediction. Subsequently, constrained predictions for TMBBs will produce more reliable topological models. Constrained predictions can also be used for the refinement of the labeling of the ends of transmembrane strands and better definition of relevant emission probabilities.

Finally, as already mentioned, some of the prediction methods use evolutionary information in the form of multiple sequence alignments [13, 15, 16, 23, 24]. The inclusion of evolutionary information in HMM based methods has been shown to substantially improve the prediction performance [16, 79]. However, most implementations used in beta barrel TM protein prediction include the construction of a sequence profile from multiple alignments that is used as an input. These methods were feasible by extending the simple HMM in a way that accepts as input a sequence profile instead of sequence. Thus, other methods such as PRED-TMBB may be difficult to adopt a similar architecture. However, a different method can be used following [71]. Briefly, given a query sequence and a multiple alignment of its homologs, predictions with the single sequence method can be obtained on each of the homologs. Then, the predicted labels can be mapped on the alignment and averaged for each position of the query sequence. This will create a "posterior label probability" (PLP) table for the query sequence that contains information from the multiple alignments. In the last step, the OAPD [71] can be applied and the final prediction is obtained. This approach, has the advantage that can be used in any single-sequence method without further modifications.

References

1. Schulz GE (2003) Transmembrane beta-barrel proteins. Adv Protein Chem 63:47–70
2. Wimley WC (2003) The versatile beta-barrel membrane protein. Curr Opin Struct Biol 13 (4):404–411
3. Bagos PG, Hamodrakas SJ (2009) Bacterial beta-barrel outer membrane proteins: a common structural theme implicated in a wide variety of functional roles. In: Daskalaki A (ed) Handbook of research on systems biology applications in medicine, pp: 182–207. doi:10.4018/978–1-60566-076-9.ch010
4. Tsirigos KD, Bagos PG, Hamodrakas SJ (2011) OMPdb: a database of {beta}-barrel outer membrane proteins from Gram-negative bacteria. Nucleic Acids Res 39(Database issue): D324–D331. doi:10.1093/nar/gkq863
5. Vogel H, Jahnig F (1986) Models for the structure of outer-membrane proteins of Escherichia coli derived from Raman spectroscopy and prediction methods. J Mol Biol 190(2):191–199, doi:0022-2836(86)90292-5 [pii]
6. Jeanteur D, Lakey JH, Pattus F (1991) The bacterial porin superfamily: sequence alignment and structure prediction. Mol Microbiol 5(9):2153–2164
7. Rauch G, Moran O (1995) Prediction of polypeptide secondary structures analysing the oscillation of the hydropathy profile. Comput Methods Programs Biomed 48(3):193–200, doi:0169260795016988 [pii]
8. Schirmer T, Cowan SW (1993) Prediction of membrane-spanning beta-strands and its application to maltoporin. Protein Sci 2 (8):1361–1363. doi:10.1002/pro.5560020820
9. Neuwald AF, Liu JS, Lawrence CE (1995) Gibbs motif sampling: detection of bacterial outer membrane protein repeats. Protein Sci 4 (8):1618–1632. doi:10.1002/pro.5560040820
10. Gromiha MM, Majumdar R, Ponnuswamy PK (1997) Identification of membrane spanning beta strands in bacterial porins. Protein Eng 10(5):497–500
11. Diederichs K, Freigang J, Umhau S et al (1998) Prediction by a neural network of outer membrane beta-strand protein topology. Protein Sci 7(11):2413–2420. doi:10.1002/pro.5560071119
12. Gromiha MM, Ahmad S, Suwa M (2004) Neural network-based prediction of transmembrane beta-strand segments in outer membrane proteins. J Comput Chem 25 (5):762–767. doi:10.1002/jcc.10386
13. Jacoboni I, Martelli PL, Fariselli P et al (2001) Prediction of the transmembrane regions of beta-barrel membrane proteins with a neural network-based predictor. Protein Sci 10 (4):779–787. doi:10.1110/ps.37201
14. Bagos PG, Liakopoulos TD, Spyropoulos IC et al (2004) A Hidden Markov Model method, capable of predicting and discriminating beta-barrel outer membrane proteins. BMC Bioinformatics 5:29. doi:10.1186/1471-2105-5-29
15. Bigelow HR, Petrey DS, Liu J et al (2004) Predicting transmembrane beta-barrels in proteomes. Nucleic Acids Res 32(8):2566–2577. doi:10.1093/nar/gkh580
16. Martelli PL, Fariselli P, Krogh A et al (2002) A sequence-profile-based HMM for predicting and discriminating beta barrel membrane proteins. Bioinformatics 18(Suppl 1):S46–S53
17. Park KJ, Gromiha MM, Horton P et al (2005) Discrimination of outer membrane proteins using support vector machines. Bioinformatics 21(23):4223–4229. doi:10.1093/bioinformatics/bti697
18. Garrow AG, Agnew A, Westhead DR (2005) TMB-Hunt: an amino acid composition based method to screen proteomes for beta-barrel transmembrane proteins. BMC Bioinformatics 6:56. doi:10.1186/1471-2105-6-56
19. Yan C, Hu J, Wang Y (2008) Discrimination of outer membrane proteins using a K-nearest neighbor method. Amino Acids 35(1):65–73. doi:10.1007/s00726-007-0628-7
20. Lin H (2008) The modified Mahalanobis Discriminant for predicting outer membrane proteins by using Chou's pseudo amino acid composition. J Theor Biol 252(2):350–356. doi:10.1016/j.jtbi.2008.02.004
21. Ou YY, Gromiha MM, Chen SA et al (2008) TMBETADISC-RBF: discrimination of beta-barrel membrane proteins using RBF networks and PSSM profiles. Comput Biol Chem 32 (3):227–231. doi:10.1016/j.compbiolchem.2008.03.002
22. Fariselli P, Savojardo C, Martelli PL et al (2009) Grammatical-restrained hidden conditional random fields for bioinformatics applications. Algorithms Mol Biol 4:13. doi:10.1186/1748-7188-4-13
23. Hayat S, Elofsson A (2012) BOCTOPUS: improved topology prediction of transmembrane beta barrel proteins. Bioinformatics 28 (4):516–522. doi:10.1093/bioinformatics/btr710
24. Natt NK, Kaur H, Raghava GP (2004) Prediction of transmembrane regions of beta-barrel

proteins using ANN- and SVM-based methods. Proteins 56(1):11–18. doi:10.1002/prot.20092

25. Bagos PG, Liakopoulos TD, Hamodrakas SJ (2005) Evaluation of methods for predicting the topology of beta-barrel outer membrane proteins and a consensus prediction method. BMC Bioinformatics 6:7. doi:10.1186/1471-2105-6-7
26. von Heijne G (1992) Membrane protein structure prediction. Hydrophobicity analysis and the positive-inside rule. J Mol Biol 225 (2):487–494
27. Bannwarth M, Schulz GE (2003) The expression of outer membrane proteins for crystallization. Biochim Biophys Acta 1610(1):37–45, doi:S0005273602007113 [pii]
28. Pautsch A, Schulz GE (1998) Structure of the outer membrane protein A transmembrane domain. Nat Struct Biol 5(11):1013–1017. doi:10.1038/2983
29. Kozma D, Simon I, Tusnady GE (2013) PDBTM: Protein Data Bank of transmembrane proteins after 8 years. Nucleic Acids Res 41(Database issue):D524–D529. doi:10.1093/nar/gks1169
30. Delano WL (2002) The PyMOL molecular graphics system. http://www.pymol.org
31. Schulz GE (2002) The structure of bacterial outer membrane proteins. Biochim Biophys Acta 1565(2):308–317, doi:S0005273602005771 [pii]
32. Wimley WC (2002) Toward genomic identification of beta-barrel membrane proteins: composition and architecture of known structures. Protein Sci 11(2):301–312. doi:10.1110/ps.29402
33. Gromiha MM, Ponnuswamy PK (1993) Prediction of transmembrane beta-strands from hydrophobic characteristics of proteins. Int J Pept Protein Res 42(5):420–431
34. Zhai Y, Saier MH Jr (2002) The beta-barrel finder (BBF) program, allowing identification of outer membrane beta-barrel proteins encoded within prokaryotic genomes. Protein Sci 11(9):2196–2207. doi:10.1110/ps.0209002
35. Bishop CM, Walkenhorst WF, Wimley WC (2001) Folding of beta-sheets in membranes: specificity and promiscuity in peptide model systems. J Mol Biol 309(4):975–988. doi:10.1006/jmbi.2001.4715
36. Gnanasekaran TV, Peri S, Arockiasamy A et al (2000) Profiles from structure based sequence alignment of porins can identify beta stranded integral membrane proteins. Bioinformatics 16 (9):839–842
37. Freeman TC Jr, Wimley WC (2010) A highly accurate statistical approach for the prediction of transmembrane beta-barrels. Bioinformatics 26(16):1965–1974. doi:10.1093/bioinformatics/btq308
38. Liu Q, Zhu Y, Wang B et al (2003) Identification of beta-barrel membrane proteins based on amino acid composition properties and predicted secondary structure. Comput Biol Chem 27(3):355–361, doi:S1476927102000853 [pii]
39. Berven FS, Flikka K, Jensen HB et al (2004) BOMP: a program to predict integral beta-barrel outer membrane proteins encoded within genomes of Gram-negative bacteria. Nucleic Acids Res 32(Web Server issue):W394–W399. doi:10.1093/nar/gkh351
40. Altschul SF, Madden TL, Schaffer AA et al (1997) Gapped BLAST and PSI-BLAST: a new generation of protein database search programs. Nucleic Acids Res 25(17):3389–3402, doi:gka562 [pii]
41. Bagos PG, Liakopoulos TD, Spyropoulos IC et al (2004) PRED-TMBB: a web server for predicting the topology of beta-barrel outer membrane proteins. Nucleic Acids Res 32 (Web Server issue):W400–W404. doi:10.1093/nar/gkh417
42. Randall A, Cheng J, Sweredoski M et al (2008) TMBpro: secondary structure, beta-contact and tertiary structure prediction of transmembrane beta-barrel proteins. Bioinformatics 24 (4):513–520. doi:10.1093/bioinformatics/btm548
43. Waldispuhl J, Berger B, Clote P et al (2006) transFold: a web server for predicting the structure and residue contacts of transmembrane beta-barrels. Nucleic Acids Res 34(Web Server issue):W189–193. doi:10.1093/nar/gkl205
44. Remmert M, Linke D, Lupas AN et al (2009) HHomp—prediction and classification of outer membrane proteins. Nucleic Acids Res 37(Web Server issue):W446–W451. doi:10.1093/nar/gkp325
45. Rabiner L (1989) A tutorial on hidden Markov models and selected applications in speech recognition. Proc IEEE 77(2):257–286
46. Eddy SR (1998) Profile hidden Markov models. Bioinformatics 14(9):755–763, doi:btb114 [pii]
47. Krogh A, Larsson B, von Heijne G et al (2001) Predicting transmembrane protein topology with a hidden Markov model: application to complete genomes. J Mol Biol 305(3):567–580. doi:10.1006/jmbi.2000.4315

48. Nielsen H, Krogh A (1998) Prediction of signal peptides and signal anchors by a hidden Markov model. Proc Int Conf Intell Syst Mol Biol 6:122–130
49. Krogh A, Mian IS, Haussler D (1994) A hidden Markov model that finds genes in E. coli DNA. Nucleic Acids Res 22 (22):4768–4778
50. Krogh A (1994) Hidden Markov models for labelled sequences. In: Proceedings of the12th IAPR international conference on pattern recognition, pp 140–144
51. Chamberlain AK, Bowie JU (2004) Asymmetric amino acid compositions of transmembrane beta-strands. Protein Sci 13(8):2270–2274
52. Slusky JS, Dunbrack RL Jr (2013) Charge asymmetry in the proteins of the outer membrane. Bioinformatics 29(17):2122–2128. doi:10.1093/bioinformatics/btt355
53. Jackups R Jr, Liang J (2005) Interstrand pairing patterns in beta-barrel membrane proteins: the positive-outside rule, aromatic rescue, and strand registration prediction. J Mol Biol 354 (4):979–993. doi:10.1016/j.jmb.2005.09.094
54. Berman HM, Westbrook J, Feng Z et al (2000) The Protein Data Bank. Nucleic Acids Res 28 (1):235–242, doi:gkd090 [pii]
55. Andreeva A, Howorth D, Brenner SE et al (2004) SCOP database in 2004: refinements integrate structure and sequence family data. Nucleic Acids Res 32(Database issue): D226–D229. doi:10.1093/nar/gkh039
56. Lomize MA, Lomize AL, Pogozheva ID et al (2006) OPM: orientations of proteins in membranes database. Bioinformatics 22 (5):623–625. doi:10.1093/bioinformatics/btk023
57. Dobson L, Lango T, Remenyi I et al (2015) Expediting topology data gathering for the TOPDB database. Nucleic Acids Res 43(Database issue):D283–D289. doi:10.1093/nar/gku1119
58. Bagos PG, Tsaousis GN, Hamodrakas SJ (2009) How many 3D structures do we need to train a predictor? Genomics Proteomics Bioinformatics 7(3):128–137. doi:10.1016/S1672-0229(08)60041-8
59. Bagos PG, Hamodrakas SJ (2009) Bacterial beta-barrel outer membrane proteins: a common structural theme implicated in a wide variety of functional roles. In: Daskalaki A (ed) Handbook of research on systems biology applications in medicine, pp 182–207. doi:10.4018/978–1-60566-076-9.ch010
60. Punta M, Coggill PC, Eberhardt RY et al (2012) The Pfam protein families database. Nucleic Acids Res 40(Database issue): D290–D301. doi:10.1093/nar/gkr1065
61. Fariselli P, Finelli M, Marchignoli D et al (2003) MaxSubSeq: an algorithm for segment-length optimization. The case study of the transmembrane spanning segments. Bioinformatics 19(4):500–505
62. Zemla A, Venclovas C, Fidelis K et al (1999) A modified definition of Sov, a segment-based measure for protein secondary structure prediction assessment. Proteins 34(2):220–223. doi:10.1002/(SICI)1097-0134(19990201)34:2<220::AID-PROT7>3.0.CO;2-K [pii]
63. Baldi P, Brunak S, Chauvin Y et al (2000) Assessing the accuracy of prediction algorithms for classification: an overview. Bioinformatics 16(5):412–424
64. Baum LE (1972) An inequality and associated maximization technique in statistical estimation for probabilistic functions of Markov processes. Inequalities 3:1–8
65. Dempster AP, Laird NM, Rubin DB (1977) Maximum likelihood from incomplete data via the EM algorithm. J R Stat Soc Ser B Methodol 39(1):1–38. doi:10.2307/2984875
66. Krogh A (1997) Two methods for improving performance of an HMM and their application for gene finding. Proc Int Conf Intell Syst Mol Biol 5:179–186
67. Bagos P, Liakopoulos T, Hamodrakas S (2004) Faster gradient descent training of hidden Markov models, using individual learning rate adaptation. In: Paliouras G, Sakakibara Y (eds) Grammatical inference: algorithms and applications, vol 3264, Lecture notes in computer science. Springer, Berlin, Heidelberg, pp 40–52. doi:10.1007/978-3-540-30195-0_5
68. Krogh A, Riis SK (1999) Hidden neural networks. Neural Comput 11(2):541–563
69. Zou L, Wang Z, Wang Y et al (2010) Combined prediction of transmembrane topology and signal peptide of beta-barrel proteins: using a hidden Markov model and genetic algorithms. Comput Biol Med 40 (7):621–628. doi:10.1016/j.compbiomed.2010.04.006
70. Schwartz R, Chow YL (1990) The N-best algorithms: an efficient and exact procedure for finding the N most likely sentence hypotheses. In: 1990 international conference on acoustics, speech, and signal processing, 1990. ICASSP-90, 3–6 Apr 1990, vol 81, pp 81–84. doi:10.1109/icassp.1990.115542
71. Kall L, Krogh A, Sonnhammer EL (2005) An HMM posterior decoder for sequence feature prediction that includes homology

information. Bioinformatics 21(Suppl 1): i251–i257. doi:10.1093/bioinformatics/bti1014

72. Fariselli P, Martelli PL, Casadio R (2005) A new decoding algorithm for hidden Markov models improves the prediction of the topology of all-beta membrane proteins. BMC Bioinformatics 6(Suppl 4):S12
73. Won KJ, Hamelryck T, Prugel-Bennett A et al (2007) An evolutionary method for learning HMM structure: prediction of protein secondary structure. BMC Bioinformatics 8:357. doi:10.1186/1471-2105-8-357
74. Won KJ, Prugel-Bennett A, Krogh A (2004) Training HMM structure with genetic algorithm for biological sequence analysis. Bioinformatics 20(18):3613–3619. doi:10.1093/bioinformatics/bth454
75. Petersen TN, Brunak S, von Heijne G et al (2011) SignalP 4.0: discriminating signal peptides from transmembrane regions. Nat Methods 8(10):785–786. doi:10.1038/nmeth.1701
76. Lin K, Simossis VA, Taylor WR et al (2005) A simple and fast secondary structure prediction method using hidden neural networks. Bioinformatics 21(2):152–159. doi:10.1093/bioinformatics/bth487
77. Martelli PL, Fariselli P, Casadio R (2004) Prediction of disulfide-bonded cysteines in proteomes with a hidden neural network. Proteomics 4(6):1665–1671. doi:10.1002/pmic.200300745
78. Bagos PG, Liakopoulos TD, Hamodrakas SJ (2006) Algorithms for incorporating prior topological information in HMMs: application to transmembrane proteins. BMC Bioinformatics 7:189. doi:10.1186/1471-2105-7-189
79. Viklund H, Elofsson A (2004) Best alpha-helical transmembrane protein topology predictions are achieved using hidden Markov models and evolutionary information. Protein Sci 13(7):1908–1917. doi:10.1110/ps.04625404

Chapter 5

Predicting Alpha Helical Transmembrane Proteins Using HMMs

Georgios N. Tsaousis, Margarita C. Theodoropoulou, Stavros J. Hamodrakas, and Pantelis G. Bagos

Abstract

Alpha helical transmembrane (TM) proteins constitute an important structural class of membrane proteins involved in a wide variety of cellular functions. The prediction of their transmembrane topology, as well as their discrimination in newly sequenced genomes, is of great importance for the elucidation of their structure and function. Several methods have been applied for the prediction of the transmembrane segments and the topology of alpha helical transmembrane proteins utilizing different algorithmic techniques. Hidden Markov Models (HMMs) have been efficiently used in the development of several computational methods used for this task. In this chapter we give a brief review of different available prediction methods for alpha helical transmembrane proteins pointing out sequence and structural features that should be incorporated in a prediction method. We then describe the procedure of the design and development of a Hidden Markov Model capable of predicting the transmembrane alpha helices in proteins and discriminating them from globular proteins.

Keywords Hidden Markov model, Algorithms, Prediction, Membrane, Transmembrane, Alpha helical, Protein

1 Introduction

Transmembrane proteins constitute ~20–30 % of fully sequenced proteomes, and they are an important class of proteins, since they are crucial for a wide variety of cellular functions [1]. Alpha helical transmembrane proteins constitute one of the two major structural classes of transmembrane proteins and the most abundant one, compared to beta-barrel transmembrane proteins and are found in nearly all cellular membranes. In order to understand their function we must acquire knowledge about their structure and topology in relation to the membrane. However, obtaining crystals of transmembrane proteins suitable for crystallographic studies is difficult and transmembrane proteins represent less than 2 % of the structures in the Protein Data Bank [2]. Therefore, during the last

David R. Westhead and M.S. Vijayabaskar (eds.), *Hidden Markov Models: Methods and Protocols*, Methods in Molecular Biology, vol. 1552, DOI 10.1007/978-1-4939-6753-7_5, © Springer Science+Business Media LLC 2017

2 decades a large number of computational methods have been developed in order to predict the structure and topology of transmembrane proteins [3]. By topology, we refer to the knowledge of the number and the exact localization of transmembrane segments, as well as their orientation with respect to the lipid bilayer.

The first prediction methods made use of hydrophobicity scales in order to predict the location of transmembrane segments along the protein sequence [4]. Later, the positive inside rule was used for the prediction of the overall topology of a transmembrane protein [5, 6]. The evolution of transmembrane topology prediction methods involved the use of several algorithmic techniques including Statistical Analyses [7, 8], Artificial Neural Networks (ANNs) [9, 10], Hidden Markov Models (HMMs) [11–15], Support Vector Machines (SVMs) [16], Dynamic Bayesian Networks (DBNs) [17], and ensemble methods (e.g., Hidden Neural Networks, HNNs) [18, 19]. In addition, there are a number of prediction methods (meta-predictors) that combine the results of several individual methods and produce a consensus prediction [20–23] in order to improve the prediction performance. Hidden Markov Models have been shown to outperform other techniques in topology prediction and are widely used [15, 24, 25]. Despite the machine learning approach used, an efficient prediction method must be designed by taking into consideration the unique structural features of alpha helical TM proteins.

2 Structural Features of Alpha Helical Transmembrane Proteins

Alpha helical transmembrane proteins are the major category of transmembrane proteins and are located in all biological membranes covering a wide variety of cellular functions. They consist of one or more alpha helices that span the membrane forming transmembrane segments (Fig. 1). Transmembrane proteins with one transmembrane helix are referred to as single-spanning TM proteins and are further classified to distinct types, whereas multi-spanning TM proteins have more transmembrane regions. In addition, certain families of transmembrane proteins have members with a fixed number of transmembrane regions. For instance, G-protein coupled receptors (GPCRs) have seven TM regions, receptor-tyrosine kinases have 1–2 TM regions, whereas sodium channels have 24 TM regions [28].

The transmembrane segments of alpha helical TM proteins are, typically, hydrophobic stretches of 15–40 residues [29–31]. The length of the transmembrane segment varies according to the inclination angle of the helix with respect to the level of the membrane and the thickness of the membrane. Based on statistical analyses, on average transmembrane segments constitute of 19–21 residues [7]. Aromatic residues like Tryptophan and

Tyrosine are more preferably located near the ends of the transmembrane segments at the boundaries with the membrane (interfacial region) [32, 33]. Loops that connect the transmembrane segments are resided either to the cytoplasmic or the extracellular side of the membrane with an average length of 1–20 residues and lower average hydrophobicity. However, longer loops are often observed in transmembrane proteins, forming in some cases large soluble domains. It has been observed that that positively charged amino acids (Arginine and Lysine) are overrepresented on the cytoplasmic loops than the extracellular ones ("positive inside rule") [34]. However this characteristic is more often present in alpha helical TM proteins from Bacteria and Archaea [6, 35, 36] and in loop regions smaller than 60 residues [35, 37]. Moreover, the orientation of the transmembrane segments within some membrane proteins is dependent on membrane lipid composition [38] and a change in lipid composition can invert the orientation of a transmembrane segment [39].

The increasing number of newly solved structures of alpha helical TM proteins has provided new information about special structural and topological characteristics of proteins (Fig. 1). It has been observed that Proline residues, when located in transmembrane helices, cause helical distortions induced by kinks that bend the helices more than 30° (disrupted helices) [32, 40, 41]. Other nontypical cases of helices in alpha helical TM proteins are,

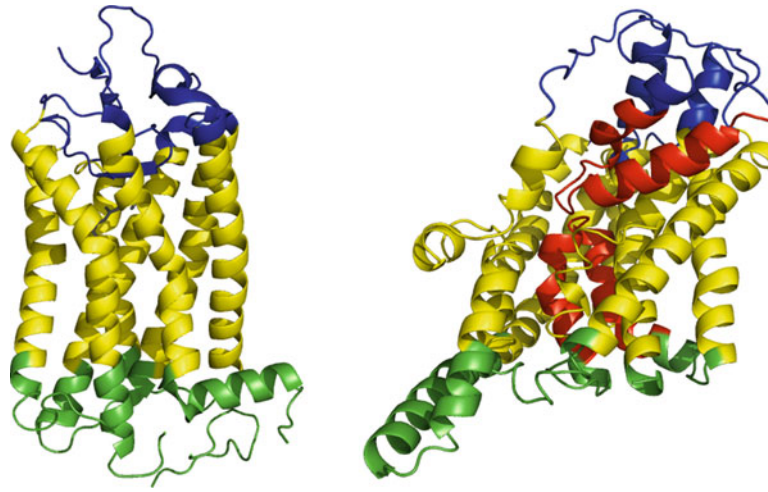


Fig. 1 On the left, a ribbon diagram of the structure of bovine rhodopsin (PDB ID: 1F88) and on the right, of the glutamate transporter homolog from *Pyrococcus horikoshii* (PDB ID: 1XFH). Cytoplasmic, extracellular, and transmembrane regions are colored *green*, *blue*, and *yellow* respectively using the annotation of PDBTM [26]. The glutamate transporter homolog contains two reentrant regions (*red*) and non-characteristic cases of transmembrane helices. The diagrams were drawn using the PyMol molecular graphics package [27]

transmembrane helices that have a great inclination angle with respect to the level of the membrane (strongly tilted helices) [42, 43], amphipathic helices that reside alongside the plane of the membrane and may immerse in the membrane–water interface region (interfacial helices) [44, 45], as well as loops/helices that enter the interior of the lipid membrane and exit along the same side (reentrant regions/helices) [46, 47]. Reentrant regions are functional regions, most commonly found in channel proteins and least commonly in signal receptors [48–51]. It is estimated that 10 % of alpha helical TM proteins have such regions [46]. Furthermore, there have been cases where the transmembrane protein occurs with two possible opposite topologies and does not follow the "positive inside rule" (dual topology proteins) [52].

3 Prediction of Alpha Helical Transmembrane Proteins

Transmembrane protein topology prediction methods predict the potential topology of a transmembrane protein from its protein sequence. In order to achieve this task, they use information "hidden" in the protein sequence. The first prediction methods made use of hydrophobicity scales in order to predict the location of transmembrane segments along the protein sequence [4]. Later, the positive inside rule was used for the prediction of the overall topology of a transmembrane protein by discriminating the regions facing the two sides of the membrane [5, 6]. The MEMSAT algorithm [8] utilized the same information along with statistical optimization methods using dynamic programming and amino acid propensity scales, in order to produce the optimum topology. Later on, similar methods, which used statistical techniques based on propensity scales, were developed [7].

The PHDhtm algorithm [53] was the first to use Artificial Neural Networks (ANNs) to predict the topology of transmembrane proteins. This algorithm incorporated evolutionary information through multiple sequence alignments to produce a consensus TM prediction for the target sequence and then, used the "positive inside rule," to predict the topology. TMHMM [11] and HMMTOP [54] were the first algorithms that used Hidden Markov Models (HMMs) to predict the topology of transmembrane proteins. Following the same concept, PRO-TMHMM and PRO-DIV-TMHMM [15] algorithms were later developed to incorporate evolutionary information. In the next years, HMMs have been widely used for the transmembrane protein topology prediction [12, 13, 55]. In some cases, Support Vector Machines (SVMs) and dynamic Bayesian networks (DBNs) have been used successfully to build transmembrane protein topology prediction algorithms, such as MEMSAT-SVM [16] and Philius [17]. Some methods as OCTOPUS [18] have extended their topological

grammar to allow the prediction of special features of alpha helical TM proteins such as reentrant regions.

Towards the improvement of the prediction performance, several modifications have been introduced including the inclusion of additional information or the combination of different predictions. An inherent problem in transmembrane protein topology prediction and signal peptide prediction is the high similarity between the hydrophobic regions of a transmembrane helix and that of a signal peptide, leading to cross-reaction between the two types of predictions. One approach to reduce false positive predictions of signal peptides as transmembrane segments is to combine the two types of predictions by adding a filtering prediction a specialized signal peptide prediction method such as SignalP [56] and removing the sequence of the predicted signal peptide before topology prediction or with the inclusion of an additional submodel in the HMM. Towards this direction, Phobius [13], was the first algorithm to combine signal peptide prediction and topology prediction for alpha helical TM proteins, in order to discriminate between the two types of prediction. Later, Phobius was further modified to incorporate evolutionary information from multiple sequence alignments [55].

Other approaches used to produce more reliable results are consensus prediction methods, which combine the results from two or more prediction algorithms [20, 21] to predict the topology of alpha helical TM proteins. More recent approaches include the use of different prediction algorithms, the results of which are combined and evaluated using HMMs or SVMs. Such examples are TOPCONS [22, 57], and MetaTM [23].

During the last few years ab initio topology prediction has been shown to be an attainable goal since it yields comparable performance [58, 59]. SCAMPI uses an experimental scale (ΔG-scale) [60] of position-specific amino acid contributions to the free energy of membrane insertion to predict the topology of alpha helical TM proteins. On the other hand, ZPRED [61] is using a physical property such as the distance between a residue and the center of the membrane (Z-coordinate) in combination with HMMs and NNs.

Importantly, several methods developed during the last few years [1, 13, 14, 18, 22, 25, 58] allow the incorporation of topological information derived from biochemical studies (constrained prediction), which results in improved topology prediction performance. Such biochemical methods include: gene fusion, using enzymes such as alkaline phosphatase, β-galactosidase, β-lactamase and various fluorescent proteins, detection of posttranslational modifications such as glycosylation, phosphorylation and biotinylation, cysteine-scanning mutagenesis, proteolysis methods, and epitope mapping techniques [62]. In addition, the occurrence of specific motifs and/or domains on a given protein may improve

the topology prediction of alpha helical transmembrane proteins [63]. These domains usually appear in specific subcellular locations and, therefore, the existence of such domains in transmembrane proteins provides additional information about their topology and can be used for the generation of constrained predictions. Such domains are be found in databases as SMART [64], INTERPRO [65], PFAM [66], and TOPDOM [67].

4 A Hidden Markov Model for Alpha Helical Transmembrane Proteins

4.1 Basic Concepts of Hidden Markov Models

Hidden Markov Models have been extensively used for pattern recognition problems, with the most known example found in the speech recognition methodology [68]. In bioinformatics, during the last few years, they have been used for generating probabilistic profiles for protein families [69], the prediction of transmembrane helices in proteins [1], the prediction of signal peptides and their cleavage sites [70], the prediction of genes [71] and for the prediction of transmembrane beta strands [72].

The Hidden Markov Model is a probabilistic model, which consists of several states, connected by means of the transition probabilities, thus forming a Markov process. If we consider an amino acid sequence of a protein with length L, denoted by

$$\mathrm{x} = x_1, x_2, \ldots, x_{L-1}, x_L$$

with a labeling (in this case corresponding to transmembrane, intracellular, and extracellular regions):

$$\mathrm{y} = y_1, y_2, \ldots, y_{L-1}, y_L$$

then the transition probability for jumping from a state k to a state l is defined as:

$$\alpha_{kl} = P\left(\pi_i = l \middle| \pi_{i-1} = k\right)$$

Where π is the "path" in the particular position of the amino acid sequence (i.e., the sequence of states, as opposed to the sequence of symbols). Each state k is associated with a distribution of emission probabilities, meaning the probabilities that any particular symbol could be emitted by the current state. Assuming an alphabet Σ, consisting of the 20 amino acids, the probability that a particular aminoacid b is emitted from state k is defined as:

$$e_k(b) = P\left(x_i = b \middle| \pi_i = k\right)$$

The term "hidden" is justified by the fact that when one observes the emitted symbols he cannot see the underlying states, thus the true state process is hidden from the observer. The total probability

of the observation sequence given the model, $P(\mathbf{x}|\theta)$, is computed using the efficient forward algorithm [68], whereas the joint probability of the sequence and the labeling denoted by $P(\mathbf{x},\mathbf{y}|\theta)$, is computed by its trivial modification proposed by Krogh [71].

4.2 The Hidden Markov Model Architecture

The architecture of the HMM is chosen so that it could fit as much as possible to the limitations imposed by the known structures. HMM-based methods use different types of states to describe the transmembrane segments, the extracellular (outer) and the cytoplasmic (inner) regions and have in general more simple architecture than HMMs used to describe the topology of beta barrel transmembrane proteins. The transmembrane region is further divided in three state compartments in order to describe the hydrophobic core of the transmembrane segment and the two interfacial regions at the end of the transmembrane helices. HMM-TM, used a Hidden Markov Model of similar architecture, which is described in Fig. 2. The model is cyclic, consisting of 114 states (including begin (B) and end (E) states) and is conceptually similar to other models described for the same task (TMHMM, HMMTOP, PHOBIUS). The HMM is further divided in three "sub-models" corresponding to the three desired labels to predict, the TM (transmembrane) helix sub-model and the inner (cytoplasmic) and outer

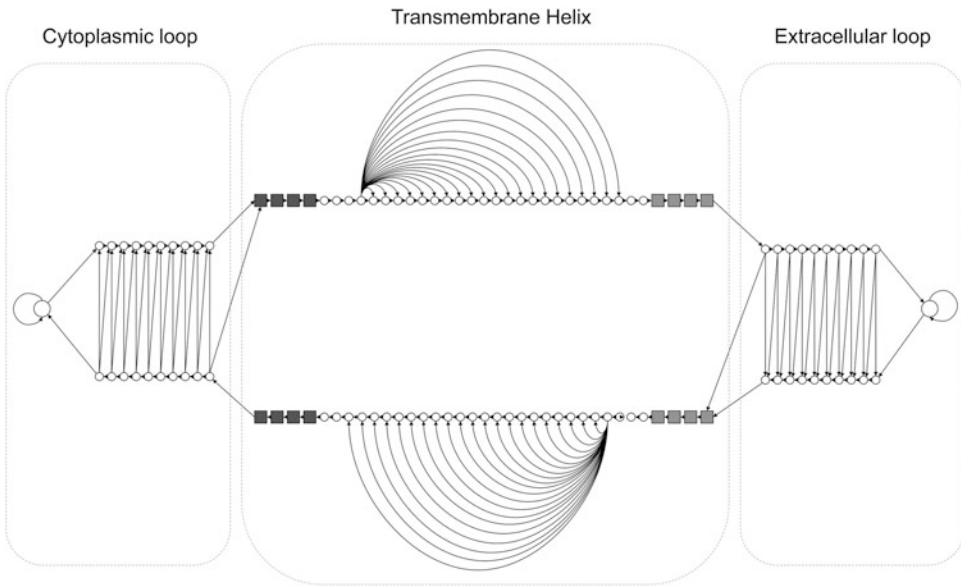


Fig. 2 A schematic representation of a HMM architecture for the prediction of the topology of alpha helical TM proteins. The model consists of three sub-models corresponding to the three labels which are shown separately. In the transmembrane sub-model different colors correspond to the tied states. *Black squares* correspond to the aromatic belt and *white circles* to the transmembrane core region. In the cytoplasmic and extracellular loop sub-models, the states forming the ladder are tied together respectively. The allowed transitions are shown with *arrows*

(extracellular) loops sub-models respectively. All states are connected with the appropriate transition probabilities in order to be consistent with the known structures, that is, to ensure appropriate length distribution. The TM helix model incorporates states to model the architecture of the transmembrane helices. Thus, there are states that correspond to the core of the helix and the cap located at the lipid bilayer interface. The minimum allowed length for the transmembrane helix (including the helix caps) is 15 residues, whereas the maximum is 35. The inner and outer loops are modeled with a "ladder" architecture. At both ends, there is a self-transitioning state corresponding to residues too distant from the membrane; these cannot be modeled as loops, hence that state is named "globular." This way long and short loop regions are both described by the model architecture. On the other hand, TMHMM used different states to model short and long loop regions.

Several modifications to this general architecture can be applied so as to improve the prediction performance and better describe the topological characteristics of alpha helical TM proteins. The HMM introduced in Phobius can discriminate between transmembrane helices and signal peptides which in many cases are falsely predicted as transmembrane segments. The model architecture of Phobius can be regarded as a combination of the modified models TMHMM and the original SignalP-HMM, with a transition from the last state of the signal peptide model in SignalP-HMM to the outer loop state in the TMHMM model.

Phosphorylation and glycosylation are the most widespread posttranslational modifications in eukaryotes [73, 74] and occur in a compartment-specific manner in the cell. In eukaryotic cells, glycosylation activity is found in the lumen of the endoplasmic reticulum (ER) and it is accomplished by the enzyme oligosaccharyl transferase (OST), which adds oligosaccharides to the amino group of Asparagine (Asn) residues of the consensus sequence Asn-X-Thr/Ser (N-linked glycosylation) [75]. In transmembrane proteins, glycosylation sites occur at parts of proteins facing the extracellular space and are located to a minimum distance away from the membrane surface [76]. It has been shown that, in some cases, glycosylation occurs only when the acceptor site (Asn residue) is located a minimum of 12 residues upstream or 14 residues downstream of a transmembrane segment ("12 + 14 rule") [76–78]. Therefore, in multi-spanning transmembrane proteins, glycosylated extracellular loops have a minimum length of approximately 30 residues [79]. Protein phosphorylation is the most important and well-studied posttranslational modification in eukaryotes and is involved in the regulation of several cellular processes such as cell growth and differentiation, signal transduction and apoptosis [80–83]. The addition of a phosphate group usually occurs in Serine (Ser), Threonine (Thr), Tyrosine (Tyr) and Histidine (His) residues in eukaryotic proteins and approximately 30–50 % of

proteins are supposed to be phosphorylated at some point [84]. In transmembrane proteins, phosphorylation sites are located at the cytoplasmic regions. Therefore, both the existence of a phosphorylation or a glycosylation site along the sequence of a transmembrane protein provides valuable information about the orientation of the modified region with respect to the membrane [62]. This information was used in HMMpTM [85] which introduced a novel HMM architecture (Fig. 3), which combines in a single model, topology prediction and phosphorylation and glycosylation site prediction. The HMM is quite similar to the one proposed by HMM-TM. It consists of five different sub-models corresponding to the five desired labels to predict, the Cytoplasmic Loop sub-model, the Transmembrane Helix sub-model, the Extracellular Loop sub-model, the Phosphorylation Site sub-model corresponding to kinase-specific phosphorylation sites and the Glycosylation Site sub-model used to model the existence of N/O-linked glycosylation sites (Fig. 3). Due to the fact that phosphorylation and glycosylation sites are compartment specific, the Phosphorylation Site and Glycosylation Site sub-models are connected with the Cytoplasmic Loop and the Extracellular Loop sub-models, respectively. The Phosphorylation Site sub-model may include additional sub-models, each one corresponding to a kinase-specific

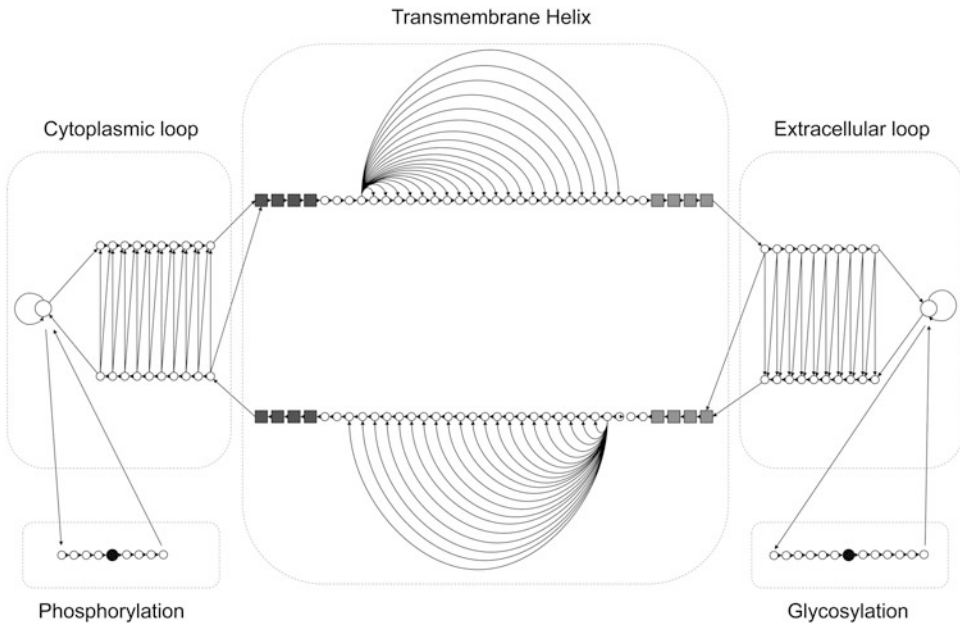


Fig. 3 A modified version of the HMM architecture presented in Fig. 2, which includes two additional sub-models for the prediction of phosphorylation and glycosylation sites. The phosphorylation site and glycosylation site sub-models are connected with the cytoplasmic loop and the extracellular loop sub-models, respectively. In both cases *black circles* represent the posttranslationaly modified site and a variable length of flanking residues can be used to specifically model the surrounding region

phosphorylation pattern. Each phosphorylation pattern includes the phosphorylation site and four flanking residues at each side of the modified site (−4/+4). For N-linked and O-linked glycosylation patterns, in the Glycosylation Site sub-model, flanking residues at each side of the modified site (−6/+6) were originally used in HMMpTM.

Some methods [16, 18, 46] have extended their topological grammar to include, describe and discriminate, regions inserted but not spanning the membrane, as reentrant regions. GPCRHMM [86], introduced a novel HMM architecture to more efficiently describe the overall topology of GPCRs, including different sub-models for each one of the seven TM helices, the three cytosolic and the three extracellular loops, the cytoplasmic C-terminal region, the extracellular N-terminal region, and the signal peptide region. A different approach was used in the development of GPCRpipe [87] where a simple modification of the original model of HMM-TM was utilized to make the method more specific for the topology of GPCRs. In specific, the N-terminal (Begin state) and C-terminal (End state) regions were connected only to the extracellular and cytoplasmic loops, respectively.

4.3 Creation of Training and Testing Datasets

For the creation of a training dataset for the HMM, a set of alpha helical TM proteins with known three dimensional structure and topology must be compiled. Three-dimensional structures of alpha helical TM proteins can be obtained from the Protein Data Bank (PDB) [2] using information deposited in PDBTM [26], OPM [88], TOPDB [89] or ExTopoDB [90] where structural and experimental data are combined. For variants of the same protein, only the structure solved at the highest resolution is collected, and multiple identical chains are removed, keeping only one chain for each structure. The sequences of the remaining structures are submitted to a redundancy check using BLAST [91], removing chains with a sequence identity above some threshold (typically 30 %). Most available HMM-based methods for alpha helical TM proteins have used approximately 72–292 proteins in their training sets. Larger sets are not expected to increase the prediction performance for a given algorithmic technique. It has been shown [92] that the relationship between the sizes of the training set with the performance of HMM-based prediction algorithms is nonlinear and reaches an upper limit, even in the case of alpha helical TM protein prediction. However, the training set should be compiled by collecting a structural representative from each known family of alpha helical TM proteins. It is important to point out that even in structures known at atomic resolution, the exact boundaries of the transmembrane helices are not obvious, and in some situations the PDB annotations for the helices are clearly extending far beyond the membrane. Therefore, the TM assignments deposited in public

databases such as PDBTM [26], which are generated by algorithms that identify the precise boundaries of the transmembrane segments can be used. Finally, in [1] an automated method for relabeling the data has been proposed, that consists of removing the labels at the boundaries of the TM strands, and then performing a constrained prediction that reassigns the boundaries. Extending this approach experimentally derived topological information of loop regions can be used to perform constrained predictions for this purpose. Such information is available in specialized resources as TOPDB [89] and ExTopoDB [90].

For the evaluation of the prediction performance of the HMM a self-consistency test and a cross-validation procedure is commonly performed. In the cross-validation procedure the training set is divided in equal subsets (folds). The model is trained with one subset of proteins and the prediction performance is evaluated against the remaining subsets. This way, prediction results for each protein in the training dataset are derived from models where the protein was not included in the training process. For small training sets measures of accuracy can be obtained using a jackknife test (leave one out cross-validation test). In addition, several test sets are compiled for the evaluation of the performance of the predictive model. An independent test set of alpha helical TM proteins having experimentally verified topologies and no sequence similarity with the proteins used in the training set can be used to evaluate the topology prediction performance.

To assess the accuracy of the predictions, several measures can be used. For the transmembrane segments predictions the number of correctly predicted segments (True Positives, TP), the number of missed segments (False Negatives, FN) and the number of the overpredicted segments (False Positives, FP) is calculated. However, the transmembrane region prediction accuracy is more efficiently measured by the segments overlap measure (SOV) [93]. As measures of the accuracy per residue, the total fraction of the correctly predicted residues (Q_α or Q_3), in a two-state model (transmembrane versus non-transmembrane), or a three-state model (transmembrane versus cytoplasmic loop versus extracellular loop), and the well known Matthews Correlation Coefficient (C_α) can be used [94]. In addition, the number of the correctly predicted transmembrane segments and topologies (i.e., when both helices localization and orientation of the loops are correctly predicted) can be calculated to evaluate the performance per protein. The discriminative power of the model can also be evaluated against a nonredundant set of globular proteins with known three-dimensional structure.

4.4 Training and Decoding Algorithms

Traditionally, the parameters of a Hidden Markov Model are optimized according to the Maximum Likelihood criterion [68],

$$\hat{\theta}^{\mathrm{ML}} = \arg\max_{\theta} P(\mathrm{x}|\theta)$$

A widely used algorithm for this task is the efficient Baum–Welch algorithm (also known as Forward–Backward) [68, 95], which is a special case of the Expectation-Maximization (EM) algorithm, proposed for Maximum Likelihood (ML) estimation for incomplete data [96] and has been widely used for the training of several HMM methods for alpha helical TM proteins [1, 11–14]. The algorithm, updates iteratively the model parameters (emission and transition probabilities), with the use of their expectations, computed with the use of the Forward and Backward algorithms. Convergence to at least a local maximum of the likelihood is guaranteed. The main disadvantage of ML training is that it is not discriminative. Alternatively, the Conditional Maximum Likelihood (CML) training for labeled data can be used, as proposed by Krogh [97]. Although CML is computationally more intensive than the ML approach, the predictive ability is better when data with good quality of labeling are used. The Conditional Maximum Likelihood criterion is:

$$\hat{\theta}^{\mathrm{CML}} = \arg\max_{\theta} P(\mathrm{y}|\mathrm{x},\theta) = \arg\max_{\theta} \frac{P(\mathrm{x},\mathrm{y}|\theta)}{P(\mathrm{x}|\theta)}$$

This kind of training, often referred to as discriminating training, seeks to maximize the probability of the correct prediction, i.e., the probability of the labeling **y** for a given sequence **x** and a model θ. Another major problem with CML is that the Baum–Welch algorithm cannot be applied and variants of the gradient descent method are needed. These methods require parameter fine-tuning (for the so-called "learning rate") and in many cases do not provide stable convergence. However, a variant that uses individual learning rates that are adapted during the process, has been presented, that offers significant advantages [98]. The CML training with the above-mentioned algorithms has been extensively described and efficiently used in [12]. The parameters of the model (transition and emission probabilities) are updated simultaneously, using the gradients of the likelihood function as described in [99], and the training process terminates when the likelihood does not increase beyond a prespecified threshold. To reduce the number of the free parameters of the model, and thus improve the generalization capability, states expecting to have the same emission probabilities, can be tied together (Fig. 2). Furthermore, to avoid over-fitting, the iterations started from emission probabilities corresponding to the initial amino acid frequencies observed in the known protein structures and small pseudocounts can be added in each step.

The decoding of an HMM can be performed using the standard Viterbi algorithm [68] or alternatively the N-best algorithm [100], as formulated in [97]. This algorithm is a heuristic that attempts to find the most probable labeling of a given sequence, as opposed to the well-known Viterbi algorithm [68], which guarantees to find the most probable path of states. Since there are several states contributing to the same labeling of a given sequence, the N-best algorithm will always produce a labeling with a probability at least as high as that computed by the Viterbi algorithm, in other words it always returns equal if not better results. Its main drawback is the memory requirements and computational complexity, resulting in a slowdown of the decoding process. A variant of the posterior decoder coupled with a post-processing step using dynamic programming has been used by Jacoboni and coworkers [101] and later presented as a general purpose algorithm [102]. However, novel decoding algorithms have been presented later, that combine the advantages of both Viterbi and posterior decoding, in a more mathematical sound way, as the optimal accuracy posterior decoder (OAPD) [55] and the Posterior-Viterbi algorithm [103].

Posterior-Viterbi decoding is based on the combination of the Viterbi and posterior algorithms. After having computed the posterior probabilities, Viterbi algorithm is used to find the best allowed posterior path through the model. Although its computational requirements are double compared to Viterbi, the Posterior-Viterbi decoding algorithm performs essentially a Viterbi-like decoding, using the Posterior probabilities instead of the emissions, and the allowed paths given by a delta function applied on the transition matrix.

Finally, Kall and coworkers [55] presented a very similar algorithm, the Optimal Accuracy Posterior Decoder. The algorithm sums for each position in the sequence the posterior label probabilities and by using the allowed transitions, calculates in a Viterbi-like manner the optimal sequence of labels. The main differences of this algorithm compared to the Posterior-Viterbi is the fact that uses the sums of the posterior label probabilities instead of the posterior probabilities, and instead of the product, it maximizes the sum of these probabilities.

For the purpose of discrimination, the information included in the prediction of the putative transmembrane segments is used. Most methods use the criterion of the number of predicted transmembrane segments to indentify transmembrane proteins. However, the fact that signal peptides of globular proteins are often predicted as transmembrane segments should be taken into account and for methods that do not include a signal peptide sub-model in the HMM architecture, potential signal peptides can be removed using the prediction results of SignalP. Apart from the number of predicted TM helices, TMHMM introduced two additional values

that could be used for discrimination between globular and transmembrane proteins [1]. Both the expected numbers of transmembrane helices and residues in transmembrane helices can be calculated and compared to threshold values that are derived from the training datasets during the cross-validation procedure. For example, the original TMHMM algorithm used a calculated threshold of 18 for the expected number of residues in a transmembrane helix. Other methods [16, 22, 58] use a pre-filtering step to decide whether a protein is transmembrane prior to the topology prediction. Therefore, the number of false positives in the discrimination prediction between globular and transmembrane proteins may vary between different prediction methods.

5 Further Considerations and Improvements

Improvements in prediction performance for alpha helical TM proteins can be achieved by designing a more plausible model architecture. Currently, the potential improvements in this respect may include modifications in order to incorporate information about the differences of the hydrophobicity of the transmembrane helices in multi-spanning transmembrane proteins [104]. Different prediction strategies for marginally hydrophobic and N- and C-terminal TM helices can be applied to better model the transmembrane regions, and discriminate transmembrane from globular proteins and more efficiently predict the topology of TM proteins with large non-membrane domains [59].

As already mentioned, constrained predictions can increase the reliability of the predicted topology by adding information about certain regions. Constrained predictions can be further used to allow the incorporation of experimental information about phosphorylation and glycosylation sites in more specific models as the one described in HMMpTM. In transmembrane proteins, glycosylation sites occur at parts of proteins facing the extracellular space and are located to a minimum distance away from the membrane surface [76]. It has been shown that, in some cases, glycosylation occurs only when the acceptor site (Asn residue) is located a minimum of 12 residues upstream or 14 residues downstream of a transmembrane segment ("12 + 14 rule") [76–78]. As a result, in multi-spanning transmembrane proteins, glycosylated extracellular loops have a minimum length of approximately 30 residues [79]. Therefore, N-linked glycosylation site prediction can also serve as a molecular ruler to define the ends of transmembrane segments. In addition, domain assignments for cytoplasmic or extracellular domain can be used as constrains in topology prediction, by constructing a pHMM library of domains with known subcellular location and adding a filtering step. This way, prediction methods

that allow constrained predictions can make use of such information without any further modifications to the model architecture.

Long-range interactions have an important role in protein folding and stability and such interactions are additional information "hidden" in the protein sequence that could be used in TM protein topology prediction. Interhelical interactions have been used in topology prediction for alpha helical TM proteins [105] and this strategy can be extended by combing a prediction method specific for long-range contacts with a topology predictor. Moreover, helix–helix interactions, residue contacts, and lipid exposure have been combined with topology prediction in order to provide additional information about the arrangement of the predicted TM helices [106]. However, using a different approach, predicted interactions (residue, helix, or lipid) may be incorporated in prediction algorithms and taken into account for topology prediction. Such information may provide more accurate topological models of transmembrane proteins along with additional information about their structure across the membrane.

Finally, some of the best prediction methods use evolutionary information in the form of multiple sequence alignments. The inclusion of evolutionary information in HMM based methods has been shown to substantially improve the prediction performance [15]. However, most implementations used in alpha helical TM protein prediction include the construction of a sequence profile from multiple alignments that is used as an input. These methods were feasible by extending the simple HMM in a way that accepts as input a sequence profile instead of sequence. Thus, other methods such as HMM-TM may be difficult to adopt a similar architecture. However, a different method can be used following [55]. Briefly, given a query sequence and a multiple alignment of its homologs, predictions with the single sequence method can be obtained on each of the homologs. Then, the predicted labels can be mapped on the alignment and averaged for each position of the query sequence. This will create a "posterior label probability" (PLP) table for the query sequence that contains information from the multiple alignments. In the last step, the OAPD [55] can be applied and the final prediction is obtained. This approach has the advantage that it can be used in any single-sequence method without further modifications.

References

1. Krogh A, Larsson B, von Heijne G et al (2001) Predicting transmembrane protein topology with a hidden Markov model: application to complete genomes. J Mol Biol 305 (3):567–580
2. Berman HM, Westbrook J, Feng Z et al (2000) The Protein Data Bank. Nucleic Acids Res 28(1):235–242, doi:gkd090 [pii]
3. Punta M, Forrest LR, Bigelow H et al (2007) Membrane protein prediction methods. Methods 41(4):460–474. doi:10.1016/j.ymeth.2006.07.026
4. Kyte J, Doolittle RF (1982) A simple method for displaying the hydropathic character of a protein. J Mol Biol 157(1):105–132, doi:0022-2836(82)90515-0 [pii]

5. Claros MG, von Heijne G (1994) TopPred II: an improved software for membrane protein structure predictions. Comput Appl Biosci 10 (6):685–686
6. Sipos L, von Heijne G (1993) Predicting the topology of eukaryotic membrane proteins. Eur J Biochem 213(3):1333–1340
7. Pasquier C, Promponas VJ, Palaios GA et al (1999) A novel method for predicting transmembrane segments in proteins based on a statistical analysis of the SwissProt database: the PRED-TMR algorithm. Protein Eng Des Sel 12(5):381–385
8. Jones DT, Taylor WR, Thornton JM (1994) A model recognition approach to the prediction of all-helical membrane protein structure and topology. Biochemistry 33 (10):3038–3049
9. Rost B, Casadio R, Fariselli P et al (1995) Transmembrane helices predicted at 95% accuracy. Protein Sci 4(3):521–533
10. Pasquier C, Hamodrakas SJ (1999) An hierarchical artificial neural network system for the classification of transmembrane proteins. Protein Eng Des Sel 12(8):631–634
11. Sonnhammer EL, von Heijne G, Krogh A (1998) A hidden Markov model for predicting transmembrane helices in protein sequences. Proc Int Conf Intell Syst Mol Biol 6:175–182
12. Bagos PG, Liakopoulos TD, Hamodrakas SJ (2006) Algorithms for incorporating prior topological information in HMMs: application to transmembrane proteins. BMC Bioinformatics 7:189. doi:10.1186/1471-2105-7-189
13. Kall L, Krogh A, Sonnhammer EL (2004) A combined transmembrane topology and signal peptide prediction method. J Mol Biol 338(5):1027–1036. doi:10.1016/j.jmb.2004.03.016
14. Tusnady GE, Simon I (2001) The HMMTOP transmembrane topology prediction server. Bioinformatics 17(9):849–850
15. Viklund H, Elofsson A (2004) Best alpha-helical transmembrane protein topology predictions are achieved using hidden Markov models and evolutionary information. Protein Sci 13(7):1908–1917. doi:10.1110/ps.04625404
16. Nugent T, Jones DT (2009) Transmembrane protein topology prediction using support vector machines. BMC Bioinformatics 10:159. doi:10.1186/1471-2105-10-159
17. Reynolds SM, Kall L, Riffle ME et al (2008) Transmembrane topology and signal peptide prediction using dynamic Bayesian networks. PLoS Comput Biol 4(11):e1000213. doi:10.1371/journal.pcbi.1000213
18. Viklund H, Elofsson A (2008) OCTOPUS: improving topology prediction by two-track ANN-based preference scores and an extended topological grammar. Bioinformatics 24(15):1662–1668. doi:10.1093/bioinformatics/btn221
19. Viklund H, Bernsel A, Skwark M et al (2008) SPOCTOPUS: a combined predictor of signal peptides and membrane protein topology. Bioinformatics 24(24):2928–2929. doi:10.1093/bioinformatics/btn550
20. Promponas VJ, Palaios GA, Pasquier CM et al (1999) CoPreTHi: a Web tool which combines transmembrane protein segment prediction methods. In Silico Biol 1(3):159–162, doi:1998010014 [pii]
21. Nilsson J, Persson B, Von Heijne G (2002) Prediction of partial membrane protein topologies using a consensus approach. Protein Sci 11(12):2974–2980. doi:10.1110/ps.0226702
22. Bernsel A, Viklund H, Hennerdal A et al (2009) TOPCONS: consensus prediction of membrane protein topology. Nucleic Acids Res 37(Web Server issue):W465–W468. doi:10.1093/nar/gkp363
23. Klammer M, Messina DN, Schmitt T et al (2009) MetaTM—a consensus method for transmembrane protein topology prediction. BMC Bioinformatics 10:314. doi:10.1186/1471-2105-10-314
24. Moller S, Croning MD, Apweiler R (2001) Evaluation of methods for the prediction of membrane spanning regions. Bioinformatics 17(7):646–653
25. Bagos PG, Liakopoulos TD, Hamodrakas SJ (2005) Evaluation of methods for predicting the topology of beta-barrel outer membrane proteins and a consensus prediction method. BMC Bioinformatics 6:7. doi:10.1186/1471-2105-6-7
26. Kozma D, Simon I, Tusnady GE (2013) PDBTM: Protein Data Bank of transmembrane proteins after 8 years. Nucleic Acids Res 41(Database issue):D524–D529. doi:10.1093/nar/gks1169
27. Delano WL (2002) The PyMOL molecular graphics system. http://www.pymol.org
28. Almen MS, Nordstrom KJ, Fredriksson R et al (2009) Mapping the human membrane proteome: a majority of the human membrane proteins can be classified according to function and evolutionary origin. BMC Biol 7:50. doi:10.1186/1741-7007-7-50

29. Bowie JU (1997) Helix packing angle preferences. Nat Struct Biol 4(11):915–917
30. Chen H, Kendall DA (1995) Artificial transmembrane segments. Requirements for stop transfer and polypeptide orientation. J Biol Chem 270(23):14115–14122
31. Nilsson I, von Heijne G (1998) Breaking the camel's back: proline-induced turns in a model transmembrane helix. J Mol Biol 284 (4):1185–1189. doi:10.1006/jmbi.1998.2219
32. Wallin E, Tsukihara T, Yoshikawa S et al (1997) Architecture of helix bundle membrane proteins: an analysis of cytochrome c oxidase from bovine mitochondria. Protein Sci 6(4):808–815. doi:10.1002/pro.5560060407
33. Weiss MS, Kreusch A, Schiltz E et al (1991) The structure of porin from Rhodobacter capsulatus at 1.8 A resolution. FEBS Lett 280 (2):379–382, doi:0014-5793(91)80336-2 [pii]
34. von Heijne G (1992) Membrane protein structure prediction. Hydrophobicity analysis and the positive-inside rule. J Mol Biol 225 (2):487–494
35. Nilsson J, Persson B, von Heijne G (2005) Comparative analysis of amino acid distributions in integral membrane proteins from 107 genomes. Proteins 60(4):606–616. doi:10.1002/prot.20583
36. Gafvelin G, Sakaguchi M, Andersson H et al (1997) Topological rules for membrane protein assembly in eukaryotic cells. J Biol Chem 272(10):6119–6127
37. Andersson H, von Heijne G (1993) Sec dependent and sec independent assembly of E. coli inner membrane proteins: the topological rules depend on chain length. EMBO J 12 (2):683–691
38. Bogdanov M, Xie J, Dowhan W (2009) Lipid-protein interactions drive membrane protein topogenesis in accordance with the positive inside rule. J Biol Chem 284 (15):9637–9641. doi:10.1074/jbc.R800081200
39. van Klompenburg W, Nilsson I, von Heijne G et al (1997) Anionic phospholipids are determinants of membrane protein topology. EMBO J 16(14):4261–4266
40. von Heijne G (1991) Proline kinks in transmembrane alpha-helices. J Mol Biol 218 (3):499–503, doi:0022-2836(91)90695-3 [pii]
41. Sansom MS (1992) Proline residues in transmembrane helices of channel and transport proteins: a molecular modelling study. Protein Eng 5(1):53–60
42. Park SH, Opella SJ (2005) Tilt angle of a trans-membrane helix is determined by hydrophobic mismatch. J Mol Biol 350 (2):310–318. doi:10.1016/j.jmb.2005.05.004
43. Yeagle PL, Bennett M, Lemaitre V et al (2007) Transmembrane helices of membrane proteins may flex to satisfy hydrophobic mismatch. Biochim Biophys Acta 1768 (3):530–537. doi:10.1016/j.bbamem.2006.11.018
44. Granseth E, von Heijne G, Elofsson A (2005) A study of the membrane-water interface region of membrane proteins. J Mol Biol 346(1):377–385. doi:10.1016/j.jmb.2004.11.036
45. Liang J, Adamian L, Jackups R Jr (2005) The membrane-water interface region of membrane proteins: structural bias and the anti-snorkeling effect. Trends Biochem Sci 30 (7):355–357. doi:10.1016/j.tibs.2005.05.003
46. Viklund H, Granseth E, Elofsson A (2006) Structural classification and prediction of reentrant regions in alpha-helical transmembrane proteins: application to complete genomes. J Mol Biol 361(3):591–603. doi:10.1016/j.jmb.2006.06.037
47. Yan C, Luo J (2010) An analysis of reentrant loops. Protein J 29(5):350–354. doi:10.1007/s10930-010-9259-z
48. Van den Berg B, Clemons WM Jr, Collinson I et al (2004) X-ray structure of a protein-conducting channel. Nature 427 (6969):36–44. doi:10.1038/nature02218
49. Dutzler R, Campbell EB, Cadene M et al (2002) X-ray structure of a ClC chloride channel at 3.0 A reveals the molecular basis of anion selectivity. Nature 415 (6869):287–294. doi:10.1038/415287a
50. Zhou Y, Morais-Cabral JH, Kaufman A et al (2001) Chemistry of ion coordination and hydration revealed by a K+ channel-Fab complex at 2.0 A resolution. Nature 414 (6859):43–48. doi:10.1038/35102009
51. Mitsuoka K, Murata K, Walz T et al (1999) The structure of aquaporin-1 at 4.5-A resolution reveals short alpha-helices in the center of the monomer. J Struct Biol 128(1):34–43. doi:10.1006/jsbi.1999.4177
52. Rapp M, Granseth E, Seppala S et al (2006) Identification and evolution of dual-topology membrane proteins. Nat Struct Mol Biol 13 (2):112–116. doi:10.1038/nsmb1057

53. Rost B (1996) PHD: predicting one-dimensional protein structure by profile-based neural networks. Methods Enzymol 266:525–539
54. Tusnady GE, Simon I (1998) Principles governing amino acid composition of integral membrane proteins: application to topology prediction. J Mol Biol 283(2):489–506. doi:10.1006/jmbi.1998.2107
55. Kall L, Krogh A, Sonnhammer EL (2005) An HMM posterior decoder for sequence feature prediction that includes homology information. Bioinformatics 21(Suppl 1):i251–i257. doi:10.1093/bioinformatics/bti1014
56. Petersen TN, Brunak S, von Heijne G et al (2011) SignalP 4.0: discriminating signal peptides from transmembrane regions. Nat Methods 8(10):785–786. doi:10.1038/nmeth.1701
57. Tsirigos KD, Peters C, Shu N et al (2015) The TOPCONS web server for consensus prediction of membrane protein topology and signal peptides. Nucleic Acids Res 43(W1): W401–W407. doi:10.1093/nar/gkv485
58. Bernsel A, Viklund H, Falk J et al (2008) Prediction of membrane-protein topology from first principles. Proc Natl Acad Sci U S A 105(20):7177–7181. doi:10.1073/pnas.0711151105
59. Peters C, Tsirigos KD, Shu N et al (2015) Improved topology prediction using the terminal hydrophobic helices rule. Bioinformatics 32:1158–1162. doi:10.1093/bioinformatics/btv709
60. Hessa T, Meindl-Beinker NM, Bernsel A et al (2007) Molecular code for transmembrane-helix recognition by the Sec61 translocon. Nature 450(7172):1026–1030. doi:10.1038/nature06387
61. Granseth E, Viklund H, Elofsson A (2006) ZPRED: predicting the distance to the membrane center for residues in alpha-helical membrane proteins. Bioinformatics 22(14): e191–e196. doi:10.1093/bioinformatics/btl206
62. van Geest M, Lolkema JS (2000) Membrane topology and insertion of membrane proteins: search for topogenic signals. Microbiol Mol Biol Rev 64(1):13–33
63. Bernsel A, Von Heijne G (2005) Improved membrane protein topology prediction by domain assignments. Protein Sci 14 (7):1723–1728. doi:10.1110/ps.051395305
64. Letunic I, Copley RR, Pils B et al (2006) SMART 5: domains in the context of genomes and networks. Nucleic Acids Res 34 (Database issue):D257–D260. doi:10.1093/nar/gkj079
65. Mulder NJ, Apweiler R, Attwood TK et al (2007) New developments in the InterPro database. Nucleic Acids Res 35(Database issue):D224–D228. doi:10.1093/nar/gkl841
66. Finn RD, Tate J, Mistry J et al (2008) The Pfam protein families database. Nucleic Acids Res 36(Database issue):D281–D288. doi:10.1093/nar/gkm960
67. Tusnady GE, Kalmar L, Hegyi H et al (2008) TOPDOM: database of domains and motifs with conservative location in transmembrane proteins. Bioinformatics 24(12):1469–1470. doi:10.1093/bioinformatics/btn202
68. Rabiner L (1989) A tutorial on hidden Markov models and selected applications in speech recognition. Proc IEEE 77 (2):257–286
69. Eddy SR (1998) Profile hidden Markov models. Bioinformatics 14(9):755–763, doi: btb114 [pii]
70. Nielsen H, Krogh A (1998) Prediction of signal peptides and signal anchors by a hidden Markov model. Proc Int Conf Intell Syst Mol Biol 6:122–130
71. Krogh A (1994) Hidden Markov models for labelled sequences. In: Proceedings of the12th IAPR international conference on pattern recognition, pp 140–144
72. Martelli PL, Fariselli P, Krogh A et al (2002) A sequence-profile-based HMM for predicting and discriminating beta barrel membrane proteins. Bioinformatics 18(Suppl 1): S46–S53
73. Khoury GA, Baliban RC, Floudas CA (2011) Proteome-wide post-translational modification statistics: frequency analysis and curation of the swiss-prot database. Sci Rep 1:90. doi:10.1038/srep00090
74. Apweiler R, Hermjakob H, Sharon N (1999) On the frequency of protein glycosylation, as deduced from analysis of the SWISS-PROT database. Biochim Biophys Acta 1473 (1):4–8, doi:S0304-4165(99)00165-8 [pii]
75. Welply JK, Shenbagamurthi P, Lennarz WJ et al (1983) Substrate recognition by oligosaccharyltransferase. Studies on glycosylation of modified Asn-X-Thr/Ser tripeptides. J Biol Chem 258(19):11856–11863
76. Nilsson IM, von Heijne G (1993) Determination of the distance between the oligosaccharyltransferase active site and the endoplasmic reticulum membrane. J Biol Chem 268 (8):5798–5801

77. Popov M, Li J, Reithmeier RA (1999) Transmembrane folding of the human erythrocyte anion exchanger (AE1, Band 3) determined by scanning and insertional N-glycosylation mutagenesis. Biochem J 339(Pt 2):269–279
78. Popov M, Tam LY, Li J et al (1997) Mapping the ends of transmembrane segments in a polytopic membrane protein. Scanning N-glycosylation mutagenesis of extracytosolic loops in the anion exchanger, band 3. J Biol Chem 272(29):18325–18332
79. Landolt-Marticorena C, Reithmeier RA (1994) Asparagine-linked oligosaccharides are localized to single extracytosolic segments in multi-span membrane glycoproteins. Biochem J 302(Pt 1):253–260
80. Pawson T, Scott JD (2005) Protein phosphorylation in signaling—50 years and counting. Trends Biochem Sci 30(6):286–290. doi:10.1016/j.tibs.2005.04.013
81. Hunter T (2009) Tyrosine phosphorylation: thirty years and counting. Curr Opin Cell Biol 21(2):140–146. doi:10.1016/j.ceb.2009.01.028
82. Wood CD, Thornton TM, Sabio G et al (2009) Nuclear localization of p38 MAPK in response to DNA damage. Int J Biol Sci 5(5):428–437
83. Zhang J, Johnson GV (2000) Tau protein is hyperphosphorylated in a site-specific manner in apoptotic neuronal PC12 cells. J Neurochem 75(6):2346–2357
84. Kalume DE, Molina H, Pandey A (2003) Tackling the phosphoproteome: tools and strategies. Curr Opin Chem Biol 7(1):64–69, doi:S1367593102000091 [pii]
85. Tsaousis GN, Bagos PG, Hamodrakas SJ (2014) HMMpTM: Improving transmembrane protein topology prediction using phosphorylation and glycosylation site prediction. Biochim Biophys Acta 1844(2):316–322. doi:10.1016/j.bbapap.2013.11.001
86. Wistrand M, Käll L, Sonnhammer EL (2006) A general model of G protein-coupled receptor sequences and its application to detect remote homologs. Protein Sci 15(3):509–521. doi:10.1110/ps.051745906
87. Theodoropoulou MC, Tsaousis GN, Litou ZI et al (2013) GPCRpipe: a pipeline for the detection of G-protein coupled receptors in proteomes. In: Joint 21st annual international conference on Intelligent Systems for Molecular Biology (ISMB) and 12th European Conference on Computational Biology (ECCB), 2013
88. Lomize MA, Lomize AL, Pogozheva ID et al (2006) OPM: orientations of proteins in membranes database. Bioinformatics 22(5):623–625. doi:10.1093/bioinformatics/btk023
89. Dobson L, Lango T, Remenyi I et al (2015) Expediting topology data gathering for the TOPDB database. Nucleic Acids Res 43(Database issue):D283–D289. doi:10.1093/nar/gku1119
90. Tsaousis GN, Tsirigos KD, Andrianou XD et al (2010) ExTopoDB: a database of experimentally derived topological models of transmembrane proteins. Bioinformatics 26(19):2490–2492. doi:10.1093/bioinformatics/btq362
91. Altschul SF, Madden TL, Schaffer AA et al (1997) Gapped BLAST and PSI-BLAST: a new generation of protein database search programs. Nucleic Acids Res 25(17):3389–3402, doi:gka562 [pii]
92. Bagos PG, Tsaousis GN, Hamodrakas SJ (2009) How many 3D structures do we need to train a predictor? Genomics Proteomics Bioinformatics 7(3):128–137. doi:10.1016/S1672-0229(08)60041-8
93. Zemla A, Venclovas C, Fidelis K et al (1999) A modified definition of Sov, a segment-based measure for protein secondary structure prediction assessment. Proteins 34(2):220–223. doi:10.1002/(SICI)1097-0134(19990201)34:2
94. Baldi P, Brunak S, Chauvin Y et al (2000) Assessing the accuracy of prediction algorithms for classification: an overview. Bioinformatics 16(5):412–424
95. Baum LE (1972) An inequality and associated maximization technique in statistical estimation for probabilistic functions of Markov processes. Inequalities 3:1–8
96. Dempster AP, Laird NM, Rubin DB (1977) Maximum likelihood from incomplete data via the EM algorithm. J R Stat Soc B Methodol 39(1):1–38. doi:10.2307/2984875
97. Krogh A (1997) Two methods for improving performance of an HMM and their application for gene finding. Proc Int Conf Intell Syst Mol Biol 5:179–186
98. Bagos P, Liakopoulos T, Hamodrakas S (2004) Faster gradient descent training of hidden Markov models, using individual learning rate adaptation. In: Paliouras G, Sakakibara Y (eds) Grammatical inference: algorithms and applications, vol 3264, Lecture notes in computer science. Springer, Berlin, Heidelberg, pp 40–52. doi:10.1007/978-3-540-30195-0_5

99. Krogh A, Riis SK (1999) Hidden neural networks. Neural Comput 11(2):541–563
100. Schwartz R, Chow YL (1990) The N-best algorithms: an efficient and exact procedure for finding the N most likely sentence hypotheses. In: 1990 international conference on acoustics, speech, and signal processing, 1990. ICASSP-90, 3–6 Apr 1990, vol 81, pp 81–84. doi:10.1109/icassp.1990.115542
101. Jacoboni I, Martelli PL, Fariselli P et al (2001) Prediction of the transmembrane regions of beta-barrel membrane proteins with a neural network-based predictor. Protein Sci 10(4):779–787. doi:10.1110/ps.37201
102. Fariselli P, Finelli M, Marchignoli D et al (2003) MaxSubSeq: an algorithm for segment-length optimization. The case study of the transmembrane spanning segments. Bioinformatics 19(4):500–505
103. Fariselli P, Martelli PL, Casadio R (2005) A new decoding algorithm for hidden Markov models improves the prediction of the topology of all-beta membrane proteins. BMC Bioinformatics 6(Suppl 4):S12
104. Virkki MT, Peters C, Nilsson D et al (2014) The positive inside rule is stronger when followed by a transmembrane helix. J Mol Biol 426(16):2982–2991. doi:10.1016/j.jmb.2014.06.002
105. Wang H, Zhang C, Shi X et al (2012) Improving transmembrane protein consensus topology prediction using inter-helical interaction. Biochim Biophys Acta 1818(11):2679–2686. doi:10.1016/j.bbamem.2012.05.030
106. Nugent T, Ward S, Jones DT (2011) The MEMPACK alpha-helical transmembrane protein structure prediction server. Bioinformatics 27(10):1438–1439. doi:10.1093/bioinformatics/btr096

Chapter 6

Self-Organizing Hidden Markov Model Map (SOHMMM): Biological Sequence Clustering and Cluster Visualization

Christos Ferles, William-Scott Beaufort, and Vanessa Ferle

Abstract

The present study devises mapping methodologies and projection techniques that visualize and demonstrate biological sequence data clustering results. The Sequence Data Density Display (SDDD) and Sequence Likelihood Projection (SLP) visualizations represent the input symbolical sequences in a lower-dimensional space in such a way that the clusters and relations of data elements are depicted graphically. Both operate in combination/synergy with the Self-Organizing Hidden Markov Model Map (SOHMMM). The resulting unified framework is in position to analyze automatically and directly raw sequence data. This analysis is carried out with little, or even complete absence of, prior information/domain knowledge.

Keywords DNA/RNA/protein sequence data, Biological chain molecule, Clustering, Visualization, Mapping, Nonlinear projection, Self-organizing map (SOM), Hidden Markov model (HMM)

1 Introduction

Deoxyribonucleic acid (DNA) , ribonucleic acid (RNA), and protein chain molecules are characteristic paradigms of sequence data which are intrinsically encoded in either the four-letter alphabet of nucleotides or the 20-letter alphabet of amino acids. Machine learning approaches that are in position to incorporate and process such symbolical sequence data, continuously expand their application range, since they have proven effective for modeling, processing, and analyzing biological molecules.

One common situation in early stages of bioinformatics projects is that the only available information comes in the form of biological sequence data (viz., DNA, RNA, and protein sequences); any other kind of prior information or domain knowledge is, or is considered to be, inexistent. Clustering's objective is to produce simplified descriptions and summaries of large data sets by adopting primordial exploration strategies which necessitate little or no prior knowledge [1, 2]. Clustering is one standard method that can

David R. Westhead and M.S. Vijayabaskar (eds.), *Hidden Markov Models: Methods and Protocols*, Methods in Molecular Biology, vol. 1552, DOI 10.1007/978-1-4939-6753-7_6, © Springer Science+Business Media LLC 2017

create higher abstractions from raw data automatically; another alternative method is to project high-dimensional data as points on a low-dimensional display. Because the Self-Organizing Map (SOM) [3] is a combination of these two methods, it has great capabilities as a data clustering, mapping and visualization algorithm [4–9].

The Self-Organizing Hidden Markov Model Map (SOHMMM) [10–14] is a hybrid unsupervised approach that can process and analyze DNA, RNA, protein chain molecules, and generic sequences of high dimensionality and variable lengths encoded directly in non-numerical/symbolical alphabets. On the contrary, various other models that extend the SOM for processing sequence data are in position to incorporate and analyze symbolical/biological sequences only after interposing a preprocessing stage (e.g., orthogonal or Euclidean encoding, weighted Levenshtein distances, local feature distances, pairwise dissimilarities), see [15–20] and references therein. The main task of this intermediate stage is to transform input sequences to numerical data spaces. Nevertheless, additional computational complexity, and frequently, loss of information are inevitable aftereffects of such preprocessing transformations.

The present study proposes cluster visualization and nonlinear projection methodologies, for symbolical sequence data (i.e., biological sequences), that are based on the SOHMMM. The devised techniques can be applied directly to raw sequence data derived from biological chain molecules, and can operate on an automated basis. Furthermore, the overall unified framework is capable of carrying out the corresponding probabilistic sequence analysis even under complete absence of prior knowledge.

The rest of this paper is organized and structured as follows. After a brief overview of the SOHMMM prototype in Subheading 2, we commence with a discussion on clustering (unsupervised classification), and in particular, on the plausibility of exploiting prior knowledge/domain information within a straightforward unsupervised framework. Also, in Subheading 3, we introduce and present in detail the two proposed visualization techniques. The corresponding subsections begin with a theoretical qualitative description of each graphic mapping methodology followed by an algorithmic realization. Experimental testing and evaluation which are based on the globin protein family follow next, coupled with characteristic and representative figures. Both subsections end with a short discussion on the functions and capabilities of the implemented visualization interface. Subheading 4 contains an additional series of experiments on splice junction sequence data which can be regarded as an exemplary application scenario of the proposed graphic mapping techniques. Finally, Subheading 5 synopsizes the paper.

2 Self-Organizing Hidden Markov Model Map

2.1 Hidden Markov Model

In essence, a Hidden Markov Model (HMM) is a stochastic process generated by two interwoven probabilistic mechanisms, an underlying one (i.e., hidden) and an observable one (which produces the symbolical sequences). Thereinafter, we may denote each individual HMM as $\lambda = (A, B, \pi)$, where $A = \{a_{ij}\}$, $B = \{b_j(k)\}$ and $\pi = \{\pi_j\}$ are the transition, emission and initial state probability stochastic matrices respectively.

Let $\{q_t\}_{t=1}^{\infty}$ be a homogeneous Markov chain, where each random variable q_t assumes a value in the state space $S = \{s_1, s_2, \ldots, s_N\}$. The conditional stationary probabilities are denoted as: $a_{ij} = P(q_t = s_j | q_{t-1} = s_i)$, $t > 1$, $1 \le i \le N$, $1 \le j \le N$. Let $\{\Upsilon_t\}_{t=1}^{\infty}$ be a random process defined over the finite space $V = \{v_1, v_2, \ldots, v_M\}$ where, in general, $M \neq N$. The processes $\{q_t\}_{t=1}^{\infty}$ and $\{\Upsilon_t\}_{t=1}^{\infty}$ are related by the conditional probability distributions: $b_j(k) = P(\Upsilon_t = v_k | q_t = s_j)$, $t \ge 1$, $1 \le j \le N$, $1 \le k \le M$. In certain cases the emission probabilities' indexes are denoted as o_t and not as k. This interchange is made whenever the exact observation symbols are insignificant for the formulation under consideration, whereas, these values are considered to be given and specific. The initial state probability distribution is defined as: $\pi_j = P(q_1 = s_j)$, $1 \le j \le N$.

2.2 The Prototype

There are two opposing tendencies in the self-organizing process. First, the set of reference patterns tends to describe the density function of the input data. Second, local interactions between processing neurons tend to preserve continuity in the low-dimensional lattice of outputs. A result of these opposing tensions is that the reference pattern distribution, tending to approximate a smooth hypersurface, also seeks an optimal orientation and form in the output space that best imitates the overall structure of the input data distributions. Although the expressive power and model fitting properties of a single HMM are unquestionable, the implementation of reference pattern distributions requires collections of HMMs. Subsequently, each individual HMM that comprises these ensembles is adjusted appropriately in such a way that the produced mapping is ordered, descriptive and representative of the input data distribution.

The SOHMMM introduces a hybrid integration of the SOM and the HMM. In essence, it implements a nonlinear structured mapping of sequence data onto the reference HMMs of a low-dimensional array. In its basic form it produces a probability distribution graph of input data that summarizes the HMMs' distributions on the sequence space; subsequently, under certain conditions, the nonlinear statistical relationships between sequence data can be visually detected, investigated and interpreted.

As the SOHMMM thereby compresses information, whereas, preserving the most significant statistical relationships of the primary data elements on the display, it may also be thought to produce some kind of abstraction. These characteristics, abstraction, dimensionality reduction and visualization in synergy with clustering, can be utilized in a number of sequence data analysis tasks.

More specifically, SOHMMM's mapping is implemented in the following way, which closely resembles the two aforementioned opposing tendencies:

1. The leading partial process, viz., competition, finds the neuron yielding the maximum likelihood for the observation sequence under consideration, namely the position of the best match identifies with the location of the highest response (usually this array element is referred to as winner and is indicated by the subscript c). Subsequently, the input sequence is mapped onto this location, like in a set of decoders.
2. The following partial process, viz., cooperation, improves the match in the neighboring neurons located in the vicinity of the winner (Fig. 1), in an effort to gain some knowledge from the input sequence. In particular, those HMMs that are topologically close according a certain geometric distance (viz., all HMMs within the winner's neighborhood NB_c) will optimize their parameters so as to best describe how a given observation comes about. This will result in a local relaxation or smoothing effect on the parameters of the HMMs in this neighborhood, which in continued learning leads to global ordering.

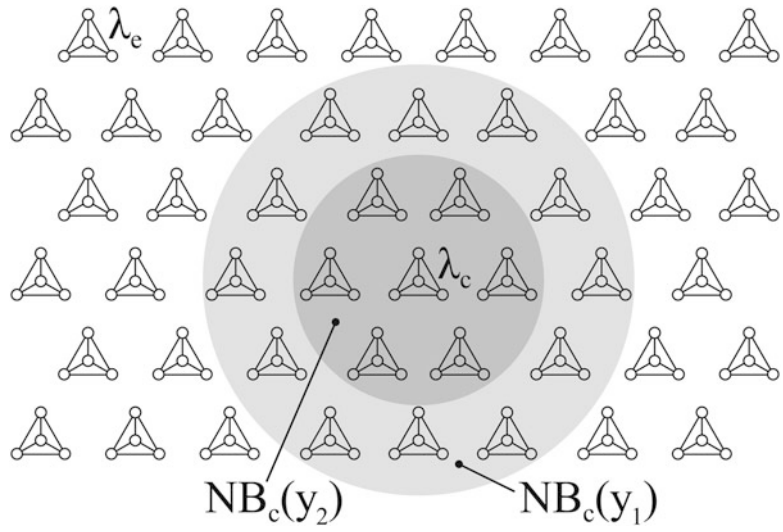


Fig. 1 Paradigm of a hexagonal 8 × 6 SOHMMM lattice where each reference HMM λ_e is depicted as a fully connected four-vertex graph. The *circular shaded areas* are an indicative winner neuron's (λ_c) closest topological neighborhoods (NB_c) at different points in time ($y_1 < y_2$). The neighborhood's radius is decreasing over time since the learning step y_2 is many iterations after the y_1 step

Each neuron in the SOHMMM array consists of a reference HMM λ_e, $1 \le e \le E$ (E represents the overall number of neurons). Because the SOHMMM is conceptualized as a self-organizing methodology its algorithmic realization fuses the corresponding competition and cooperation processes. Both complementary processes of the SOHMMM algorithm are detailed next:

for $e = 1$ to E

$$\tilde{\alpha}_t^{(e)}(i) = \left[\sum_{j=1}^{N} \hat{\alpha}_{t-1}^{(e)}(j) a_{ji}^{(e)}\right] b_i^{(e)}(o_t),\ 1 \le i \le N,\ 2 \le t \le T;$$

$$sc_t^{(e)} = \left(\sum_{i=1}^{N} \tilde{\alpha}_t^{(e)}(i)\right)^{-1},\ 1 \le t \le T;$$

$$\hat{\alpha}_t^{(e)}(i) = sc_t^{(e)} \tilde{\alpha}_t^{(e)}(i),\ 1 \le i \le N,\ 2 \le t \le T;$$

$$\tilde{\beta}_t^{(e)}(i) = \sum_{j=1}^{N} a_{ij}^{(e)} b_j^{(e)}(o_{t+1}) \hat{\beta}_{t+1}^{(e)}(j),\ 1 \le i \le N,\ t = T-1, \ldots, 1;$$

$$\hat{\beta}_t^{(e)}(i) = sc_t^{(e)} \tilde{\beta}_t^{(e)}(i),\ 1 \le i \le N,\ t = T-1, \ldots, 1;$$

end for

$$c \leftarrow \arg \max_{e} \left\{ -\sum_{t=1}^{T} \log\left(sc_t^{(e)}\right) \right\};$$

for $e = 1$ to E

$$w_{ij}^{(e)} \leftarrow$$

$$w_{ij}^{(e)} + \eta(y) h_{ce}(y) \left\{ a_{ij} \sum_{l=1}^{T-1} \left[\hat{\alpha}_l(i) \left(b_j(o_{l+1}) \hat{\beta}_{l+1}(j) - sc_l^{-1} \right.\right.\right.$$

$$\left.\left.\left. \hat{\beta}_l(i) \right)\right]\Big|_{\lambda_e} \right\},\ 1 \le i \le N,\ 1 \le j \le N;$$

$$r_{jt}^{(e)} \leftarrow r_{jt}^{(e)} + \eta(y) h_{ce}(y) \left\{ \sum_{l=1}^{T} \left[sc_l^{-1} \hat{\alpha}_l(j) \hat{\beta}_l(j) \left(I\{o_l = t | \lambda\} - b_j \right.\right.\right.$$

$$\left.\left.\left. (t) \right)\right]\Big|_{\lambda_e} \right\},\ 1 \le j \le N,\ 1 \le t \le M;$$

$$u_j^{(e)} \leftarrow u_j^{(e)} + \eta(y) h_{ce}(y) \left\{ \left[\pi_j b_j(o_1) \hat{\beta}_1(j) - \pi_j \right] \Big|_{\lambda_e} \right\},\ 1 \le j \le N;$$

$$a_{ij}^{(e)} \leftarrow e^{w_{ij}^{(e)}} \Big/ \sum_{l=1}^{N} e^{w_{il}^{(e)}},\ 1 \le i \le N,\ 1 \le j \le N;$$

$$b_j^{(e)}(t) \leftarrow e^{r_{jt}^{(e)}} \Big/ \sum_{l=1}^{M} e^{r_{jl}^{(e)}},\ 1 \le j \le N,\ 1 \le t \le M;$$

$$\pi_j^{(e)} \leftarrow e^{u_j^{(e)}} \Big/ \sum_{l=1}^{N} e^{u_l^{(e)}},\ 1 \le j \le N;$$

end for

$O = o_1 o_2 \ldots o_T$ represents an input observation sequence and T is its length. The scaled forward and backward variables of reference

HMM λ_e are denoted as $\hat{\alpha}_t^{(e)}(i)$ and $\hat{\beta}_t^{(e)}(i)$ respectively; whereas, $sc_t^{(e)}$ are its scaling coefficients. Variable $y \geq 0$ is an integer, the discrete time coordinate. The function $\eta(y)$ plays the role of a scalar learning rate factor. The function $h_{ce}(y)$ acts as the neighborhood function; the width and form of $h_{ce}(y)$ define the stiffness of the elastic surface that is fitted to the input sequence data. $I\{o_l = t|\lambda\}$ is the indicator function. The exponentially normalized variables of reference HMM λ_e are denoted as $w_{ij}^{(e)}$, $r_{jt}^{(e)}$ and $u_j^{(e)}$.

It should be noted that the SOHMMM learning algorithm is clearly on-line, and as such, it is closer to the Bayesian spirit of updating one's belief as data become available. More important, perhaps, learning after the presentation of each sample introduces a degree of stochasticity that may be useful in exploring the space of solutions and averting the phenomenon of trapping in local optima.

3 Visualization Techniques and Graphic Displays

Clustering (also called unsupervised classification) is a branch of exploratory data analysis where prior knowledge and/or domain information on the given problem space is (or is considered to be) absent or unknown. According to the rather strict formulation a methodology can be categorized as unsupervised if it yields results by incorporating raw data only and by exploiting information retrieved solely from such data. This assumption should apply to each and every stage of a clustering procedure (and of potential further visualization processes); thus, prior knowledge and/or domain information should be considered absent within a straightforward unsupervised framework. Consequently, it should be expected that clustering techniques yield results biased towards a generic qualitative description of the problem at hand rather than a precise quantitative analysis. It would be a paradox if lack of prior information came at no cost. Nevertheless, in real case problems where sequences are unclassified, orphan or of unknown origin the only means towards a primordial investigation are unsupervised methodologies.

More specifically, in the case where uncategorized biological sequences or chain molecules define the given problem space, the only provided information/data are the relative positions of monomers in the sequences (and obviously the sequences' lengths). Any type of additional prior/domain knowledge suggests a deviation from the straightforward unsupervised discipline. It has been shown that the SOHMMM is in position to carry out unsupervised tasks (i.e., clustering) by operating on both artificial and biological sequence problem spaces [10, 11]. In the present study we attempt to extend and broaden the function range of the SOHMMM by equipping it with visualization and nonlinear projection

capabilities, viz., providing the ability to map high-dimensional sequences as points on a low-dimensional display. This is achieved by devising the Sequence Data Density Display (SDDD) and the Sequence Likelihood Projection (SLP) . The former exploits, in parallel, the landslide of provided data whereas the latter uses a particular sequence for creating a certain projection. Nevertheless, both visualization techniques share the characteristic of operating on a pure unsupervised basis. Obviously, low-dimensional displays are to be preferred for visual inspection reasons.

3.1 Sequence Data Density Display

In essence, the SDDD depicts the number/amount of sequences which are assigned to each individual neuron of the SOHMMM mesh. Each sequence's winner (i.e., best matching) SOHMMM node is determined according to the maximum likelihood criterion. Furthermore, the corresponding graphic display, apart from the sequence data density, takes into consideration the type and dimensions of the SOHMMM lattice as well as the topology and positioning of the neurons. Each neuron is drawn with a size relative to the number of assigned sequences. An aftereffect of this procedure is that, in certain cases, clusters (of input sequence data) can be visually detected by searching for groups of topologically close/neighboring large size nodes separated by areas of small (or zero) size nodes. It should be noted that a neuron is not drawn at all (usually referred to as zero size) if it is not the best match for any sequence. A generic algorithmic formulation of the SDDD is given next:

/* Sequence Data Density Display */
for $d = 1$ to D
  for $e = 1$ to E

$$\tilde{\alpha}_t^{(e)}(i) = \left[\sum_{j=1}^{N} \hat{\alpha}_{t-1}^{(e)}(j) a_{ji}^{(e)}\right] b_i^{(e)}(o_t),\ 1 \le i \le N,\ 2 \le t \le T_d;$$

$$sc_t^{(e)} = \left(\sum_{i=1}^{N} \tilde{\alpha}_t^{(e)}(i)\right)^{-1},\ 1 \le t \le T_d;$$

$$\hat{\alpha}_t^{(e)}(i) = sc_t^{(e)} \tilde{\alpha}_t^{(e)}(i),\ 1 \le i \le N,\ 2 \le t \le T_d;$$

$$\log P\left(O^{(d)} | \lambda_e\right) \leftarrow -\sum_{t=1}^{T_d} \log\left(sc_t^{(e)}\right);$$

  end for
  $c \leftarrow \arg \max_e \left\{\log P\left(O^{(d)} | \lambda_e\right)\right\};$
  $numOfSeqs[c] \leftarrow numOfSeqs[c] + 1;$
end for
$maxNumOfSeqs \leftarrow \max_e \{numOfSeqs[e]\};$
$minNumOfSeqs \leftarrow \min_e \{numOfSeqs[e]\};$
for $e = 1$ to E

  SDDD_Paint($numOfSeqs[e]$, $maxNumOfSeqs$, $minNumOfSeqs$);
end for

where D is the overall/total number of available sequences, $O^{(d)} = o_1 o_2 \ldots o_{T_d}$ is the dth observation sequence which consists of T_d symbols. $P(O|\lambda)$ is the probability of sequence O conditioned on λ (likelihood). For technical reasons, these probabilities can be very small. The solution is to work with the corresponding logarithms [21, 22].

In general, the computational complexity of SDDD is equal to $DEN^2T_d + EG$ operations, where G denotes the number of calculations required by each individual call to the respective native paint () function.

Two remarks can be made by examining the SDDD algorithm. First, the resulting visualization depends on and takes into consideration the provided sequence data and the already trained SOHMMM. Second, one can defensibly claim that the SDDD is a straightforward unsupervised visualization technique since the only required/necessary information is the corpus of provided monomer sequences.

Experimental investigation/verification of the SDDD and also, of the SLP utilizes the globin protein family. Globins form a well-known family of heme-containing proteins that reversibly bind oxygen, and are involved in its storage and transport. The globin protein family is a large family which is composed of subfamilies. From crystallographic studies, all globins have similar overall three-dimensional structures but widely divergent sequences. The globin sequences used here were extracted from the iProClass integrated protein knowledgebase [23], a database that provides extensive information integration of over 160 biological databases. In total, 560 proteins belonging to the three major globin subfamilies were retrieved (namely hemoglobin α-chains, hemoglobin β-chains, and myoglobins). The resulting globin data set's composition is 194 α-globins, 216 β-globins, and 150 myoglobins.

Initially, a hexagonal 6×7 mesh of 42 SOHMMM neurons is trained on the whole protein data set; further details, experimental investigation and comparative results can be found in [11]. The corresponding SDDD is illustrated in Fig. 2. An examination of the depicted display reveals that SDDD's two main goals are met satisfactorily. First, the number of each neuron's assigned sequences (viz., sequence data density) can be visually investigated. Second, three clusters are formed onto the SDDD, these consist of regions of large size nodes interrupted by ravines and gaps of small/zero size neurons. Furthermore, one can easily verify that these three clusters correspond to each individual globin subfamily. Protein class information that is excluded from the training and visualizing procedures can be used to perform a posterior identification/labeling of each SOHMMM node. The outcome of this procedure can be found in Fig. 3, the one-to-one correspondence between a depicted cluster and a globin subfamily is evident.

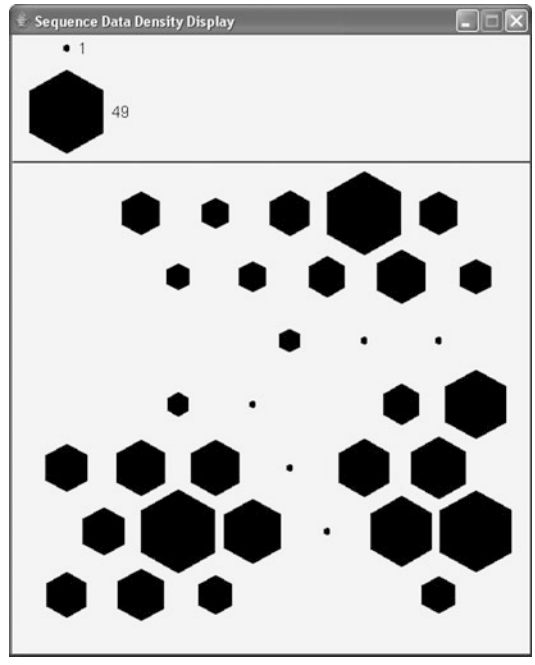


Fig. 2 SDDD for the whole globin protein family. Each hexagon, regardless of size, represents a SOHMMM neuron

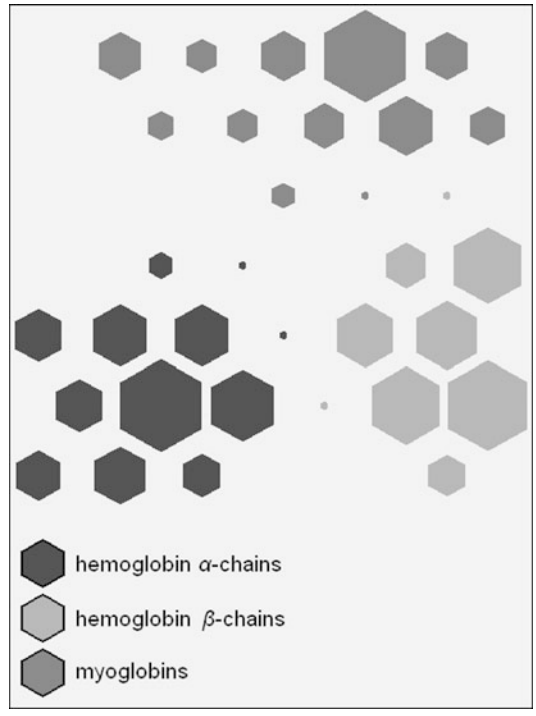


Fig. 3 SDDD's produced visualization enriched with information obtained from posterior labeling of depicted clusters

The Java language has been utilized for programming all the visualization interfaces. The main reason for this choice was the built-in platform independent support of graphics. The majority of supplied figures are screenshots of the implemented graphical user interface. The SDDD's interface (apart from the top-left legend) provides additional features and functionality. One can retrieve the amino acid or nucleotide descriptions, the total number and the identifiers of all the sequences which are assigned to each individual SOHMMM node, and also, access the corresponding HMM's parameters. This information can potentially be used for studying the profile and statistical properties of the assigned sequences [24–28].

An intrinsic drawback of the SDDD originates from its dependence from input sequence data. In certain cases where sequences that belong to specific clusters are orders of magnitude larger in numbers compared to the remaining clusters' sequences, visual detection of the latter clusters is obstructed (since these are suppressed on the display). Appropriate thresholds could be employed for overcoming this difficulty but their determination would deviate considerably from the unsupervised framework, and thus, should be avoided.

3.2 Sequence Likelihood Projection

Practically, the SLP is a graphic representation of the likelihood magnitudes (of a given sequence) with respect to each individual SOHMMM neuron. As expected, the corresponding visualization incorporates the type and dimensions of the SOHMMM array as well as the topology and positioning of the respective nodes. More specifically, each neuron's coloring is analogous to the likelihood value it yields; the color itself is selected relatively to the employed color scale. A potential aftereffect of this process is that a cluster can be visually traced (given the examined sequence belongs to a cluster which is represented) by searching for groups of topologically close/neighboring high likelihood nodes separated by areas of low likelihood nodes. A generic formulation of the SLP algorithm is the following:

/* Sequence Likelihood Projection */
$O \leftarrow$ sequenceSelection(OS);
for $e = 1$ to E

$$\tilde{\alpha}_t^{(e)}(i) = \left[\sum_{j=1}^{N} \hat{\alpha}_{t-1}^{(e)}(j) a_{ji}^{(e)}\right] b_i^{(e)}(o_t),\ 1 \le i \le N,\ 2 \le t \le T;$$

$$sc_t^{(e)} = \left(\sum_{i=1}^{N} \tilde{\alpha}_t^{(e)}(i)\right)^{-1},\ 1 \le t \le T;$$

$$\hat{\alpha}_t^{(e)}(i) = sc_t^{(e)} \tilde{\alpha}_t^{(e)}(i),\ 1 \le i \le N,\ 2 \le t \le T;$$

$$\log P\left(O|\lambda_e\right) \leftarrow -\sum_{t=1}^{T} \log\left(sc_t^{(e)}\right);$$

$$likelihood[e] \leftarrow \log P\left(O|\lambda_e\right);$$

end for
$maxLikelihood \leftarrow \max_e \{likelihood[e]\}$;
$minLikelihood \leftarrow \min_e \{likelihood[e]\}$;
for $e = 1$ to E
SLP_Paint($likelihood[e]$, max *Likelihood*, min *Likelihood*);
end for

where $OS = \left\{ O^{(1)}, O^{(2)}, \ldots, O^{(D)} \right\}$ is the set of D available observation sequences.

Overall, the computational complexity of SLP is equal to $EN^2 T_d + EG$ operations, similarly G denotes the number of calculations required by each individual call to the respective native paint() function.

Also in this case, two comments can be made. The resulting graphic display is based upon and incorporates the selected sequence and the already trained SOHMMM. Moreover, one can claim that the SLP is a direct unsupervised projection technique since the only required/necessary information is the provided monomer sequence.

The present experimental examination of the SLP was conducted by using the same SOHMMM that had been trained previously on the globin protein family. The resulting SLPs, for sequences belonging to the three different globin subfamilies, are illustrated in Figs. 4–6. The theoretically designated objectives are

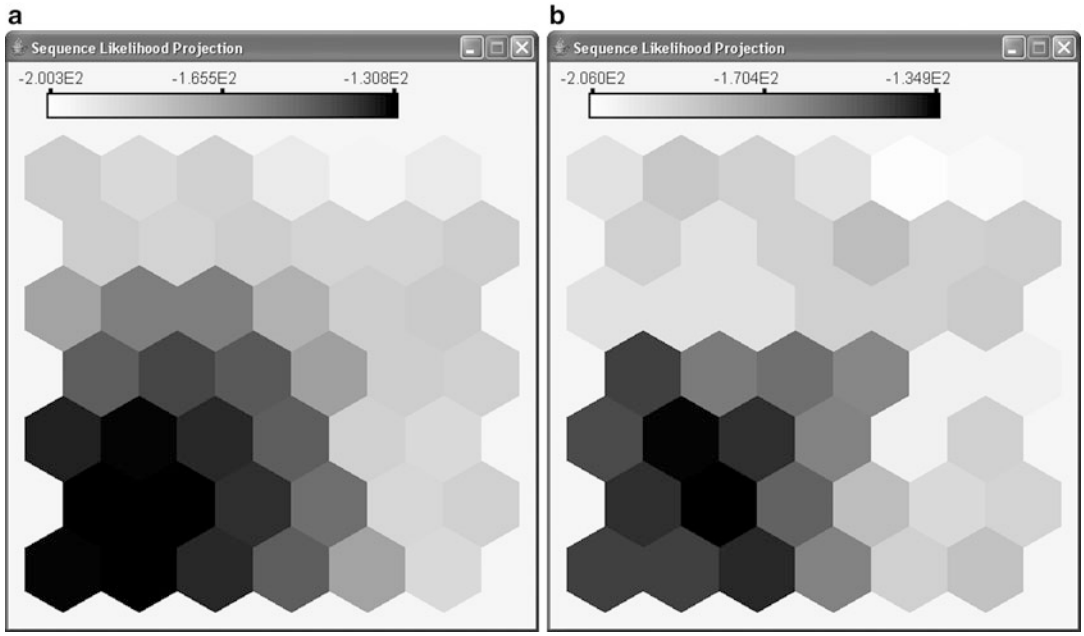


Fig. 4 Characteristic SLPs corresponding to α-globins. Each hexagon, regardless of color, represents a SOHMMM node

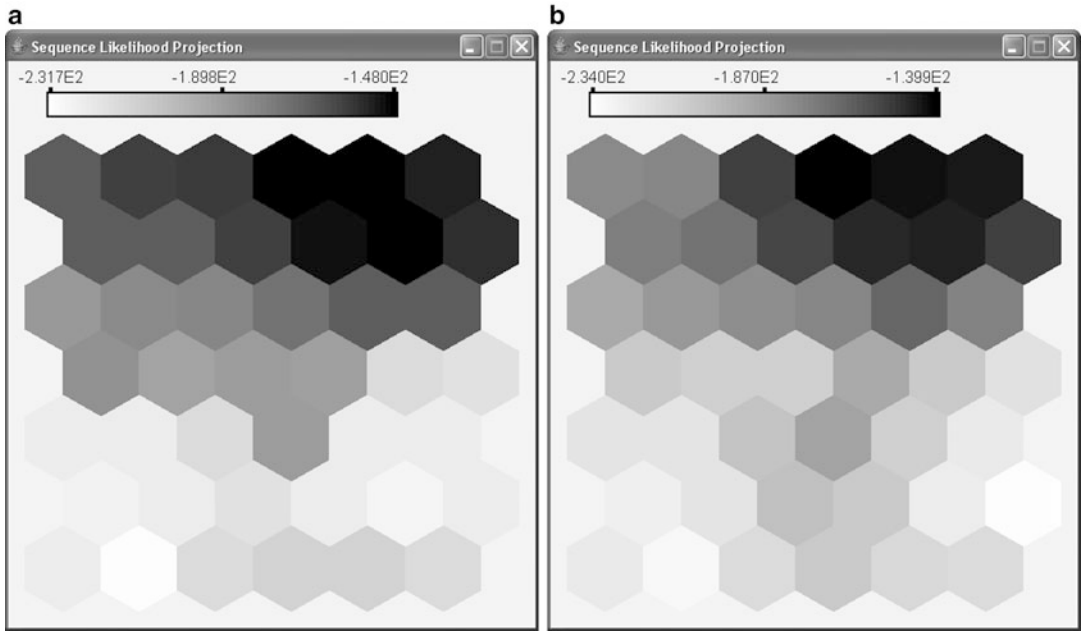


Fig. 5 Representative SLPs utilizing myoglobin sequences

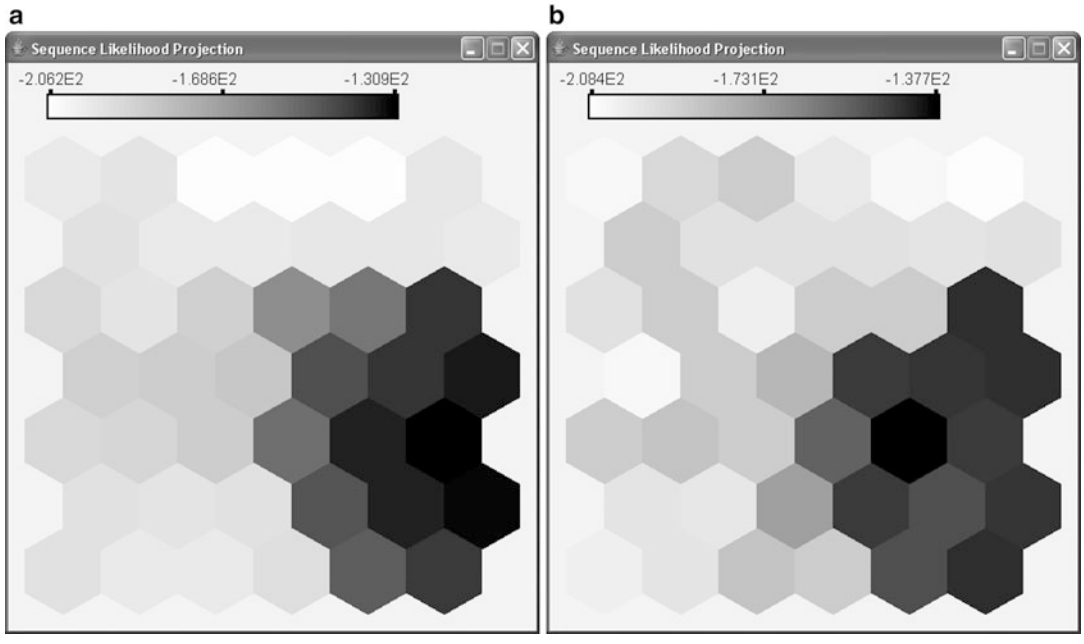


Fig. 6 Paradigms of SLP visualizations for hemoglobin β-chains

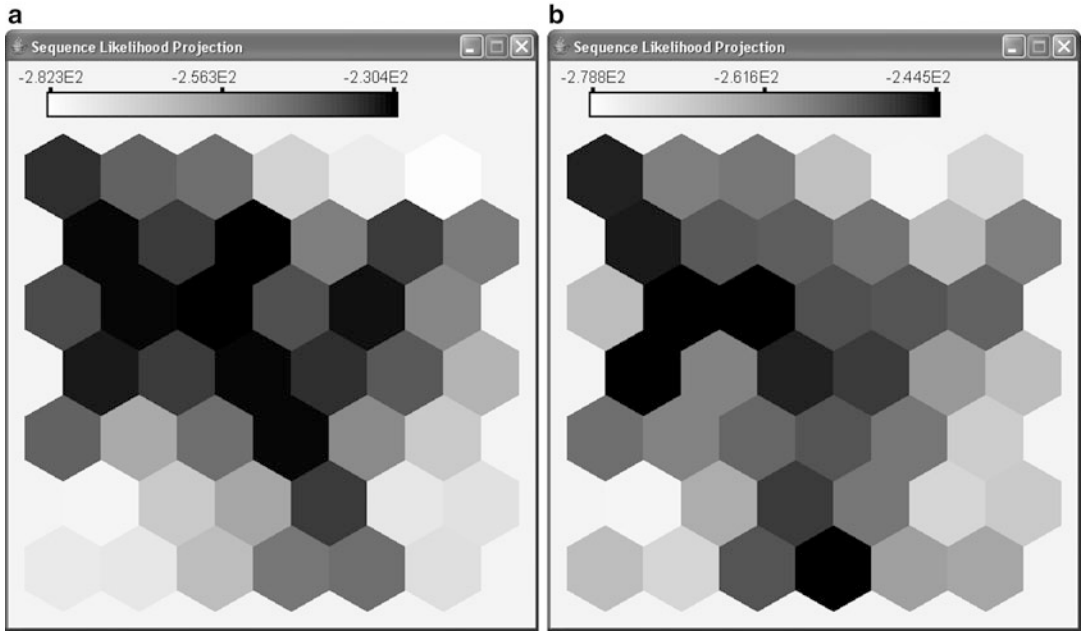


Fig. 7 Indicative SLP outcomes corresponding to a cytoglobin and a neuroglobin chain molecules

(at a certain extent) achieved. Each sequence's likelihood value distribution (or likelihood landscape) is represented visually. More important perhaps, one cluster is formed onto each SLP. This cluster consists of cohesive areas of high likelihood neurons surrounded by regions of low likelihood neurons. Furthermore, by keeping in mind the fact that the first SLPs (Fig. 4) describe hemoglobin α-sequences, the second SLPs (Fig. 5) regard myoglobins and the third SLPs (Fig. 6) represent β-globins; there is an evident analogy/correspondence with the detected α-globin, myoglobin, and hemoglobin β-chain clusters of the SDDD visualization (Fig. 2).

Per contra, a cytoglobin (Xenopus laevis) and a neuroglobin (Pan troglodytes) not used during the adaptation procedure of the SOHMMM, viz., not represented explicitly on the SOHMMM plane, are depicted in Fig. 7. In this case, instead of observing cohesive regions with well-defined boundaries, we see that sparse mosaic-like parcellations emerge. SLPs' resulting visualizations appear unstructured and unformed. This is justifiable since the specific SOHMMM has not been trained for modeling the cytoglobin and neuroglobin subfamilies or for clustering the respective chain molecules. Nevertheless, even in this case, the highest likelihood values are located in areas which do not contain any of the three known clusters (Fig. 2). We believe that this is an additional

indirect proof of the fact that the SOHMMM produces a nonlinear, ordered mapping of sequence data. Even though the cytoglobin and the neuroglobin are applied for the first time, they are not assigned falsely to a HMM neuron corresponding to one of the three previously identified clusters, instead the SOHMMM node that best describes these sequences does not represent any α-globin, β-globin, or myoglobin cluster.

As mentioned previously the SLP visualization interface has been programmed in Java. Likewise, all figures are screenshots of the implemented graphical user interface. The SLP's interface includes a color scale legend (which is in analogy to the minimum and maximum log-likelihood values). Moreover, it has functions for retrieving the examined sequence's identifier and amino acid/nucleotide chain, for displaying each neuron's likelihood magnitude/value, and for accessing the respective HMM's coefficients/parameters.

4 Application Scenario

Splice junctions mark transitions from expressed to unexpressed gene regions and vice versa, and they are thus important for the assembly of structural and regulatory proteins that constitute and control biochemical metabolisms. Splice site recognition is therefore a topic of interest for the understanding of genotype–phenotype relationships. The UCI machine learning repository [29] contains a data set for primate (eukaryotic) splice junction determination.

In an effort to put the SDDD's and the SLP's characteristics and capabilities in context we employ the devised unsupervised learning algorithm for training a hexagonal 9×4 SOHMMM array; further details, experimental investigation and comparative results can be found in [10]. The exact problem posed in this practical application is to recognize the exon-intron boundaries (frequently called donor splice sites) and the intron-exon boundaries (acceptor splice sites). The sequence data set's composition is 767 donors and 768 acceptors. The produced visualizations are shown in Figs. 8–10.

Initially, one can observe that the SDDD (in synergy with the underlying SOHMMM) identifies the existence of the two splice junction clusters (namely the acceptor cluster and the donor cluster). Moreover, alongside a strict unsupervised learning procedure, no prior knowledge nor category information are used in any stage, the recognized clusters are directly traced/identified visually (Fig. 8). The interpretation of the SLPs' results (Figs. 9 and 10) is dual. They can be considered as assignments of unknown sequences to the previously detected clusters, viz., sequences with exon-intron boundaries are assigned to the donor cluster and

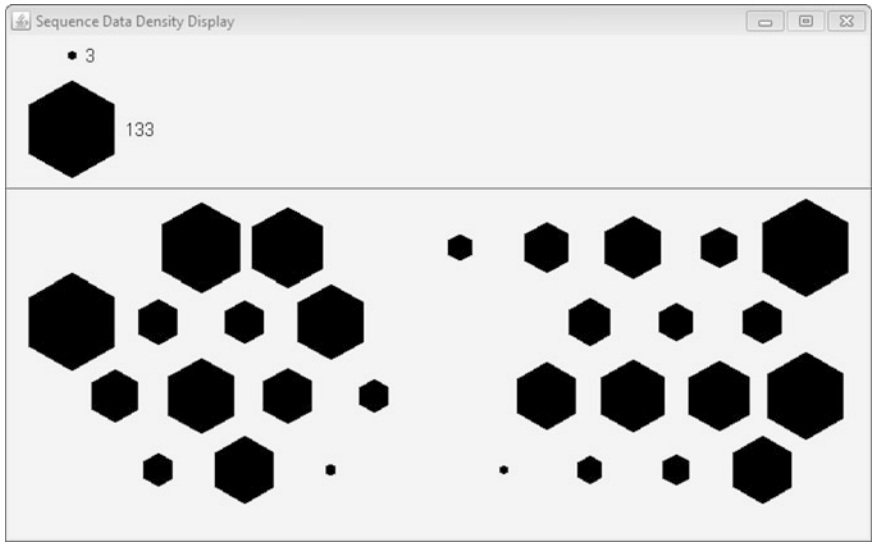


Fig. 8 SDDD for the splice junction recognition problem

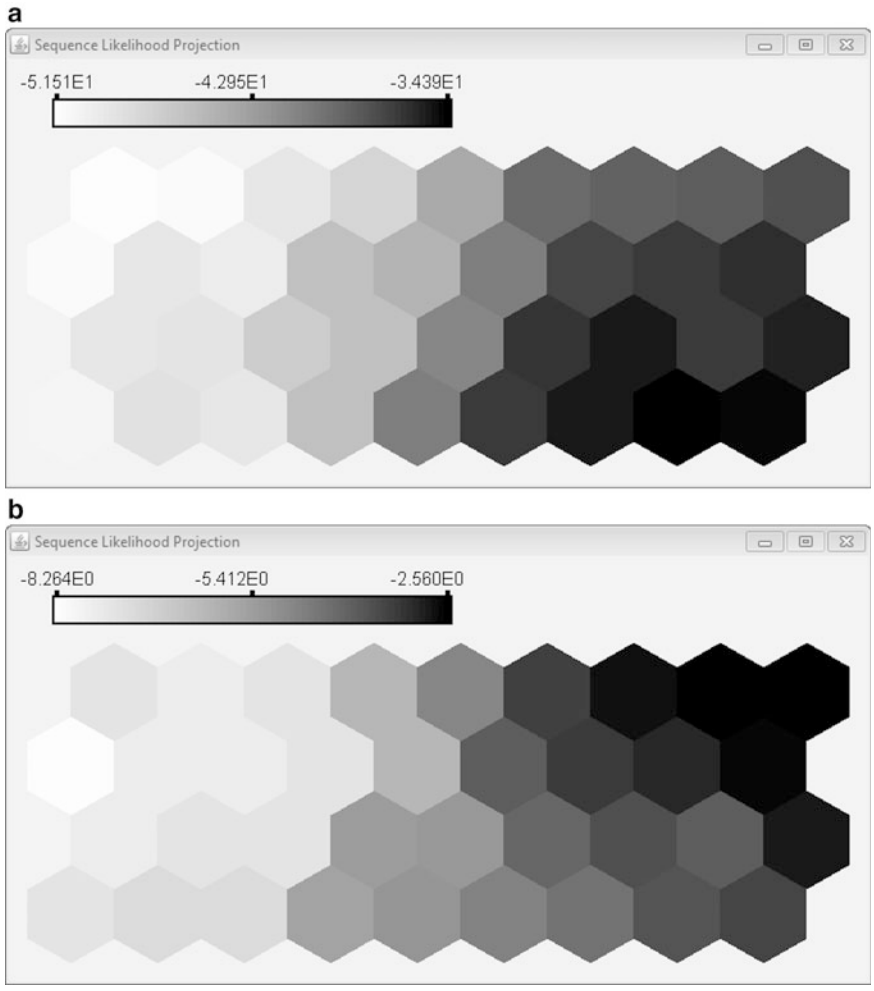


Fig. 9 Representative SLPs of sequences containing an exon-intron boundary

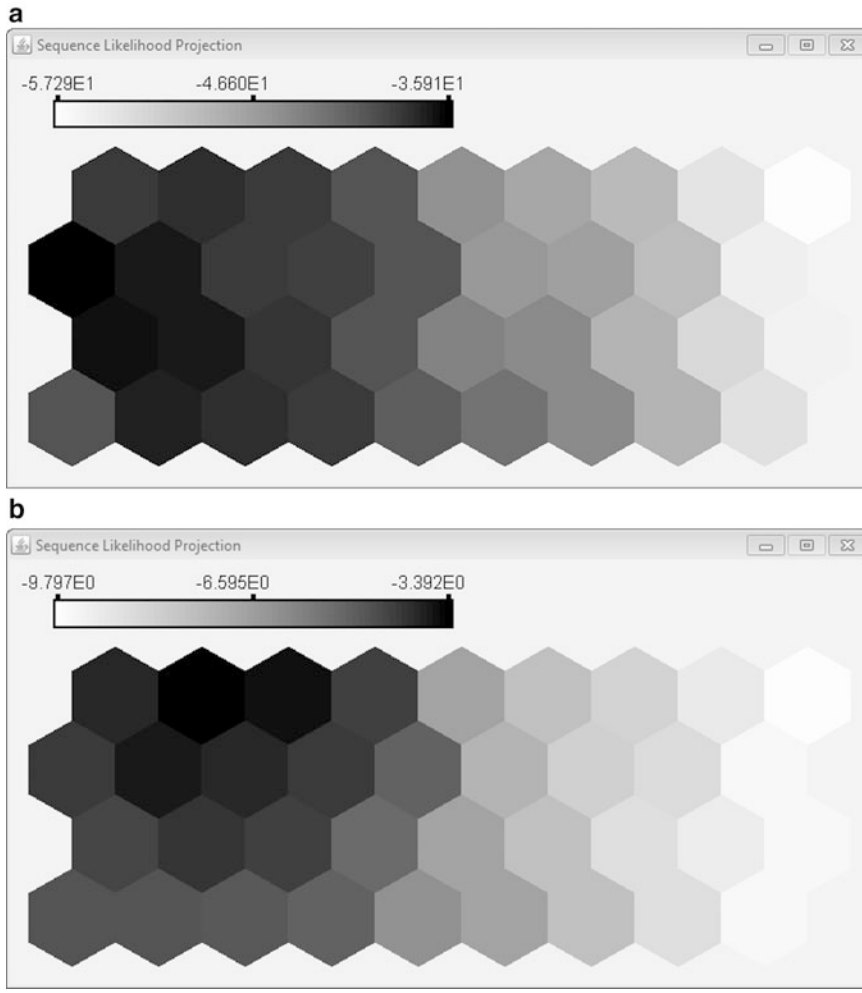


Fig. 10 Paradigms of SLP visualizations for acceptor splice site sequences

sequences with intron-exon boundaries are assigned to the acceptor cluster. Alternatively, they can be treated as an additional verification/proof of the discovered clusters since each one of the 1535 sequences produces high likelihoods in exactly one out of the two distinct areas on the SOHMMM mapping; these exact two regions coincide with the recognized clusters.

Subsequently, under the assumption that class information is given, a posterior labeling of the traced clusters can be performed on the SDDD (Fig. 11). In such a case, an unknown or orphan splice junction is classified as belonging to the category designated by the SOHMMM neuron yielding the highest likelihood, and (as shown previously) is clustered accordingly. It is interesting that the SLP's visualization goes beyond the mapping of the highest likelihood SOHMMM node by depicting the overall likelihood landscape of the analyzed sequence, thus resulting in a more rich and comprehensive description.

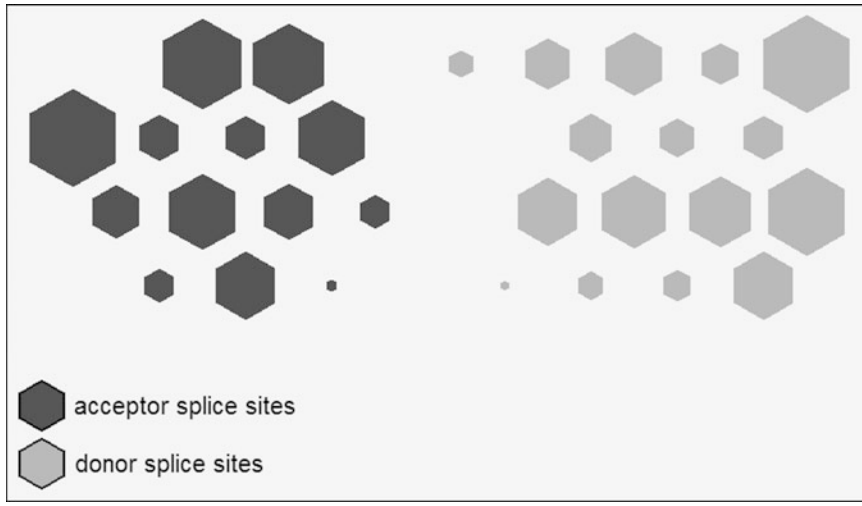


Fig. 11 SDDD visualization enhanced by posterior labeling of depicted clusters

An additional use of the SLP visualization can be revealed by employing sequences that contain neither an acceptor nor a donor splice site (Fig. 12). In this case the resulting SLPs, which employ a SOHMMM array trained solely on exxon-intron and intron-exon boundaries, consist of non-coherent regions and vague boundaries of heterogeneous neurons. Such or similar SLP graphic displays indicate sequences which do not belong to the clusters detected/identified by the SDDD. Consequently, by combining the functionality and findings of the SDDD and SLP visualizations (Figs. 8 and 12) one is in position to detect outlier sequences that belong to a problem space different from the one being which is being modeled.

5 Conclusion

The present study has developed mapping techniques and graphic displays to demonstrate and visualize sequence data clustering results. Epigrammatically, the SDDD estimates and subsequently devises a sequence data density graphic representation, whereas the SLP produces a sequence likelihood landscape. The SDDD and SLP projections represent the input sequence data in a lower-dimensional space in such a way that the clusters and statistical relations of data elements are depicted graphically. It has been demonstrated that the SDDD and the SLP in synergy with the SOHMMM integrate cluster visualization and nonlinear mapping (on a low-dimensional lattice) in a unified functional framework; thus making the produced results visually accessible, verifiable and interpretable. The proposed techniques' experimental testing and verification has been performed both on amino acid and nucleotide sequences.

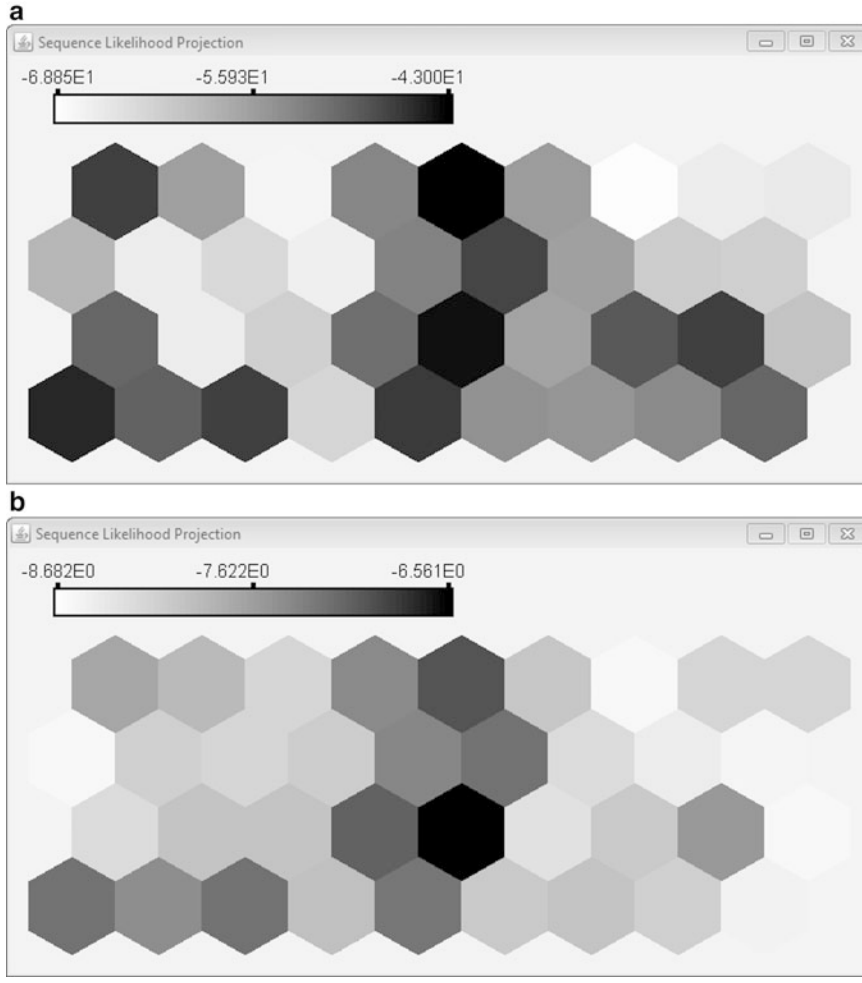


Fig. 12 Examples of SLPs depicting sequences not known to contain a splicing site

Acknowledgment

The authors would like to thank Anastasis Tzimas for his insightful remarks and comments.

References

1. Xu R, Wunsch D (2005) Survey of clustering algorithms. IEEE Trans Neural Netw 16:645–678
2. Du K-L (2010) Clustering: a neural network approach. Neural Netw 23:89–107
3. Kohonen T (2001) Self-organizing maps, 3rd edn. Springer, Berlin
4. Tasdemir K (2010) Graph based representations of density distribution and distances for self-organizing maps. IEEE Trans Neural Netw 21:520–526
5. Tasdemir K, Merenyi E (2009) Exploiting data topology in visualization and clustering of self-organizing maps. IEEE Trans Neural Netw 20:549–562

6. Brugger D, Bogdan M, Rosenstiel W (2008) Automatic cluster detection in Kohonen's SOM. IEEE Trans Neural Netw 19:442–459
7. Ultsch A (2003) Maps for the visualization of high-dimensional data spaces. In: Proc. workshop self-organizing maps, pp 225–230
8. Yin H (2002) ViSOM—a novel method for multivariate data projection and structure visualization. IEEE Trans Neural Netw 13:237–243
9. Kraaijveld MA, Mao J, Jain AK (1995) A nonlinear projection method based on Kohonen's topology preserving maps. IEEE Trans Neural Netw 6:548–559
10. Ferles C, Stafylopatis A (2013) Self-Organizing Hidden Markov Model Map (SOHMMM). Neural Netw 48:133–147
11. Ferles C, Siolas G, Stafylopatis A (2013) Scaled self-organizing map—hidden Markov model architecture for biological sequence clustering. Appl Artif Intell 27:461–495
12. Ferles C, Siolas G, Stafylopatis A (2011) Scaled on-line unsupervised learning algorithm for a SOM-HMM hybrid. In: 26th Int. symposium computer information sciences, pp 533–537
13. Ferles C, Stafylopatis A (2008) A hybrid self-organizing model for sequence analysis. In: 20th IEEE int. conf. tools artificial intell., pp 105–112
14. Ferles C, Stafylopatis A (2008) Sequence clustering with the self-organizing hidden Markov model map. In: 8th IEEE int. conf. bioinformatics bioengineering, pp 1–7
15. Barreto G de A, Araujo A, Kremer S (2003) A taxonomy of spatiotemporal connectionist networks revisited: the unsupervised case. Neural Comput 15:1255–1320
16. Hammer B, Micheli A, Strickert M et al (2004) A general framework for unsupervised processing of structured data. Neurocomputing 57:3–35
17. Hammer B, Hasenfuss A (2010) Topographic mapping of large dissimilarity data sets. Neural Comput 22:2229–2284
18. Lebbah M, Rogovschi N, Bennani Y (2007) BeSOM: Bernoulli on self-organizing map. In: Int. joint conf. neural netw., pp 631–636
19. Somervuo P (2004) Online algorithm for the self-organizing map of symbol strings. Neural Netw 17:1231–1239
20. Strickert M, Hammer B (2004) Self-organizing context learning. In: Proc. European symposium artificial neural netw., pp 39–44
21. Koski T (2001) Hidden Markov models for bioinformatics. Kluwer Academics Publishers, Dordrecht, The Netherlands
22. Rabiner L (1989) A tutorial on hidden Markov models and selected applications in speech recognition. Proc IEEE 77:257–286
23. The iProClass Protein Knowledgebase (release 4.32) [online]. Available: http://pir.georgetown.edu/
24. Sharma K (2008) Bioinformatics: sequence alignment and Markov models. McGraw-Hill, New York
25. Durbin R, Eddy SR, Krogh A et al (1998) Biological sequence analysis: probabilistic models of proteins and nucleic acids. Cambridge University Press, Cambridge
26. Krogh A, Brown M, Mian IS et al (1994) Hidden Markov models in computational biology: applications to protein modeling. J Mol Biol 235:1501–1531
27. Mount DW (2004) Bioinformatics: sequence and genome analysis, 2nd edn. Cold Spring Harbor Laboratory Press, New York
28. Baldi P, Brunak S (2001) Bioinformatics: the machine learning approach, 2nd edn. The MIT Press, Cambridge, Massachusetts
29. UCI Machine Learning Repository [online]. Available: http://archive.ics.uci.edu/ml/

Chapter 7

Analyzing Single Molecule FRET Trajectories Using HMM

Kenji Okamoto

Abstract

Structural dynamics of biomolecules, such as proteins, plays essential roles in many biological phenomena at molecular level. It is crucial to understand such dynamics in recent biology. The single-molecule Förster resonance energy transfer (smFRET) measurement is one of few methods that enable us to observe structural changes of biomolecules in realtime. Time series data of smFRET, however, typically contains significant fluctuation, making analysis difficult. On the other hand, one can often assume a Markov process behind such data so that the hidden Markov model (HMM) can be used to reproduce a state transition trajectory (STT). The common solution of the HMM can be used for smFRET data, too, while one has to define the specific model, i.e., the observable variable and the emission probability. There are several choices of the model for smFRET depending on the measurement method, the detector type, and so on. I introduce some of applicable models for smFRET time series data analysis.

Keywords Single-molecule measurement, FRET, Fluorescence microscopy, Time series data analysis, Molecular structural dynamics, State transition trajectory

1 Introduction

Experimental biology in recent decades has unveiled the nature of biomolecules. While the molecular structure and function are closely connected, mechanical motion often plays crucial roles in functional expression as proteins are often likened to "molecular machines." The understanding of structural dynamics has been increasing its importance. However, it is not easy to directly observe such dynamics since, for example, structural change is very small (~nanometers), dynamics is asynchronous among molecules and the molecule must be placed under physiological (or mimic) condition. One of few methods to realize it is the single-molecule Förster resonance energy transfer (smFRET) measurement that gives us information of molecular structural changes through the distance between two fluorescent dyes from individual molecules in realtime. However, the time series data of smFRET typically contains significant fluctuation since the signal from only a single molecule is typically too weak. On the other hand, it often

David R. Westhead and M.S. Vijayabaskar (eds.), *Hidden Markov Models: Methods and Protocols*, Methods in Molecular Biology, vol. 1552, DOI 10.1007/978-1-4939-6753-7_7, © Springer Science+Business Media LLC 2017

shows multiple signal levels and stepwise changes connecting them, which can be interpreted as the multiple molecular states and transitions between them. Therefore the data analysis method that can extract a state transition trajectory (STT) from noisy data is required and the hidden Markov model (HMM) is perfectly suitable for that purpose. The smFRET HMM can be solved by the standard procedure for HMM while the user has to define the specific model. In this chapter, after brief introduction of the principle of FRET and the sample and the apparatus for smFRET measurement, I will introduce some models that have been successfully applied to smFRET time series data.

2 Förster Resonance Energy Transfer—FRET

The FRET has been extensively used in biological experiments to detect a distance from fluorescence signals and is hence sometimes referred to as a "spectroscopic ruler." When two different kinds of dye molecules are in the close vicinity and one of them (the donor) is excited by light, the energy is transferred to another (the acceptor) dye without radiation and the acceptor emits fluorescence photon with a certain probability (Fig. 1). The probability of the energy transfer, the so-called FRET efficiency E_{FRET}, depends on the distance r between dyes as [1, 2]

$$E_{\mathrm{FRET}} = \frac{1}{1 + (r/R_0)^6}, \tag{1}$$

where R_0 is the Förster distance, at which E_{FRET} becomes 0.5, and typically 5–10 nm while it depends on various conditions including dye species. E_{FRET} can be written as the ratio of fluorescence intensity from the donor I_{D} and the acceptor I_{A} as

$$E_{\mathrm{FRET}} = \frac{I_{\mathrm{A}} - \beta I_{\mathrm{D}}}{I_{\mathrm{A}} + (\gamma - \beta) I_{\mathrm{D}}}, \tag{2}$$

where β and γ are coefficients for compensation of fluorescence leak from the donor dye to the acceptor detector channel and difference

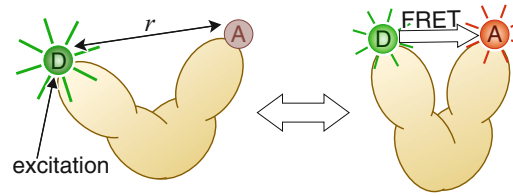


Fig. 1 When the donor dye (D) is excited by light absorption in the proximity of the acceptor dye (A), the excited energy is transferred to the acceptor with a certain probability, the FRET efficiency E_{FRET}

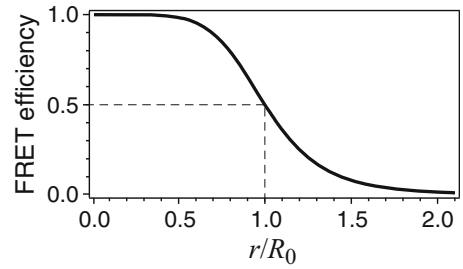


Fig. 2 Dependence of E_{FRET} on r. Horizontal axis is normalized by R_0. Dependency curve is steepest and then E_{FRET} is most sensitive to r change near $r = R_0$

in detection efficiencies of dyes, respectively. From comparison of Eqs. 1 and 2, the molecular distance can be estimated by fluorescence measurement.

As can been seen in Fig. 2, E_{FRET} drastically changes and hence is sensitive to distance change near R_0. That distance range is much shorter than spatial resolution of conventional optical microscopy (~hundreds of nanometers) and corresponds to the size of biomolecules or their motion. FRET measurement can also receive benefits of fluorescence measurements, such as single-molecule measurability or realtimeness. Especially, smFRET is advantageous in molecular dynamics measurement since dynamics proceeds independently on individual molecules and details of dynamics are missed by ensemble average in bulk measurement. The smFRET measurement is almost only way to observe asynchronous biomolecular dynamics directly in realtime.

3 Materials

3.1 Samples

The target molecule must be labeled with two fluorescent dyes of different colors. Either organic dyes or fluorescent proteins, such as a green fluorescent protein (GFP), can be used. In order to yield the FRET effectively, there has to be sufficient overlap between spectra of the donor emission and the acceptor absorption. On the other hand, difference in absorption spectra of dyes should be sufficiently large so that the direct excitation of the acceptor by donor excitation light is avoided. In single-molecule experiments, brightness and photostability of dyes are also important.

3.2 Instruments

smFRET experiments are usually executed on optical microscopes with high spatial resolution and high sensitivity. In order to collect limited number of photons from only a single molecule, the microscope also has to be equipped with highly sensitive detectors. There are two types of microscope configurations typically used in smFRET experiments as briefly introduced below.

3.2.1 Confocal Microscopy

A confocal microscope consists of a focused laser beam for fluorescence excitation and a pinhole placed before a detector in order to achieve high spatial resolution and high signal to noise (S/N) ratio by collecting light only from the focal point. For smFRET time series measurement, one first has to take an image by scanning the laser beam or the sample stage and then fix the focus on a single molecule, which must be immobilized, typically on the surface of a glass substrate. Fluorescence light is separated by a dichroic filter, which typically transmits long-wavelength light and reflects short-wavelength light, so that fluorescence from two dyes is introduced to two separate detectors.

Confocal microscope is typically equipped with the single photon counting (SPC) detectors for smFRET measurement. As commercially available SPC detectors, photomultiplier tubes (PMT) were widely used while avalanche photodiodes (APD) have been replacing them recently. An SPC detector outputs an electric pulse when a photon is detected. Since photon emission, photon detection, and thermal noise generation are all stochastic processes, resulting pulse generation is a Poisson process. The SPC signal preserves the statistical characteristics of that stochasticity.

3.2.2 TIRF Microscopy

For imaging single molecules, it is more convenient to use highly sensitive cameras. For that purpose, total internal reflection fluorescence (TIRF) microscopy is often used in order to illuminate the wide field. When the excitation light is introduced onto the glass–-water interface from the glass side with incident angle larger than the critical angle, the light is totally reflected while the evanescent field is generated in water near the surface. Since the penetration depth of the evanescent field is very thin (typically hundreds of nanometers), fluorescence emission from molecules in bulk solution, which causes the background fluorescence signal, can be sufficiently suppressed. Fluorescence light emitted by molecules on the surface is separated by a dichroic filter and introduced to two cameras or two halves of a camera. Imaging measurement enables us to observe multiple molecules simultaneously and track slowly moving molecules as well as immobile ones. In order to obtain fluorescence time series, one has to extract fluorescence intensity of a single molecule from each video frame. For examples, one can first define a region-of-interest (ROI) surrounding the single-molecule spot and, pick a maximum signal in the ROI, integrate signals within the ROI (and subtract background) or fit a two-dimensional Gaussian function. Various algorithms have been proposed to extract intensities as well as track particles and provided as software packages [3].

Commercially available cameras, which are typically used in smFRET experiments, include electron multiplying charge coupled device (EMCCD) and scientific complementary metal oxide semiconductor (sCMOS). Both types integrate the photon signal within

each pixel in each frame and electrically amplify it at readout. Both photon signals and background noise are typically approximated by a Gaussian distribution.

4 Methods

4.1 Common HMM

smFRET time series data can be analyzed by a standard procedure of the HMM. When experimental data is obtained as an N-point time series $\mathbf{X} = \{\mathbf{x}_1, \ldots, \mathbf{x}_n, \ldots, \mathbf{x}_N\}$, the corresponding latent variable $\mathbf{Z} = \{\mathbf{z}_1, \ldots, \mathbf{z}_n, \ldots, \mathbf{z}_N\}$, $\mathbf{z}_n = \{z_{n1}, \ldots, z_{nk}, \ldots, z_{nK}\}$ is defined so that it is equal to 1 when the molecule is in the i-th state at time n and 0 otherwise, where K is the number of states (NoS). The joint probability distribution over $\mathbf{X}$ and $\mathbf{Z}$ can be written as

$$p(\mathbf{X}, \mathbf{Z}|\Theta) = p(\mathbf{z}_1|\boldsymbol{\pi}) \times \prod_{n=2}^{N} p(\mathbf{z}_n|\mathbf{z}_{n-1}, \mathbf{A}) \times \prod_{n=1}^{N} \prod_{i=1}^{K} p(\mathbf{x}_n|\theta_i)^{z_{ni}} \quad (3)$$

The first term $p(\mathbf{z}_1|\boldsymbol{\pi}) = \prod_{i=1}^{K} \pi_k^{z_{1i}}$ represents the probability distribution for the initial state with a parameter $\boldsymbol{\pi} = \{\pi_i\}$. The second term $p(\mathbf{z}_n|\mathbf{z}_n\text{-}1, \mathbf{A}) = \prod_{i=1}^{K} \prod_{j=1}^{K} A_{ij}^{z_n\text{-}1, iz_{nj}}$ represents the probability distribution for state transitions with a $K \times K$ matrix $\mathbf{A}$, the element A_{ij} of which is the transition probability form the i-th to the j-th state. The third term $p(\mathbf{x}_n|\theta_i)$ is the emission probability, which relates the molecular state and the observed data, and depends on the model, with a parameter set θ_i that represents the emission probability of the i-th state. $\Theta = \{\boldsymbol{\pi}, \mathbf{A}, \boldsymbol{\theta}\}$ represents all parameters. Equation 3 is the general expression of the HMM and can be solved by a standard procedure based on, for example, the maximum likelihood estimation (MLE) using the expectation-maximization (E-M) algorithm with the Baum–Welch and the Viterbi algorithms.

Since the first and the second terms of Eq. 3 are the common forms, one has to define the model-dependent emission probability.

4.2 smFRET HMM

The experimental observables of smFRET time series measurement are fluorescence intensities I_A and I_D as shown in Fig. 3a, from which a time series of E_{FRET} (gray line in Fig. 3b) can be calculated by Eq. 2.

The purpose of smFRET HMM is to obtain a STT since it is associated with the molecular conformation (black line in Fig. 3b). There are some choices of observable variables and the form of the emission probability for the smFRET HMM, depending on the measurement method, the detector type, and so on. The user can and has to choose one, for example, from ones described in the following sections.

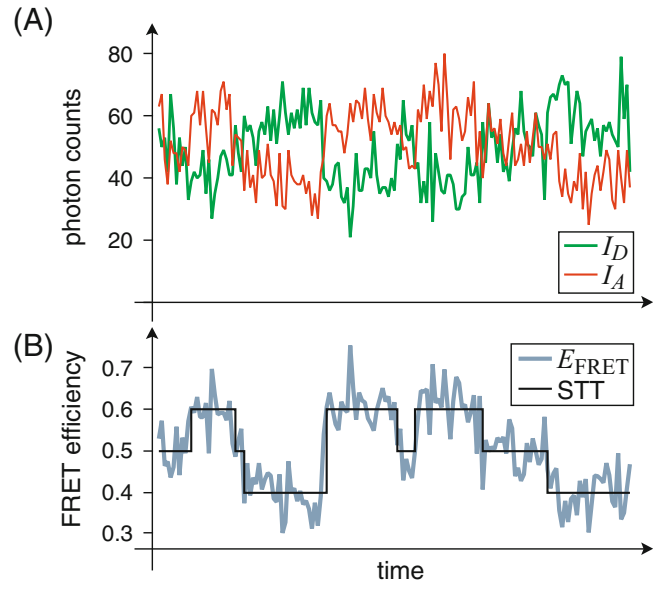


Fig. 3 Example of smFRET time series. (**a**) Two fluorescence intensities of the donor and the acceptor are the direct experimental observables. (**b**) E_{FRET} time series (*gray*) can be calculated from intensities, behind which the state transition dynamics is hidden. *Black line* represents the state transition trajectory (STT) to be reproduced by the HMM analysis, shown with the mean E_{FRET} for each state

4.2.1 Gaussian FRET Time Series [4]

The observable: $x_n = E_n$

E_n: the FRET efficiency at time n.

Parameters:

μ_i: the mean of Gaussian distribution for i-th state.

σ_i^2: the variance of Gaussian distribution for i-th state.

The emission probability:

$$p(x_n|\mu_i, \sigma_i^2) = \frac{1}{\sqrt{2\pi\sigma_i^2}} \exp\left(-\frac{(E_n - \mu_i)}{2\sigma_i^2}\right) \tag{4}$$

The simplest and most common representation of the HMM model for smFRET time series is a Gaussian distribution for a scalar observable $x_n = E_n$. This model can be generally used for any smFRET experiments and must be used when the intensity signals are also normally distributed, for example, in imaging experiments using cameras.

4.2.2 Beta FRET Time Series [5]

The observable: $x_n = E_n$

E_n: the FRET efficiency at time n.

Parameters:

α_i: the mean photon counts of the acceptor ($= \langle I_A \rangle$) for the i-th state with $\langle x \rangle$ as the time average of x.

β_i: the mean photon counts of the donor ($= \langle I_D \rangle$) for the i-th state.

The emission probability:

$$p(x_n|\alpha_i, \beta_i) = \frac{E_n^{\alpha_i - 1}(1 - E_n)^{\beta_i - 1}}{B(\alpha_i, \beta_i)} \tag{5}$$

where the normalization factor

$$B(\alpha_i, \beta_i) = \frac{\Gamma(\alpha_i)\Gamma(\beta_i)}{\Gamma(\alpha_i + \beta_i)} \tag{6}$$

is a Beta function with a Gamma function $\Gamma(x)$.

This model is valid for SPC measurements, in which the intensity signals are obtained as photon counts within a time bin and those probability distributions are Poisson distributions, and the means are large enough, e.g., $\langle I_A \rangle, \langle I_D \rangle \geq 5$.

4.2.3 Poisson Intensity Time Series [5, 6]

The observable: $\mathbf{x}_n = \{a_n, d_n\}$

a_n: the photon count (integer) on the acceptor detection channel at time n.

d_n: the photon count (integer) on the donor detection channel at time n.

Parameters:

μ_i: the mean photon counts ($= \langle a_n \rangle + \langle d_n \rangle$) for the i-th state.

E_i: the FRET efficiency for the i-th state.

The emission probability:

$$p(\mathbf{x}_n|\mu_i, E_i) = \frac{E_i^{a_n}(1 - E_i)^{d_n}}{a_n! d_n!} \mu_i^{a_n + d_n} \exp(-\mu_i) \tag{7}$$

This model is valid for SPC measurements, in which the intensity signals are obtained as photon counts represented by Poisson distributions regardless of the mean intensities.

4.2.4 Gaussian Intensity Time Series [5]

The observable: $\mathbf{x}_n = \{I_{A,n}, I_{D,n}\}$

$I_{A,n}$: the fluorescence intensity detected on the acceptor channel at time n.

$I_{D,n}$: the fluorescence intensity detected on the donor channel at time n.

Parameters:

$\boldsymbol{\mu}_i = (\langle I_A \rangle, \langle I_D \rangle)$: the mean intensities of the acceptor and the donor for the i-th state.

$\boldsymbol{\Sigma}_i$: the covariance matrix for the i-th state.

The emission probability:

$$p(\mathbf{x}_n \Big| \boldsymbol{\mu}_i, \boldsymbol{\Sigma}_i) = \frac{1}{2\pi|\boldsymbol{\Sigma}_i|^{1/2}} \exp\left(-\frac{1}{2}(\mathbf{x}_n - \boldsymbol{\mu}_i)^{\mathrm{T}} \cdot \boldsymbol{\Sigma}_i^{-1} \cdot (\mathbf{x}_n - \boldsymbol{\mu}_i)\right) \quad (8)$$

If the intensity signals can be represented by Gaussian distributions, this model can be used. Intensity signals acquired from camera images are commonly approximated by a Gaussian distribution. It is also valid when the average intensity is sufficiently strong in SPC measurement, where a Poisson distribution can be well approximated by a Gaussian distribution. In this model, E_{FRET} is not included and cannot be obtained directly from parameters but indirectly as the intensity ratio via $\boldsymbol{\mu}_i$.

4.3 Determination of the Number of States

In order to solve the HMM, we have to assume the NoS at first. However, in real experiments, especially in measurements of biomolecular dynamics, we basically have no information on the NoS. We often have to infer the likeliest NoS from data itself. One may compare the results of HMM analyses executed with different NoS. However, we cannot directly compare the results of the MLE, which are given as the optimized likelihood $p(\mathbf{X}|\Theta)$, since larger NoS generally gives better likelihood due to overfitting.

A solution to avoid this problem is to introduce a penalty for larger NoS. For that purpose, Akaike's information criterion (AIC) or Bayesian information criterion (BIC) have been successfully used as

$$\mathrm{AIC} = -2\log L + 2k \quad (9)$$

$$\mathrm{BIC} = -2\log L + k\log n \quad (10)$$

where L is the optimized likelihood, k is the number of free parameters and n is the number of data points. We can determine the NoS as the one that gives the largest AIC/BIC value [4, 5].

Another solution is the variational Bayes (VB) method that compares the model evidence, which is the marginal likelihood written as

$$p(\mathbf{X}) = \int p(\mathbf{X}|\Theta)p(\Theta)d\Theta, \quad (11)$$

where $p(\Theta)$ is the prior distribution for parameters, instead of the likelihood itself [7]. While the MLE finds the optimum values for parameters, the VB optimizes the probability distributions of parameters using the prior distributions, which the user has to give. We can directly compare the evidence between different models (NoS). In order to solve the VB-HMM, one can use the similar procedure

to the E-M algorithm for MLE. In the E-step, the Baum–Welch and the Viterbi algorithms optimize $\mathbf{Z}$ while parameters are updated in the M-step. For details of procedures, refer to, for example, Refs. 7, 8.5. Special Treatments

5 Special Treatments

5.1 Global Analysis

Elementary processes of molecular dynamics are basically stochastic phenomena. Therefore, multiple observations are important to understand those natures, especially in single-molecule experiments. For example, in order to decide a transition rate, one has to observe sufficient number of the transition events and statistically evaluate them. However, it is often difficult to acquire very long time series in smFRET experiments because of, for example, photobleach of fluorescent dyes. A single data may not contain sufficient information to portray the whole molecular dynamics. If we could assume that the same dynamics lies behind different time traces, we may be able to acquire necessary information by global analysis over a set of time series data.

The HMM solution for multiple observations was introduced by Li et al. [9] and applied to smFRET data by Liu et al. [5]. Briefly, while the E-M algorithm for a single observation evaluates

$$Q\left(\Theta,\Theta^{\text{old}}\right)=\sum_{\mathbf{Z}}p(\mathbf{Z}|\mathbf{X},\Theta^{\text{old}})\ln p(\mathbf{X},\mathbf{Z}|\Theta), \tag{12}$$

where Θ^{old} is the parameter set in the previous step of iterative calculation, to obtain updated Θ, multiple observations can be evaluated by a weighted sum

$$Q\left(\Theta,\Theta^{\text{old}}\right)=\sum_{m=1}^{M}w_m\sum_{\mathbf{Z}^{(m)}}p\left(\mathbf{Z}^{(m)}|\mathbf{X}^{(m)},\Theta^{\text{old}}\right)\ln p\left(\mathbf{X}^{(m)},\mathbf{Z}^{(m)}|\Theta\right), \tag{13}$$

where $\mathbf{X}^{(\mathrm{m})}$ and $\mathbf{Z}^{(\mathrm{m})}$ are the observable and the latent variables for m-th out of M observations and w_m is the weight. The E-M algorithm finds the optimum set of parameter values common to all M observations.

Such global optimization of parameters, however, does not always work well in single-molecule experiments. The parameters themselves, such as the FRET efficiency or the fluorescence intensity, may slightly vary among molecules because of, for example, differences in local environment or optical configuration. In order to allow such variation, the empirical Bayes (EB) method has been recently proposed [10]. The EB treats parameters as probability distributions and is solved in the VB framework. While the

common prior distribution $p(\Theta|\psi_0)$ is defined with the hyperparameters ψ_0, the posterior distribution $p(\Theta_m|\psi_m)$ is optimized for each time trace with the posterior parameters ψ_m.

5.2 Time Series with Nonuniform Time Intervals

From the physical or chemical point of view, molecular reactions are characterized by rates. In common HMMs for time series data, we can usually assume that the time interval between data points is uniform and then use the state transition probabilities, which are calculated from the rates, given by a matrix $\mathbf{A}$ as in Eq. 3. However, if one needs to analyze a time series with nonuniform time intervals, this substitution does not hold any longer. For example, the SPC detector can be run in the so-called "time stamp" detection mode, in which the detection time of every single photon is recorded. As mentioned in Subheading 3.2.1, emission of fluorescence photons is Poisson process and the time intervals are exponentially distributed. Okamoto et al. derived the transition probability distribution for time-stamp time series [6]. Briefly, the transition probability distribution $p(z_{nj}|z_{n\text{-}1,i}, \boldsymbol{\kappa}, \mathbf{I})$ can be written with rates as

$$p(z_{nj}|z_{n-1,i}, \boldsymbol{\kappa}, \mathbf{I}) = \begin{cases} \dfrac{I_i}{\kappa_{ii} + I_i} & (i = j) \\ \dfrac{\kappa_{ii}\kappa_{ij}}{\kappa_{ii} + I_i} & (i \neq j) \end{cases}, \tag{14}$$

where I_i is the fluorescence intensity of the i-th state as the average number of photons per unit time. The transition rate matrix $\boldsymbol{\kappa}$ is introduced instead of $\mathbf{A}$ so that $\kappa_{ii} = \sum_{j\neq i}^{K} k_{ij}$ represents the rate that the molecule exits from the i-th state and $\kappa_{ij} = k_{ij}/\sum_{j\neq i}^{K} k_{ij} = k_{ij}/\kappa_{ii}$ $(i \neq j)$ represents the probability that the state changes from i to j once the transition occurs, where k_{ij} is the transition rate from the state i to j $(i \neq j)$. For time-stamped smFRET experiments, the observable must be a vector to represent the photon color in addition to the detection time so that $\mathbf{x}_n = \{\Delta t_n, \rho_n\}$, where Δt_n is the time interval from the $(n-1)$-th to the n-th photons and the ρ_n is 0 (1) if the photon is detected on the donor (acceptor) channel. Then the emission probability is rewritten as

$$p(\mathbf{x}_n|I_i, E_i) = E_i^{\rho_n}(1 - E_i)^{1-\rho_n} \times I_i \exp(-I_i \Delta t_n), \tag{15}$$

where I_i and E_i are the fluorescence intensity and the FRET efficiency of the i-th state. The VB solution of time-stamped smFRET HMM is described in Ref. 6.

References

1. Förster T (1946) Energiewanderung und Fluoreszenz. Naturwissenschaften 33:166–175
2. Lakowicz JR (2006) Principles of fluorescence spectroscopy, 3rd edn. Springer, New York
3. Chenouard N, Smal I, de Chaumont F et al (2014) Objective comparison of particle tracking methods. Nat Methods 11:281–290
4. McKinney SA, Joo C, Ha T (2006) Analysis of single-molecule FRET trajectories using hidden Markov modeling. Biophys J 91:1941–1951
5. Liu Y, Park J, Dahmen KA et al (2010) A comparative study of multivariate and univariate hidden Markov modelings in time-binned single-molecule FRET data analysis. J Phys Chem B 114:5386–5403
6. Okamoto K, Sako Y (2012) Variational Bayes analysis of a photon-based hidden Markov model for single-molecule FRET trajectories. Biophys J 103:1315–1324
7. Bronson JE, Fei J, Hofman JM et al (2009) Learning rates and states from biophysical time series: a Bayesian approach to model selection and single-molecule FRET data. Biophys J 97:3196–3205
8. Bishop CM (2006) Pattern recognition and machine learning. Springer, New York
9. Li X, Parizeau M, Plamondon R (2000) Training hidden Markov models with multiple observations—a combinatorial method. IEEE Trans Pattern Anal Mach Intell 22:371–377
10. van de Meent J-W, Bronson JE, Wiggins CH et al (2014) Empirical Bayes methods enable advanced population-level analyses of single-molecule FRET experiments. Biophys J 106:1327–1337

Chapter 8

Modelling ChIP-seq Data Using HMMs

Veronica Vinciotti

Abstract

Chromatin ImmunoPrecipitation-sequencing (ChIP-seq) experiments have now become routine in biology for the detection of protein binding sites. In this chapter, we show how hidden Markov models can be used for the analysis of data generated by ChIP-seq experiments. We show how a hidden Markov model can naturally account for spatial dependencies in the ChIP-seq data, how it can be used in the presence of data from multiple ChIP-seq experiments under the same biological condition, and how it naturally accounts for the different IP efficiencies of individual ChIP-seq experiments.

Key words Hidden Markov models, ChIP-sequencing, Histone Modifications

1 Introduction

Hidden Markov models (HMMs) are popular tools in the statistical and machine learning community. There are a number of biological applications where HMMs have been adopted, such as for the identification of transcription factor binding sites, histone modification sites, and *cis*-regulatory modules [1–5].

This chapter focusses on the application of HMMs for the identification of protein binding sites and, in particular, it considers the case of chromatin modifiers. These tend to have broad regions of enrichment and do not follow the peak-like pattern which is instead typical of conventional DNA-binding transcription factors, for which more specific peak detection methods have been developed [6]. The parameters in the HMM are estimated from data generated by ChIP-seq experiments. In the next section, we describe the data with the use of an illustrative example and emphasize some key features of this type of data. In Subheading 3, we describe a HMM that can capture these features and show how this model can be used for deciding whether a region of the genome is bound by a particular protein.

David R. Westhead and M.S. Vijayabaskar (eds.), *Hidden Markov Models: Methods and Protocols*, Methods in Molecular Biology, vol. 1552, DOI 10.1007/978-1-4939-6753-7_8, © Springer Science+Business Media LLC 2017

2 ChIP-seq Data

ChIP-sequencing, also known as ChIP-seq, is a well-known biological technique to detect protein–DNA interactions, DNA methylation, and histone modifications in vivo [7]. ChIP-seq combines Chromatin ImmunoPrecipitation (ChIP) with massively parallel DNA sequencing to identify all DNA binding sites of a transcription factor or genomic regions with certain histone modification marks. The ChIP process captures cross linked and sheared DNA–protein complexes using an antibody against a protein of interest. After decrosslinking of the protein–DNA complexes, the final DNA pool is enriched in DNA fragments bound by the protein of interest, but there are always random genomic DNA fragments piggybacking on the specific DNA fragments. The degree of enrichment depends on the ChIP efficiency. A more efficient experiment will induce a higher proportion of protein-bound fragments in the mixture pool, and generate more sequence reads in bound regions and less sequence reads in non-bound regions, than an experiment with lower ChIP efficiency [8].

The final data produced by the experiment report the number of DNA fragments in the sample aligned to each location of the genome. Due to the size of the data and the level of noise, it is common to summarize counts at the level of bins, with often an ad-hoc choice for the fixed-width window for the bins. Figure 1 shows an example of data generated by ChIP-seq experiments on the CREB-binding protein (CBP) [8, 9]. The CBP binding is profiled in human T98G cells at time point 0, where cells are serum starved and where the protein is restricted to a limit set of genes, and at 30 min after stimulation with tetradecanoyl phorbol acetate. For the latter condition, there are two technical replicates. In Figure 1, the x-axis represents consecutive bins of size 200 bp for a region of chromosome 21, whereas the y-axis reports the overall number of counts for each 200 bp window. The figures show regions with a high concentration of counts and regions with a small concentration of counts. The figures also show a different ChIP efficiency for the three experiments, which is evident also amongst the two technical replicates (bottom plots). Finally, the figures show the presence of neighboring regions with high concentration of counts, suggesting a possible spatial dependency in the data. This is probably a reflection of the fact that enriched regions are likely to cover a number of 200 bp bins.

The aim of the statistical analysis is to distinguish the truly enriched regions along the genome from the background noise, while taking into account the features of the data described above, namely the spatial dependencies, the presence of replicates, and the different degrees of efficiencies of the different experiments.

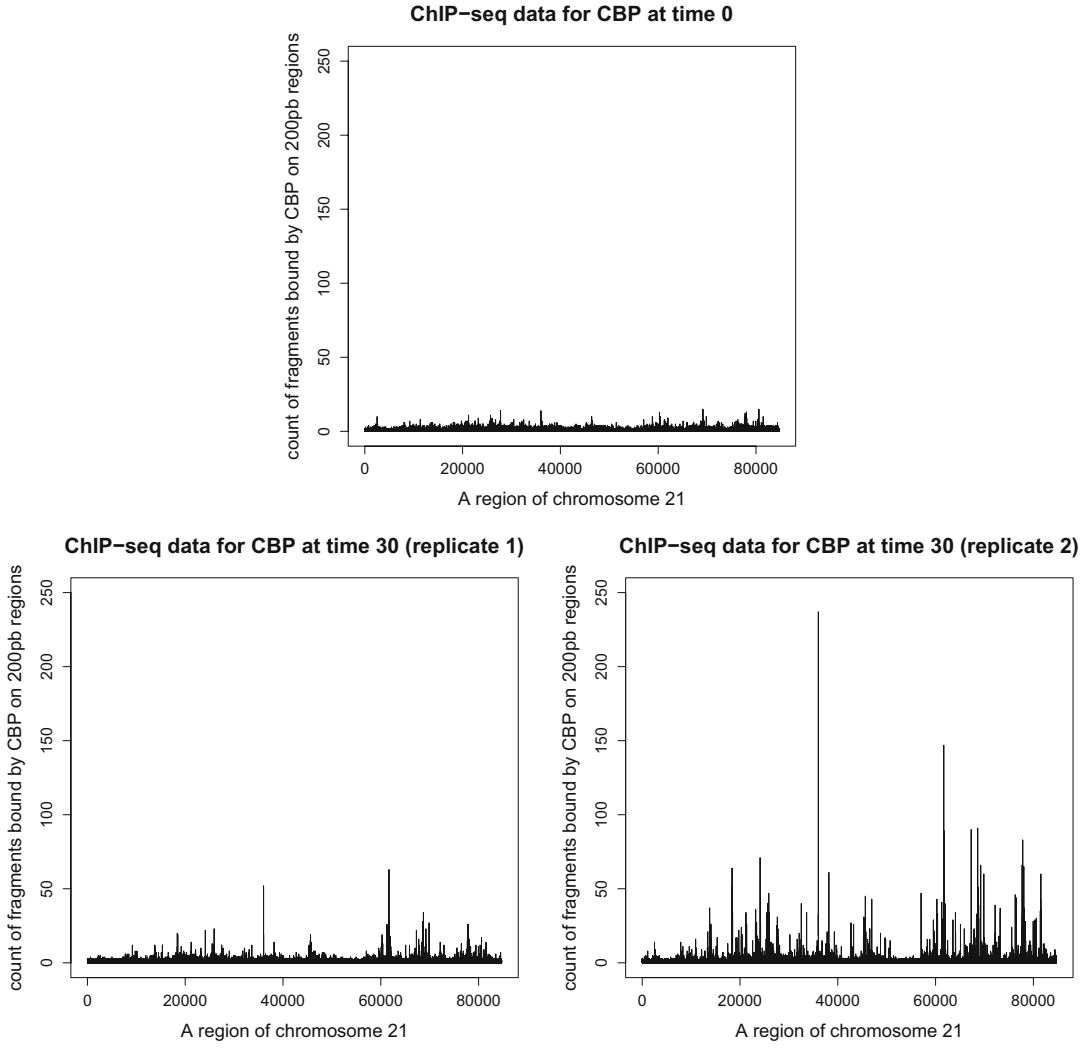


Fig. 1 Plots of bin counts for bins of size 200 bp on a region of chromosome 21 for three ChIP-seq experiments performed on the CBP chromatin modifier at two different time points

3 Methods

3.1 The Hidden Markov Model

The underlying truth says that some regions of the genome are enriched by the particular protein in question and some are not. If we can characterize the distribution of the counts for the enriched and non-enriched regions, respectively, then we can associate to each region a probability of being enriched or not given the available data and make a decision according to that. This situation is particularly suited to mixture models. The additional assumption that the probability of a region being enriched is dependent on the state of the neighboring regions, due to spatial dependencies in the data, further points to the use of a hidden Markov model.

In order to go through the technical specification of the model, let us first define some useful notation. Let M be the total number of bins and Y_{mcr} the counts in the mth bin, $m = 1, 2, \ldots, M$, under condition c and replicate r. In our case, the condition c stands for a particular protein and/or a particular time point, and $r = 1, \ldots, R_c$ is the number of replicates under condition c. The counts Y_{mcr} are either drawn from a background population (non-enriched region) or from a signal population (enriched region). Let X_{mc} be the unobserved random variable specifying if the mth bin is enriched ($X_{mc} = 1$) or non-enriched ($X_{mc} = 0$) under condition c. A mixture model for Y_{mcr} is defined as follows [8]:

$$Y_{mcr} \sim p_c f\left(y \mid \theta_{cr}^S\right) + \left(1 - p_c\right) f\left(y \mid \theta_{cr}^B\right), \tag{1}$$

where $p_c = P(X_{mc} = 1)$ is the mixture portion of the signal component and $f(y, \theta_{cr}{}^S)$ and $f(y, \theta_{cr}^B)$ are the signal and background densities for condition c and replicate r, respectively. The following aspects of the model are of particular notice:

1. The probability of a region being enriched under a particular condition, p_c, is a feature of the true underlying process and does not depend on ChIP efficiencies.
2. The signal and background distributions have parameters, θ_{cr}^S and θ_{cr}^B, respectively, that do instead depend on the replicates r. This is to account for the different ChIP-efficiencies of the individual experiments, which generate different distributions for the background and signal counts.
3. The signal and background densities can take any form. Popular choices are Poisson and Negative Binomial and their zero-inflated extensions to account for the excess number of zeros typical of this type of data.

This model does not account for spatial dependencies in the data. To further account for this, the latent variable, X_{mc}, is assumed to satisfy one dimensional Markov properties, that is,

$$P(X_{mc} = i \mid X_{-mc}) = P\left(X_{mc} = i \mid X_{m-1,c}, X_{m+1,c}\right), i \in \{0, 1\}, \tag{2}$$

where $X_{-mc} = \{X_{1c}, \ldots, X_{m-1,\,c}, X_{m+1,\,c}, \ldots, X_{Mc}\}$, i.e. the probability of a certain state at location m, given the state values at all the other locations is the same as the probability of that state at location m, given only the state values at the neighboring locations. By imposing a first-order Markov assumption on the latent variable, the model class becomes richer, since a nearest neighbor latent Markov model can induce long-range conditional dependencies on the observed data. We make a further assumption of stationarity, as typical for this type of models, and assume that the probabilities

in Eq. 2 do not depend on the region m. Thus the model (1) becomes:

$$\Upsilon_{mcr} \mid X_{m-1,c} = i, X_{m+1,c} = j \sim p_{c,ij} f\left(y, \theta^S_{cr}\right) + \left(1 - p_{c,ij}\right) f\left(y, \theta^B_{cr}\right),$$

where $p_{c,ij} = P(X_{mc} = 1 \mid X_{m-1,c} = i, X_{m+1,c} = j)$, with $i, j \in \{0, 1\}$ and for any region m. Finally, due to stationarity, it is reasonable to assume that $p_{c,01} = p_{c,10}$, resulting in only two probabilities to estimate.

3.2 Estimation of Parameters

Given the hidden Markov model of Eq. 3, the task is to estimate its parameters from data, so that a prediction can be made about which regions of the genome are enriched. In the example described in Subheading 2, we have three experiments on the CBP protein, two conditions (time 0 and time 30) and two technical replicates for the second condition. Assuming, for example, a mixture of two Negative Binomials and considering that a Negative Binomial is defined by two parameters, the mean μ and the overdispersion ϕ, the total parameters to estimate are:

$$\begin{aligned}
&\mu^B_{time0}, \phi^B_{time0}, \mu^S_{time0}, \phi^S_{time0},\\
&\mu^B_{time30,1}, \phi^B_{time30,1}, \mu^S_{time30,1}, \phi^S_{time30,1},\\
&\mu^B_{time30,2}, \phi^B_{time30,2}, \mu^S_{time30,2}, \phi^S_{time30,2},\\
&p_{time0,00}, p_{time0,11}, p_{time30,00}, p_{time30,11},
\end{aligned}$$

where μ^B and ϕ^B are the parameters of the background mixture densities, μ^S and ϕ^S are the parameters of the signal densities, and they depend both on the condition and replicates. The rest are the mixture probabilities, e.g. $p_{\text{time0},00}$ is the probability of a region between enriched at time 0 when the neighboring regions are not enriched. These probabilities depend only on the condition and not on the replicates, as they refer to the underlying binding process and not to the count data. We will denote the full set of parameters with Θ.

Estimation of parameters can be done either in a frequentist or Bayesian framework. We will now describe the Bayesian approach. For this, we need to derive the likelihood, as a function of the data and the parameters, and we also need to decide on prior distributions of the parameters. As for the likelihood, for each condition c, this is given by

$$\begin{aligned}
P(\mathbf{X}, \mathbf{Y} \mid \Theta) &= P(\mathbf{X}) P(\mathbf{Y} \mid \mathbf{X}, \Theta)\\
&= P(\mathbf{X}) \times \prod_{r=1}^{R} \prod_{m=1}^{M} \left[f\left(y_{mr}, \theta^B_r\right)\right]^{I[X_m=0]} \left[f\left(y_{mr}, \theta^S_r\right)\right]^{I[X_m=1]}.
\end{aligned}$$

Here we assume that the conditions are independent. If there is some dependency between conditions, e.g. the same protein at different time points or different proteins that have similar roles, then the model can be extended to take this into account and a joint likelihood can be written.

The term $P(\mathbf{X})$ can be fully written in terms of the mixture probabilities $p_{c,\,ij}$ or, equivalently, in terms of the transition probabilities $q_{c,\,ij} = P(X_{m+1,\,c} = j | X_{mc} = i)$ or of the joint probabilities $\delta_{c,\,ij} = P(X_{mc} = i,\, X_{m+1,\,c} = j)$ with $i, j \in \{0, 1\}$. In particular, given the first-order Markov properties, it holds that

$$P(X_{1c}, \ldots, X_{Mc}) = \frac{\prod_{m=1}^{M-1} P(X_{mc}, X_{m+1,c})}{\prod_{m=2}^{M-1} P(X_{mc})} \tag{3}$$

where $P(X_{mc},\, X_{m+1,c})$ is the joint probability of X_{mc} and $X_{m+1,c}$ and $P(X_{mc})$ is the marginal probability of X_{mc}. In particular, we have $P(X_{mc}) = \sum_{x_{m+1,c}} P(X_{mc}; X_{m+1,c} = x_{m+1,c})$. To *see* Eq. 3, consider the case of 3 regions only (i.e., $M = 3$). Then

$$P(X_1, X_2, X_3) = P(X_3 | X_1, X_2) P(X_1, X_2) = P(X_3 | X_2) P(X_1, X_2) = \frac{P(X_1, X_2) P(X_2, X_3)}{P(X_2)}.$$

Having written the likelihood function in terms of the data (unobserved X and observed Y) and in terms of the parameters Θ, the next step is to define prior distributions on Θ and to set up an MCMC routine to sample from the posterior distribution of the parameters. The steps of this procedure are described in [5] and are implemented in the R package `enRich`.

3.3 Prediction

In the final stage of the analysis, the statistical model described above is used to detect the regions in the genome that are bound by a protein of interest. Let $\mathbf{X}_c^{(1)}, \ldots, \mathbf{X}_c^{(N)}$ be N Gibbs draws of the joint states $\mathbf{X}_c$ under condition c, where $\mathbf{X}_c^{(k)} = (X_{1c}^{(k)}, \ldots, X_{Mc}^{(k)})$ is the vector of enrichment for all the regions in the genome. Under the proposed hidden Markov model, a natural estimate of the posterior probability that the mth bin is enriched is given by [10]

$$\widehat{P}(X_{mc} = 1 | \mathbf{Y}) = \sum_{k=1}^{N} I\left(X_{mc}^{(k)} = 1\right).$$

Note that, for each region, the probability depends only on the condition c, i.e. all the technical replicates for that condition contribute jointly to the likelihood function in Eq. 3. The fact that the parameters of the mixture distributions depend on the technical replicates allows to account for the different ChIP efficiencies of the

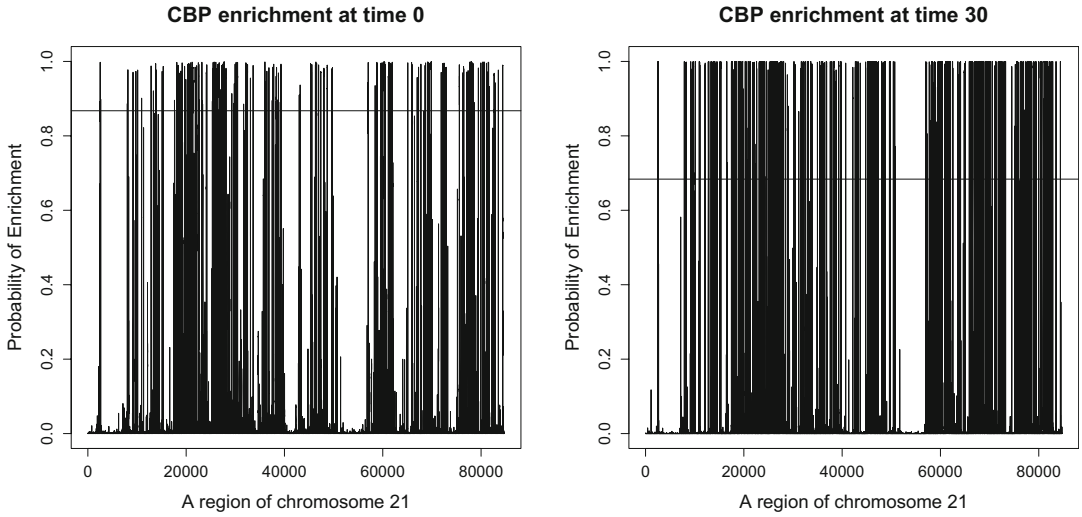


Fig. 2 Predicted probabilities of enrichment of CBP for bins of size 200 bp on a region of chromosome 21. At the time 0, the probabilities are estimated from one ChIP-seq experiment, whereas at time 30 the probabilities are estimated from two technical replicates

individual ChIP-seq experiments in this final estimate of the probability of enrichment [8].

Finally, in order to decide whether a bin is enriched or not, one sets a threshold on these probabilities. Different criteria can be used to set this cut-off. In [3], a 0.5 cut-off is used, whereby each region is assigned to the state with the highest posterior probability. In [11], a cut-off is used corresponding to a specific value of the expected posterior false discovery rate. If D is the set of declared enriched regions corresponding to a particular cut-off on the posterior probabilities, then the estimated false discovery rate for this cut-off is given by

$$\widehat{FDR} = \frac{\sum_{m \in D} \hat{P}(X_{mc} = 0 \,|\, \mathbf{Y})}{|D|}.$$

Figure 2 shows the predicted probabilities of enrichment of CBP at time 0 and time 30, respectively, from the data generated by the three ChIP-seq experiments explained in Subheading 2. The horizontal line shows the 5 % FDR cut-off. All the regions that have an estimated probability above this line are predicted as enriched by CBP at that particular time point. The 200 bp enriched regions can be further processed, for example by combining consecutive regions that belong to the same gene.

When data are available for more than one protein, the interest is also on finding the regions that are bound only by one of the proteins. In this case, one can define a probability of differential binding by [8]

$$P(X_{m1} \neq X_{m2}|\mathbf{Y}) = P(X_{m1} = 0|\mathbf{Y}_1)P(X_{m2} = 1|\mathbf{Y}_2) + P(X_{m1} = 1|\mathbf{Y}_1)P(X_{m2} = 0|\mathbf{Y}_2)$$

where $P(X_{mc} = 0|\mathbf{Y}_c) = P(X_{mc} = 0|\mathbf{Y}_{c1}, \ldots, \mathbf{Y}_{cR_c})$ is the posterior probability that the mth bin is enriched for protein c.

References

1. Wu J, Xie J (2008) Computation-based discovery of cis-regulatory modules by hidden Markov model. J Computat Biol 15(3):279–290
2. Qin Z, Yu J, Shen J, Maher C, Hu M, Kalyana-Sundaram S, Yu J, Chinnaiyan A (2010) HPeak: an HMM-based algorithm for defining read-enriched regions in chip-seq data. BMC Bioinf 11(369)
3. Spyrou C, Stark R, Lynch A, Tavare S (2009) BayesPeak: Bayesian analysis of ChIP-seq data. BMC Bioinf 10(1):299
4. Mo Q (2012) A fully Bayesian hidden Ising model for ChIP-seq data analysis. Biostatistics 13(1):113–128
5. Bao Y, Vinciotti V, Wit E, 't Hoen P (2014) Joint modelling of ChIP-seq data via a Markov random field model. Biostatistics 15 (2):296–310
6. Zhang Y, Liu T, Meyer C, Eeckhoute J, Johnson D, Bernstein B, Nussbaum C, Myers R, Brown M, Li W (2008) Model-based analysis of ChIP-Seq (MACS). Genome Biol 201(1): R137
7. Robertson G, Hirst M, Bainbridge M, Bilenky M, Zhao Y, Zeng T, Euskirchen G, Bernier B, Varhol R, Delaney A, Thiessen N, Griffith O, He A, Marra M, Snyder M, Jones S (2007) Genome-wide profiles of STAT1 DNA association using chromatin immunoprecipitation and massively parallel sequencing. Nat Methods 4 (8):651–657
8. Bao Y, Vinciotti V, Wit E, 't Hoen P (2013) Accounting for immunoprecipitation efficiencies in the statistical analysis of ChIP-seq data. BMC Bioinf 14(169)
9. Ramos Y, Hestand M, Verlaan M, Krabbendam E, Ariyurek Y, van Dam H, van Ommen G, den Dunnen J, Zantema A, 't Hoen P (2010) Genome-wide assessment of differential roles for p300 and CBP in transcription regulation. Nucleic Acids Res 38 (16):5396–5408
10. Scott S (2002) Bayesian methods for hidden Markov models: recursive computing in the 21st century. J Am Stat Assoc 97 (457):337–351
11. Broët P, Richardson S (2006) Detection of gene copy number changes in CGH microarrays using a spatially correlated mixture model. Bioinformatics 22(8):911–918

Chapter 9

Hidden Markov Models in Bioinformatics: SNV Inference from Next Generation Sequence

Jiawen Bian and Xiaobo Zhou

Abstract

The rapid development of next generation sequencing (NGS) technology provides a novel avenue for genomic exploration and research. Hidden Markov models (HMMs) have wide applications in pattern recognition as well as Bioinformatics such as transcription factor binding sites and cis-regulatory modules detection. An application of HMM is introduced in this chapter with the in-deep developing of NGS. Single nucleotide variants (SNVs) inferred from NGS are expected to reveal gene mutations in cancer. However, NGS has lower sequence coverage and poor SNV detection capability in the regulatory regions of the genome. A specific HMM is developed for this purpose to infer the genotype for each position on the genome by incorporating the mapping quality of each read and the corresponding base quality on the reads into the emission probability of HMM. The procedure and the implementation of the algorithm is presented in detail for understanding and programming.

Keywords Hidden Markov model, Single nucleotide variation, Next generation sequencing, Posterior probability, Low sequencing depth

1 Introduction

The advent of NGS technology has largely propelled the genomic research in recent years. Continuous improvement in NGS technology brings the increase of the throughput to a high extent and also lowers the cost [1]. NGS can generate millions of reads ranging from 30 to 350 base pairs (bp) based on the sequencing platform used. With abundant reads aligned, many novel inferences can be made including regulatory element identification, point mutation detection, gene fusion and gene expression estimation, and detection of RNA splicing [2–6]. NGS is expected to be a powerful tool for revealing genetic variations contributing to various complex diseases by providing sequence of a set of candidate genes, the whole exome or the whole genome. For example, whole genome sequencing can help in finding the frequency of tumor-specific point mutations for diseases such as multiple myeloma [2], while

David R. Westhead and M.S. Vijayabaskar (eds.), *Hidden Markov Models: Methods and Protocols*, Methods in Molecular Biology, vol. 1552, DOI 10.1007/978-1-4939-6753-7_9, © Springer Science+Business Media LLC 2017

whole exome sequencing can be used to discover protein-coding mutation as well as small noncoding RNAs and aberrant transcriptional regulation that may contribute to diseases such as myelodysplastic syndromes [3]. Recently, the Cancer Genome Atlas group used the latest sequencing and analysis methods to identify somatic variants across thousands of tumors. Twelve major tumor types are analyzed and the highly mutated genes and the cellular processes were studied as groundwork for developing new diagnostics and individualizing cancer treatment [7].

New tool is needed to be put forward to deal with the classification or detecting task with complex uncertainty. HMM is proved to be a powerful tool in pattern recognition for classification and detecting the hidden states behind the observations affected by multiple variables [8]. It was widely used in many fields such as speech and handwriting recognition, text classification, DNA and protein classification, detecting and characterizing transcriptional regulatory elements as well as transcription factor binding sites and cis-regulatory modules [1–8]. Recently, Bian et al. [9] proposed a specific HMM for SNVs detection of area of low sequencing depth. In this chapter, the speciality of the model is given and the detail of the implementation of HMM for SNVs inference is presented.

2 Existing Algorithms and Insufficiency

Several commercial software packages such as Roche GSMapper, Lasergene have been proposed for SNVs calling. The algorithms used in these software packages are threshold based, and thus a good threshold setting is difficult to obtain and relies heavily on the user experience [14]. In transcriptome based data, the number of reads representing a given transcript is highly variable across all genes making it difficult to determine a minimum depth. Moreover, the confidence for the prediction of each location is unavailable. Posterior probability based methods including Maq [10], SOAPsnp [11], Varscan [12], and Atlas-SNP2 [13] were proposed to deal with the uncertainty in the sequencing and alignment. Compared to the threshold based methods, posterior probability based methods achieve flexibility by considering the confidence of observation of each position on the genome. For the cancer genome sequencing data, sequencing errors, as well as the altered ploidy and tumor cellularity, are important factors affecting the accuracy of SNV calling. Although tools exist for SNVs discovery from NGS data, few are specifically suited to work with data from tumors. Recently, SNVMix [15] addressed this problem by incorporating the dependency of near-by genotypes and the posterior probability to improve the accuracy of SNVs prediction. However, the performance of SNVMix for data with low sequencing depth is not satisfactory compared to its performance with data having high

sequencing depth. It has been observed that NGS provides lower sequence coverage in certain areas of genome including regulatory regions [16]. It is necessary to improve the performance of SNVs detection for tumor data with low sequencing depth. Moreover, SNVMix has achieved a relatively high sensitivity in the Bayesian framework, but the specificity is some low. The performance of specificity is needed to be improved further.

In this chapter, a specific HMM is proposed for SNVs prediction of tumor data obtained from NGS. The motivation of HMM used here is as follows. Since non-SNVs are prevalent and continuous in the genome [17], point mutations in cancer data are relevant to certain genes and are concentrated in the corresponding area [18, 19]. The contextual information, especially for the non-SNVs, can be considered and made full use of in addition to the information from the overall distribution of traditional Bayesian framework. So the specific HMM is expected to gain more probability power from the contextual information on the genome compared to traditional Bayesian framework, and obtain better performance for SNVs prediction. Moreover, with the contextual information added to the whole distribution information, the proposed model is also expected to improve the statistical performance of Bayesian method for tumor data with low sequencing depth.

3 Problem Formulation

We denote the length of the considered genome as L. Given the aligned reads for the sequence in study, we can get the depth L_t of the stated position t on the genome. The quality of the reads covering position t and the quality of corresponding bases on the reads are denoted as $\{r_i^t\}_{i=1}^{L_t}$ and $\{q_i^t\}_{i=1}^{L_t}$ respectively (Fig. 1). We consider three genotypes for each stated position as {aa, ab, bb}, where {aa} denotes homozygous for the reference allele, {ab} denotes heterozygous, and {bb} denotes homozygous for the non-reference allele. Our aim is to predict the genotype for each position on the genome, given the aligned reads. We denote the number of the hidden states as I. The hidden state and observation for each position are noted as $\boldsymbol{S} = \{s_t\}(t = 1, 2, \ldots, L) \in \{v_i\}(i = 1, 2, \ldots, I)$ and $\boldsymbol{O} = \{\boldsymbol{o}_t\}(t = 1, 2, \ldots, L)$ respectively, where $\{v_i\}_{i=1}^{I}$ are all states considered. The underlying genotypes of the sequenced genome are taken as the hidden states, which are interpreted as follows: (1) homozygous for normal; (2) heterozygous; (3) homozygous for mutation (Fig. 2). These states are important in detecting single nucleotide polymorphism or point mutation for normal sample as well as cancer sample. The last two states are taken as SNV in our study. For simplicity, we note state {aa} as state 1, state {ab} as state 2, and state {bb} as state 3 in the following initial state distribution and state transition matrix.

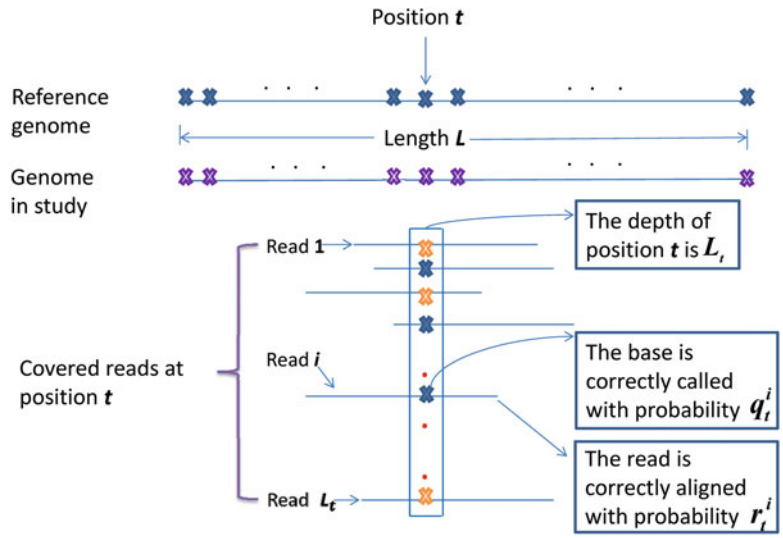


Fig. 1 Observation of HMM: the illustration for alignment at position t for the sequence in study. The observation sets $\{r_i^t\}_{i=1}^{L_t}$ and $\{q_i^t\}_{i=1}^{L_t}$ are considered as the observation of o_t in HMM. For the bases on the covered reads, *blue color* denotes the base is the same as reference allele, *yellow color* denotes the base is different from reference allele, and *purple color* denotes the base is undecided on the genome in study

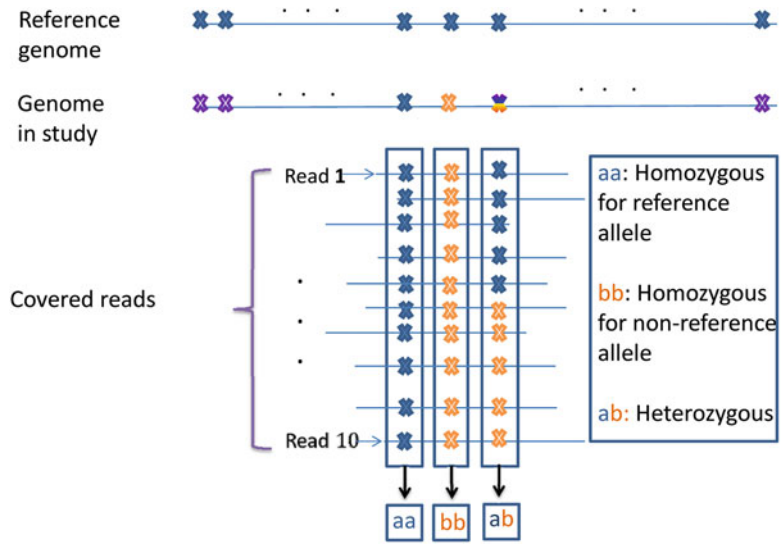


Fig. 2 States of HMM: the illustration for three states {aa}, {ab}, and {bb} in HMM. The meaning of different colors are defined the same as in Fig. 1

- Initial state distribution: $\boldsymbol{\pi} = \{\pi_1, \pi_2, \pi_3\}$, $\pi_i = P(s_1 = v_i | t = 1)$
- State transition matrix:

$$\boldsymbol{A} = \begin{pmatrix} a_{11} & a_{12} & a_{13} \\ a_{21} & a_{22} & a_{23} \\ a_{31} & a_{32} & a_{33} \end{pmatrix} \triangleq \left(\boldsymbol{A}_1^T, \boldsymbol{A}_2^T, \boldsymbol{A}_3^T\right),\ a_{ij} = P\left(s_{t+1} = v_j | s_t = v_i\right)$$

- Emission probability distribution: $\boldsymbol{B} = \{b_{s_t}(\boldsymbol{o_t})\}$

The observation to be considered for each position includes the coverage, the mapping quality of the covering reads and the base quality of the covering reads corresponding to each stated position. The observation for each position t is taken as $\boldsymbol{o_t} = \left\{q_i^t, r_i^t\right\}_{i=1}^{L_t}$. The emission probability $b_{v_i}(\boldsymbol{o_t})$ is calculated as a conditional probability, given the hidden state:

$$b_{s_t}(\boldsymbol{o_t}) = P\left(\left\{q_i^t, r_i^t\right\}_{i=1}^{L_t} | s_t = v_i\right) = f\left(\left\{q_i^t, r_i^t\right\}_{i=1}^{L_t}\right) \quad (1)$$

To make full use of the mapping quality and base quality for each position on the genome, we compute $b_{s_t}(\boldsymbol{o_t})$ using the whole probability formula by considering if the covered reads are correctly aligned and if the corresponding bases on these reads are correctly called. We use a formula motivated by Eq. 5 in [15] by introducing a generalized Binomial distribution in addition to the conditional computation of the base calling probability and aligning probability.

$$b_{v_i}(o_t) = \binom{L_t}{P_t} \prod_{j=1}^{L_t} \left\{0.25\left(1 - r_j^t\right) + 0.5 r_j^t \left[q_j^t u_i + \left(1 - q_j^t\right)(1 - u_i)\right]\right\} \quad (2)$$

where $\{u_i\}_{i=1}^{I}$ is the Binomial distribution parameter for each position on the genome and P_t is the number of reads having the same base with reference allele at position t. The detailed derivation of Eq. 2 is discussed in Bian et al. [9]. In this study, we only considered two types of nucleotides covering the stated position, which have the largest and second largest number at the stated position. In the case of rare third alleles, these reads are assumed to be errors. In this study, u_i denotes the probability of occurrence for the allele having the largest number at the stated position.

4 Prior Distribution of HMM

We take the initial distribution of $\boldsymbol{\pi}$ as Dirichlet distribution with hyper-parameter $\boldsymbol{\delta} = (\delta_1, \delta_2, \delta_3)$, and $\boldsymbol{u} = (u_1, u_2, u_3)$ according to a Beta distribution with hyper-parameter $\boldsymbol{\alpha} = (\alpha_1, \alpha_2, \alpha_3)$ and $\boldsymbol{\beta} = (\beta_1, \beta_2, \beta_3)$ as follows:

$$P(\boldsymbol{\pi}|\boldsymbol{\delta}) = \text{Dirichlet}(\boldsymbol{\pi}|\boldsymbol{\delta}) \tag{3}$$

$$\text{P}(u_k|\alpha_k, \beta_k) = \text{Beta}(u_k|\alpha_k, \beta_k) \tag{4}$$

where $\boldsymbol{\delta} = (1000, 100, 100)$ assuming that most positions will be homozygous for the reference allele. We also set $\boldsymbol{\alpha} = (1000, 500, 1)$ and $\boldsymbol{\beta} = (1, 500, 1000)$ by assuming the probability of state {aa} occurring at the stated position is much larger than that it not occurring, vice versa for state {bb}. We also assume that the probability of state {ab} occurring at the stated position is the same as that it not occurring. For the initial distribution of state transition matrix, we take the initial distribution of $\boldsymbol{A_i}$ as follows:

$$\mathbf{P}(\boldsymbol{A_i}|\gamma_i) = \text{Dirichlet}(\boldsymbol{A_i}|\gamma_i) \tag{5}$$

where $\gamma_1 = (1000, 100, 100)$, $\gamma_2 = (100, 1000, 100)$, and $\gamma_3 = (100, 100, 1000)$. Since the sum of elements in $\boldsymbol{A_i}$ should be equal to probability 1, a normalization for $\{\boldsymbol{A_i}\}_{i=1}^{I}$ is performed after each iteration of HMM.

5 Estimation of HMM Parameters

For simplicity, we denote the model parameters of HMM as $\boldsymbol{\lambda} \triangleq (\pi, u, A)$ and learn the unknown HMM by using EM algorithm and computing the maximum likelihood estimation when the observed data are incomplete [20]. The aim is to find the model parameter $\boldsymbol{\lambda}$ maximizing the observation probability, i.e., $L(\boldsymbol{O}, \boldsymbol{\lambda}) \triangleq \text{P}(\boldsymbol{O}|\boldsymbol{\lambda})$ or log $P(\boldsymbol{O}|\boldsymbol{\lambda})$,where the latter one is usually used when the length of the observation is large. We use a special case of EM algorithm, Baum–Welch algorithm [8], to learn the unknown parameters. For the training of HMM, we use the following auxiliary function $Q(\boldsymbol{\lambda}, \overline{\boldsymbol{\lambda}})$ as the objective function for the optimization of the HMM parameters.

$$Q(\boldsymbol{\lambda}, \overline{\boldsymbol{\lambda}}) = \Sigma_{\boldsymbol{s} \in \boldsymbol{S}} \log P(\boldsymbol{O}, \boldsymbol{S}|\overline{\boldsymbol{\lambda}}) P(\boldsymbol{S}|\boldsymbol{O}, \boldsymbol{\lambda}) \tag{6}$$

It has been proven that maximizing the following auxiliary function $Q(\boldsymbol{\lambda}, \overline{\boldsymbol{\lambda}})$ can lead to the increase of the likelihood $P(\boldsymbol{O}|\boldsymbol{\lambda})$, i.e., $_{\overline{\boldsymbol{\lambda}}} Q(\boldsymbol{\lambda}, \overline{\boldsymbol{\lambda}}) \rightarrow P(\boldsymbol{O}|\overline{\boldsymbol{\lambda}}) > P(\boldsymbol{O}|\boldsymbol{\lambda})$ [8].

Given model parameter set $\boldsymbol{\lambda}$, $P(\boldsymbol{O}, \boldsymbol{S}|\boldsymbol{\lambda})$ can be calculated as:

$$P(\boldsymbol{O}, \boldsymbol{S}|\boldsymbol{\lambda}) = \pi_{s_0} \prod_{t=1}^{L} a_{s_{t-1} s_t} b_{s_t}(o_t) \tag{7}$$

Replacing the term $P(\boldsymbol{O}, \boldsymbol{S}|\boldsymbol{\lambda})$ in Eq. 6 with Eq. 7, Eq. 6 can be rewritten as:

$$Q(\lambda,\bar{\lambda}) = \sum_{s\in S}\log\bar{\pi}_{s_0}P(O,S|\lambda) + \sum_{s\in S}\left(\sum_{t=1}^{L}\log\bar{a}_{s_{t-1}s_t}\right)P(O,S|\lambda) + \sum_{s\in S}\left(\sum_{t=1}^{L}\log\bar{b}_{s_t}(o_t)\right)P(O,S|\lambda) \tag{8}$$

The update of model parameters π_i and a_{ij} with constraints $\sum_{i=1}^{I}\pi_i = 1$ and $\sum_{j=1}^{I}a_{ij} = 1$ can be obtained by maximizing the first and second term of Eq. 8 with respect to π_i and a_{ij} respectively as follows:

$$\bar{\pi}_i = \frac{P(O, s_0 = i|\lambda)}{P(O|\lambda)} \tag{9}$$

$$\bar{a}_{ij} = \frac{\sum_{t=1}^{L-1}P(O, s_t = i, s_{t+1} = j|\lambda)}{\sum_{t=1}^{L-1}P(O, s_t = i|\lambda)} \tag{10}$$

The update for $\{u_i\}_{i=1}^{I}$ can be obtained by maximizing the third term with respect to $\{u_i\}_{i=1}^{I}$; however, the close-form expression is not available due to the complicated structure of the observation term $f\left(\{q_i^t, r_i^t\}_{i=1}^{L_t}\right)$. We use a Newton iteration with respect to the first and second derivation of the third term as follows:

$$u_i^{\text{new}} = u_i^{\text{old}} - \frac{\sum_{t=1}^{L}\frac{\partial f}{\partial u_i} * \frac{P(O, s_t = i|\lambda)}{f}}{\frac{\partial f}{\partial u_i}\left\{\sum_{t=1}^{L}\frac{\partial f}{\partial u_i} * \frac{P(O, s_t = i|\lambda)}{f}\right\}} \tag{11}$$

6 Implementation of the Estimation

The estimation of π_i, a_{ij} and u_i can be obtained by using the forward and backward algorithms [8].

6.1 Forward Algorithm

The forward algorithm can be used to calculate $P(O|\lambda)$ with much less order of computation load.

Define

$$f_t(i) = P(o_1, o_2, \cdots, o_t, s_t = i|\lambda) \tag{12}$$

i.e., the probability of the partial observation from 1 to t, given state i at time t and the model parameters λ. $f_t(i)$ can be calculated inductively as follows:

- Initialization: $f_1(i) = \pi_1 b_i(o_1), \quad 1 \le i \le I.$

- Induction: $f_{t+1}(i) = \left[\sum_{j=1}^{I} f_t(j)a_{ji}\right] b_i(o_{t+1}), \quad 1 \le i \le I, \; t = 1, 2, \ldots, L-1.$
- Termination: $P(\boldsymbol{O}|\boldsymbol{\lambda}) = \sum_{i=1}^{I} f_L(i).$

It has been noted that the forward algorithm can also introduce underflow errors. One approach to solving the problem is to use a scaling factor in each recursion: $l_t = \sum_{i=1}^{I} f_t(i)$, $f_t(i) = f_t(i)/l_t, \quad 1 \le i \le I$. Then the full probability of the sequence is $P(\boldsymbol{O}|\boldsymbol{\lambda}) = \sum_{i=1}^{I} f_L(i) \prod_{t=1}^{L} l_t$.

6.2 Backward Algorithm

An alternative way to calculate the full probability of the sequence, i.e., $P(\boldsymbol{O}|\boldsymbol{\lambda})$, is to use the algorithm called the backward algorithm. In contrast to the forward algorithm, the backward algorithm calculates the probability of a sequence from the tail.

Define

$$h_t(i) = P(o_{t+1}, o_{t+2}, \cdots, o_T | s_t = i, \lambda) \tag{13}$$

i.e., the probability of the partial observation from $t+1$ to the end, given state i at time t and the model parameters λ, $h_t(i)$ can be calculated inductively as follows:

- Initialization: $h_L(i) = 1, \quad 1 \le i \le I.$
- Induction: $h_t(i) = \sum_{j=1}^{I} a_{ij} b_j(o_{t+1}) f_{t+1}(j), \quad 1 \le i \le I, \quad t = L-1, L-2, \ldots, 1.$
- Termination: $P(\boldsymbol{O}|\boldsymbol{\lambda}) = \sum_{i=1}^{I} f_1(i) b_i(o_1).$

It is also noted that the backward algorithm can introduce underflow errors. The scaling factor can also be used in each recursion: $l_t = \sum_{i=1}^{I} h_t(i)$, $h_t(i) = h_t(i)/l_t, \quad 1 \le i \le I$. Then the full probability of the sequence is $P(\boldsymbol{O}|\boldsymbol{\lambda}) = \sum_{i=1}^{I} f_L(i) \prod_{t=1}^{L} l_t$.

To estimate $\overline{\pi}_i$, $\bar{a}_{ij}$ and u_i by Eqs. 9–11, the Baum–Welch [8] algorithm can be used on the basis of the forward and backward algorithm.

6.3 Baum–Welch Algorithm

If we denote

$$\varphi_t(i,j) = P(s_t = i, s_{t+1} = j | \boldsymbol{O}, \boldsymbol{\lambda}) \tag{14}$$

and

$$\eta_t(i) = P\left(s_t = i \middle| \boldsymbol{O}, \boldsymbol{\lambda}\right) \tag{15}$$

then

$$\varphi_t(i,j) = \frac{f_t(i)a_{ij}b_j(o_{t+1})h_{t+1}(j)}{P\left(\boldsymbol{O}\middle|\boldsymbol{\lambda}\right)} = \frac{f_t(i)a_{ij}b_j(o_{t+1})h_{t+1}(j)}{\sum_{i=1}^{I}\sum_{j=1}^{I} f_t(i)a_{ij}b_j(o_{t+1})h_{t+1}(j)} \tag{16}$$

and

$$\eta_t(i) = \sum_{j=1}^{I} \varphi_t(i,j) \tag{17}$$

So the update of $\overline{\pi}_i$ and $\bar{a}_{ij}$ can be obtained on the basis of $\varphi_t(i, j)$ and $\eta_t(i)$ as follows:

$$\overline{\pi}_i = \eta_1(i), \qquad \bar{a}_{ij} = \frac{\sum_{t=1}^{L-1} \varphi_t(i,j)}{\sum_{t=1}^{L-1} \eta_t(i)} \tag{18}$$

The term $P\left(\boldsymbol{O}, s_0 = i\middle|\boldsymbol{\lambda}\right)$ in Eq. 11 can be obtained by the estimation of $\overline{\pi}_i$ in Eq. 9 as follows:

$$P\left(\boldsymbol{O}, s_0 = i\middle|\boldsymbol{\lambda}\right) = \overline{\pi}_i P\left(\boldsymbol{O}\middle|\boldsymbol{\lambda}\right) = \eta_1(i) P\left(\boldsymbol{O}\middle|\boldsymbol{\lambda}\right) \tag{19}$$

where $P\left(\boldsymbol{O}\middle|\boldsymbol{\lambda}\right)$ can be obtained by the forward algorithm or the backward algorithm.

Once u_i is updated as in Eq. 11, $\eta_t(i)$ can be updated and used to update the emission probability of HMM as:

$$b_i(k) = \frac{\sum_{\text{s.t. } o_t = v_k} \eta_t(i)}{\sum_{t=1}^{T} \eta_t(i)}, \quad 1 \le i, k \le I \tag{20}$$

$\overline{\pi}_i$ and $\bar{a}_{ij}$ can also be updated as in Eq. 18 by the update of $\varphi_t(i, j)$ and $\eta_t(i)$. The iteration will not stop unless the difference of the value of $P\left(\boldsymbol{O}\middle|\boldsymbol{\lambda}\right)$ between consecutive iterations is less than a given threshold.

Finally, the Viterbi algorithm [8] is used to estimate the hidden states of the model by maximizing the posterior probability, i.e., $P\left(\boldsymbol{Q}\middle|\boldsymbol{O},\boldsymbol{\lambda}\right)$ or $P\left(\boldsymbol{Q},\boldsymbol{O}\middle|\boldsymbol{\lambda}\right)$. To save computation cost, a recursive computation similar with the forward algorithm can be used for the computation of $P\left(\boldsymbol{Q},\boldsymbol{O}\middle|\boldsymbol{\lambda}\right)$.

6.4 Viterbi Algorithm

Define

$$\sigma_t(i) = \max_{s_1, s_2, \ldots, s_{t-1}} P\left[s_1, s_2, \ldots, s_{t-1}, s_t = i, o_1, o_2, \ldots, o_t \middle| \lambda\right] \quad (21)$$

i.e., $\sigma_t(i)$ is the best score along a single path for the first t observations and ends in state i at time t. Then

$$\sigma_{t+1}(j) = \left[\max_i \sigma_t(i) a_{ij}\right] b_j(o_{t+1}) \quad (22)$$

If we denote the state maximizing $\sigma_t(i)$ as $\rho_t(i)$. The optimal state sequence can be obtained by backtracking $\rho_T(s^*)$ from time T to 1 where s^* is the state which maximizes $\rho_T(s)$.

- Initialization: $\sigma_1(i) = \pi_i b_i(o_1), \quad i = 1, 2, \ldots, I.$
- Induction: $\sigma_{t+1}(j) = \left[\max_i \sigma_t(i) a_{ij}\right] b_j(o_{t+1}),\ t = 1, 2, \ldots, T-1;$ $i = 1, 2, \ldots, I;\ \rho_{t+1}(j) = \operatorname{argmax}_i \left[\sigma_t(i) a_{ij}\right],\ t = 1, 2, \ldots, T-1;$ $i = 1, 2, \ldots, I.$
- Termination: $P^* = \max_i \sigma_T(i);\ s_T^* = \operatorname{argmax}_i \sigma_T(i).$
- States backtracking: $s_t^* = \rho_{t+1}\left(s_{t+1}^*\right), \quad t = T-1, T-2, \ldots, 1.$

In practice, the Viterbi algorithm is calculated in log space, i.e., calculating $\log[\sigma_t(i)]$ instead of $\sigma_t(i)$. In this way, one can avoid the underflow problem caused by multiplying many probabilities in the induction steps.

References

1. Shendure J, Ji H (2008) Next-generation DNA sequencing. Nat Biotechnol 26:1135–1145
2. Chapman MA et al (2011) Initial genome sequencing and analysis of multiple myeloma. Nature 471:467–472
3. Beck D, Ayers S, Wen J et al (2011) Integrative analysis of next generation sequencing for small non-coding RNAs and transcriptional regulation in Myelodysplastic Syndromes. BMC Med Genomics 4:4–19
4. Wu J, Xie J (2008) Computation-based discovery of cis-regulatory modules by hidden markov model. J Comput Biol 15:279–290
5. Wang H, Zhou X (2013) Detection and characterization of regulatory elements using probabilistic conditional random field and hidden Markov model. Chin J Cancer 32:186–194
6. Liu C, Ma J, Chang CJ et al (2013) FusionQ: a novel approach for gene fusion detection and quantification from paired-end RNA-Seq. BMC Bioinformatics 14:193. doi:10.1186/1471-2105-14-193#_blank
7. Kandoth C, Kandoth MD, Vandin F et al (2013) Mutational landscape and significance across 12 major cancer types. Nature 502:333–339
8. Rabiner LR (1989) A tutorial on hidden Markov models and selected applications in speech recognition. Proc IEEE 77:257–286
9. Bian J, Liu C, Wang H et al (2013) SNVHMM: predicting single nucleotide variants from next generation sequencing. BMC Bioinformatics 14:225
10. Li H, Ruan J, Durbin R (2008) Mapping short DNA sequencing reads and calling variants using mapping quality scores. Genome Res 18:1851–1858
11. Li R, Li Y, Yang H et al (2009) SNP detection for massively parallel whole-genome resequencing. Genome Res 19:1124–1132
12. Koboldt DC, Chen K, Wylie T et al (2009) VarScan: variant detection in massively parallel sequencing of individual and pooled samples. Bioinformatics 25:2283–2285

13. Shen Y, Wang Z, Coarfa C et al (2010) A SNP discovery method to assess variant allele probability from next-generation resequencing data. Genome Res 20:273–280
14. Martin ER, Kinnamon DD, Schmidt MA et al (2010) SeqEM: an adaptive genotype-calling approach for next generation sequencing studies. Bioinformatics 26:2803–2810
15. Goya R, Sun MG, Morin RD et al (2010) SNVMix: predicting single nucleotide variants from next generation sequencing of tumors. Bioinformatics 26:730–736
16. Wang W, Wei Z, lam TW et al (2011) Next generation sequencing has lower sequence coverage and poorer SNP-detection capability in the regulatory regions. Sci Rep 1:1–7
17. The International SNP Map Working Group (2001) A map of human genome sequence variation containing 1.42 million single nucleotide polymorphisms. Nature 409:928–933
18. Bejar R, Stevenson K, Abdel-Wahab O et al (2011) Clinical effect of point mutations in Myelodysplastic Syndromes. N Engl J Med 364:2496–2506
19. Thol F, Kade S, Schlarmann C et al (2012) Frequency and prognostic impact of mutations in SRSF2, U2AF1, and ZRSR2 in patients with myelodysplastic syndromes. Blood 119:3578–3584
20. Dempster AP, Laird NM, Rubin DB (1977) Maximum likelihood from incomplete data via the EM algorithm. J R Stat Soc B 39:1–38

Chapter 10

Computationally Tractable Multivariate HMM in Genome-Wide Mapping Studies

Hyungwon Choi, Debashis Ghosh, and Zhaohui Qin

Abstract

Hidden Markov model (HMM) is widely used for modeling spatially correlated genomic data (series data). In genomics, datasets of this kind are generated from genome-wide mapping studies through high-throughput methods such as chromatin immunoprecipitation coupled with massively parallel sequencing (ChIP-seq). When multiple regulatory protein binding sites or related epigenetic modifications are mapped simultaneously, the correlation between data series can be incorporated into the latent variable inference in a multivariate form of HMM, potentially increasing the statistical power of signal detection. In this chapter, we review the challenges of multivariate HMMs and propose a computationally tractable method called sparsely correlated HMMs (scHMM). We illustrate the method and the scHMM package using an example mouse ChIP-seq dataset.

Keywords Hidden Markov model, Genome-wide mapping study

1 Introduction

Genome-wide mapping studies aim to identify binding sites of DNA-bound molecules such as transcription factors or loci harboring epigenetic modifications (acetylation/methylation of histones) based on the sequencing data. Chromatin immunoprecipitation (ChIP) followed by microarrays (ChIP-chip) or sequencing (ChIP-seq), which isolates DNA-bound proteins by an antibody-based purification, is the method of choice in the current literature to produce quantitative readings for mapping regulatory/epigenetic activities on genomic loci in a high-throughput manner [1, 2]. The computational analysis of these large-scale datasets involves nontrivial data processing steps for experimental scientists, including sequence alignment, read counting and signal extraction, and summarization of the resulting data. Moreover, as the sequencing technology advances, it is becoming increasingly common to simultaneously perform ChIP experiments for multiple regulatory proteins or epigenetic events. Although localization of binding or

David R. Westhead and M.S. Vijayabaskar (eds.), *Hidden Markov Models: Methods and Protocols*, Methods in Molecular Biology, vol. 1552, DOI 10.1007/978-1-4939-6753-7_10, © Springer Science+Business Media LLC 2017

modification sites can be done separately, these datasets are profiles of functionally co-regulated factors and are thus expected to be positively or negatively correlated, and joint analysis of multiple factors is therefore mutually beneficial for the localization of all the events covered in the entire set of experiments.

2 Basic Concepts of Univariate and Multivariate HMM

HMM is a probability model designed for inference of latent states in spatially correlated data [3]. In the context of genome-wide mapping studies, the frequency or count of aligned sequences at each genomic locus is observed, and the observed data (e.g., read counts in a 200 bp block) is assumed to come from either of the two latent states: enriched with binding/modification events (state 1), or background (state 0). Instead of modeling the observed series as multivariate data, the spatial correlation is assumed for the hidden states in adjacent loci, which can be easily modeled as a Markov process. Conditional on the hidden state at a locus, read count data are assumed to be statistically independent and follow identical distributions (emission). This probabilistic framework naturally suits the data from ChIP experiments, which led to development of a variety of approaches with the computational backbone based on HMMs [4–10].

To review the basic concept of HMM, the key components of HMM consist of the probability distribution $\pi(o_t|h_t)$ for the sequence of actual observations O (emission), the transition probabilities (q) defining the Markov process for the hidden states $H = h_0 \rightarrow h_1 \rightarrow \ldots \rightarrow h_T$, and finally the distribution of the initial state $\pi(h_0)$. Assuming a two-state HMM, the probability of observing a particular sequence of read count data at $(T + 1)$ loci, denoted by $O = \{o_0, \ldots, o_T\}$, conditional on hidden states $H = \{h_0, \ldots, h_T\}$, can be written as

$$P(O, H) = \pi(h_0)P\left(o_0\middle|h_0\right)\prod_{t=1}^{T}\pi(h_t)P(o_t|h_t)q\left(h_t\middle|h_{t-1}\right)$$

where $q\left(h_t\middle|h_{t-1}\right)$ is the transition probability between $t-1$ and t.] The goal of the analysis is to infer the hidden states $H = \{h_t\}_{t=0}^{T}$, and a dynamic programming approach termed the Viterbi algorithm is typically used to find the most probable solution H^*, which maximizes $P(O, H)$. Since the model parameters for this distribution are unknown, a practical approach is to iteratively update the parameters of the emission probability (e.g., by the EM algorithm), transition probabilities, and locus-specific *a posteriori* estimates of the hidden states (posterior probability), until all relevant parameters converge. This is the Baum–Welch algorithm, which employs the forward–backward equations to achieve all the updates [3].

3 scHMM as a Computationally Tractable Alternative to Multivariate HMM

With rapid advances and increasing access to sequencing technology, it has become a common practice to perform genome-wide mapping studies for multiple ChIP four major trimethylation events to delineateexperiments. For example, Mikkelsen et al. have performed ChIP-seq experiments for genome-wide mapping of four major trimethylation events to delineate chromatin state changes in embryonic stem (ES) cells, neural progenitor (NP) cells, and embryonic fibroblasts (MEF) in mice [11]. Wang et al. have surveyed the combinatorial pattern of 39 histone acetylations and methylations in human CD4+ T cells using ChIP-seq [12] and the same group reported a global mapping of histone acetyltransferase and deacetylase binding sites for complete characterization of the chromatin states in active and inactive genes along with active RNA polymerase activity [13]. A key characteristic commonly shared by these datasets is that the regulatory proteins and epigenetic modifications are functionally co-regulated factors and therefore it is reasonable to expect a high degree of correlation. If binding sites or epigenetic modification sites are localized by separate analysis for each of K ChIP samples, this correlation will not be incorporated in the analysis.

To address the issue, we posit a fully multivariate HMM, i.e., a single HMM with multivariate hidden states for all possible combinatorial states (2^K state combinations assuming two states in each sample) and independent (or dependent) emission. However, for even modest K, the dynamic programming algorithm in the forward–backward equations becomes computationally intractable since $2^K \times 2^K$ transitions must be followed. Figure 1 illustrates

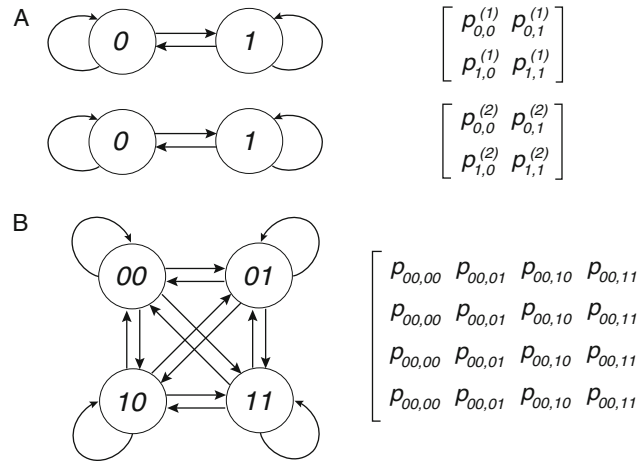


Fig. 1 HMM transitions and transition probabilities for the analysis of two samples ($K = 2$) in independent HMM (**a**) and multivariate HMM (**b**)

this problem for $K = 2$. By coupling the hidden states in two ChIP samples, the number of transitions to be followed increased significantly, and this computational burden will soon become restrictive as K increases. Since the order of computation for forward and backward equations for updating the locus specific probabilities is proportional to the number of loci (in addition to the number of hidden states), the exponentially increasing number of transitions makes the entire inference algorithm intractable.

While there are other approximation-based approaches to solve this problem [14], the soaring computational cost remains a challenge when the number of loci is extremely large, unless the hidden state inference itself is reduced to the dynamic programming algorithm in each ChIP sample. This motivated us to develop the sparsely correlated HMMsparsely correlated HMMs (scHMM), termed scHMM. In this model, the transitions are modeled in each sample separately ($2K$ states), but the transition in one sample depends on the transitions in other *correlated* samples at the same locus. Suppose that we express the transition probabilities in the jth ChIP sample at locus t as

$$K_j(t) = \begin{pmatrix} 1 - p_{jt} & p_{jt} \\ 1 - q_{jt} & q_{jt} \end{pmatrix}$$

where

$$\log\left(\frac{p_{jt}}{1 - p_{jt}}\right) = \beta_{j0}^{\mathrm{p}} + \sum_{k \neq j} \left(\beta_{jk}^{\mathrm{p}} h_{k,t-1} + \beta_{jk}^{\mathrm{c}} h_{k,t}\right)$$

$$\log\left(\frac{q_{jt}}{1 - q_{jt}}\right) = \gamma_{j0}^{\mathrm{p}} + \sum_{k \neq j} \left(\gamma_{jk}^{\mathrm{p}} h_{k,t-1} + \gamma_{jk}^{\mathrm{c}} h_{k,t}\right)$$

where β_{jk} and γ_{jk} are the logistic regression coefficients for the transition moves ($0 \rightarrow 1$ and $1 \rightarrow 1$ transitions) in ChIP sample j respectively, the superscripts p and c indicate the effects of previous and current loci, and $(h_{k,t-1}, h_{k,t})$ indicate the hidden states for ChIP sample k ($k \neq j$) at loci $t - 1$ and t. This formulation allows us to incorporate the transitions of hidden states from locus $(t - 1)$ to t in other samples into the transition probability of the current sample j, borrowing statistical information across samples. As the number of jointly analyzed ChIP samples K increases, however, it is necessary to choose carefully which other samples should contribute to the Markov process of the current sample j, and scHMM achieves this with the least absolute shrinkage selection operator (LASSO) [15],

$$\sum_{k \neq j} \left(\left|\beta_{jk}^{\mathrm{p}}\right| + \left|\beta_{jk}^{\mathrm{c}}\right|\right) \leq \lambda \text{ and } \sum_{k \neq j} \left(\left|\gamma_{jk}^{\mathrm{p}}\right| + \left|\gamma_{jk}^{\mathrm{c}}\right|\right) \leq \rho$$

This yields automatic selection of correlated samples for each ChIP sample at different loci (t), and partly achieves the sensitivity of fully multivariate HMM at a reduced cost of computation.

4 Step-Wise Estimation Procedure for scHMM

Here we employ the Baum–Welch algorithm for estimating the HMM parameters, which uses forward–backward variables to dynamically update both the locus specific posterior probability for hidden states (to be in the enriched state 1) and the transition probabilities, while iteratively updating the emission parameters at the same time. In genome-wide mapping studies, the read count data is collected on a very long sequence (typically millions of windows), and thus the iteration of forward–backward equations comes with a heavy computational cost. Moreover, in scHMM, despite the benefit of not updating $2^K \times 2^K$ transitions, the regression coefficients with LASSO penalty must be obtained for each iteration. To reduce the computation burden at the expense of a minor loss in precision, the scHMM software runs through the following steps in one cycle:

- **Step 1**: Estimate the emission parameters by a mixture model fitting.
- **Step 2**: Use the forward–backward algorithm to compute the locus-specific posterior probability (of being in the enriched state) separately for each sample.
- **Step 3**: Take a subset of the full dataset as a training dataset, and estimate the logistic regression coefficients with LASSO penalty with coordinate descent algorithm [16], where the variables are selected based on by the Akaike information criterion (AIC).
- **Step 4**: Recompute the forward–backward equations based on the transition probabilities calculated with the logistic regression model to update the posterior probabilities at all loci for one sample at a time.

This algorithm has been tested using both simulation datasets and several large-scale ChIP-seq datasets. It has consistently yielded good performance in the sense that additional iterations resulted in minimal change in the final posterior probability scores [17].

Step 1. Mixture Model-Based Estimation of Emission Parameters

In each sample (omitting sample subscripts), initialize parameters $(\rho_0, \lambda_0, \lambda_1, \lambda_2)$ where the background state emission is zero inflated Poisson distribution

$$g_0(O) = P(O|h = 0) = \rho_0\delta(O) + (1 - \rho_0)e^{-\lambda_0}\frac{{\lambda_0}^O}{O!}$$

and the enriched state emission is the generalized Poisson distribution

$$g_1(O) = P(O|h = 1) = \lambda_1 e^{-(\lambda_1+\lambda_2 O)}\frac{(\lambda_1 + \lambda_2 O)^{O-1}}{O!}$$

Compute the expectation of latent variables $\{h_{j,t}\}$ by the Bayes theorem

$$P(h = 1|O) = \theta g_1(O)/\{(1 - \theta)g_0(O) + \theta g_1(O)\}$$

where $\theta = P(h = 1)$.

REPEAT

- Find optimal solutions of $(\rho_0, \lambda_0, \lambda_1, \lambda_2)$ given $\tilde{h} = 1\{h \geq 0.5\}$ (e.g., the method of moment estimator)
- Update the latent variables $\{h\}$

UNTIL convergence

Step 2. Forward–Backward Algorithm to Compute Locus Specific Probabilities

For all loci $\{t = 0, 1, \ldots, T\}$, compute the forward and backward variables

$$f_t(z) = P(o_0, o_1, \ldots, o_t, h_t = z)$$

$$b_t(z) = P(o_{t+1}, \ldots, o_T|h_t = z)$$

by forward and backward dynamic recursions, respectively (see [3]).

Derive the posterior probability at a given locus

$$P(h_t = z|O) = \frac{P(h_t = z, O)}{P(O)} = \frac{f_t(z)b_t(z)}{P(O)}$$

where the last equality holds true due to the Markov property of the hidden state process.

Step 3. Logistic Regression for Transition Probability with LASSO Penalty

SET $y_t = 1\{h_{j,t} \geq 0.5\}$ for all $t = 0, 1, \ldots, T$.

Initialize the regression coefficients $\{\beta_{j0}^{\mathrm{p}}\}, \{\beta_{jk}^{\mathrm{p}}\}_{k\neq j}, \{\beta_{jk}^{\mathrm{c}}\}_{k\neq j}$.

Perform iterative weighted least squares as follows:

REPEAT

- Compute the weights

$$w_t = \tilde{p}(\boldsymbol{h_t})(1 - \tilde{p}(\boldsymbol{h_t}))$$

$$z_t = \tilde{\beta}_{j0} + \boldsymbol{h}'_{t,-j}\tilde{\boldsymbol{\beta}}_{j,-j} + \frac{y_t - \tilde{p}(\boldsymbol{h_t})}{\tilde{p}(\boldsymbol{h_t})(1 - \tilde{p}(\boldsymbol{h_t}))}$$

where tilde indicates the current estimate of other regression coefficients, $-j$ denotes vectors without element j, and $\tilde{p}(\boldsymbol{h_t})$ is the posterior probability of enriched state at locus t given the hidden state vector $\boldsymbol{h_t}$ across all the samples such that

$$\log\left(\frac{\tilde{p}(\boldsymbol{h_t})}{1 - \tilde{p}(\boldsymbol{h_t})}\right) = \tilde{\beta}_{j0} + \boldsymbol{h}'_t\tilde{\boldsymbol{\beta}}_j$$

- Update each coefficient by

$$\beta_{j0} = \frac{\sum_{t\in\varepsilon} w_t\left(z_t - \boldsymbol{h}'_t\tilde{\boldsymbol{\beta}}_{j,-j}\right)}{\sum_{t\in\varepsilon} w_t}$$

$$\beta_{jk} = \frac{S\left(|\varepsilon|^{-1}\sum_{t\in\varepsilon} w_t h_{k,t}\left(z_t - \beta_{j0} - \sum_{l\notin(j,k)} h_{l,t}\beta_{jl}\right), \lambda\right)_+}{\sum_{t\in\varepsilon} w_t h_{l,t}^2 + \lambda}$$

where $\varepsilon = \{t : h_{j,t} \geq 0.5\}$ and $S(\cdot)_+$ is the soft-thresholding operator defined in [16].

UNTIL convergence

Step 4. Reapply the Forward–Backward Algorithm to Compute the Scores

REPEAT **Step 2**.

5 Application to ChIP-seq Data Analysis

5.1 Computing Environment and Resources

In this section, we describe the data analysis steps for analyzing the mouse ChIP-seq data [11]. It is important to make sure that all the required software tools and the minimal computational resource are available. While experienced bioinformaticians are expected to have a more advanced pipeline and resources, the following materials are required for basic analysis:

1. A desktop computer or preferably a server with Unix/Linux operating system with a sufficient disk space (typically tens of gigabytes are consumed for one dataset) and standard ANSI C compiler.
2. SRA toolkit (http://eutils.ncbi.nih.gov/Traces/sra/?view=software) and Bowtie (http://bowtie-bio.sourceforge.net/index.shtml).
3. R (http://www.r-project.org/).

4. scHMM package from SourceForge (http://sourceforge.net/projects/schmm/) and the GNU Scientific Library (GSL) C library (http://www.gnu.org/software/gsl/).

5.2 Data Processing: Sequence Read Alignment and Counting

For ChIP-seq datasets, the analysis starts with processing of the sequence read data, typically stored in a vendor specific format, which can be converted into other commonly accepted formats such as fasta and fastq. In the case of the mouse dataset, the data was stored as .sra files in the Sequence Read Archive (SRA) and thus they were converted into fastq format using fastq-dump program in the SRA toolkit (version 2.6.1) by the following script:

```
#!/bin/bash
for sra in *.sra; do
        cmd="fastq-dump $sra"
        echo "$cmd";
        $cmd
done
```

Since processing of each file may take a long time, we recommend that the scripts be run in 'screen' utility, which runs the commands in the background and allows monitoring of the progress with 'screen –r' command. Upon completion, fastq files with extention '.fastq' will be generated. These sequences need to be aligned onto the reference genome (e.g., bowtie mm10 index, ftp://ftp.ccb.jhu.edu/pub/data/bowtie2_indexes/mm10.zip), which can be achieved with computationally efficient tools such as Bowtie.

```
#!/bin/bash
for files in *.fastq; do
        bn=$(basename $files .fastq);
        cmd="bowtie -k 3 -v 2 mm10 –suppress 1,5,6,7 -q $files"
        echo "$cmd";
        $cmd > $bn.map;
done
```

Note that the reference genome data, produced by the genome builder in Bowtie, must be placed in the directory containing the fastq files (mm10). This script will produce aligned sequence output file (.map files) and the size of output depends on the alignment stringency criterion applied in the optional arguments specified for the bowtie software (the options used above are standard; see Bowtie manual). The next step is to obtain read count data along the chromosomes, typically in the form of a tile of small windows of prefixed size.

```
#!/bin/bash
for files in *.map; do
        bn=$(basename $files .map);
        cmd="getReadCount $files mm10chrom 500 300 27"
        echo "$cmd";
        $cmd > $bn.txt;
done
```

The script above uses getReadCount, a function built in the scHMM package and it expects several key parameters to be provided by the user in addition to the .map file:

- Chromosome length information.
- The size of the windows to summarize the read counts.
- Expected average DNA fragment length.
- Length of sequence tag (read length).

More experienced users can write their own script to do this step. Once the read count files are generated from all .map files, it is important to note that multiple raw sequence files may be generated for the same regulatory protein or epigenetic modification. Thus an additional step may be necessary to aggregate read counts for each ChIP experiment (*sumCount* command) and merge files into one data frame (*mergeData* command). The following is an example:

```
#!/bin/bash
sumCount SRR006805.txt SRR006848.txt SRR006849.txt > NP_H3K9me3
sumCount SRR006888.txt SRR006889.txt > NP_H3K4me3
sumCount SRR007333.txt SRR007334.txt > NP_H3K36me3
sumCount SRR007425.txt SRR007426.txt > NP_H3K27me3
mergeData NP_H3K27me3 NP_H3K36me3 NP_H3K4me3 NP_H3K9me3 >\
mouseData
```

5.3 Separate HMM Fitting for Individual ChIP Samples

The *mouseData* file generated above is the input file for all subsequent HMM analyses. The first step to the model fitting is to obtain emission parameters. As indicated earlier, concurrent updates of all the model parameters via the Baum–Welch algorithm are feasible in short series data, which updates the emission, transition probabilities, and latent state probabilities iteratively. This is not feasible in long series datasets with large T and thus we obtain the emission parameters by the EM algorithm first and infer the other model parameters later. The current version of *getEmission* program in the scHMM package fits a two component mixture distribution consisting of zero inflated Poisson distribution

(background state) and generalized Poisson distribution (enriched state).

```
#!/bin/bash
getEmis mouseData 100 > emisT
```

The second argument "100" indicates the upper limit to the count data used for modeling. This is necessary to stabilize the emission parameter estimates since there are genomic regions with extremely large counts, especially in repeat regions, which results in unreasonably large variance estimates for the generalized Poisson distribution for the enriched state (state 1).

The next step is to estimate the transition probabilities and the latent state inference at individual loci, which can be done with the *iHMM-count* program for each chromosome as follows:

```
#!/bin/bash
for i in {1..19}
do
        getChrData mouseData $i > chr$i
        iHMM-count chr$i emisT
done
```

This command will first extract each chromosome data, fit a HMM, and produce an output file named as the chromosome number with a suffix "_HMM". The output file contains the chromosome number, starting position of each window, and the posterior probabilities of all the ChIP samples provided in the input file.

5.4 scHMM Fitting

As described above, scHMM analysis is conducted in a few steps following the emission parameter estimation step. First, a training dataset (a randomly selected subset of the mouse data) must be selected for the estimation of logistic regression coefficients. *get_train_data* command randomly samples m regions of w consecutive windows. See the following example where segments with $m = 10$, $w = 1000$ are extracted in each chromosome and then concatenated into a single file.

```
#!/bin/bash
for i in {1..19}
do
        get_train_data chr${i} 10 1000 > chr${i}tr
done
cat chr1tr chr2tr chr3tr chr4tr chr5tr chr6tr chr7tr chr8tr
chr9tr chr10tr chr11tr chr12tr chr13tr chr14tr chr15tr chr16tr
chr17tr chr18tr chr19tr > chr_tr
```

The second step is to use this training dataset and the emission parameters obtained in the previous HMM fitting to estimate the logistic regression parameters.

```
#!/bin/bash
scHMM-count chr_tr emisT
```

The command returns the regression coefficients in an output file 'chr_tr_scHMMcoef', which will be used for the final step of the forward–backward algorithm (*scHMM-count-fb* command):

```
#!/bin/bash
for i in {1..19}
do
        scHMM-count-fb chr${i} chr_tr_scHMMcoef emisT
done
```

This completes the scHMM fitting process and the output files will be saved as the chromosome number appended with the suffix '_scHMM'.

5.5 Full Multivariate HMM Fitting

The current implementation of the scHMM package also offers a routine for fitting *fullHMM*. To run this analysis, run *fullHMM-count* command:

```
#!/bin/bash
for i in {1..19}
do
        fullHMM-count chr${i} emisT 5
done
```

The last argument '5' is the number of iterations in the Baum–-Welch algorithm (without the update of the emission parameters). The algorithm typically converges to the final solution in a few cycles, but it may keep iterating through updates to achieve pre-specified precision if the criterion is too stringent. This safeguard is necessary since each iteration is highly time consuming when the number of ChIP sample is large (>5).

5.6 Post Hoc Analysis and Interpretation of HMM/scHMM Output

The file format of the output generated from both HMM fitting process is identical to that of the input files, with the raw read count data replaced by the posterior probabilities. Each window associated with a genomic locus (for starting position on the + strand) will be assigned the locus-specific marginal probability score for all ChIP samples. Based on a reasonable threshold (e.g., 0.9), the genomic regions of consecutive windows with scores above the threshold can be extracted and referenced against the known genome annotation with respect to the transcription start sites

(TSS) and transcription end sites (TES), upstream regions of TSS, and exons. Figure 2 below shows the read count data and the peaks selected with the posterior probability threshold 0.9, marking the regions enriched with varyinh combination of four histone trimethylations (H3K27me3, H3K36me3, H3K4me3, H3K9me3) in three different cell types. The red bars in the bottom of each panel indicate the location of annotated genes, green bars are the additional peaks reported by the scHMM method. As expected, these additional peaks tend to appear in the regions where other correlated series data showed significant peaks in the nearby windows, indicating that the scHMM effectively utilizes this information to mimick the fully multivariate HMM in relevant regions. This can be further validated by the fact that the additional peaks tend to be located near the TSS, TES, and near exons (not shown in the figure).

6 Conclusion

In this chapter we introduce a computationally efficient algorithm for multivariate HMM inference, termed sparsely correlated HMM (scHMM) and provided a tutorial to apply the method to a genome-wide location study of epigenetic regulation via histone modification based on chromatin immunoprecipitation experiments. Despite the computational intractability of simultaneously inferring multiple HMMs, scHMM can borrow statistical information across different spatially correlated series data and still achieve a superior statistical power comparable to multivariate HMM. In addition to the reduction in computational cost, the sparse logistic regression models reveal the underlying dynamics of hidden state transitions across different regulatory elements, and are thus able to highlight the inter-relationship of various epigenetic regulations and shed important biological insights. Hence we expect this methodology to be widely applicable to similar types of genome-scale location studies in the future.

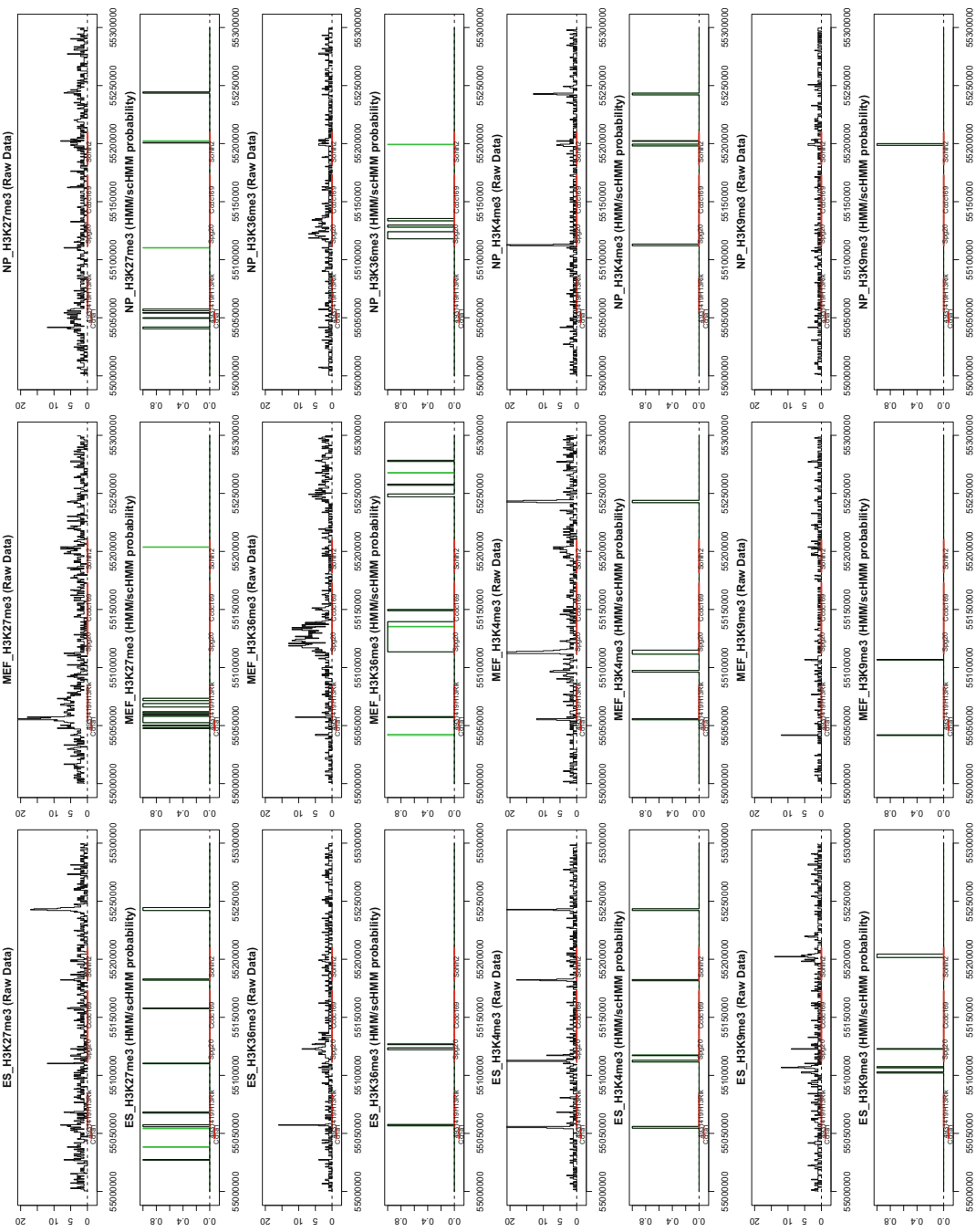


Fig. 2 Four trimethylations in three cell types. Raw data with HMM/scHMM probabilities. Additional peak regions reported by scHMM are shown in *green bars*

References

1. Furey TS (2012) ChIP-seq and beyond: new and improved methodologies to detect and characterize protein-DNA interactions. Nat Rev Genet 13(12):840–852. doi:10.1038/nrg3306
2. Park PJ (2009) ChIP-seq: advantages and challenges of a maturing technology. Nat Rev Genet 10(10):669–680. doi:10.1038/nrg2641
3. Rabiner L (1989) A tutorial on hidden Markov models and selected applications in speech recognition. Proc IEEE 77:257–286
4. Humburg P, Bulger D, Stone G (2008) Parameter estimation for robust HMM analysis of ChIP-chip data. BMC Bioinformatics 9:343. doi:10.1186/1471-2105-9-343
5. Ji H, Wong WH (2005) TileMap: create chromosomal map of tiling array hybridizations. Bioinformatics 21(18):3629–3636. doi:10.1093/bioinformatics/bti593
6. Li W, Meyer CA, Liu XS (2005) A hidden Markov model for analyzing ChIP-chip experiments on genome tiling arrays and its application to p53 binding sequences. Bioinformatics 21(Suppl 1):i274–i282. doi:10.1093/bioinformatics/bti1046
7. Qin ZS, Yu J, Shen J, Maher CA, Hu M, Kalyana-Sundaram S, Yu J, Chinnaiyan AM (2010) HPeak: an HMM-based algorithm for defining read-enriched regions in ChIP-Seq data. BMC Bioinformatics 11:369. doi:10.1186/1471-2105-11-369
8. Rashid N, Sun W, Ibrahim JG (2014) Some statistical strategies for DAE-seq data analysis: variable selection and modeling dependencies among observations. J Am Stat Assoc 109:78–94
9. Spyrou C, Stark R, Lynch AG, Tavare S (2009) BayesPeak: Bayesian analysis of ChIP-seq data. BMC Bioinformatics 10:299. doi:10.1186/1471-2105-10-299
10. Yau C, Holmes CC (2013) A decision-theoretic approach for segmental classification. Ann Appl Stat 7:1814–1835
11. Mikkelsen TS, Ku M, Jaffe DB, Issac B, Lieberman E, Giannoukos G, Alvarez P, Brockman W, Kim TK, Koche RP, Lee W, Mendenhall E, O'Donovan A, Presser A, Russ C, Xie X, Meissner A, Wernig M, Jaenisch R, Nusbaum C, Lander ES, Bernstein BE (2007) Genome-wide maps of chromatin state in pluripotent and lineage-committed cells. Nature 448(7153):553–560. doi:10.1038/nature06008
12. Wang Z, Zang C, Rosenfeld JA, Schones DE, Barski A, Cuddapah S, Cui K, Roh TY, Peng W, Zhang MQ, Zhao K (2008) Combinatorial patterns of histone acetylations and methylations in the human genome. Nat Genet 40(7):897–903. doi:10.1038/ng.154
13. Wang Z, Zang C, Cui K, Schones DE, Barski A, Peng W, Zhao K (2009) Genome-wide mapping of HATs and HDACs reveals distinct functions in active and inactive genes. Cell 138(5):1019–1031. doi:10.1016/j.cell.2009.06.049
14. Ghahramani Z, Jordan M (1997) Factorial hidden Markov models. Mach Learn 29:245–273
15. Tibshirani R (1996) Regression shrinkage and selection via the lasso. J R Stat Soc B 58:267–288
16. Friedman J, Hastie T, Tibshirani R (2010) Regularization paths for generalized linear models via coordinate descent. J Stat Softw 33:1–22
17. Choi H, Fermin D, Nesvizhskii AI, Ghosh D, Qin ZS (2013) Sparsely correlated hidden Markov models with application to genome-wide location studies. Bioinformatics 29(5):533–541. doi:10.1093/bioinformatics/btt012

Chapter 11

Hidden Markov Models in Population Genomics

Julien Y. Dutheil

Abstract

With the advent of sequencing techniques population genomics took a major shift. The structure of data sets has evolved from a sample of a few loci in the genome, sequenced in dozens of individuals, to collections of complete genomes, virtually comprising all available loci. Initially sequenced in a few individuals, such genomic data sets are now reaching and even exceeding the size of traditional data sets in the number of haplotypes sequenced. Because all loci in a genome are not independent, this evolution of data sets is mirrored by a methodological change. The evolutionary processes that generate the observed sequences are now modeled spatially along genomes whereas it was previously described temporally (either in a forward or backward manner). Although the spatial process of sequence evolution is complex, approximations to the model feature Markovian properties, permitting efficient inference. In this chapter, we introduce these recent developments that enable the modeling of the evolutionary history of a sample of several individual genomes. Such models assume the occurrence of meiotic recombination, and therefore, to date, they are dedicated to the analysis of eukaryotic species.

Keywords Population genomics, Coalescence theory, Recombination

1 Introduction

Molecular population genetics aims to analyze data in the form of sequences of several individuals from one or more populations. As full genome sequences have become available due to the development of new sequencing techniques, the field has evolved toward "population genomics" (for example see [1]). The change in data sets has resulted in a new paradigm for the underlying models, building on existing theories, such as the so-called "coalescent" models but accounting for the "spatial" realization of processes along genomes, a dimension that can no longer be ignored. Hidden Markov models (HMMs) play a fundamental role in this new paradigm, and their combination with the coalescence theory is referred to as "coalescent hidden Markov models." The term was initially coined by Hobolth et al. [2] in reference to a particular method (CoalHMM, see also [3]), but it has gained broader usage as new methods have been developed [4].

David R. Westhead and M.S. Vijayabaskar (eds.), *Hidden Markov Models: Methods and Protocols*, Methods in Molecular Biology, vol. 1552, DOI 10.1007/978-1-4939-6753-7_11, © Springer Science+Business Media LLC 2017

The coalescence theory provides analytical tools to describe the evolution of sequences in populations [5, 6]. Briefly, the coalescent models the genealogy of a sample of sequences, in a retrospective manner. Figure 1a shows a simple example of a population of five sequences (haplotypes) evolving at a constant size during four generations. In this simple example, the population evolves neutrally (with no selection of any kind) and reproduction is fully random, with no recombination. In this most simplistic model of sequence evolution, the only processes at stake are mutation and genetic drift. The parental relationships between haplotypes show that the five haplotypes in the last generation have a common ancestor in the third haplotype of the first generation (it is said that the lineages *coalesce*, going backward in time). In such a simple model a single tree is sufficient to describe the genealogy of the sample, where inner nodes represent coalescence events at the most recent common ancestors (MRCAs) of the haplotypes underneath. The coalescence theory allows the prediction of several properties of this tree, such as the distribution of coalescence times (number of generations necessary for two haplotypes to find a common ancestor when going backward in time).

The consequence of adding recombination to this simple model is that the genealogy of a set of haplotypes can no longer be described by a single tree. Depending on the loci under consideration, the last common ancestral haplotype of two extant haplotypes may differ. Although each position in the sequence may be described by a tree, the complete history of the sequences is represented by a more complex network called the "ancestral recombination graph" (ARG) [7]. Figure 1b shows the evolution of five haplotypes through four generations, with two recombination events defining three regions along the sequence, each with a distinct tree. If full genomes are involved, the ARG is a highly complex unknown structure that must be reconstructed from the data. As a result, statistical inference under the coalescent with recombination is impossible at the genomic scale, and simplifications are required. McVean and Cardin [8] and Marjoram and Wall [9] first introduced an approximation to the coalescent with recombination process that permits the calculation of a likelihood function in an efficient manner, making possible maximum likelihood inference using such models. This model follows the pioneer work of Wiuf and Hein [10], who first described the coalescent along genomes as a spatial rather than temporal process. More recently, it was further developed by Hobolth and Jensen [11].

An important underlying concept is that of *ancestral material* (reviewed in [9]), defined as a material that is found in at least one descendant in the sample of extant sequences under consideration. In our example (Fig. 1b), the blue chromosome is ancestral, and the dark green, pink and orange ones are not because they are lost by genetic drift. The light green chromosome is an intermediate

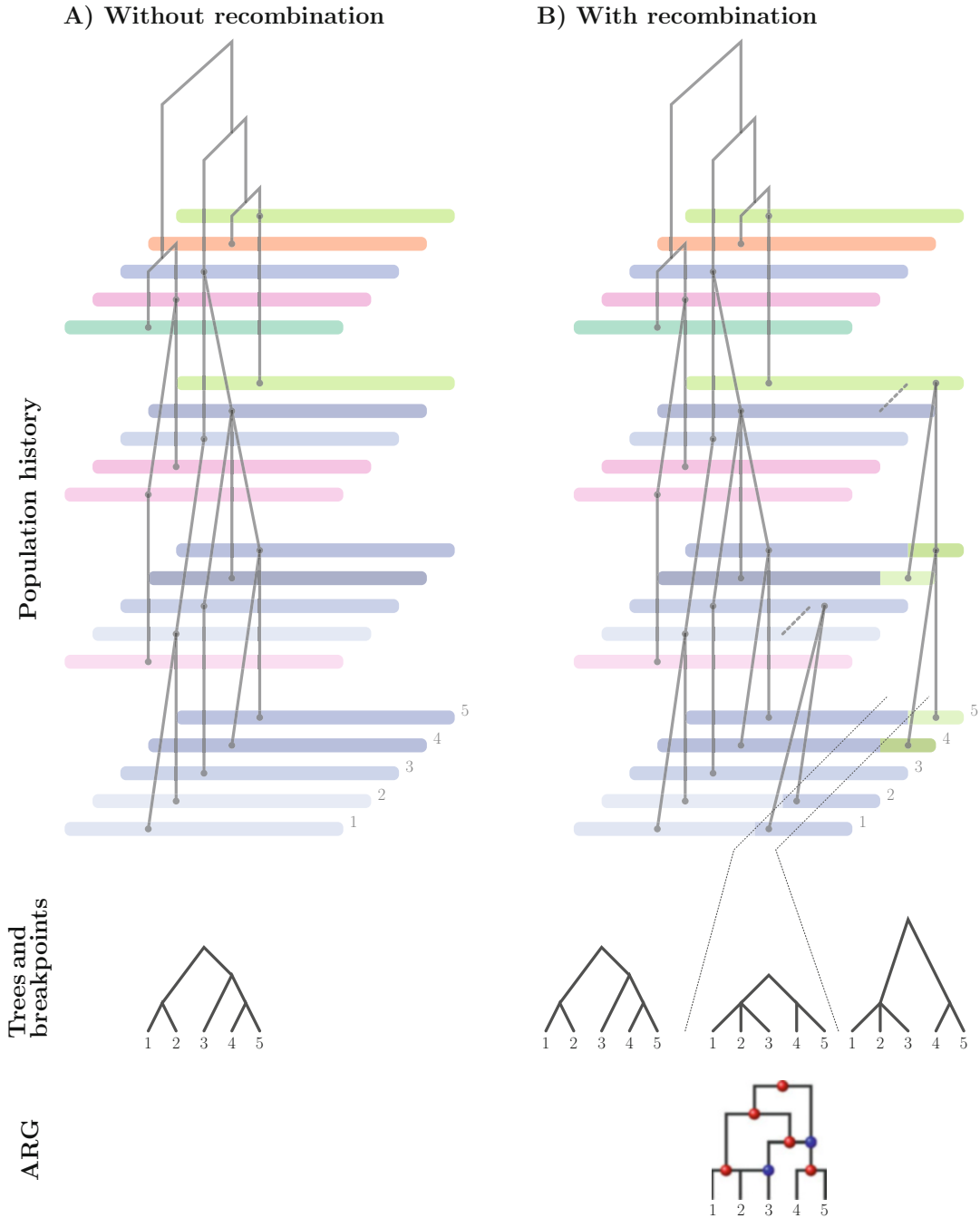


Fig. 1 The coalescent process along sequences. The trees depict the genealogical relationships between each haplotype in each generation. The initial population contains five haplotypes depicted with five distinct colors. The genealogies of these five haplotypes are shown as a cladogram. Parental relationships between each generation are indicated by *solid lines*. Mutation is indicated with tint changes. (**a**) Evolution without recombination. (**b**) Evolution with recombination. *Dashed lines* between haplotypes within a generation indicate crossing-over events

case: the material on the right of the first recombination point is ancestral, as it is found in extant haplotypes 4 and 5, whereas the left part of the haplotype is non-ancestral. As a result, when large regions with multiple recombination events are considered, all ancestral haplotypes are most likely a mosaic of ancestral and non-ancestral material, and recombination can occur in both. As Wiuf and Hein demonstrated, the spatial recombination process is not Markovian, because the genealogy at a given loci cannot be predicted based on the genealogy at the previous positions only. This property results from the fact that recombination events are not independent, inducing long-range dependencies between the local genealogies [12]. Such dependencies are the consequence of a particular class of recombination events that occurred in non-ancestral material flanked by ancestral material [9], also called *trapped* sequences. The process studied by McVean and Cardin [8] specifically ignores this class of recombination events. As a result, observed recombination events become independent, and the genealogy at a given loci can be generated from the genealogy at the previous position. While the resulting process is no longer equivalent to the coalescent with recombination, it has the property of being Markovian and was named as the sequentially Markovian coalescent (SMC, and SMC′ for the improved method of Marjoram and Wall [9]). McVean and Cardin demonstrated that many characteristics of sequence data sets generated by the SMC process are very similar to those of data sets generated under the "real" coalescent with recombination [8]. Interestingly, even estimates of the recombination rate are rather robust to the approximation, despite the fact that much less recombination events are accounted for in the SMC than in the coalescent with recombination. As a result, the SMC model approximation serves as the basis for many of the models currently in use in the field of population genomics.

2 The Pairwise Sequentially Markovian Coalescent (PSMC)

The SMC was initially used as a procedure to simulate genome sequences under an evolutionary model allowing for recombination. The first application of the SMC model for statistical inference was proposed by Li and Durbin [13] to model the evolution of a phased diploid genome (or equivalently, two haploid genomes), and was therefore named as the pairwise sequentially Markovian coalescent (PSMC). With two genomes, loci-specific trees are rather simple and only differ by their branch lengths, namely the time to the most recent common ancestor (TMRCA) of the two genomes. Li and Durbin therefore designed an HMM with hidden states corresponding to the distinct possible TMRCA, taken from their theoretical (discretized) distribution (Fig. 2a). Such a distribution is exponential, with a rate depending on the effective

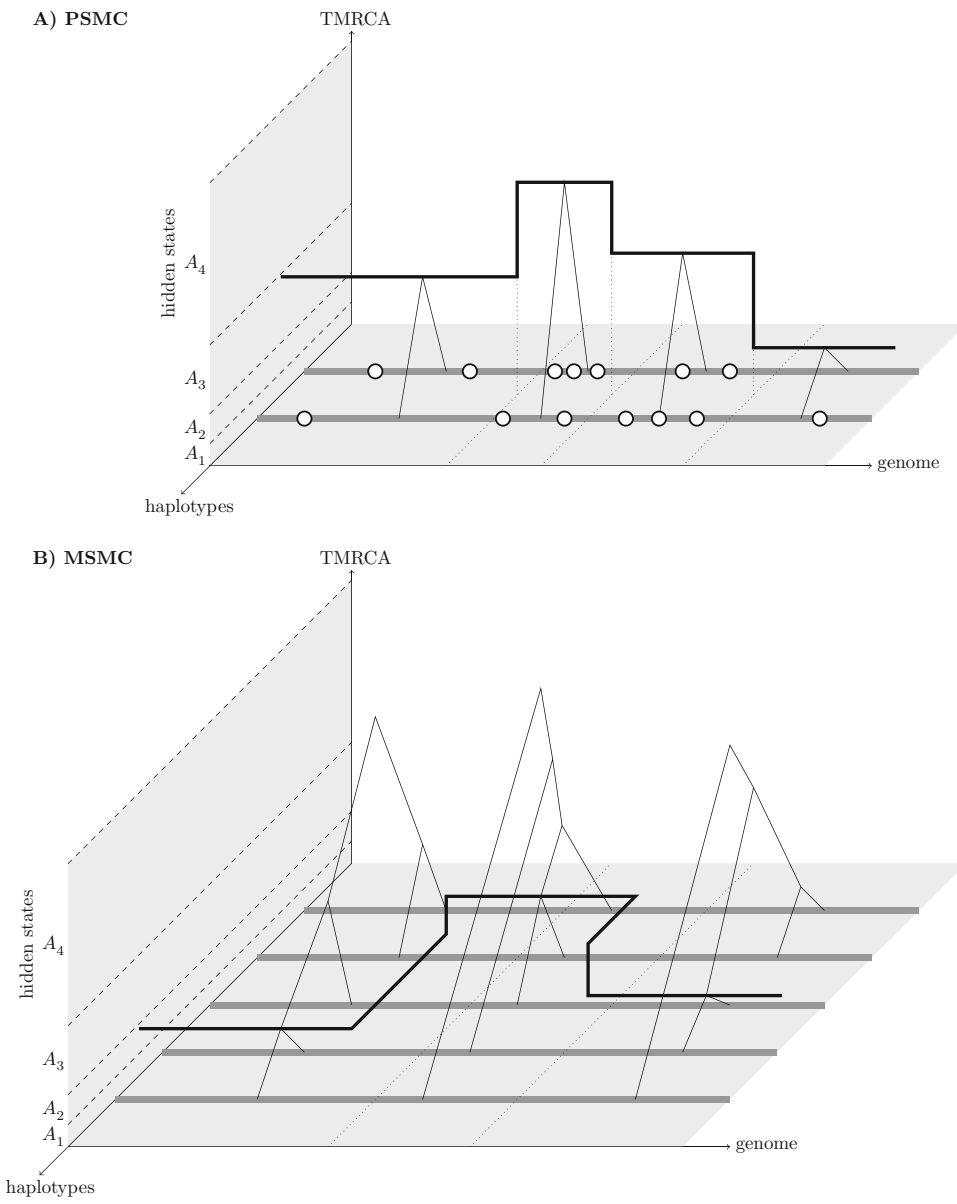


Fig. 2 The pairwise and multiple sequentially Markov coalescent, as implemented by Durbin and colleagues. (**a**) The pairwise SMC for two haplotypes. Hidden states correspond to the discretized local time to the most recent common ancestor (TMRCA). Recombination events leading to a change of hidden states are shown as *dotted lines. White circles* represent mutation events that occurred after divergence with the ancestor, figuring observed states. (**b**) The multiple SMC: as the local relationships between haplotypes can no longer be described by a single value, hidden states represent the minimum time to the most recent coalescence event among all pairs of haplotypes

population size. The authors however considered a more complex model in which the population size varies over time. In addition to the discretized TMRCA, they considered several epochs with distinct effective population sizes, in which the coalescence rate varies.

The parameters of this model therefore consist of the collection of effective population sizes, the mutation rate and the recombination rate.

Emission probabilities are computed for a given TMRCA under the infinite site model (that is, assuming that a given position cannot have more than one mutation). If all of the mutation events are considered as likely, the probability of observing a heterozygous site given a TMRCA equal to t is $1 - e^{-\theta.t}$ and the probability of observing a homozygous site is $e^{-\theta.t}$ (note: these quantities are erroneously inverted in the original publication [13]). More elaborate models can be used, for instance allowing distinct rates of transitions and transversions. Transition probabilities between hidden states can be computed under the sequentially Markov assumption. A change of hidden state along the genome alignment indicates that a recombination event occurred at that particular position. The TMRCA of the flanking material before and after that position are therefore different. The probability of changing the TMRCA from s to t can be computed analytically in the SMC model [13]. As discretized time is considered, all of the emission and transition probabilities are integrated over each time class. The time intervals can be computed in various ways, but a recommended design is to set them according to a logarithmic scale, with larger interval sizes in the distant past where less events can be inferred [12, 15]. The Markov chain is used to compute a likelihood function by integrating in an efficient manner over all possible HMM paths, that is, all possible ARGs. An optimization procedure is then used to perform maximum likelihood inference of parameters using this likelihood function. It is also possible to use the Viterbi algorithm to infer the genealogy at each position, and to compute the posterior probability of each hidden state at each position (posterior decoding). Posterior decoding can be used to infer the genealogy with the highest posterior probability and to compute posterior estimates of divergence times.

This approach has been used with success to reconstruct human demography [13]. However, because of the restricted amount of individuals under consideration, the PSMC lacks resolution for very recent times because the density of available recombination events is too low [13]. This limitation can be overcome by analyzing more individuals simultaneously.

3 The SMC for Multiple Genomes

While adding more genomes is a natural extension of pairwise methods, it is a challenging task due to the increasing complexity of the underlying hidden state space. A sample of only three genomes requires two MRCA and can be described by three possible topologies, depending on the order of coalescence events

(haplotypes 1 and 2, 1 and 3, or 2 and 3 can coalesce first). The timing of these events (TMRCA) can also differ for a given topology, making the hidden state space continuous. As a result, even in discretized time, the complexity of the hidden states space is no longer tractable, and further simplifications are required.

Schiffels and Durbin introduced an elegant extension of the PSMC to multiple genomes, namely the multiple sequentially Markovian coalescent (MSMC) [16]. They simplified the underlying hidden state space by considering the time to the first coalescence event only (Fig. 2b). As shown with simulations, this method successfully increases the resolution of demographic inference for recent times. When applied to human genomes, the statistical power was found to be maximal with eight genomes. When more genomes are added to the analysis, the time to the most recent coalescence event within n genomes typically becomes very small, and the length of segments in a given hidden state becomes very long (Fig. 3). At the time of writing this method has not been applied to nonhuman species. As the optimal number of genomes depends on species-specific parameters such as the recombination rate, it is most likely to differ in other species.

Another approach to handling the multiple genomes modeling issue was suggested by Paul and Song [17]. The authors propose to use composite likelihood to approximate the likelihood of the full sample. Let n be the number of haplotypes in the data set under consideration. The approximation consists of evaluating the likelihood of a single haplotype conditioned on one or several other haplotypes, under a given demographic model. This approach results in transforming the hidden state space of n haplotypes into a maximum of n independent spaces for one haplotype, or equivalently, converting a putative HMM for n haplotypes into n independent HMMs of reduced complexity, which can even be run in parallel. The computation of the conditional likelihood for one haplotype is based on the so-called conditional sampling distribution (CSD), originally described by Stephens and Donnelly [18–20]. The principle of the CSD consists of modeling the distribution of additional DNA sequences based on a given set of already observed sequences (see Box 1). A single haplotype HMM is built by singling out one haplotype of the n haplotypes available. The likelihood that it has been generated from a given demographic model, knowing that the $n-1$ other haplotypes have already been observed is then computed. The hidden state space of such single-haplotype HMMs resembles the one from the PSMC because it consists of a TMRCA (continuous) and the index of one of the $n-1$ haplotypes it shares the MRCA with. As in the previous approaches, the continuous variable TMRCA has to be discretized in order to be able to use the forward and backward algorithms to compute the likelihood and perform posterior decoding. Several approaches have been proposed to approximate the global

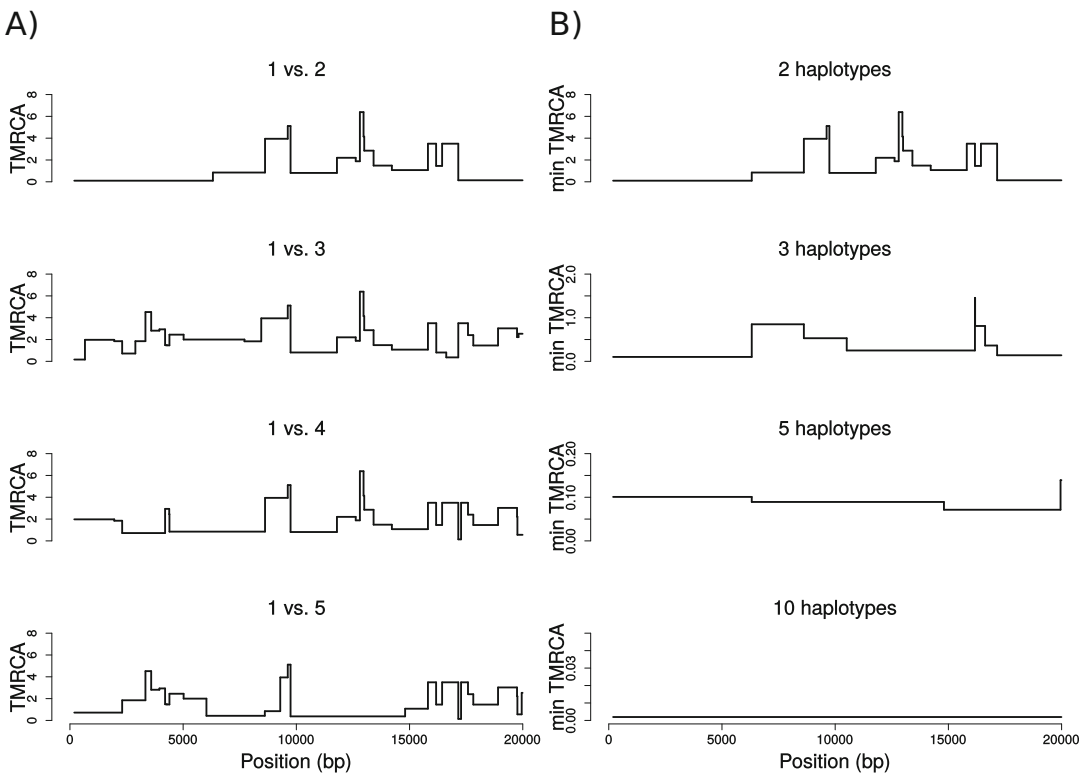


Fig. 3 The distribution of the time to the most recent common ancestor (TMRCA) along the genome. Haplotypes are generated using the MACS simulator and a recombination rate equal to 0.001 [39]. (**a**) The TMRCA between two haplotypes. The identity of haplotypes compared is shown on each graph. (**b**) Distribution of the minimum TMRCA when two, three, five, or ten haplotypes are considered. Note the difference in scale on the *y*-axis

likelihood from the likelihoods of the single-haplotype HMM [15]. A method that was reported to work well for several models is the so-called product of approximate conditionals, based on the following formula:

$$\begin{aligned} Pr\left(h_1, h_2, \ldots, h_n \vee \Theta\right) = & Pr(h_1 \vee \Theta) \times Pr(h_2 \vee h_1; \Theta) \\ & \times Pr(h_3 \vee h_1, h_2; \Theta) \times \cdots \\ & \times Pr(h_n \vee h_1, \ldots, h_{n-1}; \Theta) \end{aligned}$$

where $h_{1\ldots n}$ represents the n haplotypes and Θ is the set of model parameters. As the individual likelihoods are approximated by the CSD, the global likelihood computed by this method depends on the order in which the haplotypes are considered. To remove this effect, the final likelihood is further averaged over several permutations of the haplotypes [15, 19].

The first CSD-based method was implemented in diCal software, with a model allowing the effective population size to vary across time [21]. This approach was also used to study the recent human demography [21, 22].

A third approach named ARGweaver and based on a similar principle as the CSD was proposed by Rasmussen et al. [12]. The authors modeled the ARG of n haplotypes by conditioning it on the ARG of $n-1$ haplotypes, an operation they referred to as "threading". As opposed to CSD-based approaches, the threading approach explicitly models the local genealogy of the haplotypes. The unknown genealogy of the "observed" n haplotypes is sampled from its posterior distribution via a Markov chain Monte Carlo (MCMC) sampler. While the use of an MCMC sampler is computationally more demanding than direct likelihood inference, the ARGweaver method permits direct inference of the ARG and enables the study of properties that are not explicitly modeled. This approach was used for instance to infer footprints of natural selection, visible as local distortions of the ARG [12].

4 More Complex Demographic Scenarios

As more haplotypes are considered, more complex demographic scenarios can be modeled. A desired extension is to allow population structure in the form of two or more subpopulations (also known as "demes") that can exchange migrants. Such structured population models have been implemented in both the MSMC [16] and diCal methods [15]. Briefly, these models account for "migration" events, which occur in addition to "coalescence", "recombination" and "absorption" (for the CSD-based method diCal) events. This adds another level of complexity to the hidden states space because one has to keep track of the deme to which the absorbing haplotype belongs. In contrast with simpler demographic models where the SMC has proved to be a good working approximation to the full coalescent process, differences in the correlation structure of genealogies along the genome have been reported for models with strong population structure (i.e., with low migration rates) [25]. It is not clear however to what extent this matters for the estimation of model parameters such as migration rates [15].

5 Multiple-Species Analyses

Another family of coalescent HMMs has been developed independently of the SMC, although it is based on similar approximations. These models are used in a somewhat distinct context because they model the ancestry of closely related species. The first model was proposed by Hobolth et al. [2] and extended by Dutheil et al. [3]. The approach aims at modeling the relationships between three species and can be seen as a particular case of structured population with no migration: the ancestral species splits in two

subpopulations which evolve independently, one splitting again so that three independent subpopulations (= species) evolve until the present time. The unique aspect of the approach of Hobolth et al. is that only one haplotype is modeled per subpopulation. In addition, given the speciation times involved, it is generally assumed that all extant haplotypes in one species have an MRCA much more recent than the common ancestor of two distinct species. Whilst this is not a requirement of the model, it permits the use of unphased or composite genomes, such as the reference human genome. Let $\phi = \big((s_1, s_2), s_3\big)$ be the phylogeny of the three species s_1, s_2, and s_3, and let s_{12} and s_{123} denote the ancestral species of (s_1, s_2) and (s_1, s_2, s_3), respectively. Under the above mentioned conditions, four types of marginal genealogies can be distinguished (Fig. 4) [26]:

- s_1 and s_2 have an MRCA in the ancestral species s_{12}, in which case the genealogy is congruent with the phylogeny
- s_1 and s_2 have a MRCA in the ancestral species s_{123}, implying that the two lineages persisted in the form of ancestral polymorphism in species s_{12}. The order of coalescence events within the s_{123} ancestral species follows the three-haplotype rule, with three equilikely possibilities that are s_1 and s_2, s_1 and s_3 or s_2 and s_3 coalesce first. In the two latter cases, the resulting genealogy is incongruent with the phylogeny, a phenomenon referred to as incomplete lineage sorting (ILS) [2, 3, 26].

These four types define four hidden states of a HMM whose parameters are the recombination rate, two speciation times, and two ancestral effective population sizes. Emission probabilities can be computed using the standard Felsenstein recursion on a tree and the corresponding genealogy and a model of nucleotide substitutions [14, 27]. The branch lengths of each genealogy depicted in Fig. 4 are determined by the demographic scenario and can be computed using standard coalescence theory [2, 3]. The transition

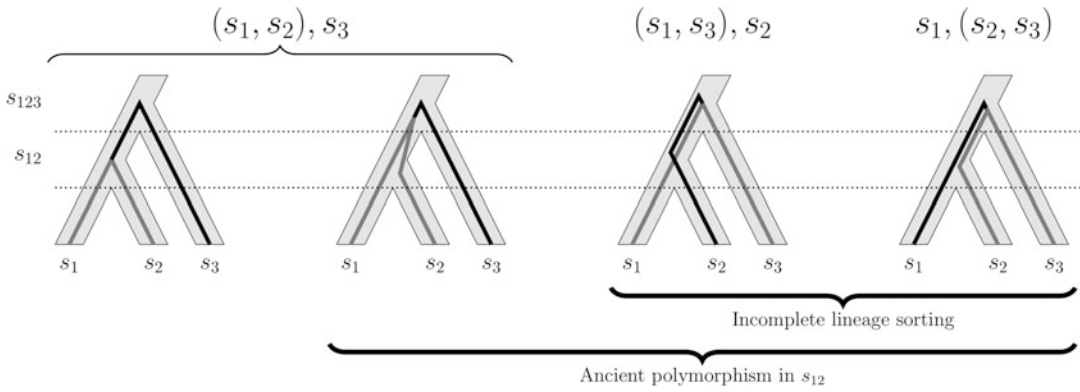


Fig. 4 Four categories of genealogies accounted for by the CoalHMM model for three species s_1, s_2, and s_3

probabilities are more cumbersome to compute, but analytical solutions have been proposed for the most simple models [3], and automatic procedures exist for more complex models [28, 29]. Just as for the PSMC, MSMC, and CSD-based methods, the underlying marginal genealogies form a continuous state space because of the time component (the branch lengths in the marginal genealogy). For the CoalHMM method, this distribution is "discretized" in one category only so that only the mean of the distribution is considered. This led to a bias in the estimation of some of the population parameters, which can potentially be reduced by extending the model to more hidden states, by increasing the number of classes in the discretization of the TMRCAs [3]. In addition, in the original method, the emission and transition probabilities were not integrated over the full distribution, but based on the mean only.

A two-species version of the CoalHMM model has been designed by Mailund et al. [30] and applied to two individual genomes from closely related species. In this model, the continuous state space is discretized in a given number of categories corresponding to distinct TMRCAs for the two species. It is very similar in principle to the PSMC, but it differs by adding a speciation time parameter, after which the two species evolved independently in the absence of genetic exchange. As a result, the distribution of the TMRCA for the two species is shifted by a quantity equal to the speciation time. This model has three population parameters, a single ancestral and two extant effective population sizes. The model was later extended to account for migration between diverging populations [28].

6 Specific Challenges to HMMs in Population Genomics

6.1 Genome Architecture

Typical HMMs in biology are based on a DNA/protein sequence as observed states. When used in the context of comparative sequence analysis, the observed states are a column (= site) of homologous positions in a sequence alignment (see for instance [31, 32]). Contrary to single gene alignments, full genome alignments are only locally ordered. Because of genomic rearrangements such as chromosome fission or fusion or DNA inversion or translocation, two genomes might not be collinear (a property called *synteny*, with rupture in collinearity referred to as *synteny breaks*). Breaks of synteny are expected to be more frequent as more distantly related sequences are compared. At the intra-specific level, synteny breaks are typically ignored because they are presumably rare, and because single nucleotide polymorphisms (SNPs) are identified by mapping sequences to a reference genome, preventing the proper assessment of synteny conservation. As sequencing techniques improve, de novo genome assembly will become more common practice and

synteny will be more thoroughly assessed and accounted for. When more distant genomes are compared, for instance at the interspecific level, breaks in synteny become more frequent. While it is possible to order the genome alignment according to a reference species and ignore synteny breaks, a better alternative is to reset the hidden Markov chain at break points [33–35].

Genome-wide HMMs involve very long sequences of observed states, from millions to billions of positions. Estimating likelihoods for such large data sets requires not only large amounts of computational time but also prohibitive amounts of memory, particularly when posterior decoding is to be performed. Genome architecture can be used to split data sets in a biologically relevant manner, per chromosome or even per contiguous regions on chromosomes. In addition to permitting independent parallel optimization of the resulting sub-data sets, this approach allows the independent estimation of parameters along the genome. Parameters such as the recombination rate are known to vary in a spatial manner, but such variation in not accounted for by current models. Although current models have shown good robustness properties regarding the heterogeneity of such parameters [16], having spatially independent estimates has proved to be informative [33, 34, 36].

6.2 Optimization of HMM Implementation

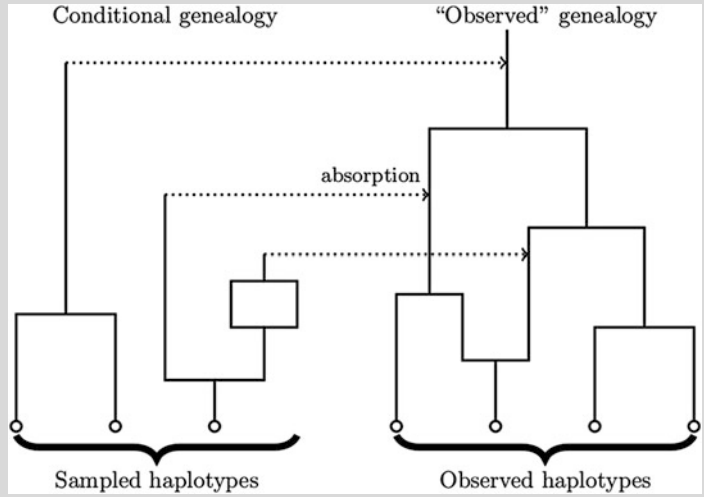
The computation of the likelihood for an HMM has a linear complexity in the length of the observed sequence l, but quadratic in the number of hidden states considered d. As discussed previously, d typically grows super-exponentially with the number of modeled haplotypes. While all models employ approximations to reduce d, it can remain rather high. As l is also very large, the execution time of the forward algorithm is a limiting step for all population genomic HMMs. Improving the efficiency of the likelihood calculation is therefore the target of dedicated research. Because the transition probabilities between hidden states result from coalescence theory, some symmetry in the process can be exploited to reduce the complexity up to linear in number of hidden states [2, 4]. Furthermore, the observed data set is typically highly repetitive, and some calculations must be redone many times. A preprocessing of the data can therefore be used to save on computational time [37].

Population genomics models are based on the coalescence theory. In contrast with traditional HMMs, an evolutionary (demographic) model is first developed, and subsequently used to derive expressions for the emission and transition probabilities. This has two consequences: first, the hidden states of the model are dependent on the demographic parameters and second, the transition probabilities are not explicit parameters of the model. The first point means that no "training" data set exist for population genomic HMMs and that likelihood optimization procedures are

required to infer model parameters. The second point implies that the computation of the expected parameter values from the observed transition and emission probability are not straightforward (the expectation step in the Baum-Welch expectation-maximization procedure [38]). Dedicated estimation procedures have therefore been developed to fit HMMs in population genomics [3, 16, 21].

Box 1: Conditional Sampling Distribution

The conditional sampling distribution (CSD) models the distribution of additional sequence sampled from a given demographic model, conditioned on a set of already observed sequences [15]. The CSD uses coalescence theory to model the genealogy of the observed sequence set, as well as how the new sampled sequences branch on this genealogy:



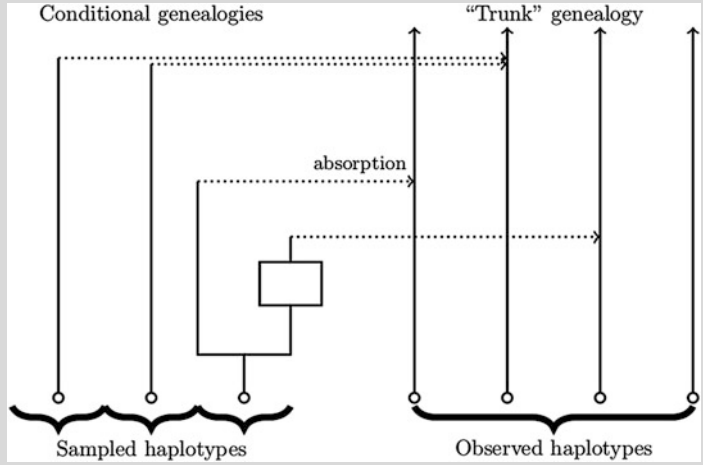
In this form of the CSD, a lineage coming backward in time can undergo distinct types of event:

- coalescence within the observed genealogy;
- recombination within the observed genealogy;
- coalescence within the sampled genealogy;
- recombination within the sampled genealogy;
- coalescence between the sampled and the observed genealogy, also referred to as an "absorption" event [23].

Modeling the full set of such events is not manageable, given that the genealogy of the observed sampled is unknown, and the MRCA of the sampled set of sequences might be older than the MRCA of the observed set. Approximations are therefore required [23, 24]:

(continued)

- The diffusion approximation: the genealogy of the observed haplotypes is modeled by a so-called "trunk" genealogy, where the lineages of the n observed haplotypes go infinitely back in time without coalescing or recombining [23].
- The serially Markov assumption: coalescence events in the conditional genealogy within the sampled haplotypes are ignored. As a result, a set of m haplotypes can be obtained by sampling m haplotypes independently, and the distribution of marginal genealogies of the sampled and observed sequence becomes Markovian along the genome [24].



Using these approximations, single haplotypes conditional genealogies can be formulated as an HMM in which the hidden states correspond to the time to lineage absorption and the observed absorption haplotype, and the changes between the hidden states correspond to a recombination event [24].

Acknowledgments

The author would like to thank Asger Hobolth for discussing the SMC model, and Yun Song for clarifying some aspects related to the implementation of the CSD. This publication is the contribution no. 2015-048 of the Institut des Sciences de l'Évolution de Montpellier (ISE-M).

References

1. 1000 Genomes Project Consortium, Abecasis GR, Auton A et al (2012) An integrated map of genetic variation from 1,092 human genomes. Nature 491:56–65
2. Hobolth A, Christensen OF, Mailund T et al (2007) Genomic relationships and speciation times of human, chimpanzee, and gorilla inferred from a coalescent hidden Markov model. PLoS Genet 3:e7

3. Dutheil JY, Ganapathy G, Hobolth A et al (2009) Ancestral population genomics: the coalescent hidden Markov model approach. Genetics 183:259–274
4. Harris K, Sheehan S, Kamm JA et al (2014) Decoding coalescent hidden Markov models in linear time. Res Comput Mol Biol 8394:100–114
5. Hein J, Schierup MH, Wiuf C (2005) Gene genealogies, variation and evolution: a primer in coalescent theory. Oxford University Press, Oxford
6. Wakeley J (2008) Coalescent theory: an introduction. Roberts and Company Publishers, Bloxham, Reading, PA
7. Hudson RR (1991) Gene genealogies and the coalescent process. Oxford Surv Evol Biol 7:1–44
8. McVean GAT, Cardin NJ (2005) Approximating the coalescent with recombination. Philos Trans R Soc Lon B Biol Sci 360: 1387–1393
9. Marjoram P, Wall JD (2006) Fast "coalescent" simulation. BMC Genet 7:16
10. Wiuf C, Hein J (1999) Recombination as a point process along sequences. Theor Popul Biol 55:248–259
11. Hobolth A, Jensen JL (2014) Markovian approximation to the finite loci coalescent with recombination along multiple sequences. Theor Popul Biol 98:48–58
12. Rasmussen MD, Hubisz MJ, Gronau I et al (2014) Genome-wide inference of ancestral recombination graphs. PLoS Genet 10: e1004342
13. Li H, Durbin R (2011) Inference of human population history from individual whole-genome sequences. Nature 475:493–496
14. Yang Z (2006) Computational molecular evolution. Oxford University Press, Oxford
15. Steinrücken M, Paul JS, Song YS (2013) A sequentially Markov conditional sampling distribution for structured populations with migration and recombination. Theor Popul Biol 87:51–61
16. Schiffels S, Durbin R (2014) Inferring human population size and separation history from multiple genome sequences. Nat Genet 46:919–925
17. Paul JS, Song YS (2012) Blockwise HMM computation for large-scale population genomic inference. Bioinformatics 28:2008–2015
18. Stephens M, Donnelly P (2000) Inference in molecular population genetics. J R Stat Soc Series B Stat Methodology 62:605–635
19. Li N, Stephens M (2003) Modeling linkage disequilibrium and identifying recombination hotspots using single-nucleotide polymorphism data. Genetics 165:2213–2233
20. Fearnhead P, Donnelly P (2001) Estimating recombination rates from population genetic data. Genetics 159:1299–1318
21. Sheehan S, Harris K, Song YS (2013) Estimating variable effective population sizes from multiple genomes: a sequentially markov conditional sampling distribution approach. Genetics 194:647–662
22. Raghavan M, Steinrücken M, Harris K et al (2015) Genomic evidence for the Pleistocene and recent population history of Native Americans. Science 349:3884
23. Paul JS, Song YS (2010) A principled approach to deriving approximate conditional sampling distributions in population genetics models with recombination. Genetics 186:321–338
24. Paul JS, Steinrücken M, Song YS (2011) An accurate sequentially Markov conditional sampling distribution for the coalescent with recombination. Genetics 187:1115–1128
25. Eriksson A, Mahjani B, Mehlig B (2009) Sequential Markov coalescent algorithms for population models with demographic structure. Theor Popul Biol 76:84–91
26. Dutheil JY, Hobolth A (2012) Ancestral population genomics. Methods Mol Biol 856:293–313
27. Felsenstein J (2003) Inferring phylogenies. Sinauer Associates, Sunderland, MA
28. Mailund T, Halager AE, Westergaard M et al (2012) A new isolation with migration model along complete genomes infers very different divergence processes among closely related great ape species. PLoS Genet 8:e1003125
29. Mailund T, Halager AE, Westergaard M (2012) Using colored petri nets to construct coalescent hidden markov models: automatic translation from demographic specifications to efficient inference methods. In: Haddad S, Pomello L (eds) Application and theory of petri nets. Springer, Berlin, Heidelberg, pp 32–50
30. Mailund T, Dutheil JY, Hobolth A et al (2011) Estimating divergence time and ancestral effective population size of Bornean and Sumatran orangutan subspecies using a coalescent hidden Markov model. PLoS Genet 7:e1001319
31. Felsenstein J, Churchill GA (1996) A Hidden Markov Model approach to variation among sites in rate of evolution. Mol Biol Evol 13:93–104

32. Goldman N, Thorne JL, Jones DT (1996) Using evolutionary trees in protein secondary structure prediction and other comparative sequence analyses. J Mol Biol 263:196–208
33. Locke DP, Hillier LW, Warren WC et al (2011) Comparative and demographic analysis of orang-utan genomes. Nature 469:529–533
34. Scally A, Dutheil JY, Hillier LW et al (2012) Insights into hominid evolution from the gorilla genome sequence. Nature 483:169–175
35. Prüfer K, Munch K, Hellmann I et al (2012) The bonobo genome compared with the chimpanzee and human genomes. Nature 486:527–531
36. Stukenbrock EH, Bataillon T, Dutheil JY et al (2011) The making of a new pathogen: insights from comparative population genomics of the domesticated wheat pathogen Mycosphaerella graminicola and its wild sister species. Genome Res 21:2157–2166
37. Sand A, Kristiansen M, Pedersen CNS et al (2013) zipHMMlib: a highly optimised HMM library exploiting repetitions in the input to speed up the forward algorithm. BMC Bioinformatics 14:339
38. Durbin R, Eddy SR, Krogh A et al (1998) Biological sequence analysis: probabilistic models of proteins and nucleic acids. Cambridge University Press, Cambridge
39. Chen GK, Marjoram P, Wall JD (2009) Fast and flexible simulation of DNA sequence data. Genome Res 19:136–142

Chapter 12

Differential Gene Expression (DEX) and Alternative Splicing Events (ASE) for Temporal Dynamic Processes Using HMMs and Hierarchical Bayesian Modeling Approaches

Sunghee Oh* and Seongho Song*

Abstract

In gene expression profile, data analysis pipeline is categorized into four levels, major downstream tasks, i.e., (1) identification of differential expression; (2) clustering co-expression patterns; (3) classification of subtypes of samples; and (4) detection of genetic regulatory networks, are performed posterior to preprocessing procedure such as normalization techniques. To be more specific, temporal dynamic gene expression data has its inherent feature, namely, two neighboring time points (previous and current state) are highly correlated with each other, compared to static expression data which samples are assumed as independent individuals. In this chapter, we demonstrate how HMMs and hierarchical Bayesian modeling methods capture the horizontal time dependency structures in time series expression profiles by focusing on the identification of differential expression. In addition, those differential expression genes and transcript variant isoforms over time detected in core prerequisite steps can be generally further applied in detection of genetic regulatory networks to comprehensively uncover dynamic repertoires in the aspects of system biology as the coupled framework.

Key words Differential expression, Alternative splicing event, Genetic regulatory network, Time series expression profile, Temporal dynamics, HMMs and hierarchical Bayesian modeling

1 Introduction

In the past few years, sequencing technology has triggered the advanced powerful methodologies to apply for biological expression data profile by rapidly replacing from array-based platforms. Profiling gene expression data has been conducted as a commonplace to identify differential expression and to uncover its underlying mechanisms on genetic regulation between tissues/conditions and on disease progression. In general, genes that change expression between conditions are of utmost interest and

*Author contributed equally with all other contributors.

David R. Westhead and M.S. Vijayabaskar (eds.), *Hidden Markov Models: Methods and Protocols*, Methods in Molecular Biology, vol. 1552, DOI 10.1007/978-1-4939-6753-7_12, © Springer Science+Business Media LLC 2017

hence differential expression is a prerequisite step to unravel unknown interactions with cis-regulatory elements. Dynamic processes are basically studied to explore their classification of temporal patterns either altered expression patterns in single transient time course experimental settings or stimulated by extended factors at each time point in multifactorial time course experimental designs. Thanks to the advances of techniques and reduced costs to sequencing, in RNA-Seq, well-balanced experimental designs with appropriate numbers of replicates and time points have been recently feasible and would be more popularly performed in the near future in developmental biology, small and moderate size of stimulated disease progression data and large-scale of repeated clinical longitudinal datasets. Compared to static data, where all different samples are independent, the most remarkable feature in dynamic processes of time series expression profiles is that two consecutive profiles are highly correlated and the current level is affected by previous expression level as a dependent structure in horizontal chain.

Differential expression has been predominantly studied in order to identify candidates of biomarkers to play a key role on changes of gene expression under given distinct tissues and conditions, or over a series of time points. Somehow routinely, differential expression methods in RNA-seq are employed for static data in which all samples are independently collected, e.g., Fisher exact test, edgeR, DEGSeq, DESeq, and bayseq [1–6]. As extended structures, multi-group comparisons are allowed to perform in latest version of edgeR and DESeq using generalized linear models on the basis of Poisson and negative binomial distribution.

More recently, specific time course experimental designs in RNA-Seq have been exploited in different two types of data, single series of time course which only includes time factor in the model and multi-series of time course which contain other experimental and biological conditions (e.g., stress, trauma, drug treatment, different tissues or cell lines) at a time. Dynamic gene expression profiles are of high interest to aim at exploration of cellular and molecular processes during the transcriptional and posttranscriptional regulation at system levels and structural modifications. In particular, temporally differentially expressed genes could be further investigated into subsequent analyses whether they are activated coupled with upstream region's regulatory elements such as precursors, enhancers, transcriptional factors (TFs), and epigenetic factors to target genes or are irresponsibly altered changes with those factors. In addition, they are further classified into groups of gene expression patterns based on temporally differential expression (TDEX) to characterize various co-expression patterns in groups. Analogous expression patterns are grouped together in time-varying patterns in single series of time course data and induced time-varying patterns by external stimuli across a series of

time points. It is not necessarily that co-expression patterns frequently preserve co-regulatory functionality and groups which are clustered together in co-expression patterns may have different functional pathways and regulatory gene sets, albeit exceptional cases are often shown as well in the data [7–11].

It is therefore one of the most crucial parts in downstream analyses of large scale expression analyses to find out reliable detection of differential expression and further identification of corresponding regulatory modules by making use of relational structures on TDEX genes and network element sets. Hereafter, we refer differential expression as DEX in static data and temporally differential expression as TDEX in dynamic data and equal expression as EEX for both data. The necessity of dynamic methods to take into account explicitly the intricate and inherent nature of time course data structure has not been thoroughly appreciated thus far and little has been known how to properly analyze temporal dynamic expression and interpret it.

In the beginning, a few of the research groups in next generation sequencing community have proposed differential analysis methods which are originally implemented for static data. As an alternative, those have been utilized, though time dependency feature is not explicitly considered for time course RNA-seq. As endeavors of investigators to develop methods continued, dynamic specific methods have been gradually proposed in the following methods: AR (autoregressive model), adjusted HMM (Hidden Markov Models) for RNA-seq, and Next magsigpro to better understand correlated dependent structure between neighboring two time points [12–20]. Along with differential expression at gene level, one of the attractive natures of RNA-Seq transcriptome makes it possible to define alternative splicing events that undergo the variety of outcomes through selection of exons of a given gene to generate different protein structures and also their own functionalities. As it has been shown that more than 95 % of human genes have transcript variant isoform structures and for an extreme example, it is spliced with more than around 1000 isoforms for one gene, implicating that gene level specific analysis in differential expression and regulatory network modeling might be limited and lead to partially detected temporal patterns in complicated transcriptome data [21–23].

In this chapter we describe hidden markov models for dynamic processes in Seq data, by focusing on multifactorial time course where samples are consecutively collected over time, and other experimental multi-factors are contained for each time point at gene level quantification. We next demonstrate the framework of temporally differential expression for alternative splicing events using hierarchical Bayesian techniques that might be more specifically the clues of transcriptional regulatory networks to represent time (and/or condition)-specific isoforms that are not comprehensively detected at gene levels in general.

2 Methods

2.1 HMM (Hidden Markov Models) in Identification of Differential Expression Analysis for Genes

2.1.1 Time Course Expression Profile

A gene expression profile matrix in single series of time course contains $g = \{1, \ldots, G\}$ genes, $g = \{1, \ldots, R\}$ replicates, $t = \{1, \ldots, T\}$ different time stages and for sake of simplicity, multi-series of time course profile data additionally contains a condition $c = \{1, \ldots, C\}$ at each time point $t = T$on the structure. The gth gene expression profile vector, $\Upsilon_g = \left[y_g(t_1), \ldots, y_g(t_T)\right]^t$ corresponds to a sequential vector of T time points. And biological replicates for each time point and a particular given condition such as external stimulus, $y_g(T = t_{TC}) = \left[y_{gt,c=1,r=1,\ldots}, y_{gt,c=1,r=R}\right]$ is composed of intra-expression measurements by R replicates. Raw mapped read counts for each transcript in RNA-seq experiment quantified against reference genome correspond to Υ_g individual sequential vector over time period. To fully decipher count property and horizontal dependency structure, hidden markov models which are well fitted and can be applied for wide range of sequencing specific data, e.g., RNA-seq data and other types of 'omic count data are demonstrated in this section. With best selected conjugate priors, Poisson (when there are no replicates at each condition and a time point) and negative binomial distribution (when replicates are available) are proposed, respectively. In order to describe a detailed procedure of HMM for both distributions, for the sake of convenience and consistency, in the simplest example, more than two intra-conditions measured at each time point are considered to represent transcriptional stimulus-response data induced by transcriptional response as a role of suppressor or enhancer such as disease progression over time with external factors [24]. Suppose that there are multiple biological replicates for each condition group. These biological individuals should enable to evaluate reproducibility and variability for significant difference on replicate effect before getting into major inference tasks on HMMs in downstream differential expression analysis and regulatory network modules. Given observed expression read counts and underlying parametric models, hidden states to be inferred are either equally expressed or temporally differential expression amongst distinct conditions. The goal of utilizing HMMs in temporal dynamic processes is to fundamentally estimate hidden states in the content of all possible cases from distinct conditions [12–20]. Briefly, if there are only given two biological or experimental conditions, we ought to infer and compare the posterior probabilities of corresponding two expected states, either EEX (equal expression) for state I or TDEX (temporally differential expression) for state II from the given two distinct conditions. As an extended experimental design, if there are four different conditions, then all possible candidate states are incremented upto 15 by the bell number of rules as given below,

For the multiple replicates within a group per condition, assume that we take a median or mean value appropriately.

State 1 $[1111] = \mu_{gTc=1} = \mu_{gTc=2} = \mu_{gTc=3} = \mu_{gTc=4}$

...

State 12 $[1123] = \mu_{gTc=1} = \mu_{gTc=2} \neq \mu_{gTc=3} \neq \mu_{gTc=4}$

State 13 $[1213] = \mu_{gTc=1} = \mu_{gTc=3} \neq \mu_{gTc=2} \neq \mu_{gTc=4}$

State 14 $[1231] = \mu_{gTc=1} = \mu_{gTc=4} \neq \mu_{gTc=2} \neq \mu_{gTc=3}$

State 15 $[1234] = \mu_{gTc=1} = \mu_{gTc=2} \neq \mu_{gTc=3} \neq \mu_{gTc=4}$

in which these states are not observed beforehand and should be inferred by underlying HMMs. We leave a simpler example when having three distinct conditions at a time point as a practice by yourselves here. To address the various biological questions by preserving RNA-Seq count property on expression level and inherent dependency characteristic on time X-axis, we first estimate the probability of each state $\pi_{gt}^{k} = \Pr(S_{gt} = k)$, $k = 1$ to 15. And the most likely path of sequential states over time is based on Viterbi-Algorithm an extension of EM algorithm as shown in microarray studies. This modeling approach has the constraint to first order markovian chain dynamic process, in which the current state is only affected by the right previous state. It is good enough so as to characterize RNA-Seq multi-series of time course, where there are relatively small and moderate numbers of time points. Of course, the higher order of markovian chain property can be employed straightforwardly when having the ample size of time points such as a large-scale of longitudinal repeated measurements for Seq dynamic processes. To preserve elegant Seq-specific count property on expression levels, we demonstrate two types of parametric distributions poisson with conjugate gamma prior (PG) and negative binomial with beta prior (NBB) when there are no replicates or multiple replicates available, respectively. The underlying distributions and joint predictive density (JPD) are incorporated to derive posterior probability of states. The detailed notations for each model are given in the following:

For Poisson–Gamma (PG) model and Negative Binomial Beta (NBB) distribution,

PG model:

$$\frac{\nu^{\alpha}}{\Gamma(\alpha_0)\prod_{i=1}^{n} z_i!}\int_0^{\infty}\mu^{(\Sigma z_i+\alpha_0)-1}\exp(-\mu(n+\nu))\,d\mu$$

Suppose $\mu' = \mu(n+\nu), d\mu' = d\mu(n+\nu)$

$$\frac{\nu^{\alpha}}{\Gamma(\alpha_0)\prod_{i=1}^{n} z_i!}\int_0^{\infty}\mu^{(\Sigma z_i+\alpha_0)-1}\exp(-\mu(n+\nu))\,d\mu$$

$$= \frac{\nu^{\alpha}}{\Gamma(\alpha_0)\prod_{i=1}^{n} z_i!}\int_0^{\infty}\frac{\mu'}{n+\nu}^{(\Sigma z_i+\alpha_0)-1}\exp\{-(\mu')\}\,d\mu'(n+\nu)$$

$$= \frac{\nu^{\alpha}}{\Gamma(\alpha_0)\prod_{i=1}^{n} z_i!}\,\frac{1}{n+\nu^{(\Sigma z_i+\alpha_0)-1}}\int_0^{\infty}(\mu')^{(\Sigma z_i+\alpha_0)-1}\exp\{-(\mu')\}\,d\mu'$$

$$= \frac{\nu^{\alpha}}{\Gamma(\alpha_0)\prod_{i=1}^{n} z_i!}\,\frac{1}{n+\nu^{(\Sigma z_i+\alpha_0)}}\Gamma^{(\Sigma z_i+\alpha_0)}$$

NBB model:

$$\int_0^1\prod_{i=1}^{n}\binom{z_i+k-1}{k-1}\frac{1}{B(\alpha,\beta)}\int_0^1 p^{nk+\alpha-1}(1-p)^{z_i-\beta-1}\,dp$$

$$= \prod_{i=1}^{n}\binom{z_i+k-1}{k-1}\frac{B(nk+\alpha,\sum z_i+\beta)}{\beta(\alpha,\beta)}$$

Thus, one of the important advantages using of HMMs is able to estimate posterior probabilities of given states, since a gene shows either differential expression or equal expression between distinct conditions, though it does not necessarily mean two conditions are perfectly distinguished on a mixture form of distributions.

Under EEX State I,

$f_{gT}^{1}(x_{gt}) = f_0^{1}(x_{gt}|\lambda_{gt})\,dG_t(\lambda_{gt})$,

where f^0 gives all sample are colleced from an identical distribution,

$f^0\left(\cdot|\mu_{gtc} = \mu_{gtc=1} = \mu_{gtc=2}\right)$

Under TDEX State II,

$$f_{gT}^{2}(x_{gt}) = \int f_t^0(x_{gt(1,\ldots,n1)}|\lambda_{gt1})\,dG_{t1}(\lambda_{gt1}) + \int f_t^0\left(x_{gt\left((n1+1),\ldots,(n1+2)\right)}|\lambda_{gt2}\right)dG_{t2}(\lambda_{gt2})$$

where 1 to $n1$ *are* i.i. d samples from $f_t^0(\cdot|\mu = \lambda_{gt1})$

and the rest of samples (n_1+1) to (n_1+n_2) are collected from $f_t^0(\cdot|\mu = \lambda_{gt2})$

The entire mixture distribution to particular gene of interest is given by marginal distributions of two states,

$(1-\phi)f^1_{gt}\left(y_{gt}\right)+\phi f^2_{gt}\left(y_{gt}\right)$, where ϕ represents the proprotion of TDEX genes,

$g=1,\ldots,\ G$ genes

$t=1,\ldots,\ T$ genes

$$f^0_{gt}(y|\lambda_{gt})=\frac{\lambda_{gt}e^{-\lambda_{gt},y}}{y!},\ \ y>0$$

As an adoption of conjugate prior on PG model, $\Theta\sim_{gt}=\left(\lambda_{gt},\alpha_{gt},\beta_{gt}\right)$ needs to be estimated in prior distribution with shape and rate parameter α_{gt}, β_{gt}. Owing to dramatically reduced costs for sequencing and the increasing need for temporal dynamic experimental studies in a wide rage of genetic and genomic areas, well-balanced experimental designs in which the optimal number of time points and replicates for each time stage are contained, have become feasible and more widely adopted in the wet-laboratories and clinical applications as microarrays. As an alternative, RNA-Seq experiments followed by PG model, negative binomial or over-dispersed poisson distribution has been proposed in classical parametric distribution settings. As presented, for Bayesian settings, negative binomial beta prior can be applied when having replicates within a group.

For given parametric assumptions, we restrict ourselves to the homogeneous HMMs in which the transition matrix A does not change across a series of time points, whereas nonhomogeneous HMMs depending upon t can also be applied for the situation, as dynamic biological processes are prone to be vastly heterogeneous over time depending on precautious decision on model selection by investigators. In order to estimate the most likely paths for given hidden states for each gene, parameter set of initial state $\pi_{gt}{}^0$, transition matrix A, and conditional distribution $f^0_{gt}\left(y_{gt}|S_{gt}=k\right)$ are inferred through Baum-Welch Algorithm [25, 26] as given below.

Let S_{gt} be a hidden random variable with N possible values relying on bell numbers under conditions. Since we assumed homogeneous HMMs to lead to following statement of transition matrix A, $A=\{a_{ij}\}=\Pr\left(S_{gt}=j|S_{g(t-1)}=i\right)$, initial state distribution $\pi^0_{gt}=\Pr\left(S_{g1}=i\right)$, y_{gt} is the observation representing gene expression temporal dynamic processes with K possible values, a conditional distribution of a certain observation $f^0_{gt}\left(y_{gt}\right)=\Pr\left(y_{gt}|S_{gt}=j\right)$ at time t for state j and an observation sequence on t, $\Upsilon_g=\left[y_{gt=1},\ldots,y_{gt=T}\right]$. Through Baum-Welch Algorithm, we estimate a local maxima markov chain process by the parameter set of $\varepsilon=\left(A,f^0_{gt},\pi_0\right)$ to be held by $\varepsilon^*=\max_{\varepsilon}\Pr\left(\Upsilon_{gt}|\varepsilon\right)$. A brief description of Baum-Welch Algorithm [27] is given,

$\varepsilon = \left(A, f^0_{gt}, \pi_0\right)$ with random initial conditions.

Forward Procedure:

Let $\alpha_i(t) = \Pr\left(\Upsilon_1 = y_1, \ldots, \Upsilon_t = y_t, X_t = i \middle| \varepsilon\right)$ represent probability of $y_1, \ldots, y_t$ and being in state i at time t. This is recursively found,

$$\alpha_i(1) = \pi_i f_i(y_i)$$
$$\alpha_j(t+1) = f_j(y_{t+1}) \sum_{i=1}^{N} \alpha_i(t) a_{ij}$$

Backward Procedure:

Let $\beta_i(t) = \Pr\left(\Upsilon_{t+1} = y_{t+1}, \ldots, \Upsilon_T = y_T | x_t = i, \varepsilon\right)$ represent the probability of ending partial sequence $y_{t+1}, \ldots, y_t$ given starting state i at time t.

$\beta_i(t)$ is computed as

1. $\beta_i(T) = 1$
2. $\beta_i(t) = \sum_{j=1}^{N} \beta_j(t+1) a_{ij} f_j(y_{t+1})$

Update:

The temporary variables,

$$\gamma_i(t) = \Pr\left(S_i = i \middle| \Upsilon, \varepsilon\right) = \frac{\alpha_i(t)\beta_i(t)}{\sum_{j=1}^{N} \alpha_j(t)\beta_j(t)}$$

is the probability of being state i at time t given observed temporal dynamic gene expression profile Υ and the parameter set of ε.

$$\zeta_{ij}(t) = \Pr\left(S_t = i, S_{t+1} = j \middle| \Upsilon, \varepsilon\right) = \frac{\alpha_i\left(t \middle| \alpha_{ij}\beta_j(t+1) f_j(y_{t+1})\right)}{\sum_{k=1}^{N} \alpha_k(t)\beta_k(t)}$$

is the probability of being in state i and j at times t and $t+1$, respectively given observed temporal dynamic gene expression profile Υ and the parameter set of ε.

$$\alpha^*_{ij} = \frac{\sum_{t=1}^{T-1} \zeta_{ij}(t)}{\sum_{t=1}^{T-1} \gamma_i(t)}, \text{ where } \pi^*_{ii} = \gamma_i(1)$$

is the expected frequency spent in state i at time 1,

which is the expected number of transitions from state i to state j compared to the expected total # of transitions away from state i (loop transition is also considered here)

It is equivalent to the number of times state is observed in the sequence from $t = 1$ to $t = T - 1$

$$f_i^{\star}(k) = \frac{\sum_{t=1}^{T-1} 1_{y_t = v_k} \gamma_i(t)}{\sum_{t=1}^{T-1} \gamma_i(t)}, \quad \text{where} \quad 1_{y_t = v_k} = \begin{cases} 1, \text{if } y_k = v_k \\ 0, \text{otherwise} \end{cases}$$

these steps are repeated until it is converged to any of user specifically desired levels.

2.2 Hierarchical Bayesian Modeling Approaches in Identification of Differential Expression Analysis for Alternative Splicing Events

A beneficial and peculiar nature of RNA-Seq data, temporally differential expression at gene level quantification can be extended to isoform levels which can better capture systematically the transcriptional dynamic processes. The proposed isoform modeling approaches for temporal dynamic processes in the current section are based on hierarchical Dirichlet Bayesian model framework.

Let γ_{ijt} be a latent variable to be estimated in a hierarchical mixture model whether EEX or TDEX is at a particular isoform of interest. Under the experimental or biological external conditions over time in stimulus-response time course data, if prior knowledge on parameters is not much given in the data, Bayesian nonparametric model, i.e., Dirichlet Process (DP) prior has been adopted commonly in the previous studies including Bayesian gene expression models [28, 29]. For the sake of consistency in formulas as denoted at gene level quantification, we consider a hierarchical Bayesian mixture model to estimate the posterior probabilities of differentially expressed (DE) isoform or equally expressed (EE) isoform using latent variables, r_{ijt}, under experimental conditions over time. In addition, in case that we do not have much information to prior information on parameters, Bayesian nonparametric model, i.e., Dirichlet Process (DP) prior, has been most widely used in many literatures including Bayesian gene expression models, e.g., Do et al. [28], Guindani et al. [29]. Let y_{ijkt} be an isoform expression of the ith isoform of the jth biological or experimental condition with the kth replicate at time t. At a given time t, τ_{it} denotes the ith isoform effect, β_{jt} the effect of the jth condition and $(\tau\beta)_{ijt}$ the interaction effect between isoform and condition. The latent variable r_{ijt} would be interpreted as indicator for isoform I being differentially expressed ($r_{ijt} = 1$, DE; $r_{ijt} = 0$, EE).

$$\begin{aligned} y_{ijtk} | r_{ijt} &= d \sim \text{Poi}_{\text{d}}\left(\lambda_{ijt}\right), \text{ independently} \\ \log\left(\lambda_{ijt}\right) &= \tau_i + \beta_j + F(\beta) + \omega_{ijt} \\ \beta_j | \left(c_j = g\right) &\sim F\left(\beta_g\right) \\ F(\beta) &\sim \text{DP}(F_0(\beta); \eta_0) \end{aligned}$$

and

$$\omega_{ijt}\big|\left(r_{ijt}=d,\ c_j=g\right)\sim \mathrm{DP}\left(G_d^*;M\right)$$

where DP(H; a) stands for the Dirichlet process having its baseline distribution $H(\cdot)$ and mass parameter a, F_0 is the joint distribution of β with $p(\beta)=N_{G-1}(0,\Psi)$ we consider $G_0^*\sim N\left(-\delta,\sigma^2\right)$ for EE and $G_1^*\sim\frac{1}{2}N\left(-\delta,\sigma^2\right)+\frac{1}{2}N\left(-\delta,\sigma^2\right)$ in a given biological condition g and η_0, ψ_0 and M are fixed hyperparameters based on prior information of them. Based on the above model, $p\left(r_{ijt}=1\big|c_j=g,y_{ij1t},\ldots,y_{ijKt}\right)$ is the posterior probabilities of differential expression.

With the advance of ultrahigh-throughput RNA-Seq technology, new various types of experimental design have allowed us to explore more complicated data formats such as time course expression profile data measured for biological temporal dynamic processes (e.g., disease progression over a time period) that have inherently intricate horizontal dependency structure and rich information compared to static data. In the methodological aspects, nonetheless, simply existing static methods have been commonly applied for the dynamic data as alternatives, albeit they do not explicitly take into account time dependent property. Yet, in reality of temporal dynamic expression profiles, it is observed that two consecutive expression profiles between neighboring time points are closely related to each other suggesting that Bayesian strategies are more suitable to address issues for the complicated data structure. Moreover, another attractive nature of RNA-Seq data enables us to decipher alternative splicing complexity that has been shown to play more crucial roles in transcriptional regulatory network modules from system biology prespective compared to gene level quantification studies. The multiple exons linked on a gene are characterized into several different groups, aka isoforms, to be a variety of combination of the exons and the different transcript variant isoforms eventually construct different protein structures and their own functionalities [30–34]. Thus, to better understand and fully capture the unique features of time series expression profiles, we endeavor to demonstrate proper modeling approaches based on HMMs and hierarchical Bayesian techniques at gene and isoform levels thus far, in detail, in this chapter.

References

1. Fisher RA (1941) The interpretation of experimental four-fold tables. Science 94:210–211. doi:10.1126/science.94.2435.210
2. Wang L, Feng Z, Wang X, Zhang X (2010) DEGseq: an R package for identifying differentially expressed genes from RNA-seq data. Bioinformatics 26:136–138. doi:10.1093/bioinformatics/btp612
3. Robinson MD, Mccarthy DJ, Smyth GK (2010) edgeR: a Bioconductor package for differential expression analysis of digital gene expression data. Bioinformatics 26:139–140. doi:10.1093/bioinformatics/btp616

4. Anders S, Huber W (2010) Differential expression analysis for sequence count data. Genome Biol 11:R106. doi:10.1186/gb-2010-11-10-r106
5. Hardcastle TJ, Kelly KA (2010) baySeq: empirical Bayesian methods for identifying differential expression in sequence count data. BMC Bioinformatics 11:422. doi:10.1186/1471-2105-11-422
6. Bullard JH, Purdom E, Hansen KD, Dudoit S (2010) Evaluation of statistical methods for normalization and differential expression in mRNA-Seq experiments. BMC Bioinformatics 11:94. doi:10.1186/1471-2105-11-94
7. Bar-Joseph Z, Gerber G, Simon I, Gifford DK, Jaakkola TS (2003) Comparing the continuous representation of time-series expression profiles to identify differentially expressed genes. Proc Natl Acad Sci U S A 100(18):10146–10151
8. Ramoni MF, Sebastiani P, Kohane IS (2002) Cluster analysis of gene expression dynamics. Proc Natl Acad Sci U S A 99(14):9121–9126. doi:10.1073/pnas.132656399
9. Zhu F, Shi L, Li H, Eksi R, Engel JD, Guan Y (2014) Modeling dynamic functional relationship networks and application to ex vivo human erythroid differentiation. Bioinformatics 30 (23):3325–3333
10. Jo K, Kwon HB, Kim S (2014) Time-series RNA-seq analysis package (TRAP) and its application to the analysis of rice, Oryza sativa L. ssp. Japonica, upon drought stress. Methods 67(3):364–372. doi:10.1016/j.ymeth.2014.02.001
11. Sîrbu A, Kerr G, Crane M, Ruskin HJ (2012) RNA-Seq vs dual- and single-channel microarray data: sensitivity analysis for differential expression and clustering. PloS One 7(12): e50986. doi:10.1371/journal.pone.0050986
12. Oh S, Song S, Grabowski G, Zhao H, Noonan JP (2013) Time series expression analyses using RNA-seq: a statistical approach. Biomed Res Int. doi:10.1155/2013/203681
13. Lu ZK, Allen, OB, Desmond AF (2012) An order estimation based approach to identify response genes for microarray time course data. Stat Appl Genet Mol Biol 11(65). doi:10.1515/1544-6115.1818
14. Sundar AS, Varghese SM, Shameer K, Karaba N, Udayakumar M, Sowdhamini R (2008) STIF: Identification of stress-upregulated transcription factor binding sites in Arabidopsis thaliana. Bioinformatics 2(10):431–437
15. Newton R, Hinds J, Wernisch L (2006) A Hidden Markov model web application for analysing bacterial genomotyping DNA microarray experiments. Appl Bioinformatics 5 (4):211–218
16. Lu J, Bushel PR (2013) Dynamic expression of 3′ UTRs revealed by Poisson hidden Markov modeling of RNA-Seq: implications in gene expression profiling. Gene 527(2):616–623. doi:10.1016/j.gene.2013.06.052
17. Thorne T, Stumpf MP (2012) Inference of temporally varying Bayesian networks. Bioinformatics 28(24):3298–3305. doi:10.1093/bioinformatics/bts614
18. Schliep A, Schönhuth A, Steinhoff C (2003) Using hidden Markov models to analyze gene expression time course data. Bioinformatics 19:255–263
19. Nueda MJ, Tarazona S, Conesa A (2014) Next maSigPro: updating maSigPro bioconductor package for RNA-seq time series. Bioinformatics 30(18):2598–2602. doi:10.1093/bioinformatics/btu333
20. Yuan M, Kendziorski C (2006) Hidden Markov models for microarray time course data in multiple biological conditions. J Am Stat Assoc 101(476):1323–1332. doi:10.1198/016214505000000394
21. Niu L, Huang W, Umbach DM, Li L (2014) IUTA: a tool for effectively detecting differential isoform usage from RNA-Seq data. BMC Genomics 15(1):862
22. Yuan X, Zhao Y, Liu C, Bu D (2011) Lex-SVM: exploring the potential of exon expression profiling for disease classification. J Bioinform Comput Biol 9(2):299–316
23. Nagalakshmi U, Wang Z, Waern K, Shou C, Raha D, Gerstein M, Snyder M (2008) The transcriptional landscape of the yeast genome defined by RNA sequencing. Science 320 (5881):1344–1349. doi:10.1126/science.1158441
24. Cho S, Lee JW, Heo JS, Kim SY (2014) Gene expression change in human dental pulp cells exposed to a low-level toxic concentration of triethylene glycol dimethacrylate: an RNA-seq analysis. Basic Clin Pharmacol Toxicol 115 (3):282–290. doi:10.1111/bcpt.12197
25. Rezaei V, Pezeshk H, Pérez-Sa'nchez H (2013) Generalized Baum-Welch algorithm based on the similarity between sequences. PloS One 8(12):e80565. doi:10.1371/journal.pone.0080565
26. Vogl C, Futschik A (2010) Hidden Markov models in biology. Methods Mol Biol 609:241–253. doi:10.1007/978-1-60327-241-4_14
27. Wikipedia Baum-Welch Algorithms
28. Do K, Ml P, Tang F (2005) A Bayesian mixture model for differential gene expression. Appl Stat 54(3):627–644
29. Guindani M, Sepúlveda N, Paulino CD, Müller P (2014) A Bayesian semi-parametric approach

for the differential analysis of sequence counts data. J R Stat Soc C 63(3):385–404

30. Nance T, Smith KS, Anaya V, Richardson R, Ho L, Pala M, Mostafavi S, Battle A, Feghali-Bostwick C, Rosen G, Montgomery SB (2014) Transcriptome analysis reveals differential splicing events in IPF lung tissue. PLoS One 9(5). doi:10.1371/journal.pone.0097550
31. Nance T, Smith KS, Anaya V, Richardson R, Ho L, Pala M, Mostafavi S, Battle A, Feghali-Bostwick C, Rosen G, Montgomery SB (2014) Transcriptome analysis reveals differential splicing events in IPF lung tissue. PLoS One 9(3). doi:10.1371/journal.pone.0092111
32. Iacobucci I, Ferrarini A, Sazzini M, Giacomelli E, Lonetti A, Xumerle L, Ferrari A, Papayannidis C, Malerba G, Luiselli D, Boattini A, Garagnani P, Vitale A, Soverini S, Pane F, Baccarani M, Delledonne M, Martinelli G (2012) Application of the whole-transcriptome shotgun sequencing approach to the study of Philadelphia-positive acute lymphoblastic leukemia. Blood Cancer J 2(3):e61. doi:10.1038/bcj.2012.6
33. Atkins N, Miller CM, Owens JR, Turek FW (2011) Non-laser capture microscopy approach for the microdissection of discrete mouse brain regions for total RNA isolation and downstream next-generation sequencing and gene expression profiling. J Vis Exp (57). doi:10.3791/3125
34. Twine NA, Janitz K, Wilkins MR, Janitz M (2011) Whole transcriptome sequencing reveals gene expression and splicing differences in brain regions affected by Alzheimer's disease. PLoS One 6(1):e16266. doi:10.1371/journal.pone.0016266

Chapter 13

Finding RNA–Protein Interaction Sites Using HMMs

Tao Wang, Jonghyun Yun, Yang Xie, and Guanghua Xiao

Abstract

RNA-binding proteins play important roles in the various stages of RNA maturation through binding to its target RNAs. Cross-linking immunoprecipitation coupled with high-throughput sequencing (CLIP-Seq) has made it possible to identify the targeting sites of RNA-binding proteins in various cell culture systems and tissue types on a genome-wide scale. Several Hidden Markov model-based (HMM) approaches have been suggested to identify protein–RNA binding sites from CLIP-Seq datasets. In this chapter, we describe how HMM can be applied to analyze CLIP-Seq datasets, including the bioinformatics preprocessing steps to extract count information from the sequencing data before HMM and the downstream analysis steps following peak-calling.

Key words Hidden Markov models, RNA-binding proteins, Interaction sites

1 Introduction

RNA-binding proteins (RBPs) participate in RNA translation, splicing, and editing events, and play essential roles in mRNA maturation and downstream regulation of cellular events [1]. Thus, accurate identification of RBP binding targets is important to a systematic understanding of transcription, translation and other biological processes within cells.

A technique known as cross-linking immunoprecipitation coupled with high-throughput sequencing (CLIP-Seq) has been developed to study genome-wide RNA–protein interactions [2–4]. CLIP-Seq experiments are further subdivided into three types: (1) high-throughput sequencing of RNA isolated by cross-linking immunoprecipitation (HITS-CLIP) [2, 5, 6]; (2) photoactivatable-ribonucleoside-enhanced cross-linking and immunoprecipitation (PAR-CLIP) [7–9]; and (3) individual-nucleotide resolution CLIP (iCLIP). HITS-CLIP utilizes UV cross-linking of proteins with RNA and introduces mutations in the sequencing data. More specifically, the mutations are induced on the cDNAs generated in the reverse transcription step from the RNA fragments, when the

David R. Westhead and M.S. Vijayabaskar (eds.), *Hidden Markov Models: Methods and Protocols*, Methods in Molecular Biology, vol. 1552, DOI 10.1007/978-1-4939-6753-7_13, © Springer Science+Business Media LLC 2017

reverse transcription enzyme incorporates an incorrect nucleotide at the site of the cross-linked nucleotide due to attachment of the remaining residues of the covalently bound protein. However, the type of mutation is not well-defined and may vary for different proteins [5, 10]. PAR-CLIP utilizes photoreactive ribonucleoside analogs for incorporation into RNA, and some of the analogs that are cross-linked by UV treatment at a later step will result in specific nucleotide substitution events. For example, 4-thiouridine treatment will induce T->C mutations at crosslinking sites [9]. The iCLIP experiment, which involves a self-circularization step to capture the truncation of cDNA reads, is less commonly used than HITS-CLIP and PAR-CLIP [11]. In iCLIP experiments, cDNA counts, rather than total tag count and mutant tag count, are used to infer protein binding sites.

The analysis of CLIP-Seq data mainly consists of three steps: the bioinformatics analysis that extracts count information from the raw sequencing data, the statistical analysis that carries out peak calling, and the downstream analysis that relates the significant interaction sites to their biological meanings. While this chapter centers on application of HMM in the peak calling step, we also briefly touch upon the other two steps in analysis of CLIP-Seq data.

2 Sequencing Data Preprocessing

CLIP-Seq analysis usually starts with SAM/BAM files, which contain the mapped reads from high-throughput sequencing data. Since CLIP-Seq experiments involve PCR amplification from cDNA libraries with limited complexities, removal of PCR duplicates is an essential step for CLIP-Seq experiments. The iCLIP technique explicitly introduces a random barcode, thus making it relatively easy to define PCR artifacts from the sequencing data. But the duplicate removal step for HITS-CLIP and PAR-CLIP is not trivial. One popular approach that has been adopted in many studies [12–14] is to define reads that have exactly the same mapping coordinates as PCR duplicates. Unfortunately there is no single best answer to this question, and the scenario is even more complicated for paired-end sequencing reads.

For HITS-CLIP and PAR-CLIP, unique tags overlapping by at least one nucleotide are grouped together to form CLIP clusters after PCR duplicates are collapsed, and those not overlapping with any other tags are discarded. Then the clusters are divided into bins of small length (for example, 5 bp) and the total tag count in each bin is calculated. If mutant information is utilized in the statistical analysis, mutant tag counts will also be calculated for each bin.

However, for iCLIP datasets, the preprocessing step differs substantially. After removing PCR duplicates, the first nucleotide upstream of each mapped cDNA, defined as the "cross-link

nucleotide," is counted and the counts are binned. Since the cross-link nucleotides exist in a sparse, discontinuous manner around RBP binding sites, it might be a good idea to expand each cross-link nucleotide by a few base pairs to smooth the tag data, as is done in dCLIP [14].

3 Hidden Markov Model (HMM)

The bins coming from one single CLIP cluster form an observational sequence. There are many CLIP clusters identified from the sequencing data, which form multiple observational sequences. The HMM needs to analyze all the observational sequences simultaneously. Our introduction to HMM is subdivided into four parts.

3.1 Transcript Abundance Normalization

Currently, most CLIP-Seq studies do not conduct transcript abundance measurements [6, 15] and, accordingly, most current CLIP-seq analysis tools, such as PARalyzer [16], also do not consider transcript abundance. However, taking background transcript abundance into account will be very helpful for more accurately identifying RBP binding sites, since the size of each CLIP cluster is affected by both the transcript level and the binding strength of the RBP.

3.2 Model Specification

We assume that for the t-th bin of each region d, the spatially associated tag count $n_t^{(d)} > 0$ is governed by a Markov latent variable $q_t^{(d)}$, which spans two exclusive states as follows:

$$q_t^{(d)} = \begin{cases} 1 & \text{if unenriched;} \\ 2 & \text{if enriched.} \end{cases}$$

Markov chains in different clusters are assumed to be independent, but the chains share identical forms of transition and emission probabilities. A graphical representation of the HMM is shown in Fig. 1.

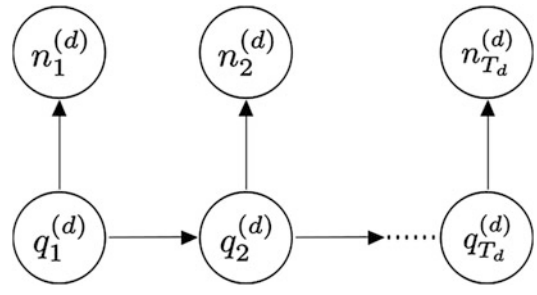


Fig. 1 Graphical representation of the HMM for cluster d: $\left\{n_t^{(d)}\right\}$ is the observable process (tag count), and $\left\{q_t^{(d)}\right\}$ is the hidden Markov chain. Independent Markov chains with identical transition rules are equipped for different clusters

3.2.1 Transition Probabilities

An initial state of Markov chains are described by an initial distribution $\xi\left(q_1^{(d)}\right)$. The Markov transition matrix Λ describes stochastic transition behaviors of $q_{t+1}^{(d)}$ relying on the previous state $q_t^{(d)}$.

$$\Lambda = \begin{pmatrix} \Lambda_{11} & \Lambda_{12} \\ \Lambda_{21} & \Lambda_{22} \end{pmatrix},$$

where Λ_{rv} denotes the transition probability from state r to v. That is,

$$\Lambda_{rv} \equiv \Pr\left(q_{t+1}^{(d)} = v | q_t^{(d)} = r\right).$$

3.2.2 Emission Probabilities

A mixture of negative binomial (NB) distributions have been commonly used to fit the tag count in next-generation sequencing data due to over-dispersion. The NB density on $n_t^{(d)} - 1 \big| q_t^{(d)}$ is assumed. The parametrization of the NB(a, b) involves the shape parameter $a > 0$ and the inverse scale parameter $b > 0$. Assuming a to be identical for all states, the conditional density of $n_t^{(d)}$ given $q_t^{(d)}$ is

$$p\left(n_t^{(d)} = n | a, b_v, q_t^{(d)} = v\right) = \binom{n+a-2}{a} \left(\frac{b_v}{1+b_v}\right)^a \left(\frac{1}{1+b_v}\right)^{n-1}, n > 0,$$

where $b_1 > b_2$. Here, $\mu_v \equiv \mathbb{E}\left[n_t - 1 | q_t^{(d)} = v\right] = a \cdot b_v^{-1}$, and $\mathbb{V}\text{ar}\left[n_t - 1 | q_t^{(d)} = v\right] = \mu_v + \mu_v^2 \cdot a^{-1}$, where a^{-1} is the dispersion parameter for the NB distribution. According to the definition of the HMM states, it is expected that $\mu_1 < \mu_2$, and so the ordering on b_v's is imposed.

3.2.3 Likelihood

Let $\phi = (\pi, \Lambda, a, b_1, b_2)$, $\mathbf{n}_d = \left(n_1^{(d)}, \ldots, n_{T_d}^{(d)}\right)$, and $\mathbf{q}_d = \left(q_1^{(d)}, \ldots, q_{T_d}^{(d)}\right)$. The likelihood of ϕ is given as

$$\prod_d p_\phi(\mathbf{n}_d) = \prod_d \sum_{\mathbf{q}_d} p_\phi(\mathbf{q}_d, \mathbf{n}_d),$$

where the summation is taken over all possible configurations of $\mathbf{q}_d$. The sequential structure of HMMs facilitates the computation of the joint density.

$$p_\phi(\mathbf{q}_d, \mathbf{n}_d) = \pi\left(q_1^{(d)}\right) \prod_{t=1}^{T_d} \Lambda\left(q_{t+1}^{(d)} | q_t^{(d)}\right) p_\phi\left(n_t^{(d)} | q_t^{(d)}\right),$$

where we use $\Lambda(v|\ r)$ to denote Λ_{rv}.

3.3 Model Inferences

3.3.1 Baum–Welch Expectation Maximization Algorithm

The Baum–Welch expectation maximization algorithm [17] is used to estimate parameters in the HMM. The iterative algorithm repeats two steps (E-step and M-step) to update a current estimate ϕ^c until a stopping criterion is met. For simplicity, we drop the cluster index d, and observation sequences are denoted by $n_{1:t} = (n_1, \ldots, n_t)$.

Forward–backward recursions are essential in implementing the E-step. In the E-step, we evaluate an expectation

$$G(\phi|\phi^c) \equiv \mathbb{E}_{\phi^c}\left[\sum_d \log p_\phi(\mathbf{q}_d, \mathbf{n}_d)\right],$$

where the expectation is taken with respect to the conditional density $\prod_d p_{\phi^c}(\mathbf{q}_d|\mathbf{n}_d)$. The filtering distributions $\pi_t(q_t) \equiv p_{\phi^c}(q_t|n_{1:t}, q_0)$ can be evaluated recursively as

$$\pi_t(q_t) = \begin{cases} \dfrac{1}{M_t}\psi_t(q_t)\xi^c(q_t) & \text{if } t = 1; \\ \dfrac{1}{M_t}\psi_t(q_t)\sum_{r\in Q}\Lambda^c(q_t|r)\pi_{t-1}(r) & \text{if } t > 1, \end{cases}$$

where $\psi_t(q_t) \equiv p_{\phi^c}(n_t|q_t)$, M_t is the normalizing constant reconciled by $\sum_{q_t\in Q}\pi_t(q_t) = 1$, and $Q = \{1, 2\}$ is the state space. The backward probability is computed as

$$\kappa_{t,T}(q_t) = \begin{cases} M_{t+1}^{-1}\sum_{v\in Q}\kappa_{t+1,T}(v)\psi_{t+1}(v)\Lambda^c(v|q_t) & \text{if } t < T; \\ 1 & \text{if } t = T. \end{cases}$$

Then, the smoothing distribution $\pi_{t|T}(q_t) \equiv p_{\phi^c}(q_t|n_{1:T})$ can be computed recursively as

$$\pi_{t|T}(q_t) = \frac{1}{M}\kappa_{t,T}(q_t)\pi_t(q_t),$$

where $M = \sum^{r\in Q}\Lambda^c(q_{T+1}|r)\pi_T(r)$.

The conditional maximization method is used to implement the M step. The initial distribution is updated by $\xi(v) \propto \sum_d \pi^{(d)}_{1|T_d}(v)$. The transition probabilities are updated on the constrained space ($\sum_r \Lambda_{rv} = 1$) as

$$\Lambda_{rv} = \frac{\sum_d \sum_{t=1}^{T_d} P_{\phi^c}(v, r|\mathbf{n}_d)}{\sum_{v\in Q}\text{numerator}},$$

where the numerator can be expressed as.

$$P_{\phi^c}(v, r|\mathbf{n}_d) = \begin{cases} \pi^{(d)}_{1|T_d}(v) & \text{if } t = 1; \\ \left(M^{(d)} \cdot M_t^{(d)}\right)^{-1}\kappa^{(d)}_{t,T_d}(v)\psi_t^{(d)}(v)\Lambda^c(v|r)\pi^{(d)}_{t-1}(r) & \text{if } 1 < t \le T_d. \end{cases}$$

The inverse scale parameter b_v can be updated by

$$b_v = \frac{a^c\sum_d\sum_t \pi^{(d)}_{t|T_d}(v)}{\sum_t\left(n_t^{(d)} - 1\right)\pi^{(d)}_{t|T_d}(v)}.$$

Updates for a need to be done using numerical optimization methods such as the Newton–Raphson.

3.3.2 Viterbi Algorithm

After the convergence of the EM is observed, the Viterbi algorithm [18] is implemented along with the EM estimate $\hat{\phi}$ to obtain the most likely sequence of states that generates tag counts.

The algorithm recursively finds $q_t^{(d)}$ that maximizes

$$\omega\left(q_t^{(d)} = v\right) = \psi_t^{(d)} \cdot \max_{q_{t-1}^{(d)}} \omega\left(q_{t-1}^{(d)}\right) \cdot \Lambda\left(v | q_{t-1}^{(d)}\right),$$

where $\omega\left(q_1^{(d)} = v\right) = \xi(v)\psi_1^{(d)}(v)$.

From the *maximum* a posteriori path, the set of the enriched bins $E = \left\{(t, d) | q_t^{(d)} = 2\right\}$ is obtained.

3.4 Mixture Models

Mutation counts can be used to pinpoint binding sites among enriched bins. Let n_i and x_i denote tag and mutation counts for bins in E. We assume a mixture of binomial distributions with a latent variable s_i, where $s_i = 1$ if the bin contains binding sites; 0 otherwise.

$$x_i | n_i \sim \begin{cases} Bin(n_i, p_1) & s_i = 1; \\ Bin(n_i, p_0) & s_i = 0, \end{cases}$$

where $p_1 > p_0$ are mutation probabilities. The EM algorithm can be easily implemented to estimate parameters $\theta = (p_0, p_1,$ and $w = p(s_i = 1))$ in the mixture model. Then, a user-defined cut-off for the posterior probability $p_\theta(s_i = 1 |$ observations) is applied to identify RBP binding sites over bins in E.

4 Downstream Analysis

1. For PAR-CLIP and HITS-CLIP data, after identification of significantly enriched RBP binding sites, it is interesting to identify single bases with high ratios of characteristic mutations within these sites. These bases pinpoint the location of RBP-RNA interaction at almost single-nucleotide resolution. However, one difficulty with this analysis is that the mutation ratios are usually very low. This is especially true in some HITS-CLIP datasets, where the highest observation mutation ratios are only around 1–2%. In addition, SNPs that exist in the cell culture or tissue could obscure this analysis. Such false positives can be distinguished by conducting background sequencing experiments as control or by comparing them to SNP databases in the public domain.
2. One of the downstream analyses that can be performed here is to relate the significant RBP binding sites to genes or annotation features (CDS, intron, 5′ UTR, 3′ UTR, etc.). When relating RBP binding sites to genes, some researchers prefer to mask tRNAs, rRNAs and other repetitive sequences from this analysis, since such binding sites usually represent uninteresting or

artifact binding events. Most CLIP-Seq experiments do not distinguish nucleus RNAs from cytoplasmic RNAs. Since libraries made in this way could contain cDNAs converted from nascent mRNAs, it is possible that a significant portion of CLIP reads will be mapped to reference gene introns. In some studies, intron-locating peaks even constitute the majority of the RBP binding targets.

3. Another popular way to analyze CLIP-Seq peaks is discovering sequence motifs or predicting secondary structures around significant RBP binding sites. Since the mutant tag counts give much more precise information regarding RBP binding locations, it is advisable to extract RNA sequences in a short window (for example, 50 bp) surrounding bases with high mutation ratios in significant clusters for analysis purpose. To discover sequence motifs, MEME or Homer can be used. If the dataset is PAR-CLIP, it could be interesting to investigate whether T->C mutations occur on the T bases within the significant motifs more often than the genome wide background. To predict secondary structures, software like RNAfold can be used. It is possible the RBPs recognize binding targets through secondary structures, rather than through the primary sequence.

5 Notes

1. Some of the sequencing reads could be mapped across splicing junctions. However, usually less than 5% of all the CLIP reads are mapped across splice junctions. The reads mapped across splice junctions represent different binding events from those reads mapped within these introns, and therefore need to be handled separately. One possible solution is to map reads to genome and then map again the unaligned reads to transcriptome. Then the CLIP clusters identified from the genome and transcriptome can be analyzed at the same time.
2. Nowadays, sequencing data can reach exceptionally large sizes (many >50 M reads). Therefore, the CLIP clusters that form from overlapping tags could be huge in number, and the length of the clusters can also be very long. These data present a lot of computational challenges. Therefore, it is advisable to place a limit on the minimum number of tags for a cluster and focus analysis on the larger CLIP clusters.

References

1. Keene JD (2007) RNA regulons: coordination of post-transcriptional events. Nat Rev Genet 8:533–543
2. Licatalosi DD, Mele A, Fak JJ, Ule J, Kayikci M et al (2008) HITS-CLIP yields genome-wide insights into brain alternative RNA processing. Nature 456:464–469
3. Ule J, Jensen K, Mele A, Darnell RB (2005) CLIP: a method for identifying protein-RNA interaction sites in living cells. Methods 37:376–386

4. Ule J, Jensen KB, Ruggiu M, Mele A, Ule A et al (2003) CLIP identifies Nova-regulated RNA networks in the brain. Science 302:1212–1215
5. Zhang C, Darnell RB (2011) Mapping in vivo protein-RNA interactions at single-nucleotide resolution from HITS-CLIP data. Nat Biotechnol 29:607–614
6. Chi SW, Zang JB, Mele A, Darnell RB (2009) Argonaute HITS-CLIP decodes microRNA-mRNA interaction maps. Nature 460:479–486
7. Hafner M, Lianoglou S, Tuschl T, Betel D (2012) Genome-wide identification of miRNA targets by PAR-CLIP. Methods 58:94–105
8. Hafner M, Landthaler M, Burger L, Khorshid M, Hausser J et al (2010) PAR-CliP–a method to identify transcriptome-wide the binding sites of RNA binding proteins. J Vis Exp 41. doi:10.3791/2034. pii: 2034
9. Hafner M, Landthaler M, Burger L, Khorshid M, Hausser J et al (2010) Transcriptome-wide identification of RNA-binding protein and microRNA target sites by PAR-CLIP. Cell 141:129–141
10. Granneman S, Kudla G, Petfalski E, Tollervey D (2009) Identification of protein binding sites on U3 snoRNA and pre-rRNA by UV cross-linking and high-throughput analysis of cDNAs. Proc Natl Acad Sci U S A 106:9613–9618
11. Konig J, Zarnack K, Rot G, Curk T, Kayikci M et al (2010) iCLIP reveals the function of hnRNP particles in splicing at individual nucleotide resolution. Nat Struct Mol Biol 17:909–915
12. Ince-Dunn G, Okano HJ, Jensen KB, Park WY, Zhong R et al (2012) Neuronal Elav-like (Hu) proteins regulate RNA splicing and abundance to control glutamate levels and neuronal excitability. Neuron 75:1067–1080
13. Boudreau RL, Jiang P, Gilmore BL, Spengler RM, Tirabassi R et al (2014) Transcriptome-wide discovery of microRNA binding sites in human brain. Neuron 81:294–305
14. Wang T, Xie Y, Xiao G (2014) dCLIP: a computational approach for comparative CLIP-seq analyses. Genome Biol 15:R11
15. Neumann M, Bentmann E, Dormann D, Jawaid A, DeJesus-Hernandez M et al (2011) FET proteins TAF15 and EWS are selective markers that distinguish FTLD with FUS pathology from amyotrophic lateral sclerosis with FUS mutations. Brain 134:2595–2609
16. Corcoran DL, Georgiev S, Mukherjee N, Gottwein E, Skalsky RL et al (2011) PARalyzer: definition of RNA binding sites from PAR-CLIP short-read sequence data. Genome Biol 12:R79
17. Welch LR (2003) Hidden Markov models and the Baum-Welch algorithm. IEEE Inform Theory Soc Newslett:1–14
18. Viterbi AJ (1967) Error Bounds for Convolutional Codes and an Asymptotically Optimum Decoding Algorithm. IEEE Trans Inform Theory 13:260–269

Chapter 14

Automated Estimation of Mouse Social Behaviors Based on a Hidden Markov Model

Toshiya Arakawa, Akira Tanave, Aki Takahashi, Satoshi Kakihara, Tsuyoshi Koide, and Takashi Tsuchiya

Abstract

Recent innovations in sensing and Information and Communication Technology (ICT) have enabled researchers in animal behavior to collect an enormous amount of data. Consequently, the development of an automated system to substitute for some of the observations and analyses that are performed currently by expert researchers is becoming a crucial issue so that the vast amount of accumulated data can be processed efficiently. For this purpose, we introduce a process for the automated classification of the social interactive status of two mice in a square field on the basis of a Hidden Markov model (HMM). We developed two models: one for the classification of two states, namely, indifference and interaction, and the other for three states, namely, indifference, sniffing, and following. The HMM was trained with data from 50 pairs of mice as provided by expert human observers. We measured the performance of the HMM by determining its rate of concordance with human observation. We found that sniffing behavior was segmented well by the HMM; however, following behavior was not segmented well by the HMM in terms of percentage concordance. We also developed software called DuoMouse, an automated system for the classification of social interactive behavior of mice, that was based on the HMM. Finally, we compared two implementations of the HMM that were based on a histogram and a Gaussian mixture model.

Key words Mouse, Behavior, Social interaction, Human observation, Hidden Markov model

1 Introduction

Recently, the development of sensing technology and Information and Communication Technology (ICT) has enabled researchers in various fields of science and technology to collect an enormous amount of spatiotemporal data from moving subjects. For example, in automotive engineering, data on velocity, acceleration, brake pressure, and steering angle can be collected via a Controller Area Network (CAN). The same situation exists in the analysis of animal behavior. However, even if data are collected and accumulated efficiently, the interpretation of and drawing of inferences from the data are intellectual activities that are difficult to automate,

David R. Westhead and M.S. Vijayabaskar (eds.), *Hidden Markov Models: Methods and Protocols*, Methods in Molecular Biology, vol. 1552, DOI 10.1007/978-1-4939-6753-7_14, © Springer Science+Business Media LLC 2017

and thus could limit the speed at which behavioral research can be performed. Consequently, the development of an automated system to assist researchers in the observation and analysis of behaviors is becoming a crucial issue so that the vast amount of accumulated data can be processed efficiently.

In this chapter, we present an analysis of the interactive behavior of two mice in a square field and introduce an automated classification system that is based on a Hidden Markov Model (HMM). This system can differentiate mouse behavior into two states, indifference and interaction, or three states, indifference, sniffing, and following. We have integrated this technology into the software DuoMouse. We explain the background of study, the HMM, and the DuoMouse software, and discuss the validity of the classification that can be achieved using DuoMouse. We also compare two implementations of the HMM based on a histogram and a Gaussian mixture model.

2 Materials and Methods

Previously, we had conducted a genetic study of mice to understand the mechanisms by which individuals develop differences in behavior [1, 2]. The previous study used C57BL/6J (B6)-based intersubspecific consomic strains that had been established from Japanese wild-mouse-derived strain MSM/Ms and B6 mice. Here, a consomic strain refers to a strain in which one chromosome of B6 is replaced with the corresponding chromosome of MSM/Ms. By comparing the social interactive behaviors of each consomic strain with those of the parental strains, B6 and MSM/Ms, we were able to speculate on the role of each chromosome in determining the social interactive nature of mice [2].

In the study, two animals from the same consomic strain and of the same sex, aged 10 weeks, were introduced into a novel open square field of 60 × 60 cm for 10 min and their behavior was recorded using a video tracking system (Image SI; O'Hara & Co., Tokyo, Japan). The tracking system was based on the public domain image processing and analysis program NIH Image, from the National Institutes of Health (USA). The locations of the two mice were recorded every third of a second, and thus the video tracking data consisted of 1800 frames. A total of 530 sets of video data were collected, which included approximately ten sets of video data for each consomic strain and sex. In the previous study by Takahashi et al. [2], genetic mapping was conducted based on the total duration and frequency of contact for each consomic strain and sex. In the analysis, it was necessary to identify the state of the pair at each tracking time point. This was a laborious manual task that is performed ideally by a human expert. To overcome this difficulty, we employed an HMM to automate this task.

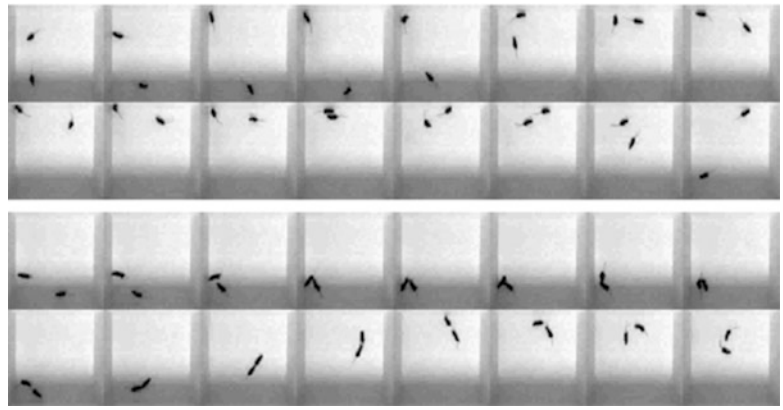


Fig. 1 Sequential photographs of a pair of mice (*top*: indifference; *bottom*: interaction). Modified from Arakawa et al. [3]

We analyzed 50 sets of video data, which included data for both sexes from nine strains, as the training data to develop a two-state or three-state HMM [3, 4]. We considered that the pair was in (1) either the "indifferent" or "interactive" state (two-state case, Fig. 1), or (2) the "indifferent," "sniffing," or "following" state (three-state case), where the state sequence obeys on the basis of a time series of observable characteristics.

3 Hidden Markov Model (HMM)

An HMM is composed of a Markov process that updates "states" and a (conditional) probability distribution of observable parameters in a given state [5–7]. Here, the main purpose of the HMM was to segment the interactive states of mice given a time series of observable parameters. We developed HMMs for the classification of two states (indifference and interaction), and three states (indifference, sniffing, and following). First, we explain the two-state model (Fig. 2a).

In our previous studies [3, 4], we considered that a pair of mice takes one of the two states s_k at each time point k (1, …, 1800), namely,

$$s_k = 0\text{: in difference (state 0)}, \quad s_k = 1\text{: (social) interaction (state 1)},$$

where s_k is the state of the pair at time point k. We considered that the state was updated in accordance with a Markov process. We defined the state of "interaction" as when animals were engaged in social behaviors, which included social sniffing, genital grooming, following, and aggressive behaviors. In contrast, "indifference" referred to all other states; namely, when the mice showed no interactive behaviors (Fig. 2a).

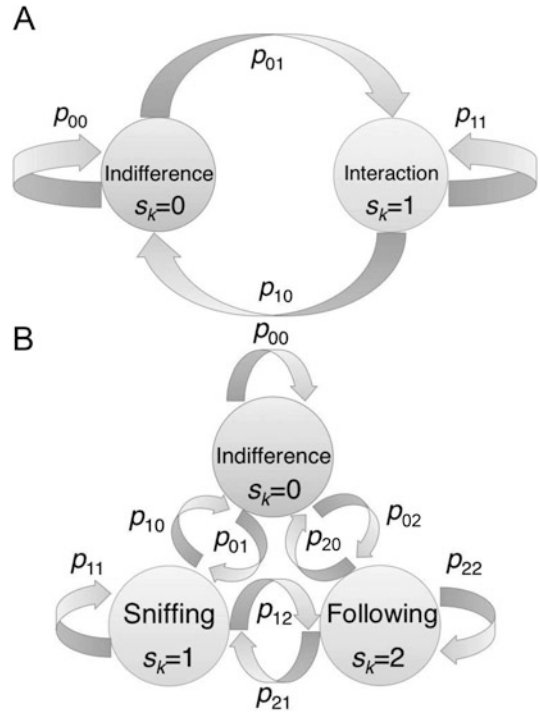


Fig. 2 (**a**) The two-state HMM. "p_{00}," "p_{01}," "p_{10}," and "p_{11}" denote "the probability of retaining indifference," "the probability of transitioning from indifference to interaction," "the probability of transitioning from interaction to indifference," and "the probability of retaining interaction." (**b**) The three-state HMM. "p_{00}," "p_{01}," "p_{02}," "p_{10}," "p_{11}," "p_{12}," "p_{20}," "p_{21}," and "p_{22}" denote "the probability of retaining indifference," "the probability of transitioning from indifference to sniffing," "the probability of transitioning from indifference to following," "the probability of transitioning from sniffing to indifference," "the probability of retaining sniffing," "the probability of transitioning from sniffing to following," "the probability of transitioning from following to indifference," "the probability of transitioning from following to sniffing," and "the probability of retaining following"

Next, we explain the conditional distribution $p(\boldsymbol{y}|s)$ of observable parameters for a given state, where $\boldsymbol{y}$ is the set of observable parameters and s is the state. In our previous studies [3, 4], $\boldsymbol{y}$ was taken as a four-dimensional vector that consisted of the following components (a)–(d):

(a) *Distance between the two mice*: The distance between the centers of the two mice (cm) at time point k was calculated as follows:

$$L(k) = 0.5\sqrt{(x_2(k) - x_1(k))^2 + (y_2(k) - y_1(k))^2}, \qquad (1)$$

where $x_1(k)$ and $x_2(k)$ are the x coordinates (pixels) of mice 1 and 2, respectively, and $y_1(k)$ and $y_2(k)$ are their y coordinates (pixels). Given that the size of the square field of 60 × 60 cm is recognized as being 120 × 120 pixels in the software, we multiply the pixel value by 0.5 to obtain $L(k)$ in the unit of cm.

(b) *Relative angle of the velocities of two mice*: The angles of mouse 1 and mouse 2 from time point $k-1$ to time point k were calculated as follows:

$$\theta_1(k) = \arctan\frac{y_1(k) - y_1(k-1)}{x_1(k) - x_1(k-1)} \tag{2}$$

$$\theta_2(k) = \arctan\frac{y_2(k) - y_2(k-1)}{x_2(k) - x_2(k-1)}. \tag{3}$$

Then, the relative angle between mouse 1 (θ_1) and mouse 2 (θ_2) was calculated as follows:

$$\theta(k) = \left|\theta_1(k) - \theta_2(k)\right|. \tag{4}$$

(c) *Relative speed of the two mice*: The relative speed was calculated as the change in the distance between the two mice from time point $k-1$ to time point k as follows:

$$V_r(k) = 3(L(k) - L(k-1)). \tag{5}$$

To use cm/s as the unit for $V_r(k)$, we multiplied by 3 to convert 1800 frames to 600 s.

(d) *Average speed of the two mice*: The speeds of mouse 1 and mouse 2 from time point $k-1$ to time point k were calculated as follows:

$$V_1(k) = 0.5 \times 3 \times \sqrt{(x_1(k) - x_1(k-1))^2 + (y_1(k) - y_1(k-1))^2} \tag{6}$$

$$V_2(k) = 0.5 \times 3 \times \sqrt{(x_2(k) - x_2(k-1))^2 + (y_2(k) - y_2(k-1))^2}. \tag{7}$$

Then, the average speed was calculated as $\bar{V}(k) = \frac{V_1(k)+V_2(k)}{2}$.

The observable parameter y_k at time point k was assumed to obey the distribution $p(\boldsymbol{y}|s_k = 0)$ if the state was indifference and $p(\boldsymbol{y}|s_k = 1)$ if the state was interaction. The distribution was then estimated using a histogram that was based on the training data.

While the observable parameter obeyed $p(\boldsymbol{y}|s_k)$, s_k was updated in accordance with the Markov model determined by the Markov transition matrix $T = \begin{bmatrix} p_{00} & p_{01} \\ p_{10} & p_{11} \end{bmatrix}$. This is an outline of our HMM.

To implement this model, we needed the Markov transition matrix T and the conditional distributions of the observable parameters $p(\boldsymbol{y}|s_k = 0)$ and $p(\boldsymbol{y}|s_k = 1)$. We estimated T simply by counting the transition of states and the conditional distributions $p(\boldsymbol{y}|s_k = 0)$ and $p(\boldsymbol{y}|s_k = 1)$ with a four-dimensional histogram from the training data from 50 pairs.

Now, given the tracking data of a pair of mice, we estimated $s_1, \ldots. s_{1800}$ as follows. We denoted the sequence $\boldsymbol{y}_1, \ldots, \boldsymbol{y}_k$ of observable parameters up to time point k by $\boldsymbol{\Upsilon}_k$. First, the entire sequence $\boldsymbol{Y}_{1800} = (\boldsymbol{y}_1, \ldots \boldsymbol{y}_k)$ of observable parameters was computed from the tracking data. Applying the standard formulas of the HMM, we computed the *a posteriori* distribution $p(s_k|\boldsymbol{Y}_{1800})$ of the state as explained below, and determined the state s_k at time point k as the mode:

$$s_k = \arg\max_{s_k} p(s_k|\boldsymbol{Y}_{1800}). \tag{8}$$

To compute $p(s_k|\boldsymbol{Y}_{1800})$, first, we computed $p(s_k|\boldsymbol{Y}_k)$ in accordance with the following recursive formula:

$$p(s_k|\boldsymbol{Y}_{k-1}) = \sum_{s_{k-1}} p(s_k|s_{k-1}) p(s_{k-1}|\boldsymbol{Y}_{k-1}) \tag{9}$$

$$p(s_k|\boldsymbol{Y}_k) = \frac{p(\boldsymbol{y}_k|s_k) p(s_k|\boldsymbol{Y}_{k-1})}{\sum_{s_k} p(\boldsymbol{y}_k|s_k) p(s_k|\boldsymbol{Y}_{k-1})}. \tag{10}$$

Starting from $k = 0$, Eqs. 8 and 9 were applied repeatedly to compute $p(s_k\text{-}1|\boldsymbol{Y}_k\text{-}1) \to p(s_k|\boldsymbol{Y}_k\text{-}1) \to p(s_k|\boldsymbol{Y}_k) \to p(s_{k+1}|\boldsymbol{Y}_k) \to \ldots$. Here, we note that $p(s_k|s_k\text{-}1)$ is just the elements of the Markov transition matrix T, and $p(\boldsymbol{y}_k|s_k)$ is the conditional distribution of observable parameters estimated in advance. After $p(s_k|\boldsymbol{Y}_k)$ was computed, we computed $p(s_k|\boldsymbol{Y}_{1800})$ in accordance with the following backward recursive formula:

$$p(s_k|\boldsymbol{Y}_{1800}) = p(s_k|\boldsymbol{Y}_k) \sum_{s_{k+1}} \frac{p(s_{k+1}|\boldsymbol{Y}_{1800}) p(s_{k+1}|s_k)}{p(s_{k+1}|\boldsymbol{Y}_k)}. \tag{11}$$

See, for instance, Kitagawa [8] for the derivation of Eqs. 9–11.

Now, we will explain the three-state HMM. In the three-state HMM, the states were "indifference," "sniffing," and "following" (Fig. 2b). "Sniffing" included both social sniffing and genital grooming, whereas "following" included following, aggressive chasing, and attack. Aggressive behaviors were rare in our dataset and contributed little to the category of "following." In the three-state HMM, we considered that a pair of mice took one of the three states s_k at each time point k (1, …, 1800), namely,

$s_k = 0$: indifference (state 0), $s_k = 1$: sniffing (state 1), $s_k = 2$: following (state 2).

Accordingly, the observable parameter y_k at time point k was assumed to obey the distribution $p(y|s_k = 0)$ if the state was indifference, $p(y|s_k = 1)$ if the state was sniffing, and $p(y|s_k = 2)$ if the state was following, aggressive chasing or attack. The Markov transition matrix

$$T = \begin{bmatrix} p_{00} & p_{01} & p_{02} \\ p_{10} & p_{11} & p_{12} \\ p_{20} & p_{21} & p_{22} \end{bmatrix}$$

was as defined in Fig. 2b. These conditional distributions and the Markov transition matrix were estimated in a similar manner to those in the two-state case from the same training data. The procedure for computing $p(s_k|\boldsymbol{Y}_{1800})$ was the same as Eqs. 8–11, except for the definitions described in this paragraph.

4 DuoMouse

We developed the software DuoMouse in order to track, quantify, and estimate the behavior of mice on the basis of the above-mentioned HMM algorithm [4]. DuoMouse was developed as free software with cross-platform compatibility for Windows, Mac OS X, and Linux. Apache Flex SDK 4.9.1 (http://flex.apache.org/) was used to build an Adobe AIR application with an easy-to-use graphical user interface for this software. PHP 5.4.19 (http://www.php.net/) and Zend Framework 1.12.3 (http://framework.zend.com/) were used to exchange data between DuoMouse and an Apache 2.2.15 web server running on a Linux operating system. GNU Octave 3.4.3 (http://www.gnu.org/software/octave/) was used for HMM analysis on the server.

The HMM analysis in DuoMouse requires a dataset that contains four observable parameters as input: (1) distance between the two mice, (2) change in distance between the two mice (relative speed), (3) relative angle of the velocities of the two mice, and (4) average speed of the two mice. The dataset is provided in the comma-separated values (CSV) file format, and prepared automatically using the video tracking system that is incorporated into DuoMouse (Fig. 3a).

To examine the results of the analysis of behavioral states, DuoMouse includes a visualization system that can display the estimated behavioral states and four observable parameters as charts that are synchronized temporally with the video. The time ranges of the charts shift automatically as the video is viewed. Thus, users can compare the estimated behavioral states and video images easily at

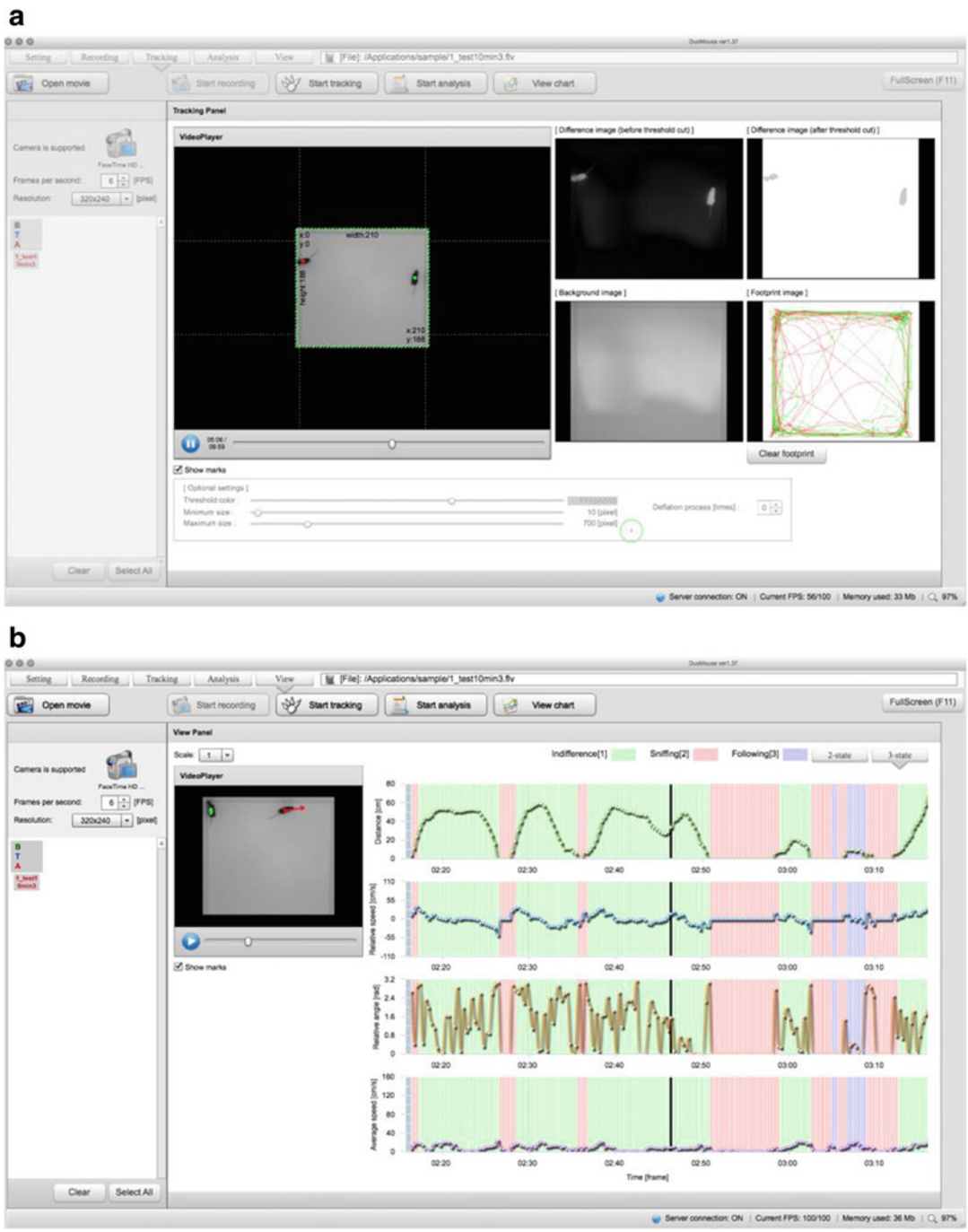


Fig. 3 Operation screen of DuoMouse. (**a**) Mouse movements are tracked and converted to data after a movie of the mice had been loaded. (**b**) The state of the mice is estimated by the HMM and the results of the estimation are shown as a time-series graph. In the time-series graph for the two-state estimation, indifference is shown as a *green bar* and interaction is shown as a *red bar*. On the other hand, in the time-series graph for the three-state estimation, indifference is shown as a *green bar*, sniffing is shown as a *red bar*, and following is shown as a *blue bar*

each time point. The direction in which each mouse will move subsequently is shown by an arrow. This is helpful in the observation of the mouse movements, particularly when they exhibit the behavioral state of following (Fig. 3b). This software package is now available freely online (http://www.nig.ac.jp/labs/MGRL/DuoMouse.html).

5 Reliability of HMM Estimation

To examine the reliability of the HMM estimation, the rate of concordance between it and detection by human observation was estimated by focusing on the occurrence of a social behavior at 1800 time points. Here, a social behavior means "interaction" in the two-state model and "sniffing" or "following" in the three-state model. We defined "concordance" as an event for which the HMM estimate of behaviors matched the occurrence of a social behavior as determined by a human expert. For instance, if we focused on interaction in the two-state model, an event was determined to be concordant when both the HMM and the human observer detected an interaction. In calculating the rate of concordance, we allowed the HMM a six-point window, in which the event was considered to be concordant if the HMM estimated a behavior to have occurred within the range of three time points (which corresponded to one second each) before or after the time point at which the behavior was noted by human observation [4]. This is required to take into account the slight difference in timing between human cognitive response and machine response to a video image. We defined a "false negative" to be an event recorded as social behavior by the human observer but not by the HMM. In contrast, a "false positive" was defined as an event that the HMM, but not the human observer, identified as social behavior. We examined the concordance at each of the 1800 time points, and then calculated the percentage concordance using the following formula:

$$(\text{percentage concordance}) = \mathrm{TP}/(\mathrm{TP} + \mathrm{FN} + \mathrm{FP}),$$

where TP is the number of "concordant" events (TP is an abbreviation of "true positive"), FN is the number of "false negative" events, and FP is the number of "false positive" events.

The average percentage concordance between the results obtained by human observation and the HMM analyses was calculated for 50 pairs of mice, to be used as the training data for the HMM. The percentage concordance for two states is shown in Fig. 4. In the results, the minimum, maximum, and average percentage concordance levels for two states for 50 pairs of mice were 55.2, 95.5, and 76.6, respectively. Thus, it can be said that the

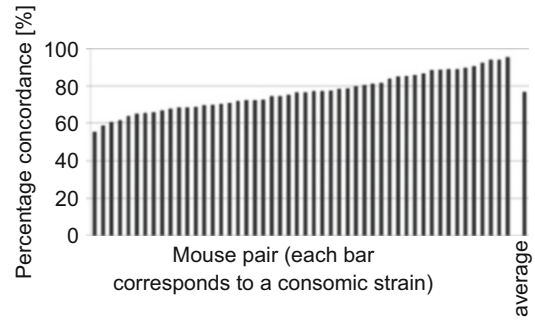


Fig. 4 Percentage concordance for the two-state HMM. Each bar represents the percentage concordance between the results from human observation and the HMM analyses for each pair of mice

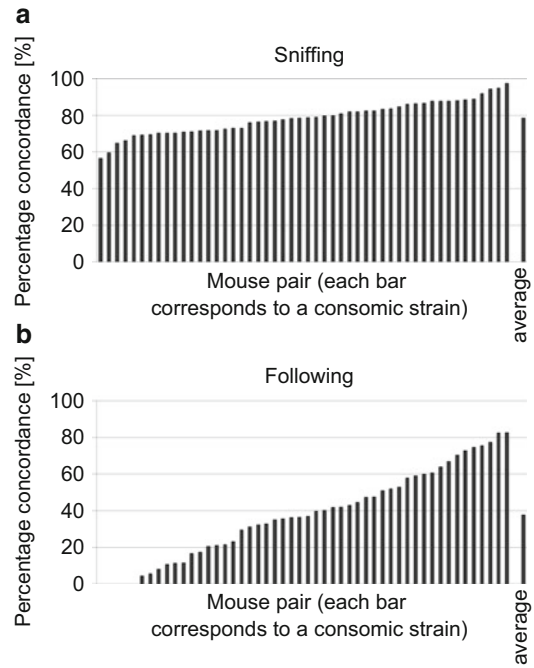


Fig. 5 Percentage concordance for the three-state HMM. Percentage concordance for the sniffing state (**a**) and following state (**b**). Each bar represents the percentage concordance between the results from human observation and the HMM analyses for each pair of mice

results of estimation by the HMM coincided well with the results of estimation by human observation. The percentage concordance for three states is shown in Fig. 5. From Fig. 5a, we can see that the minimum, maximum, and average percentage concordance levels for sniffing were 56.7, 97.4, and 78.6, respectively. In contrast, the minimum, maximum, and average percentage concordance levels

for "following" were 0, 82.8, and 37.7, respectively (Fig. 5b). Thus, in the three-state model, sniffing behavior was segmented well by the HMM, whereas the classification of "following" behavior by the HMM could still be improved substantially in terms of percentage concordance. A possible reason for this is that "following" is exhibited rarely as compared with sniffing behavior. We also observed that, although this scenario did not occur often, the current system regarded two mice running in parallel as interactive behavior. However, it should not be difficult to eradicate such errors by post-processing. Thus, we anticipate that the percentage concordance with respect to "following" will be improved further.

6 Gaussian Mixture Model

So far, we have estimated the conditional density $p(\boldsymbol{y}_k|s_k)$ on the basis of a histogram, which seemed to work reasonably well. This is partly because the number of dimensions of $\boldsymbol{y}$ in our model was relatively low, four. It is known that a histogram density estimate suffers from the problem of dimensionality and it is difficult to estimate a higher-dimensional density distribution accurately by this approach. To detect more complex behavioral states, we might need to increase the number of dimensions of observable parameters. However, this would increase the number of dimensions of conditional density $p(\boldsymbol{y}_k|s_k)$, so that a histogram density estimate might no longer be applicable. A well-known alternative to density estimation, which might be applicable at higher dimensions, is the Gaussian mixture model. As a preliminary step before further development of the approach, we implemented a version of the HMM that used a Gaussian mixture model to estimate $p(\boldsymbol{y}_k|s_k)$, instead of a histogram. We used an object of the *gmdistribution* class of MATLAB to estimate the Gaussian mixture model, where the variance–covariance matrix of each component Gaussian was assumed to be diagonal. We selected the number of components by Akaike's Information Criterion (AIC). The results show that the number of Gaussian elements of state 0 (indifference) was 10, that of state 1 (sniffing) was 9, and that of state 2 (following) was 10. To enable comparison of the estimates with the histogram and Gaussian mixture model, we show the two-dimensional marginal conditional density of $p(\boldsymbol{y}_k|s_k)$ as estimated by both methods in Fig. 6. It can be seen that the two profiles agreed reasonably well.

Figure 7 shows the difference in percentage concordance for the three-state model between the HMM based on a histogram and the HMM based on the Gaussian mixture model. It can be seen that the percentage concordance for sniffing for the HMM based on the Gaussian mixture model was slightly better than for that based on a histogram. The average percentage concordance levels for sniffing for the HMM based on the Gaussian mixture model and

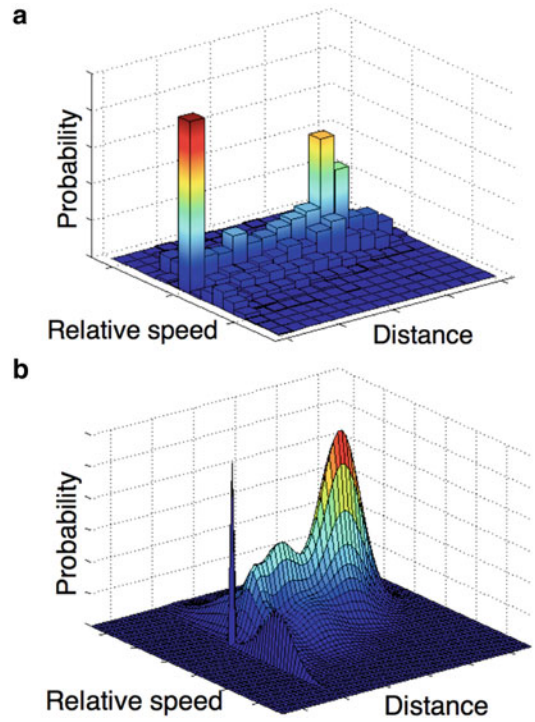


Fig. 6 Estimated conditional densities using a histogram (**a**) and the Gaussian mixture model (**b**) for indifference

for the HMM based on a histogram were 80.3 and 78.6, respectively. In contrast, the average percentage concordance levels for "following" for the HMM based on the Gaussian mixture model and for the HMM based on a histogram were 30.0 and 37.7, respectively. Thus, the percentage concordance for following for the HMM based on the Gaussian mixture model was lower than that for the density histogram by 7.7 %. Although the performance with the Gaussian mixture model for following was slightly worse than that with the histogram in the current model, it is certainly worthwhile applying an HMM that is based on a Gaussian model with higher-dimensional state space when detecting the state of following. In our opinion, if the state space is designed properly, there is a good chance that an HMM based on a Gaussian model with higher-dimensional state space can outperform an HMM based on a histogram.

7 Concluding Remarks

In this chapter, we introduce a system for the automated classification of two social interactive states of two mice that is based on an HMM. The system worked reasonably well for the classification of indifference, sniffing, and following. In general, an expert observer

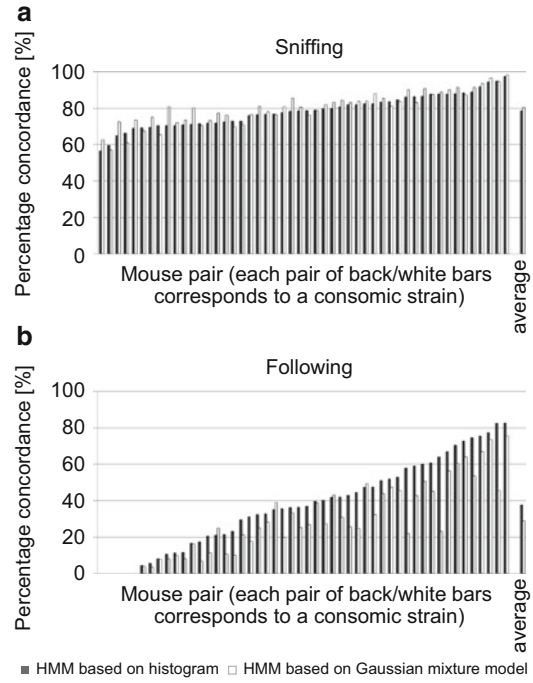


Fig. 7 The percentage concordance for HMMs based on a histogram and the Gaussian mixture model, respectively, for sniffing (**a**) and for following (**b**). Each bar represents the percentage concordance for each pair of mice

can distinguish further social aggressive states such as "chasing" and "attacking." The development of automated detection of these states would thus be an interesting topic for further research. The incorporation of additional information in terms of new parameters for mouse movements might help in the classification of chasing and attacking.

References

1. Takahashi A, Nishi A, Ishii A et al (2008) Systematic analysis of emotionality in consomic mouse strains established from C57BL/6J and wild-derived MSM/Ms. Genes Brain Behav 7:849–858
2. Takahashi A, Tomihara K, Shiroishi T et al (2010) Genetic mapping of social interaction behavior in B6/MSM consomic mouse strains. Behav Genet 40:366–376
3. Arakawa T, Takahashi A, Tanave A et al (2012) A Markov transition score for characterizing interactive behavior of two animals and its application to genetic background analysis of social behavior of mouse. Proc Measur Behav 2012:279–282
4. Arakawa T, Tanave A, Ikeuchi S et al (2014) A male-specific QTL for social interaction behavior in mice mapped with automated pattern detection by a hidden Markov model incorporated into newly developed freeware. J Neurosci Methods 234:127–134
5. Rabiner LR (1989) A tutorial on hidden Markov models and selected applications in speech recognition. Proc IEEE 77:257–286
6. Eddy SR (1996) Hidden Markov models. Curr Opin Struct Biol 6:361–365
7. Eddy SR (2004) What is a hidden Markov model? Nat Biotechnol 22:1315–1316
8. Kitagawa G (1987) Non-Gaussian state-space modeling of nonstationary time series. J Am Stat Assoc 82:1032–1063

Chapter 15

Modeling Movement Primitives with Hidden Markov Models for Robotic and Biomedical Applications

Michelle Karg and Dana Kulić

Abstract

Movement primitives are elementary motion units and can be combined sequentially or simultaneously to compose more complex movement sequences. A movement primitive timeseries consist of a sequence of motion phases. This progression through a set of motion phases can be modeled by Hidden Markov Models (HMMs). HMMs are stochastic processes that model time series data as the evolution of a hidden state variable through a discrete set of possible values, where each state value is associated with an observation (emission) probability. Each motion phase is represented by one of the hidden states and the sequential order by their transition probabilities. The observations of the MP-HMM are the sensor measurements of the human movement, for example, motion capture or inertial measurements. The emission probabilities are modeled as Gaussians. In this chapter, the MP-HMM modeling framework is described and applications to motion recognition and motion performance assessment are discussed. The selected applications include parametric MP-HMMs for explicitly modeling variability in movement performance and the comparison of MP-HMMs based on the loglikelihood, the Kullback–Leibler divergence, the extended HMM-based F-statistic, and gait-specific reference-based measures.

Key words Hidden Markov models, Movement analysis, Gait analysis, Rehabilitation, Robotics

1 Introduction

Automatic analysis of human movement data has attracted increasing interest during the last decade, with applications in human–computer interaction [1, 2], affective computing [3–5], computer graphics [6], surveillance monitoring [7, 8], robotics [9, 10], sports monitoring [11], and biomechanical, clinical, and physiotherapeutic motion assessment [12–14]. This wide range of applications can be subdivided into two broad problem types: *recognizing movement primitives* or *assessing movement performance*. Movement primitives (MP) are elementary motion units and can be interpreted as motor schemas or motion control modules [15]. MPs can be combined sequentially or simultaneously to compose complex tasks or movement sequences [15]. Examples of

David R. Westhead and M.S. Vijayabaskar (eds.), *Hidden Markov Models: Methods and Protocols*, Methods in Molecular Biology, vol. 1552, DOI 10.1007/978-1-4939-6753-7_15, © Springer Science+Business Media LLC 2017

MPs are motions of individual body parts, e.g., gestures, or more complex full-body motions, such as walking, standing, or falling.

When addressing the MP recognition problem, the goal is to recognize which MP is currently being performed, given a library of known MPs. A key challenge with MP recognition is to identify the common pattern of the MP, in the face of multiple sources of variability. Sources of variability in motion performance include both intrapersonal and interpersonal factors. Intrapersonal factors include stochastic components in muscle recruitment and motion control, the influence of factors such as fatigue, age, gender, mood, emotions, and medical conditions, the influence of external factors, e.g., different footwear, obstacle avoidance, or weight of an object being lifted, and the use of different motion strategies or compensatory movements. Interpersonal sources of variability include differences in body size and ability, age, and gender. Algorithms have been developed for action and gesture recognition for human–computer interaction [1, 2], surveillance monitoring [16–18], and human movement imitation in robotics [9, 10].

When movement performance is being assessed, differences in the way a task or a motion is completed are detected and analyzed. Here, the goal is to disambiguate the effect of the factor of interest (e.g., the effect of treatment) from the other variability sources. Because of the many possible sources causing motion variability, different applications have evolved. Individual gait patterns are subject-specific and it is possible to identify subjects by observing their gait [3, 7, 8]. Emotions influence motion performance and the affective state can be recognized from observing changes in functional (e.g., gait), communicative (e.g., gestures), or artistic movements (e.g., dancing) [3–5]. Algorithms assessing exercise performance can monitor fatigue or estimate patient progress during the rehabilitation [11, 13]. In clinical gait analysis, gait conspicuities can be automatically detected from the motion capture measurements [12, 19].

The inputs to movement primitive modeling algorithms are measurements of human movement, consisting of time-series data of the position of each limb at each time instant of the movement. Human movement data can be recorded with video cameras, motion capture systems, inertial measurement units, and on rare occasions with equipment such as pressure sensors [1, 4, 11–13, 17, 18]. Depending on the measurement source, the time series data can describe the body joint angles, the Cartesian position of the limbs, the ground contact forces, or the effective joint torques. The recorded timeseries datasets are characterized by high dimensionality, temporal dependencies, high variability, correlations between timeseries, and nonlinear relationships [19]. Considering the many sources of variability in motion performance, the number of observed motion samples is often small. An efficient and comprehensive representation for movement primitives, which can

be learned on a small number of samples and can model known and unknown variabilities, temporal dependencies, and correlations between timeseries of the joints is desired. Hidden Markov Models (HMMs) are efficient graphical models for sequential datasets and have been applied to speech recognition, handwriting recognition, and genome sequence analysis [20–22]. An HMM consists of a sequence of hidden states and observation distributions for each hidden state. When an HMM is used to model a movement primitive, the hidden states are the phases of the movement primitive and the observations are the sensor measurements. The temporal evolution of the phases of a movement primitive is encoded in the state transitions of such a movement primitive HMM (MP-HMM). The observation distributions model the variability in movement performance and the correlations between the observations of the joints or limb positions. Furthermore, the observation distributions can be extended to explicitly model the influence of individual factors on the movement by using a parametric HMM. HMMs are generative models and the most likely timeseries can be generated from a learned MP-HMM. This enables a visual analysis of the learned MP-HMMs and the ability to reproduce the movement on a robot or exoskeleton.

This chapter is structured as follows: Subheading 2 discusses the recording, characteristics, and preprocessing of movement datasets needed for learning MP-HMMs. Subheading 3 introduces the modeling framework of MP-HMMs. Several techniques for comparing MP-HMMs are described with their application to movement analysis. Additionally, parametric HMMs are discussed for explicitly modeling the influence of individual factors on motion performance. An overview of automatic human movement analysis is provided in Fig. 1.

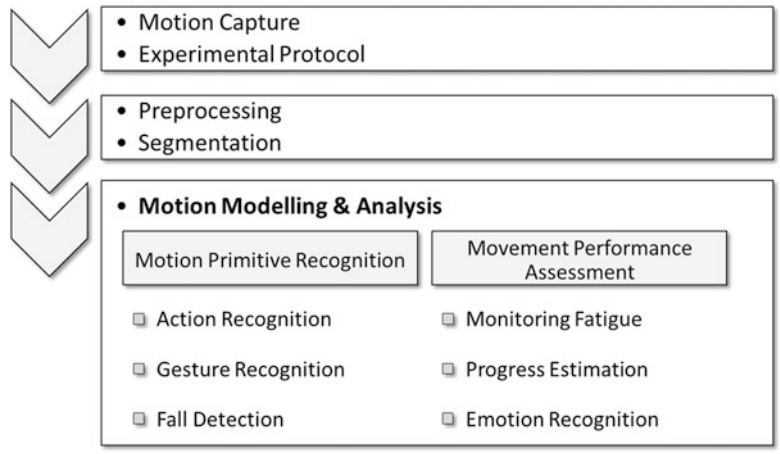


Fig. 1 Automatic analysis of human movement primitives

2 Data Collection and Preprocessing

In the following, an overview of human movement dataset data collection, experimental protocol design, and preprocessing is provided. The primary field of application is briefly summarized for the three most commonly used sensor technologies. Common attributes of motion datasets are listed, and a function of the experimental protocol design. The necessity of low-pass filtering and segmentation for modeling movement primitives with HMMs is discussed.

2.1 Motion Recording

Motion data can be recorded with video cameras, motion capture systems, and inertial measurement units [1, 4, 11–13, 17, 18, 23]. Video, stereo, and structured-light cameras are easily available and depth camera systems such as the Kinect system already include the estimation of joint angle data from the camera data. Camera based recording systems are often highly sensitive to changes in the illumination and motion reconstruction can be affected by occlusion of body parts. For this reason, video recording is chosen for unobtrusive motion recording and for stationary environments, e.g., for human computer interactions [1, 4, 7, 24]. Optical motion capture systems are the golden standard in motion capture and provide highest data accuracy. They are based on either passive or active markers that are placed on the subject. These markers are tracked by several infrared cameras and the motion is reconstructed from the measured marker positions. These sophisticated systems are expensive and require a setup time prior to recording. For these reasons, optical motion capture finds application in clinical and scientific studies [9, 12, 19]. Inertial measurement units (IMU) measure angular velocity and linear acceleration. The use of IMU sensors for motion capture has attracted large interest because IMU sensors are light-weight, cheap, and can be easily placed on the limbs. Recently IMU-based devices have been developed as an accurate unobtrusive motion capture system for biomechanical, physiotherapeutic, and clinical applications [23, 25, 26].

The sensor technology is selected based on the desired temporal and spatial resolution of the data, the study purpose, and the intended application. The raw sensor measurements can be either 2D or 3D image coordinates, marker positions, velocity, or acceleration measurements. From these measurements, bodily configurations can be computed and the joint angle time series can be reconstructed [26]. Joint angle timeseries are often more easily interpreted than the raw sensor measurements. The observation distributions of the HMM can model either the raw sensor measurements or the reconstructed joint angle timeseries, e.g., [12, 13, 24].

2.2 Dataset Characteristics and Experimental Protocol

Recorded motion capture datasets are characterized by

1. the number of factors (sources of movement variability) considered in the study,
2. the number of subjects and demographic properties of the study group,
3. the number of recorded movement primitives, and the number of repetitions of each movement primitive.

The demographic information includes the age, weight, height, and gender of the subjects and is collected as part of the experimental protocol. Additional information can be limb length, level of fitness, BMI, and further study-related information. During the design of the experimental protocol, the internal and external factors to be included in the study are selected. Examples include healthy versus pathological movement, normal versus fatigued movement, or slow versus fast movement. Studies are usually limited to a small number of factors. Ground truth labels for the factors are either collected during the data collection or afterwards by manual or automatic annotation. For the evaluation of the performance of HMMs, these ground truth labels are desirable, but they may be only partly available, available only as qualitative information, or not available in real-life situations, e.g., during inconspicuous recording of patient progress as part of the patient's rehabilitation program [13]. Methodologies to learn condition-dependent information with an HMM-based approach are addressed in Subheadings 3.3–3.7.

2.3 Preprocessing and Segmentation

The data reprocessing required depends on the sensor technology used for data collection. As human motion occurs in a known frequency range, a common preprocessing step is low-pass filtering to remove high-frequency sensor noise. The cut-off frequency for position data is approximately 6–10 Hz [27]. Higher-frequency components and spikes can be observed for acceleration measurements during high-impact movements, e.g., vertical acceleration spikes at the ankle during walking or jumping when the foot contacts the ground.

To extract an individual movement primitive from a continuous recording of multiple primitives, motion timeseries can be segmented into windows with fixed or variable length. In the following, it is assumed that a segment corresponds to the length of a movement primitive. This can be achieved by segmenting the data into *movement primitive segments*, e.g., separating gait cycles by the detection of foot-ground-contact events [3, 12] or separating squats by the detection of extended knee positions [11]. Furthermore, HMMs can be employed to segment timeseries, to recognize movement primitives, or to perform both tasks simultaneously [28, 29]. When each state of the HMM represents a movement primitive, the

Viterbi algorithm is used to find the sequence of movement primitives for a given timeseries [29]. When the states of the HMM represent the phases of a movement primitive, subsections of the given timeseries are first identified using velocity-based features and afterwards HMM template matching is employed, e.g., for segmenting repetitions of rehabilitation exercises [28]. The required accuracy of the start and end points of the movement primitive segments depends on the application. Motion performance assessment can result in less reliable estimates when parts of a movement primitive segment are missing.

3 Movement Primitive HMMs (MP-HMM) Modeling Framework

The MP-HMM modeling framework is described in the following. Learning MP-HMMs includes the configuration, initialization, and optimization of the model parameters. Different methods for comparing MP-HMMs are discussed, based on the application area. Furthermore, an approach for explicitly modeling individual factors influencing movement through a parametric HMM is described.

3.1 MP-HMMs

An HMM is a graphical model for sequential data analysis [20, 30]. The model consists of a number of hidden states $q_1, \ldots, q_n$, that are not directly observable. These *hidden states* can represent either action primitives [10] or phases of a movement [2, 9, 11, 12, 24, 28, 31]. A sequence of hidden states models the temporal evolution of a timeseries. The transitions between the hidden states are defined by the *transition probability matrix* A with $a_{ij} = P[q_{t+1}|q_t]$. It is assumed that the hidden states are unknown and can be estimated from a set of observations. The observations are the preprocessed data streams of the sensor recordings. In an HMM, the occurrence of an observation is modeled by the conditional probability $P(v_j|q_t)$ of observing v_j in state q_t. For discrete observation symbols, an emission matrix B can be learned that contains the probabilities for observing each symbol when in one of the hidden states q. However, motion data is recorded over a continuous range. Discretization of this range enables the use of contingency tables, but this is only practicable for a small number of observations. To enable a more detailed representation of movements, the *emission probabilities* of MP-HMMs are formulated as Gaussian distributions [11, 13, 31]. For each state q_i, a multivariate Gaussian distribution $N(\mu_i, \Sigma_i)$ is learned and the variability in movement is modeled by Σ_i. The off-diagonal elements of Σ_i further include information about the correlations between different observations, e.g., between the joint angles. Given k observations, the dimensionality of the mean μ_i is $\mathbb{R}^k$ and the covariance matrix $\Sigma_i \in \mathbb{R}^{k \times k}$. The observation distribution can be extended to a

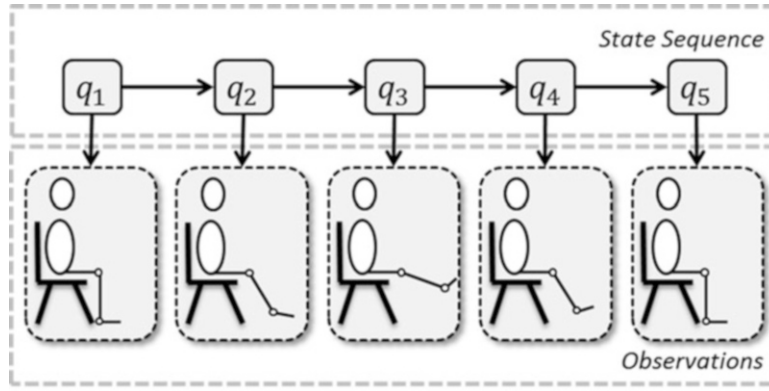


Fig. 2 Five-state MP-HMM representing the phases of a knee-extension exercise

mixture of Gaussians if the observations are distributed over several clusters, or can be parameterized, e.g., by formulating a parametric HMM [2, 11].

The sequential order of different actions can be modeled by a single HMM, where each state is an action primitive and the emission probabilities are learnt for each action. Learning an HMM in such a way is suitable for activity detection. For a detailed analysis of movement performance or detection of similar activities, it is advantageous to model each movement primitive by an MP-HMM [2, 11-13, 24, 28, 31]. In the MP-HMM as illustrated in Fig. 2, the states model the phases of a movement primitive and the temporal development of a movement primitive can be modeled in more detail.

3.2 MP-HMM Learning

An MP-HMM λ with continuous observation densities is characterized by the following parameters:

1. the number of states n,
2. the transition probability matrix A including the topology of the Markov chain,
3. the multivariate observation distributions $N(\mu_i, \Sigma_i)$ for each state q_i,
4. the initial state distribution $\pi \in \mathbb{R}^n$.

The Baum–Welch optimization finds the optimal values for the prior π, A, and $N(\mu_i, \Sigma_i)$ given the number of states n, a training set $V = [v^1, v^2, \ldots]$, and the initial values for the prior π, A, and $N(\mu_i, \Sigma_i)$ [20]. The training set V for a model λ contains observation vectors $v^q \in \mathbb{R}^{k \times t_{\max,q}}$ of length $t_{\max,q}$ of the same movement primitive. The length $t_{\max,q}$ can be different for each sequence q. The Baum–Welch algorithm is an expectation-minimization optimization that finds a local optimum that depends on the

initialization. The stopping criterion of the iterative Baum–Welch algorithm is either a maximum number of iterations, a threshold for the loglikelihood $P(V|\lambda)$ that the observation sequences are generated by the learned MP-HMM, or a threshold for the rate of change of $P(V|\lambda)$. The values of $P(V|\lambda)$ depend on the length of the sequences and the accuracy of the model λ. The better the model λ fits the training set V, the higher $P(V|\lambda)$. As the Baum–Welch optimization depends on its initialization, these parameters are discussed in the following:

3.2.1 Number of States

The states of the MP-HMM represent the phases of a movement primitive. Biomechanical interpretations can be used to associate the states with functional phases of a movement, e.g., with the swing, the stance, and the ground contact phase during walking. The larger the number of states, the more detailed the HMM-based representation of the movement primitive. The number of states can be defined a priori, e.g., based on a biomechanical subdivision of the movement primitive into a set of functional phases [12] or the optimal number of states is determined computationally [11]. For a squat exercise, the optimal number of states n ranges from 6 to 12 [11]. A small n facilitates a biomechanical interpretation of the states and faster computation of the Baum–Welch algorithm.

3.2.2 Transition Probability Matrix **A** and Initial State Distribution $\boldsymbol{\pi}$

The state transition probability $A \in \mathbb{R}^{n \times n}$ contains the probabilities $a_{ij} = P\left[s_{t+1} = q_j | s_t = q_i\right]$, that the system will transition to state q_j at a given timestep if its current state is q_i. The expected state duration $\bar{d}_{ii} = \frac{1}{1-a_{ii}}$ can be estimated from the diagonal elements of A. When every state is reachable $a_{ij} \neq 0 \ \forall i, j$, the system is called *ergodic*. The initial state distribution $\pi \in \mathbb{R}^n$ is a vector of the probabilities $P\left[s_{t=0} = q_i\right]$ that the system is initially in one of the states q_i. Either a random or uniform initialization are suitable for A and π [20].

For sequential data of human movement primitives, it can be assumed that the temporal evolution of the state follows a sequence and transitions occur only to adjacent states. This can be described by a *left-to-right model*, where the elements of A are non-zero on the main diagonal, the first diagonal above the main diagonal, and a_{1n}:

$$A = \begin{bmatrix} a_{11} & a_{12} & & & 0 & 0 \\ 0 & a_{22} & \cdots & & 0 & 0 \\ \vdots & & \ddots & & \vdots & \\ 0 & 0 & & a_{n-1,n-1} & a_{n,n-1} & \\ a_{1n} & 0 & \cdots & 0 & a_{n,n} & \end{bmatrix}$$

The left-to-right model is illustrated for a knee extension exercise in Fig. 2. If $a_{1n} \neq 0$, the left-to-right model is *cyclic* and the data sequence can include repetitions of the movement primitive. The prior π can be uniformly distributed. If $a_{1n} = 0$, the left-to-right model is *non-cyclic* and it is necessary to segment the data sequence into movement primitives after preprocessing. In this case, the prior is usually defined as $\pi = [1, 0, 0, \ldots]$ assuming that each segment begins with an observation of the first state q_1. When initializing the state transition matrix A for the left-to-right model, the elements of the main diagonal a_{ii} can be either uniformly distributed or approximated by biomechanical estimates of the state durations. Subsequently, the elements on the first diagonal above the main diagonal are $a_{i,i+1} = 1 - a_{ii}$.

3.2.3 Emission Probabilities

The multivariate observation distributions $N(\mu_i, \Sigma_i)$ are learned for each state q_i. The Baum–Welch algorithm finds solutions in the neighborhood of the initial values for the emission probabilities. For this reason, the Gaussian distributions are either initialized based on a few selected exemplars of the movement primitive or a set of segments. When a few exemplars are selected, the sequences can be manually subdivided into regions corresponding to one of the states and an initial estimate for $N(\mu_i, \Sigma_i)$ can be computed for the observations of each state q_i. When a larger set of movement primitive segments is available, these segments can be automatically subdivided into k regions given an initial estimate of A for a left-to-right MP-HMM. The length of each region i is defined by its relative state duration $\frac{\bar{d}_{ii}}{\sum_{i=1}^{n} \bar{d}_{ii}} \cdot t_{\max}$ for a sequence of total length $t_{\max}$. The data of each sequence is distributed into n bins. For each bin, the mean μ_i and the covariance Σ_i are computed providing an initial estimate for (μ_i, Σ_i). For a small number of training samples, the reestimation of the covariance Σ_i as part of the Baum–Welch algorithm can be improved by (1) assuming independence between the observations, $\sigma_{ij} = 0 \forall i \neq j$, or (2) constraining the elements of Σ_i with upper and lower limits.

3.3 Movement Primitive Recognition based on the Conditional Probability $P(v|\lambda)$

When the goal is to use MP-HMM models to recognize a newly observed movement from a set of known movements, an MP-HMM $\lambda^c = \{A^c, N(\mu_i{}^c, \Sigma_i{}^c), \pi^c\}$ is learned for each movement primitive $c = 1, \ldots c_{\max}$,, e.g., walking, running, and squatting. This set of MP-HMMs $\{\lambda^1, \ldots, \lambda^{c_{\max}}\}$ is defined as the *movement primitive dictionary*. Given a new observation sequence $v = [v_1, \ldots, v_{t_{\max}}]$, the conditional probability $P(v|\lambda^c)$ is computed that the observation sequence v is generated by each MP-HMM λ^c. As the straight-forward computation of $P(v|\lambda^c)$ is computationally intensive, the computationally more efficient forward-backward algorithm is applied [20]. The conditional probabilities $P(v|\lambda^c)$

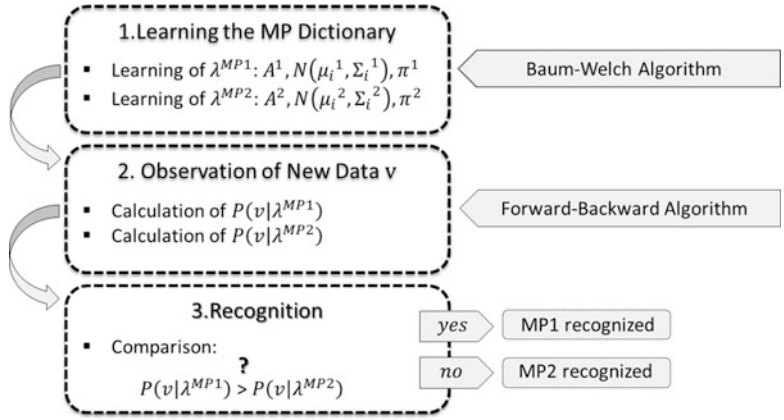


Fig. 3 MP-HMM concept for recognizing motions illustrated for two movement primitives (MP)

are computed for all MP-HMM λ^c and are compared with each other. It is most likely that the observation sequence v has been generated by the model MP-HMM λ^c with the highest $P(v|\lambda^c)$. Figure 3 illustrates the procedure. This approach can be also used for movement performance assessment, if the assessment can be discretized into a set of classes. In this case, an MP-HMM is learned for each class of movement performance considered, e.g., happy, sad, neutral, and angry gait [32].

3.4 Comparison of MP-HMMs based on the Kullback–Leibler Divergence

The similarity between two HMMs λ^1 and λ^2 can be estimated by the Kullback–Leibler (KL) divergence:

$$D(\lambda^1, \lambda^2) = \frac{1}{t_{\max}} \left[\log P(v^1|\lambda^2) - \log P(v^1|\lambda^1)\right]$$

where v^1 is an observation generated by the model λ^1 [20]. The KL-divergence compares the loglikelihood that an observation v^1 has been generated either by the model λ^1 or by a different model λ^2. When the two models λ^1 and λ^2 are similar, $D(\lambda^1, \lambda^2)$ is small. The evaluation of the similarity is based only on the observation of λ^1 and $D(\lambda^1, \lambda^2)$ is non-symmetric. The symmetric version is

$$D_s(\lambda^1, \lambda^2) = \frac{D(\lambda^1, \lambda^2) + D(\lambda^2, \lambda^1)}{2}.$$

The KL-divergence can be used for recognition and clustering of movement primitives [9]. MP-HMMs are learned using the training data of each movement primitive. For recognition, a new MP-HMM $\widetilde{\lambda}$ is learned for each new observation sequence v. The KL-divergence is computed between the new MP-HMM $\widetilde{\lambda}$ and the previously learned MP-HMMs λ^c. The sequence v will be recognized as the movement primitive c for which $D_s\left(\widetilde{\lambda}, \lambda^c\right)$ is smallest. When the similarities $D_s\left(\widetilde{\lambda}, \lambda^c\right)$ between the newly observed MP-HMM $\widetilde{\lambda}$ and the MP-HMMs λ^c included in the movement primitive

dictionary exceed a defined threshold, $\tilde{\lambda}$ can be added as a new movement primitive to the dictionary. This enables online learning of movement primitive dictionaries and trees [9]. In [10], a movement primitive is represented by a single state and the KL-divergence can be simplified to the comparison of two Gaussian distributions $D_{\mathrm{KL}}(\lambda^1, \lambda^2) = \frac{1}{2}\left(\log\frac{|\Sigma_2|}{|\Sigma_1|} + \mathrm{tr}(\Sigma_2^{-1}\Sigma_1) + (\mu_2 - \mu_1)^T \Sigma_2^{-1}(\mu_2 - \mu_1) - k\right)$.

3.5 Parametric HMM

Human movement exhibits a large variability in motion performance. This variability is caused by external and internal factors. Within the MP-HMM framework, variability in movement performance is modeled by the covariance of the emission probabilities. In addition, sources of variability can be explicitly modeled by a parametric HMM (PHMM). In a PHMM, the observation distribution $N(\mu_i, \Sigma_i)$ depends linearly on the parameter Θ

$$\hat{\mu}_i(\Theta) = W_i\Theta + \overline{\mu}_i$$

with the weight matrix $W_i \in \mathbb{R}^{k\times d}$ [2, 10, 11]. The parameter $\Theta \in \mathbb{R}^d$ can be a single factor with $d = 1$ (e.g., the level of fatigue [11]) or a vector combining several factors with $d > 1$ (e.g., final reaching position in 2D-coordinates [2]). The weight matrix W_i is learnt separately for each state i so that a state-dependent influence of the parameter Θ on the observation distributions can be modeled. To extend the Baum–Welch algorithm, the linear dependence of $N(\mu_i, \Sigma_i)$ on Θ is formulated in the following form:

$$\hat{\mu}_i(\Theta) = Z_i\Omega_i \text{ with } Z_i = [W_i, \mu_i],\ \Omega_i = [\Theta\ 1]^T.$$

Given a set of training sequences $V = [v_1, \ldots, v_l]$, Z_i is estimated in each iteration of the Baum–Welch algorithm

$$Z_i = \left[\sum_{q=1\ldots l}\sum_t \gamma_{qi}(t) v_q(t) \Omega_q^T\right]\left[\sum_{q=1\ldots l}\sum_t \gamma_{qi}(t)\Omega_q\Omega_q^T\right]^{-1}$$

with the probability $\gamma_i(t)$ of being in state i at time t. The covariance is estimated based on $\hat{\mu}_i$:

$$\Sigma_i = \sum_{q=1\ldots l}\sum_t \frac{\gamma_{qi}(t)}{\sum_t \gamma_{qi}(t)}\left[v_q(t) - \hat{\mu}_i(\theta_q)\right]\left[v_q(t) - \hat{\mu}_i(\theta_q)\right]^T$$

After learning a PHMM, the parameter Θ can be reestimated iteratively from the observation sequence v by

$$\theta = \frac{\sum_{i,t}\gamma_i(t)\, W_i^T\Sigma_i^{-1}[v(t) - \mu_i(t)]}{\sum_{i,t}\gamma_i(t)\, W_i^T\Sigma_i^{-1} W_i}$$

As the reestimation of Θ is an iterative procedure, the initial value Θ_{init} must be specified. PHMMs have been applied to model reaching tasks [2], the influence of fatigue on performing squat exercises [11], and the manipulation of objects [10].

3.6 HMM-based Extension of the F-statistic

The F-statistic is the ratio of two variances and is widely used for hypothesis testing in applied science. For the analysis of variance (ANOVA), the between-groups variability S_{B}^2 is compared with the within-groups variability S_{W}^2. In [31], an approach is proposed to extend the F-statistic to timeseries data. Given the timeseries data of $\xi_{\max} \geq 2$ groups, an HMM is learned for each group λ_ξ, for each set of timeseries $\lambda_{\xi j}$, and a general model λ. Considering a movement primitive dataset containing $\xi_{\max}$ primitives recorded from $j_{\max}$ subjects, these models correspond to

1. a general MP-HMM λ using the full movement primitive dataset for training,
2. $\xi_{\max}$ MP-HMMs for each movement primitive λ_ξ,
3. and $\xi_{\max} \cdot j_{\max}$ individual MP-HMM $\lambda_{\xi j}$ using only the $m_{\max}$ recorded movement primitives of each subject for training.

The MP-HMMs can be compared using the KL-divergence, see Subheading 3.4. The similarity of $\xi_{\max} \geq 2$ groups of timeseries can be assessed by computing the following *within-groups variability* D_{B} and *between-groups variability* D_{W}

$$D_{\text{W}} = \frac{1}{l-c} \sum_{\xi,j} D\left(\lambda_\xi, \lambda_{\xi j}\right)$$

with $D\left(\lambda_\xi, \lambda_{\xi j}\right) = \frac{1}{m_{\max}} \sum_{m} \frac{1}{t_{\max,m}} \left| \log P\left(v^{(m,\xi,j)}\right) \middle| \lambda_\xi \right) - \log P\left(\left(v^{(m,\xi,j)}\right) \middle| \lambda_{\xi j}\right)|$

$$D_{\text{B}} = \frac{1}{c-1} \sum_{\xi} D(\lambda, \lambda_\xi)$$

with $D(\lambda, \lambda_\xi) = \sum_{j} \frac{1}{m_{\max}} \sum_{m} \frac{1}{t_{\max,m,j}} \left| \log P\left(v^{(m,\xi,j)}\right) \middle| \lambda \right) - \log P\left(\left(v^{(m,\xi,j)}\right) \middle| \lambda_\xi\right)|$

where $l = m_{\max} \xi_{\max} j_{\max}$ is the total number of timeseries included in the dataset. The HMM-based extension of the F-statistic $F = \frac{D_{\text{B}}}{D_{\text{W}}}$ using the above formulation for the within-groups and between-groups-variability can be used to compare the similarity of sets of timeseries. This approach has been applied to analyze the difference between normal walking and walking after exhaustion [31]. For this application, the HMM-based extension of the F-statistic provides information about which joint angles during walking are affected by exhaustion. Higher F-statistics are observed for the head and neck backward tilt, shoulder abduction, and shoulder rotation. This observation can be explained by an increased inward posture of the shoulders and the head after exhaustion [31].

3.7 Clinical Gait Analysis Using Reference-Based Measures

In clinical gait analysis, gait patterns are recorded with optical motion capture. The timeseries data is accurately segmented into gait cycles based on force plate data. The gait cycle of patients is compared with the gait of a healthy control group. In [12], reference-based measures have been introduced to compare the gait cycles on three levels: The gait phase deviation reference based index (GPD-RBI) compares individual phases of a gait cycle, the joint function deviation index (JFD-RBI) compares on the joint angle level, and the gait cycle deviation index (GCD-RBI) assesses the similarity of two complete gait recordings. To compute these indices, the gait cycle is represented by an explicit state duration HMM λ given accurate gait cycle segmentation. The number of states n is selected as a multiple of eight considering the biomechanical subdivision of a gait cycle into eight functional phases [33]. Given l gait cycles of a subject $V = [v_1, \ldots, v_l]$, the timeseries are subdivided into n equal parts and $\left\lfloor \frac{t_{\max}}{n} \right\rfloor$ frames of l gait cylces are pooled into n bins. Gaussian distributions are learnt for the observations of each state i. For the computation of the reference-based indices, it is assumed that the observations for the joint angles are independent and $\mathcal{N}(\mu_{ik}, \sigma_{ik}^2)$ is estimated for each joint angle k. The variance estimate σ_{ik}^2 includes both between-strides and within-stride variability. A model λ is learned for each subject and for the healthy reference group λ_{ref}. The GPD-RBI is $\delta_{ik} = \frac{|\mu_{ik} - \mu_{ik,\text{ref}}|}{\sigma_{ik,\text{ref}}}$, the JFD-RBI is $\Delta_k = \frac{1}{n}\sum_{i=1}^{n} \delta_{ik}$, and the GCD-RBI is $\overline{\Delta} = \frac{1}{k_{\max}} \sum_k \Delta_k$. The smaller the RBIs are, the more similar are λ and λ_{ref}. The RBIs can be used for clinical interpretation and biomechanical analysis of individual gait patterns and group comparisons. Using $\overline{\Delta}$ as the input feature for a Naïve Bayes classifier, gait data from healthy and patients suffering from rheumatism can be recognized with 87 % accuracy, 71 % specificity, and 95 % sensitivity [12].

4 Summary

HMMs are graphical models for timeseries data. They can be used to describe movement primitives, where the states of the MP-HMM are the phases of a motion. MP-HMMs can model variations in movement performance, correlations between the observations, and the influence of known factors on movement performance. The HMM framework assumes independence between successive observations and a first order Markov chain modeling the state transitions. It can be further extended to higher-order Markov chains, parallel HMMs, or hierarchical HMMs at the cost of higher computational complexity.

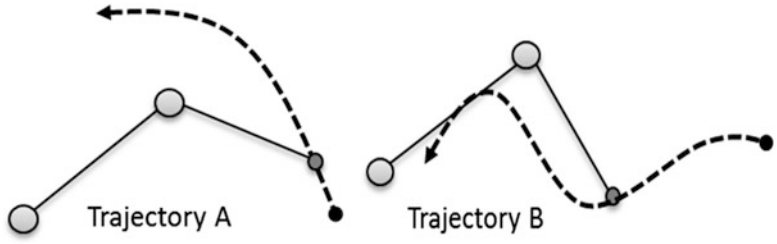


Fig. 4 The 2-dof planar arm model follows either trajectory A or B

5 Programming Exercise

In the following, two motion trajectories of a two degree of freedom (dof) planar arm model are simulated. The arm follows the trajectory A or B illustrated in Fig. 4.

The position of the arm $[x;\ y] = [l_1 \cos\alpha + l_2 \cos(\alpha+\beta);\ l_1 \sin\alpha + l_2 \sin(\alpha+\beta)\ ]$ is given by the limb lengths l_1 and l_2, and the joint angles α and β.

1. Generate sample data for each movement using the following parameterization of the trajectories $t = 0 \ldots .100$;
   Trajectory A: $\alpha = \left(\frac{90^\circ}{100}\right) * t;\ \beta = 0^\circ$;
   Trajectory B: $\alpha = \left(\frac{90^\circ}{100}\right) * t;\ \beta = -(180^\circ/100) * t + \cos(t * 1/10) * 5^\circ$;
2. Add a small noise term to each movement primitive and generate a data set Γ of ten primitives for each movement.
3. Learn an HMM λ^{A} and λ^{B} based on the observations α and β of each movement primitive using the datasets Γ_{A} and Γ_{B}. Model each primitive by a left-to-right HMM with n hidden states.
4. Take a sample observation v^{A} of α and β of Γ_{A} and compute the conditional probabilities $P\left(v^{\mathrm{A}}|\lambda^{\mathrm{A}}\right)$ and $P\left(v^{\mathrm{A}}|\lambda^{\mathrm{B}}\right)$. Compare $P\left(v^{\mathrm{A}}|\lambda^{\mathrm{A}}\right)$ with $P\left(v^{\mathrm{A}}|\lambda^{\mathrm{B}}\right)$.
5. Repeat **steps 2–4** and vary the number of states n of the HMM. How do $P\left(v^{\mathrm{A}}|\lambda^{\mathrm{A}}\right)$ and $P\left(v^{\mathrm{A}}|\lambda^{\mathrm{B}}\right)$ change?
6. Repeat **steps 2–4** and increase the magnitude of the noise. How does the difference between $P\left(v^{\mathrm{A}}|\lambda^{\mathrm{A}}\right)$ and $P\left(v^{\mathrm{A}}|\lambda^{\mathrm{B}}\right)$ change?

References

1. Mitra S, Acharya T (2007) Gesture recognition: a survey. IEEE Trans Syst Man Cybern Part C Appl Rev 37(3):311–324
2. Wilson AD, Bobick AF (1999) Parametric hidden markov models for gesture recognition. IEEE Trans Pattern Anal Mach Intell 21 (9):884–900
3. Karg M, Kuehnlenz K, Buss M (2010) Recognition of affect based on gait patterns. IEEE Trans Syst Man Cybern B Cybern 40 (4):1050–1061
4. Karg M, Samadani A, Gorbet R, Kuehnlenz K, Hoey J, Kulic D (2013) Body movements for affective expression: a survey of automatic

recognition and generation. IEEE Trans Affect Comput 4(4):341–359

5. Kleinsmith A, Bianchi-Berthouze N (2013) Affective body expression perception and recognition: a survey. IEEE Trans Affect Comput 4(1):15–33
6. Lee Y, Wampler K, Bernstein G, Popovic J, Popovic Z (2014) Motion fields for interactive character locomtion. Commun ACM 57 (6):101–108
7. Boulgoris N, Hatzinakos D, Plataniotis K (2005) Gait recognition: a challenging signal processing technology for biometric identification. IEEE Signal Process Mag 22(6):78–90
8. Sarkar S, Phillips P, Liu Z, Vega I, Grother P, Bowyer K (2005) The humanid gait challenge problem: data sets, performance, and analysis. IEEE Trans Pattern Anal Mach Intell 27 (2):162–177
9. Kulić D, Takano W, Nakamura Y (2008) Incremental learning, clustering and hierarchy formation of whole body motion patterns using adaptive hidden markov chains. Int J Rob Res 27(7):761–784
10. Kruger V, Herzog D, Baby S, Ude A, Kragic D (2010) Learning actions from observations. IEEE Robot Autom Mag 17(2):30–43
11. Karg M, Venture G, Hoey J, Kulic D (2014) Human movement analysis as a measure for fatigue: a hidden markov-based approach. IEEE Trans Neural Syst Rehabil Eng 22 (3):470–481
12. Karg M, Seiberl W, Kreuzpointner F, Haas JP, Kulić D (2015) Clinical gait analysis: comparing explicit state duration HMMs using a reference-based index. IEEE Trans Neural Syst Rehabil Eng 23(2):1812–1826
13. Houmanfar R, Karg M, Kulić D (2016) Movement analysis of rehabilitation exercises: distance metrics for measuring patient progress. IEEE Syst J 10(3):1014–1025
14. Simon S (2004) Quantification of human motion: gait analysis—benefits and limitations to its application to clinical problems. J Biomech 37(12):1869–1880
15. Flash T, Hochner B (2005) Motor primitves in vertebrates and invertebrates. Curr Opin Neurobiol 15(6):660–666
16. Wang L, Hu W, Tan T (2003) Recent developments in human motion analysis. Pattern Recogn 36(3):585–601
17. Poppe R (2007) Vision-based human motion analysis: an overview. Comput Vis Image Underst 108(1–2):4–18
18. Moeslund T, Hilton A, Krueger V (2006) A survey of advances in vision-based human motion capture and analysis. Comput Vis Image Underst 104(1–2):90–126
19. Chau T (2001) A review of analytical techniques for gait data. Part 1: Fuzzy, statistical and fractal methods. Gait Posture 13(1):49–66
20. Rabiner L (1989) A tutorial on hidden markov models and selected applications in speech recognition. Proc IEEE 77(2):257–286
21. Plamondon R, Srihari S (2000) On-line and off-line handwriting recognition: a comprehensive survey. IEEE Trans Pattern Anal Mach Intell 22(1):63–84
22. Wu J, Xie J (2010) Hidden Markov model and its applications in motif findings. Statistical methods in molecular biology. Humana Press, New York, pp 405–416
23. Zheng Y, Ding X, Poon C, Lo B, Zhang H, Zhou X, Yang G, Zhao N, Zhang Y (2014) Unobtrusive sensing and wearable devices for health informatics. IEEE Trans Biomed Eng 61(5):1538–1554
24. Kale A, Sundaresan A, Rajagopalan AN, Cuntoor NP, Roy-Chowdhury AK, Kruger V, Chellappa R (2004) Identification of humans using gait. IEEE Trans Image Process 13 (9):1163–1173
25. Brigante C, Basile A, Faulisi A, Sessa S (2011) Towards miniaturization of a MEMS-based wearable motion capture system. IEEE Trans Ind Electron 58(8):3234–3241
26. Lin J, Kulić D (2012) Human pose recovery using wireless inertial measurement units. Physiol Meas 33:2099–2115
27. Winter D (1990) Biomechanics and motor control of human movement. John Wiley & Sons, NJ
28. Lin J, Kulić D (2014) On-line segmentation of human motion for automated rehabilitation exercise analysis. IEEE Trans Neural Syst Rehabil Eng 22:168–180
29. Sanmohan B, Krueger V (2009) Primitive based action representation and recognition. Image Anal LNCS 5575:31–40
30. Bishop C (2006) Pattern recognition and machine learning. Springer, New York
31. Karg M, Seiberl W, Hoey J, Kulic D (2013) Human movement analysis: extension of the F-statistic for time series data using HMM. In: IEEE int. conf. on systems, man and cyberbetics
32. Karg ME (2012) Pattern recognition algorithms for gait analysis with application to affective computing. Doctoral dissertation, Technische Universität München, München
33. Perry JBJ (2010) Gait analysis: normal and pathological function. SLACK Incorporated

INDEX

David R. Westhead and M.S. Vijayabaskar (eds.), *Hidden Markov Models: Methods and Protocols*, Methods in Molecular Biology, vol. 1552, DOI 10.1007/978-1-4939-6753-7, © Springer Science+Business Media LLC 2017

D

E

F

G

H

N

O

P

Q

R

S

T

U

V

W

X

Z