Chapter 14
Biological Sequence Analysis

Simon E. Cawley

Shortly after the start of my graduate studies at the U.C. Berkeley Statistics department in 1995, I had the good fortune to meet Terry and learn about some of his work in the area of the application of statistics to genetics and molecular biology. Not having thought about biology since high school, I was very impressed by the large impact statistical approaches were making in a field I had naively considered as one that had little to do with quantitative analysis. I eagerly dove in to a collaboration that Terry had put in place with the Human and Drosophila Genome Projects at Lawrence Berkeley National Laboratories and spent the next few years having a great time working on interesting and practical statistical problems that arose in the context of the ongoing genome sequencing efforts.

In this section we present some of Terry's contributions in the area of Sequence Analysis – generally speaking, the area of analysis of biological sequences such as DNA or protein sequences. The papers presented here relate to the interpretation of DNA sequences.

DNA sequence analysis has been an area of growing importance since DNA sequencing techniques started to emerge in the early 1970s. The chain-terminator method developed by Frederick Sanger at the University of Cambridge [7] was a pivotal moment, enabling the first rapid scaling up of DNA sequencing capabilities. The rate of sequencing was further accelerated through the 1980s and 1990s as ever-greater levels of automation were brought to bear on Sangers original concept.

As the level of automation increased, it became possible to sequence entire genomes of successively more complex organisms with larger genomes, ranging from bacteriophage phiX174 in the late 1970s, various microbial genomes in the early 1990s through to the draft of the human genome sequence published in 2001. The Sanger method showed remarkable longevity and was at the core of the vast majority of sequencing efforts through to the early 2000s.

The dominance of Sanger sequencing finally ended in the early 2000s with the advent of a renaissance of sorts as multiple new massively parallel technologies such

S.E. Cawley
Ion Torrent
e-mail: simon.cawley@lifetech.com

S. Dudoit (ed.), *Selected Works of Terry Speed*, Selected Works in Probability and Statistics,
DOI 10.1007/978-1-4614-1347-9_14, © Springer Science+Business Media, LLC 2012

as 454 pyrosequencing, followed soon after by Solexa (Illumina), SOLiD, polony, DNA nanoball and Ion Torrent sequencing.

As DNA sequencing technologies scaled up, huge opportunities arose along the way for the application of statistics, both in the area of analysis of the signals generated from each of the various instruments and technologies to improve DNA sequencing accuracy (the subject of Chapter 13), and in the downstream analysis of the DNA sequence collected. In particular, as the volumes of sequence generated started to exceed what an expert molecular biologist could manually browse and interpret, it became crucial to develop statistical models for assembling and interpreting the sequences.

The papers presented in this chapter cover two important areas in the interpretation of DNA sequences. The first, Cawley et al. [3], addresses the problem of analyzing stretches of DNA to search for the collections of sub-sequences that correspond to gene transcripts. The model presented was not the first of its kind; similar Hidden Markov Models (HMMs) had been published before [2, 4, 5]. Its novel contributions were various observations about computational shortcuts that can be made, at no cost to accuracy, taking advantage of some of the structure of the problem of applying HMMs to gene finding. This paper was also the first instance where the probabilistic formulation of the HMM gene finder was used to derive posterior probabilities of bases being part of the gene; previous attempts focused exclusively on the use of the Viterbi algorithm to predict gene structures. The software implementing the gene finder was also the first HMM gene finder made available as open-source software, something of value given the rate at which new organisms were then being sequenced.

As an interesting side note, while doing some of the work that was described in the publication, I had a near-death experience with the very Malaria parasite that was the subject of the work. A pure coincidence – the work had involved nothing more than electronic interaction with the parasite!

The second paper, Zhao et al. [8], introduced the novel concept of a Permuted Variable Length Markov Model (PVLMM), a generalization of the VLMM [1, 6]. VLMMs themselves are a generalization of Markov models. When applied to sequence analysis, they have the advantage of allowing for modeling of long context dependencies without necessarily coming at the cost of an exponential increase in the number of parameters to estimate. However, the dependencies that VLMMs best model are still relatively local dependencies and they are ill-suited to describe long-range dependencies between particular positions in a sequence as sometimes occurs. PVLMMs offer a way around that limitation by providing a framework in which the modeled sequence can be permuted to bring dependent positions together, turning long-range dependencies into local ones.

The paper provides some impressive work, putting the new theory into practice in two substantial applications: modeling of splice sites, a sub-component of gene sequences; and modeling of Transcription Factor Binding Sites (TFBS), important regions of DNA to which regulatory molecules known as transcription factors bind as part of the regulation mechanism for gene expression. By showing effective performance in two different sequence analysis problems, a strong case is made for the PVLMM as a general tool that will be well suited to a broad range of applications.

These papers, along with the diverse range of publications reviewed in the other chapters, provide a sense of the amazing breadth of Terry's work. I am a direct beneficiary of his diverse interests – when he introduced me to the field of statistics applied to molecular biology, I enjoyed it so much that it ended up being the basis of my career to-date. I will always be grateful to him for how selflessly he shared his time and insights, and for the patient guidance he provided during my graduate years and beyond.

References

[1] P. Bühlmann and A. J. Wyner. Model selection for variable length Markov chains and tuning the context algorithm. *Ann. Inst. Stat. Math.*, 52(2):287–315, 2000.

[2] C. Burge and S. Karlin. Prediction of complete gene structures in human genomic DNA. *J. Mol. Biol.*, 268:78–94, 1997.

[3] S. Cawley, A. Wirth, and T. P. Speed. Phat—a gene finding program for *Plasmodium falciparum*. *Mol. Biochem. Parasit.*, 118:167–174, 2001.

[4] A. Krogh. Two methods for improving performance of an HMM and their application for gene finding. In *Proc. Int. Conf. Intell. Syst. Mol. Biol.*, volume 5, pages 179–186. ISMB, 1997.

[5] D. Kulp, D. Haussler, M. G. Reese, and F. H. Eeckman. A generalized hidden Markov model for the recognition of human genes in DNA. In *Proc. Int. Conf. Intell. Syst. Mol. Biol.*, volume 4, pages 134–142. ISMB, 1996.

[6] J. Rissanen. Complexity of strings in the class of Markov sources. *IEEE Trans. Inform. Theory*, 32:526–532, 1986.

[7] F. Sanger and A. R. Coulson. A rapid method for determining sequences in DNA by primed synthesis with DNA polymerase. *J. Mol. Biol.*, 94(3):441–448, 1975.

[8] X. Zhao, H. Huang, and T. P. Speed. Finding short DNA motifs using permuted Markov models. *J. Comput. Biol.*, 12(6):894–906, 2005.

ELSEVIER

Molecular & Biochemical Parasitology 118 (2001) 167–174

MOLECULAR
& BIOCHEMICAL
PARASITOLOGY

www.parasitology-online.com.

Phat—a gene finding program for *Plasmodium falciparum*

Simon E. Cawley [a,b,*], Anthony I. Wirth [c], Terence P. Speed [a]

[a] *Department of Statistics, U.C. Berkeley, Berkeley, CA 94720, USA*
[b] *Affymetrix, Emeryville, CA 94608, USA*
[c] *Department of Computer Science, Princeton University, Princeton, NJ 08544, USA*

Abstract

We describe and assess the performance of the gene finding program pretty handy annotation tool (Phat) on sequence from the malaria parasite *Plasmodium falciparum*. Phat is based on a generalized hidden Markov model (GHMM) similar to the models used in GENSCAN, Genie and HMMgene. In a test set of 44 confirmed gene structures Phat achieves nucleotide-level sensitivity and specificity of greater than 95%, performing as well as the other *P. falciparum* gene finding programs Hexamer and GlimmerM. Phat is particularly useful for *P. falciparum* and other eukaryotes for which there are few gene finding programs available as it is distributed with code for retraining it on new organisms. Moreover, the full source code is freely available under the GNU General Public License, allowing for users to further develop and customize it. © 2001 Elsevier Science B.V. All rights reserved.

Keywords: *Plasmodium falciparum*; Gene-finding; Generalized hidden Markov model; Viterbi algorithm

1. Introduction

Sequencing of the *Plasmodium falciparum* genome is proceeding apace. Two completely sequenced chromosomes have been published [1,2] as well as the mitochondrion, and substantial amounts of the sequence of other chromosomes are already available [3–6]. The two published chromosomes have been annotated extensively, in each case making use of a gene-finding program. GlimmerM [7,8], a eukaryotic gene-finding program based on Glimmer [9], was used in the analysis of chromosome 2, while chromosome 3 was annotated with the help of Hexamer [10] and Genefinder [11]. Furthermore, chromosome 3 was revisited later with GlimmerM [12].

Before either of these chromosome sequences was published, there was no publicly available gene-finding program trained on *P. falciparum* sequence, which is known to have a base composition different enough from other organisms to preclude simply using an existing program. Since some of our colleagues had a desire to analyze the sequence then available for genes, one of us wrote a gene-finding program [13]. This paper is about a descendent of that original program which we call pretty handy annotation tool (Phat).

Broadly speaking, there are now four publicly available *Plasmodium* gene-finding programs: Genefinder, GlimmerM, Hexamer and Phat. They each differ somewhat in the way in which they seek to exploit sequence features to find genes, in their availability, and in the extent to which they can be re-trained on new data and used by people other than their authors. As well as introducing Phat, we compare and contrast it with the other programs.

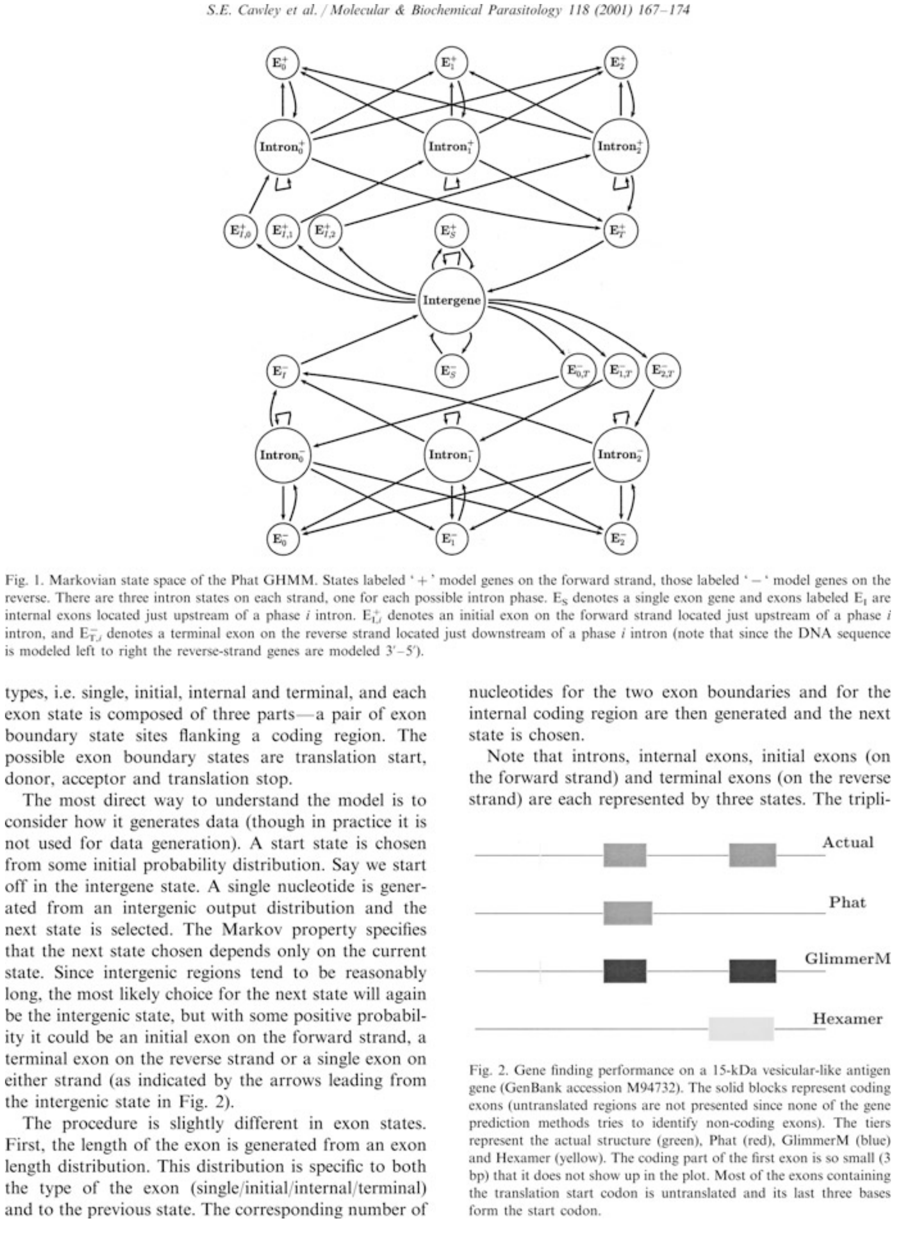
2. Methods

2.1. The model

Phat models genomic DNA with a generalized hidden Markov model (GHMM), similar to existing GHMM gene models such as GENSCAN [14] Genie [15,16] and HMMgene [17]. There is an underlying state space consisting of three main types of states: exons, introns and intergenic regions (Fig. 1). Introns are classified as phase 0, 1 or 2 according to the number of bases of the final codon generated in the previous exon (where previous means the last exon in the 5′ direction, on the coding strand). Exons are classified into four

* Corresponding author. Tel.: + 1-510-428-8534; fax: + 1-510-428-8585.
E-mail address: simon_cawley@affymetrix.com (S.E. Cawley).

0166-6851/01/$ - see front matter © 2001 Elsevier Science B.V. All rights reserved.
PII: S0166-6851(01)00363-2



S.E. Cawley et al. / Molecular & Biochemical Parasitology 118 (2001) 167–174

E_0^+
E_1^+
E_2^+
$Intron_0^+$
$Intron_1^+$
$Intron_2^+$
$E_{I,0}^+$
$E_{I,1}^+$
$E_{I,2}^+$
E_S^+
E_T^+
Intergene
E_I^-
E_S^-
$E_{0,T}^-$
$E_{1,T}^-$
$E_{2,T}^-$
$Intron_0^-$
$Intron_1^-$
$Intron_2^-$
E_0^-
E_1^-
E_2^-

Fig. 1. Markovian state space of the Phat GHMM. States labeled ' + ' model genes on the forward strand, those labeled ' – ' model genes on the reverse. There are three intron states on each strand, one for each possible intron phase. E_S denotes a single exon gene and exons labeled E_I are internal exons located just upstream of a phase i intron. $E_{I,i}^+$ denotes an initial exon on the forward strand located just upstream of a phase i intron, and $E_{T,i}^-$ denotes a terminal exon on the reverse strand located just downstream of a phase i intron (note that since the DNA sequence is modeled left to right the reverse-strand genes are modeled 3′–5′).

types, i.e. single, initial, internal and terminal, and each exon state is composed of three parts—a pair of exon boundary state sites flanking a coding region. The possible exon boundary states are translation start, donor, acceptor and translation stop.

The most direct way to understand the model is to consider how it generates data (though in practice it is not used for data generation). A start state is chosen from some initial probability distribution. Say we start off in the intergene state. A single nucleotide is generated from an intergenic output distribution and the next state is selected. The Markov property specifies that the next state chosen depends only on the current state. Since intergenic regions tend to be reasonably long, the most likely choice for the next state will again be the intergenic state, but with some positive probability it could be an initial exon on the forward strand, a terminal exon on the reverse strand or a single exon on either strand (as indicated by the arrows leading from the intergenic state in Fig. 2).

The procedure is slightly different in exon states. First, the length of the exon is generated from an exon length distribution. This distribution is specific to both the type of the exon (single/initial/internal/terminal) and to the previous state. The corresponding number of nucleotides for the two exon boundaries and for the internal coding region are then generated and the next state is chosen.

Note that introns, internal exons, initial exons (on the forward strand) and terminal exons (on the reverse strand) are each represented by three states. The tripli-

Fig. 2. Gene finding performance on a 15-kDa vesicular-like antigen gene (GenBank accession M94732). The solid blocks represent coding exons (untranslated regions are not presented since none of the gene prediction methods tries to identify non-coding exons). The tiers represent the actual structure (green), Phat (red), GlimmerM (blue) and Hexamer (yellow). The coding part of the first exon is so small (3 bp) that it does not show up in the plot. Most of the exons containing the translation start codon is untranslated and its last three bases form the start codon.

cate representation is to keep track of the frame. When arriving in a given exon state from a particular intron state, the next state is fully determined and the length of the exon must have the appropriate remainder after division by three.

The possibility of the length of exon output sequences following an arbitrary distribution makes the model slightly more flexible than a regular HMM (in which the output will always have length 1 for each state in the hidden state space). Hence the name 'generalized' HMM. For intron and intergene sequences we only allow the generation of one base of output at a time (so the output length is always fixed at 1) but unlike the exon states we allow self-transitions. Accordingly, intron and intergene lengths will follow a geometric distribution. This restriction of the model allows for large decreases in both running time and memory requirements, and turns out to be a reasonable approximation for many organisms.

2.2. Gene predictions

In practice the aim is to use the model to predict the location of genes in sequence data, which involves the estimation of the hidden states and their duration given an observed genomic sequence. A reasonable approach is to determine the sequence of states and duration that maximizes the joint probability of the hidden and observed data. This approach is the one usually adopted in HMM gene finders [14–16] (though see [17] for a nice alternative) and is the one we use in Phat. The approach is popular not only because it is effective, but also because it can be implemented in an efficient manner using the *Viterbi* algorithm [18].

The key idea in the Viterbi algorithm is to record, for each hidden state and sequence position, the maximum joint probability of hidden and observed data up to that position. The actual algorithm is a dynamic programming procedure that computes recursively the single most likely sequence.

The standard Viterbi algorithm applies to any GHMM, but there are certain features of the state space of the Phat model that can be exploited to yield savings in time and memory. Firstly, the state space can be decomposed into the set of exon states and the set of intron and intergene states. This decomposition has the special property that no exon state can jump directly to another exon state, which implies that an intron or intergene state must be preceded by another state of the same kind either one or two states back.

By modifying the Viterbi algorithm to maximize over the previous two states rather than just the previous one we can achieve a 70% reduction in the storage space required. Other features that help in reducing computation include the fact that exon states have only one possibility for the next state, that intron or intergenic states always have a duration of 1, and that the only way for a state of the latter kind to be followed directly by another such state is via a self-transition.

There is a trick that can be used to further reduce the number of computations by a significant factor, at the expense of a modest extra storage requirement. The distributions we use for exons all consist of three parts, i.e. a pair of exon boundary distributions and a distribution for the coding portion. For the last we use a three-periodic Markov model of order five. There are thus three sets of probabilities used for coding sequence, one corresponding to each codon position. Certain quantities related to these probabilities can be computed in advance and can be stored in a look-up table, after which exon probabilities can be computed when needed with only a single division. Such a look-up table approach reduces the runtime of the program by orders of magnitude.

One of the attractive features of using a GHMM for gene prediction is that it provides a natural way of computing the probability of a predicted exon, given the observed data. In addition we are often interested in the probability of a particular base or region being part of an exon. The probability that a particular sequence position is non-coding can be calculated, and subtracting it from 1, we get the probability that the position is part of some coding state. While useful in predicting potential exons that the Viterbi reconstruction may have missed, this probability says nothing about the possible strand of the coding region.

2.3. Training the model

There is a number of standard techniques for training hidden Markov models. Perhaps the best-known is the Baum–Welch method (also known as the expectation maximization, or EM algorithm) presented by Rabiner [18]. Given a collection of training sequences and initial values for the model parameters a single iteration of the Baum–Welch method provides new model parameters under which the training sequences have greater or equal likelihood. Repeated iterations yield maximum likelihood parameter estimates.

Though reasonably straightforward to write down, actual implementation of a maximum likelihood training method is a tricky and time-consuming task. The common approach, and the one we adopt, is to obtain parameter estimates independently from categorized training data. See [17] for an alternative approach, where parameters are estimated by a conditional maximum likelihood approach. We now describe our training in a little more detail.

First consider the transition probabilities between the underlying states. Any pair of states not connected in Fig. 1 has a transition probability of zero. As we have constructed the state space so that every exon state is followed by a unique state, we can now restrict our

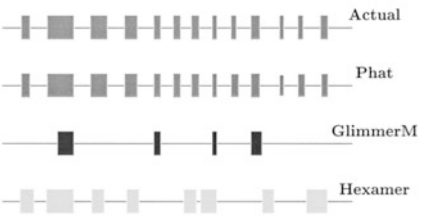


Fig. 3. Gene finding performance on a chloroquine resistance transporter (GenBank accession AF030694). The scheme is as in Fig. 2. The actual structure (green), Phat (red), GlimmerM (blue) and Hexamer (yellow). The complete coding region is shown 5′–3′.

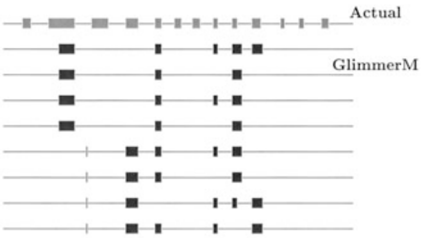


Fig. 5. GlimmerM's alternative predictions for the chloroquine resistance transporter from Fig. 3. Note that some of the predictions include the fourth and ninth exons, which are among the exons missing from the original prediction.

attention to transitions from non-coding states. In fact, the previous discussion implies that we can consider effectively such transitions as being from non-coding to non-coding states. Since we assume that all non-coding states emit one nucleotide at a time, the self-transition probabilities parameterize the length of the non-coding sections. Using our training data, we can obtain frequency counts for each of these transitions, which can be used to compute maximum likelihood estimates of the transition probabilities. One slight complication is that transitions from the Intergene state could go to either the forward or reverse strand states. For convenience, we assume that the next gene is equally likely to be on either strand.

We assume that the initial state is not an exon state and set the initial probability of the intergene state to be the fraction of all non-coding sequence that is intergenic in the training set. The initial probability of each type of intron is set to its relative frequency and we assume that each intron type is equally likely on both strands. We use different initial probabilities for the three intron phases to allow for the observed fact that phase 2 introns in *P. falciparum* are relatively rare.

Non-coding states have geometric length distributions, a result of the model's restriction that their state durations are always 1. The mean lengths are fully determined by the transition probabilities, which are estimated from a training data set of intron/intergene lengths. For coding state lengths we use a shifted γ-distribution, whose three parameters allow for a reasonable fit to observed data. Given the characteristically different types of distributions for single, initial, internal and terminal exons in *P. falciparum*, a different distribution is estimated for each. The reliance on both the previous and current states in the term length distribution is a feature of the frame constraints. A single exon must have a length that is a multiple of

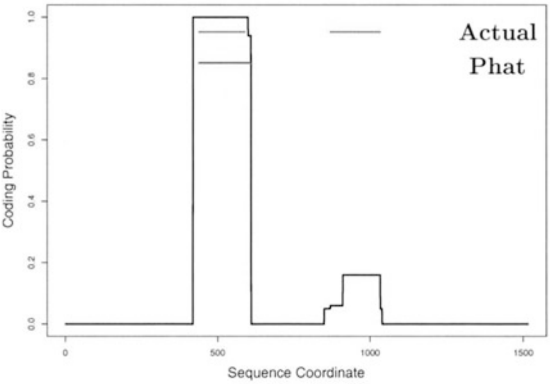
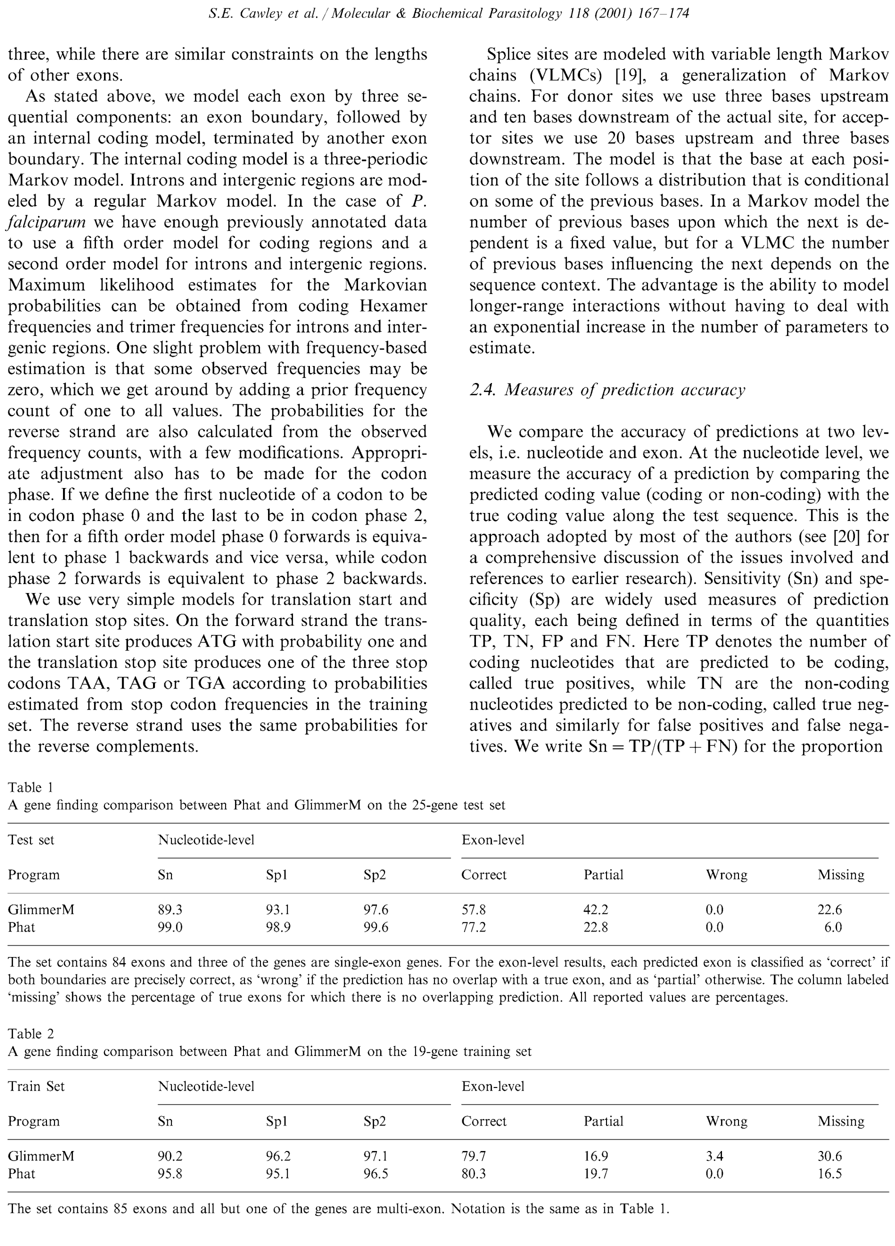


Fig. 4. Phat's coding probability plot for the vesicular-like antigen of Fig. 2. The green bars near the top represent the actual structure (the first 3 bp coding exon is invisible) and the red bar represents the single exon predicted by Phat. Note that even though the terminal exon was not predicted, its presence is suggested by the coding probability plot.



three, while there are similar constraints on the lengths of other exons.

As stated above, we model each exon by three sequential components: an exon boundary, followed by an internal coding model, terminated by another exon boundary. The internal coding model is a three-periodic Markov model. Introns and intergenic regions are modeled by a regular Markov model. In the case of *P. falciparum* we have enough previously annotated data to use a fifth order model for coding regions and a second order model for introns and intergenic regions. Maximum likelihood estimates for the Markovian probabilities can be obtained from coding Hexamer frequencies and trimer frequencies for introns and intergenic regions. One slight problem with frequency-based estimation is that some observed frequencies may be zero, which we get around by adding a prior frequency count of one to all values. The probabilities for the reverse strand are also calculated from the observed frequency counts, with a few modifications. Appropriate adjustment also has to be made for the codon phase. If we define the first nucleotide of a codon to be in codon phase 0 and the last to be in codon phase 2, then for a fifth order model phase 0 forwards is equivalent to phase 1 backwards and vice versa, while codon phase 2 forwards is equivalent to phase 2 backwards.

We use very simple models for translation start and translation stop sites. On the forward strand the translation start site produces ATG with probability one and the translation stop site produces one of the three stop codons TAA, TAG or TGA according to probabilities estimated from stop codon frequencies in the training set. The reverse strand uses the same probabilities for the reverse complements.

Splice sites are modeled with variable length Markov chains (VLMCs) [19], a generalization of Markov chains. For donor sites we use three bases upstream and ten bases downstream of the actual site, for acceptor sites we use 20 bases upstream and three bases downstream. The model is that the base at each position of the site follows a distribution that is conditional on some of the previous bases. In a Markov model the number of previous bases upon which the next is dependent is a fixed value, but for a VLMC the number of previous bases influencing the next depends on the sequence context. The advantage is the ability to model longer-range interactions without having to deal with an exponential increase in the number of parameters to estimate.

2.4. Measures of prediction accuracy

We compare the accuracy of predictions at two levels, i.e. nucleotide and exon. At the nucleotide level, we measure the accuracy of a prediction by comparing the predicted coding value (coding or non-coding) with the true coding value along the test sequence. This is the approach adopted by most of the authors (see [20] for a comprehensive discussion of the issues involved and references to earlier research). Sensitivity (Sn) and specificity (Sp) are widely used measures of prediction quality, each being defined in terms of the quantities TP, TN, FP and FN. Here TP denotes the number of coding nucleotides that are predicted to be coding, called true positives, while TN are the non-coding nucleotides predicted to be non-coding, called true negatives and similarly for false positives and false negatives. We write $\mathrm{Sn} = \mathrm{TP}/(\mathrm{TP} + \mathrm{FN})$ for the proportion

Table 1
A gene finding comparison between Phat and GlimmerM on the 25-gene test set

Test set	Nucleotide-level			Exon-level			
Program	Sn	Sp1	Sp2	Correct	Partial	Wrong	Missing
GlimmerM	89.3	93.1	97.6	57.8	42.2	0.0	22.6
Phat	99.0	98.9	99.6	77.2	22.8	0.0	6.0

The set contains 84 exons and three of the genes are single-exon genes. For the exon-level results, each predicted exon is classified as 'correct' if both boundaries are precisely correct, as 'wrong' if the prediction has no overlap with a true exon, and as 'partial' otherwise. The column labeled 'missing' shows the percentage of true exons for which there is no overlapping prediction. All reported values are percentages.

Table 2
A gene finding comparison between Phat and GlimmerM on the 19-gene training set

Train Set	Nucleotide-level			Exon-level			
Program	Sn	Sp1	Sp2	Correct	Partial	Wrong	Missing
GlimmerM	90.2	96.2	97.1	79.7	16.9	3.4	30.6
Phat	95.8	95.1	96.5	80.3	19.7	0.0	16.5

The set contains 85 exons and all but one of the genes are multi-exon. Notation is the same as in Table 1.

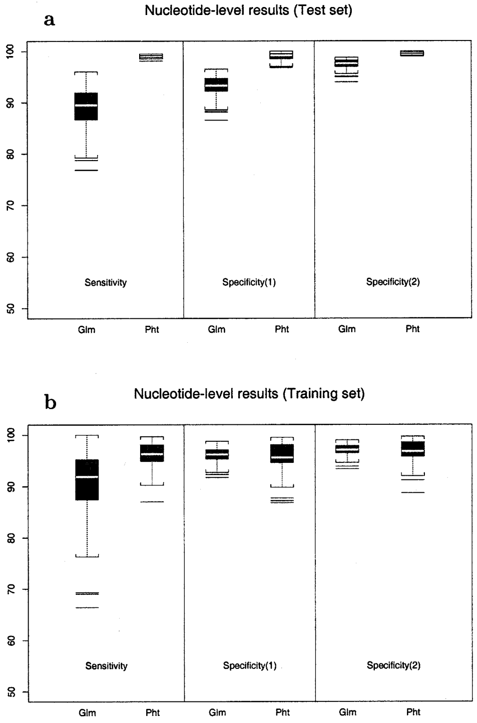


Fig. 6. Box plots of bootstrapped sensitivities and specificities (as defined in Table 1) for GlimmerM (Glm) and Phat (Pht) on the test set (a) and the training set (b). For each box plot, the solid box covers the inter-quartile range of the data, with the line within the box representing the median. The 'whiskers' extend to the nearest values not beyond 1.5 times the inter-quartile range from the quartiles. Remaining points are represented by isolated dashes.

of positives that are correctly annotated, Sp1 = TN/(TN + FP) for the proportion of negatives correctly annotated, and Sp2 = TP/(TP + FP) for the proportion of coding predictions which are correct (also called the positive predictive value).

In many contexts Sp1 is the more natural measure of specificity, but in discussion of the accuracy of genefinders, it has generally been replaced by Sp2. This is because the typically high proportion of non-coding sequence predicted readily as such can dominate Sp1, and thus make the measure less sensitive. We present both quantities below.

Exon level results are also reported. Predicted exons are classified as either correct (both boundaries correct), partial (overlapping a true exon), or wrong. A missing exon is one which the genefinder did not detect at all.

3. Results

We have conducted a study to compare the performance of Phat [21,13] with other gene finding programs on *P. falciparum* sequence. Currently the other main programs are Hexamer [10], Genefinder [11] and GlimmerM [7,8]. Hexamer operates quite differently to the others using only Hexamer frequencies to predict individual coding regions. It does not attempt to detect exon boundaries, nor does it assemble its predicted coding regions together into whole genes.

The remaining three programs all attempt to predict whole gene structures where possible, and can be applied to large sequences containing multiple genes. GlimmerM has models for coding regions and splice sites. Genefinder models coding regions, splice sites, introns, intergenic regions and has a model for the transcription start site. Phat models all the aforementioned features, save for the transcription start site, also using explicit state length distributions to model feature lengths.

For the purpose of comparing gene finding programs it is important to train the programs on a common data set. Phat and Hexamer are distributed along with code for retraining on new data sets. The GlimmerM version used here (obtained from the authors in August 2000) comes pre-trained on a set of around 300 genes and there are no means available to re-train GlimmerM on new data sets. Genefinder also comes with code for retraining on new data, however, we experienced technical difficulties getting the retraining code to work. The end result is that the only way to do a fair comparison is to drop Genefinder out of the analysis and train the others on GlimmerM's training set.

P. falciparum researchers from the Sanger Centre and from the Walter and Eliza Hall Institute (WEHI) were asked to provide a list of genes, which have been confirmed biologically by reverse transcription-polymerase chain reaction (RT-PCR) experiments, leading to an evaluation data set of 44 genes. Of these 44 genes it turned out that 19 were already in the GlimmerM training set. In what follows we refer to these 19 genes as the training set, and to the remaining 25 in the evaluation set as the test set.

Comparing the gene finders on the evaluation set, it is clear that Phat and GlimmerM often provide accurate predictions. Hexamer is a very simple model and while it does a reasonable job of generally indicating regions of coding potential it has no model for splice sites nor for how to join the regions together as genes, so performs much worse than Phat and GlimmerM.

Looking at Figs. 2 and 3 there are cases when Phat outperforms GlimmerM, and vice versa. Each program also provides some useful features for detecting possibly missed exons. Fig. 4 is a plot of the coding probability computed by Phat (as earlier) for the gene in 2. Phat missed two exons in the optimal prediction, but the coding probability plot is suggestive of the larger of the two exons missed. For each gene predicted by GlimmerM, a list of alternative gene predictions is also provided—these are genes achieving high scores in GlimmerM's model. Fig. 5 shows GlimmerM's alternative predictions for the same gene in Fig. 3, the alternative predictions suggest an extra two exons missing from the original prediction.

Tables 1 and 2 present the nucleotide-level and exon-level results for the test and training sets. An understanding of the variability of these estimates can be helpful, and we address this using a bootstrap study. For one bootstrap iteration, we draw with replacement a new test data set from the original, then evaluate the Genefinder on this bootstrapped data set and compute the performance measures. This is repeated many times and the results are collected. Fig. 6 presents results for the test and training data sets. Across both sets GlimmerM's performance is clearly more variable. It is important to note that the extent to which the results on this evaluation set can be extrapolated to the set of all *P. falciparum* genes will depend on the extent to which the evaluation set is a representative sample.

The time and memory requirements of the programs are important, particularly if they are to be used in a high throughput environment. We compared the time and memory requirements of the two programs on a 700 MHz Pentium III processor running under Linux. For a sequence of 100 kbp Phat requires 39 Mb of memory and takes 20 s of CPU time. For the same sequence GlimmerM requires 12 Mb of memory and takes 34 s. Phat runs faster than GlimmerM, its chief gain probably coming from the use of look-up tables for fast computation of exon probabilities, but at the expense of increased memory requirements. Both programs scale roughly linearly in the length of the sequence being analyzed and have been used to analyze sequences of up to 1 Mbp.

4. Discussion

Both genefinders displayed relatively high sensitivity and specificity on both the training and test sets of genes. It is a little surprising that both gene finders performed better on the examples on which they had not been trained, perhaps the genes in the training set are in some sense more difficult to predict accurately. A reviewer with extensive experience in the field has found that GlimmerM tends consistently to under-annotate while Phat tends consistently to over-annotate. He also found Phat sometimes returned abnormally short introns and abnormally long exons. These aspects of Phat are perhaps general features of the GHMM approach, and all we can say is that they have their advantages as well, the example of Fig. 3 being a clear example.

As mentioned earlier, the use of our specificity measure Sp1 has all but ceased in the assessment of genefinding algorithms, due to its being dominated by large values of TN. Nonetheless, we have included it along with Sp2, because in our case the two measures are comparable, indeed Sp2 is slightly larger than Sp1. The reason for this is that *P. falciparum* has a relatively high coding content, (chromosomes 2 and 3 are about 50% coding), so the values of TP and TN are much more similar than in other organisms (e.g. human, Drosophila) for which these measures have been calculated previously.

In conclusion, we have demonstrated that Phat performs well upon *P. falciparum* sequence, and compares favorably with GlimmerM. There is thus a good case for making use of both GlimmerM and Phat for new *P.*

falciparum data as there are examples where one predicts correctly what the other misses. Each program also has useful functionality to try to detect exons that may have been missed in the single best prediction (Figs. 4 and 5). These gene finders should prove useful as the *P. falciparum* genome approaches completion.

Acknowledgements

This work was made possible with advice and data from many others. The authors would like to thank Mauro De Lorenzi, Alan Cowman and Tony Triglia at WEHI, Mihaela Pertea and Steven Salzberg at TIGR, Allan Saul and Robert Heustis at QIMR, Sharen Bowman and Neil Hall at the Sanger Centre, Winston Hide and Ralhston Muller at SANBI and Jane Carlton at NCBI. S.C. was supported by DOE grant DE-FGO3-97ER62387.

References

[1] Gardner MJ, Tettelin H, Carucci DJ, Cummings LM, Aravind L, Koonin EV, Shallom S, et al. Chromosome 2 sequence of the human malaria parasite *Plasmodium falciparum*. Science 1998;282:1126–32.

[2] Bowman S, Lawson D, Basham D, Brown D, Chillingworth T, Churcher CM, Craig A, et al. The complete nucleotide sequence of chromosome 3 of *plasmodium falciparum*. Nature 1999;400:532–8.

[3] The Malaria genome project at Stanford University, URL: http://sequence-www.stanford.edu/group/malaria/.

[4] The *Plasmodium falciparum* Genome Database at the University of Pennsylvania, URL: http://PlasmodiumDB.cis.upenn.edu/.

[5] The *Plasmodium falciparum* genome database at The Institute for Genomic Research, URL: http://www.tigr.org/tdb/edb2/pfa1/htmls/.

[6] The *Plasmodium falciparum* genome project at the Sanger Centre, URL: http://www.sanger.ac.uk/Projects/P_falciparum/.

[7] GlimmerM, http://www.tigr.org/softlab/glimmerm.

[8] Salzberg S, Pertea M, Deicher A, Gardner M, Tettelin H. Interpolated Markov models for eukaryotic gene finding. Genomics 1999;59:24–31.

[9] Salzberg S. Decision trees and Markov chains for gene finding. In: Salzberg S, Searls D, Kasif S, editors. Computational methods in molecular biology. Amsterdam: Elsevier, 1998:187–203.

[10] Durbin R. Hexamer. 1995. Source code available at ftp://ftp.sanger.ac.uk/pub/pathogens/software/hexamer.

[11] Green P. Genefinder. 1994. Contact phg@u.washington.edu for details.

[12] Pertea M, Salzberg S, Gardner M. Finding genes in *Plasmodium falciparum*. Nature 2000;404:34–5.

[13] Wirth A. A *Plasmodium falciparum* genefinder. Honours thesis, University of Melbourne; 1998.

[14] Burge C, Karlin S. Prediction of complete gene structures in human genomic DNA. J Mol Biol 1997;268:78–94.

[15] Kulp D, Haussler D, Reese MG, Eeckman FH. A generalized hidden Markov model for the recognition of human genes in DNA. In: States D, Agarwal P, Gaasterland T, Hunter L, Smith RF, editors. ISMB-96: proceedings of the fourth international conference on intelligent systems for molecular biology. AAI Press, 1996:134–41.

[16] Reese MG, Kulp D, Tammana H, Haussler D. Genie—gene finding in *Drosophila melanogaster*. Genome Res 2000;10:529–38.

[17] Krogh A. Using database matches with HMMGene for automated gene detection in *Drosophila*. Genome Res 2000;10:523–8.

[18] Rabiner LR. A tutorial on hidden Markov models and selected applications in speech recognition. Proc IEEE 1989;77:257–86.

[19] Buhlmann P, Wyner AJ. Variable length Markov models. Ann Statistics 1999;27:480–513.

[20] Burset M, Guigo R. Evaluation of gene structure prediction programs. Genomics 1996;34:353–67.

[21] Wirth A, Cawley S, Speed T. Phat. 1998. Source code available at http://www.stat.berkeley.edu/users/scawley/Phat.

ICM 2002 · Vol. III · 1–3

Biological Sequence Analysis

T. P. Speed*

Abstract

This talk will review a little over a decade's research on applying certain stochastic models to biological sequence analysis. The models themselves have a longer history, going back over 30 years, although many novel variants have arisen since that time. The function of the models in biological sequence analysis is to summarize the information concerning what is known as a motif or a domain in bioinformatics, and to provide a tool for discovering instances of that motif or domain in a separate sequence segment. We will introduce the motif models in stages, beginning from very simple, non-stochastic versions, progressively becoming more complex, until we reach modern profile HMMs for motifs. A second example will come from gene finding using sequence data from one or two species, where generalized HMMs or generalized pair HMMs have proved to be very effective.

2000 Mathematics Subject Classification: 60J20, 92C40.
Keywords and Phrases: Motif, Regular expression, Profile, Hidden Markov model.

1. Introduction

DNA (deoxyribonucleic acid), RNA (ribonucleic acid), and proteins are macromolecules which are unbranched polymers built up from smaller units. In the case of DNA these units are the 4 nucleotide residues A (adenine), C (cytosine), G (guanine) and T (thymine) while for RNA the units are the 4 nucleotide residues A, C, G and U (uracil). For proteins the units are the 20 amino acid residues A (alanine), C (cysteine) D (aspartic acid), E (glutamic acid), F (phenylalanine), G (glycine), H (histidine), I (isoleucine), K (lysine), L (leucine), M (methionine), N (asparagine), P (proline), Q (glutamine), R (arginine), S (serine), T (threonine), V (valine), W (tryptophan) and Y (tyrosine). To a considerable extent, the chemical properties of DNA, RNA and protein molecules are encoded in the linear sequence of these basic units: their primary structure.

*Department of Statistics, University of California, Berkeley, CA 94720, USA; Division of Genetics and Bioinformatics, Walter and Eliza Hall Institute of Medical Research, VIC 3050, Australia. E-mail: terry@stat.berkeley.edu

The use of statistics to study linear sequences of biomolecular units can be descriptive or it can be predictive. A very wide range of statistical techniques has been used in this context, and while statistical models can be extremely useful, the underlying stochastic mechanisms should never be taken literally. A model or method can break down at any time without notice. Further, biological confirmation of predictions is almost always necessary.

The statistics of biological sequences can be global or it can be local. For example, we might consider the global base composition of genomes: *E. coli* has 25% A, 25% C, 25% G, 25% T, while *P. falciparum* has 82%A+T. At the very local, the triple ATG is the near universal motif indicating the start of translation in DNA coding sequence. A major role of statistics in this context is to characterize individual sequences or classes of biological sequences using probability models, and to make use of these models to identify them against a background of other sequences. Needless to say, the models and the tools vary greatly in complexity.

Extensive use is made in biological sequence analysis of the notions of motif or domain in proteins, and site in DNA. We shall use these terms interchangeably to describe the recurring elements of interest to us. It is important to note that while we focus on the sequence characteristics of motifs, domains or sites, in practice they also embody (biochemical) structural significance.

2. Deterministic models

The C2H2 (cysteine-cysteine histidine-histidine) zinc-finger DNA binding domain is composed of 25-30 amino acid residues including two conserved cysteines and two conserved histidines spaced in a particular way, with some restrictions on the residues in between and nearby. Of course the arrangement reflects the three-dimensional molecular structure into which the amino-acid sequence folds, for it is the structure which has the real biochemical significance, see Figure 1, which was obtained from `http://www.rcsb.org/pdb/`. An example of this motif is the 27-

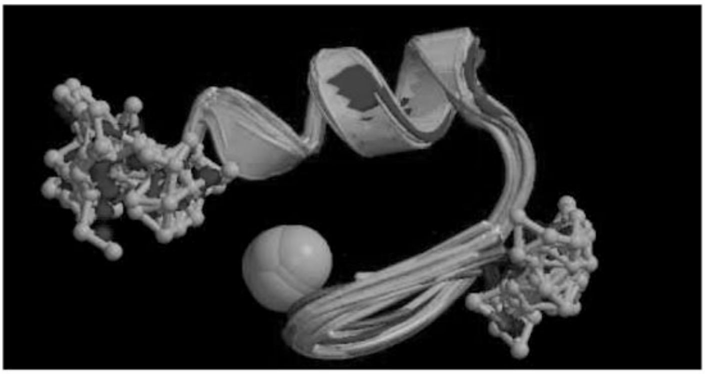


Figure 1: A C2H2 zinc finger DNA binding domain

letter sequence known as 1ZNF, this being a Protein Data Bank identifier for the structure XFIN-31 of *X. laevis.* Its amino acid sequence is

```
1ZNF:  XYKCGLCERSFVEKSALSRHQRVHKNX
```

Note the presence of the two Cs separated by 2 other residues, and the two Hs separated by 3 other residues. Here and elsewhere, X denotes an arbitrary amino acid residue. A popular and useful summary description of C2H2 zinc fingers which clearly includes our example, is the regular expression

$$C - X(2,4) - C - X(3) - [LIVMFYWC] - X(8) - H - X(3,5) - H$$

where $X(m)$ denotes a sequence of n unspecified amino acids, while $X(m,n)$ denotes from m to n such, and the brackets enclose mutually exclusive alternatives. There is a richer set of notation for *regular expressions* of this kind, but for our purposes it is enough to note that this representation is essentially deterministic, with uncertainty included only through mutually exclusive possibilities (e.g. length or residue) which are not otherwise distinguished.

Simple and efficient algorithms exist for searching query sequences of residues to find every instance of the regular expression above. In so doing with sequence in which all instances of the motif are known, we may identify some sub-sequences of the query sequence which are not C2H2 zinc finger DNA binding domains, i.e. which are false positives, and we may miss some sub-sequences which are C2H2 zinc fingers, i.e. which are false negatives. Thus we have essentially deterministic descriptions and search algorithms for the C2H2 motifs using regular expressions. Their performance can be described by the frequency of false positives and false negatives, equivalently, their complements, the specificity and sensitivity of the regular expression. We do not have space for an extensive bibliography, so for more on regular expressions and on most of the other concepts we introduce below, see [2].

3. Regular expressions can be limiting

Most protein binding sites are characterized by some degree of sequence specificity, but seeking a consensus DNA sequence is often an inadequate way to recognize their motifs. Simply listing the alternatives seen at a position may not be very informative, but keeping track of the frequencies with which the different alternatives appear can be very valuable. Thus position-specific nucleotide or amino acid distributions came to represent the variability in DNA or protein motif composition. This is just the set of marginal distribution of letters at each position. Rather than present an extensive tabulation of frequencies for our C2H2 zinc finger example, we present a pictorial representation: a sequence logo coming from `http://blocks.fhcrc.org`.

Sequence logos are scaled representation of position-specific nucleotide or amino acid distributions. The overall height at a given position is proportional to information content, which is a constant minus the entropy of the distribution at that

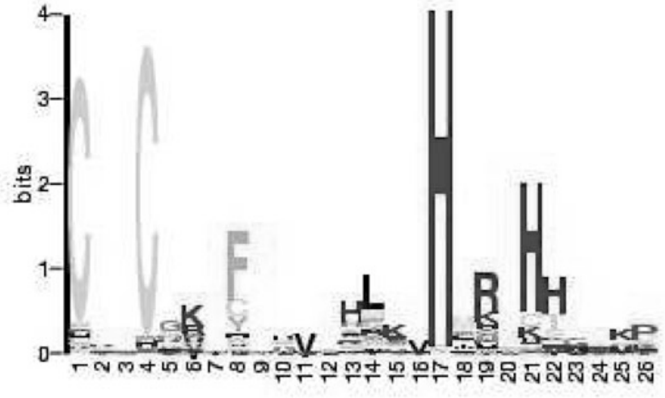


Figure 2: Sequence logo for C2H2 zinc finger

position. The proportions of each nucleotide or amino acid at a position are in relation to their observed frequency at that position, with the most frequent on top, the next most frequent below, etc.

4. Profiles

It is convenient for our present purposes to define a profile as a set of position-specific distributions describing a motif. (Traditionally the term has been used for the derived scores.) How would we use a set of such distributions to search a query sequence for instances of the motif? The answer from bioinformatics is that we *score* the query sequence, and for suitably large scores, declare that a candidate subsequence is an instance of our motif.

There are a number of approaches for deriving profile scores, but the easiest to explain here is this: scores are *log-likelihood ratio test statistics*, for discriminating between a probability model M for the motif and a model B for the background. The model M will be the direct product of the position-specific distributions, (i.e. the independent but not identical distribution model), while the background model B will be the direct product of a set of relevant background frequencies (i.e. the independent and identical distribution model). Thus, if f_{al} is the frequency of residue a at position l of the motif, and f_a background frequency of the same residue, then the profile score assigned to residue a at position l in a possible instance of the motif will be $s_{al} = \log f_{al}/f_a$. These scores are then summed across the positions in the motif, and compared to a suitably defined threshold. Note that proper setting of the threshold requires a set of data in which all instances of the motif are known. The false positive and false negative rate could then be determined for various thresholds, and a suitable choice made.

We briefly discuss variants of the log-likelihood ratio scores. In many contexts, it will matter little whether a position is occupied by a leucine (L) rather than an isoleucine (I), as each can evolve in time to or from the other rather more readily than from other residues. Thus it might make sense to modify the scores to take this and similar evolutionary patterns into account. Indeed the first use of profiles involved scores of this kind, using the position specific amino acid distribution of an alignment of instances of the motif and entries from what are known as PAM matrices, which embody patterns of molecular evolution. In addition, the background distribution of residues may be modelled more detailed manner, e.g. using the so-called Dirichlet mixture models.

It is also possible to include position-specific scores for insertion and deletion of residues, relative to a consensus pattern. When these are used, the scoring becomes a little more subtle, as the problem is then quite analogous to pairwise sequence alignment, but with position dependent scoring parameters for matches, mismatches, insertions and deletions.

We summarise this section by noting that probability has entered into our description through the use of frequencies, and scores based on them, but so far we do not have global statistical models, at least not ones embodying insertions and deletions, on which we base our estimation and testing. These are all part of the use of profile HMMs, but first we introduce HMMs.

5. Hidden Markov models

Hidden Markov models (HMMs) are processes $(S_t, O_t), t = 1, \ldots, T$, where S_t is the hidden state and O_t the observation at time t. Their probabilistic evolution is constrained by the equations

$$\begin{aligned} pr(S_t|S_{t-1}, O_{t-1}, S_{t-2}, O_{t-2}, \ldots) &= pr(S_t|S_{t-1}), \\ pr(O_t|S_{t-1}, O_{t-1}, S_{t-2}, O_{t-2}, \ldots) &= pr(O_t|S_t, S_{t-1}). \end{aligned}$$

The definitions and basic facts concerning HMMs were laid out in a series of beautiful papers by L. E. Baum and colleagues around 1970, see [2] for references. Much of their formulation has been used almost unchanged to this day. Many variants are now used. For example, the distribution of O may not depend on previous S, or it may also depend on previous O values,

$$\begin{aligned} pr(O_t|S_t, S_{t-1}, O_{t-1}, \ldots) &= pr(O_t|S_t), \quad \text{or} \\ pr(O_t|S_t, S_{t-1}, O_{t-1}, \ldots) &= pr(O_t|S_t, S_{t-1}, O_{t-1}). \end{aligned}$$

Most importantly for us below, the times of S and O may be decoupled, permitting the observation corresponding to state time t to be a string whose length and composition depends on S_t (and possibly S_{t-1} and part or all of the previous observations). This is called a hidden semi-Markov or generalized hidden Markov model.

Early applications of HMMs were to finance, but these were never published, to speech recognition, and to modelling ion channels. In the mid-late 1980s HMMs entered genetics and molecular biology, where they are now firmly entrenched. One of the major reasons for the success of HMMs as stochastic models is the fact that although they are substantial generalizations of Markov chains, there are elegant dynamic programming algorithms which permit full likelihood calculations in many cases of interest. Specifically, there are algorithms which permit the efficient calculation of a) $pr(sequence|M)$, where *sequence* is a sequence of observations and M is an HMM; b) the maximum over *states* of $pr(states|sequence, M)$, where *states* is the unobserved state sequence underlying the observation *sequence*; and c) the maximum likelihood estimates of parameters in M based on the observation *sequence*. Step c) is carried out by an iterative procedure which in the case of independent states was later termed the EM algorithm.

6. Profile HMMs

In a landmark paper A. Krogh, D. Haussler and co-workers introduced profile HMMs into bioinformatics. An illustrative form of their profile HMM architecture is given in Figure 3. There we depict the underlying state space of the hidden

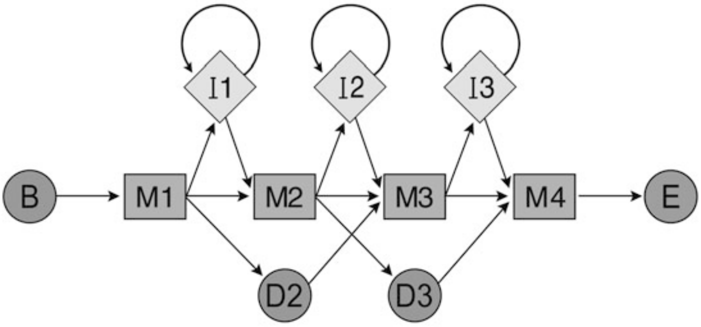


Figure 3: State space of a simple profile HMM

Markov chain of a profile HMM of length 4, with M denoting *match* states, I *insert* states and D *delete* states, while B and E are *begin* and *end* states, respectively. Encircled states (D, B and E) do not emit observations, while each of the match and insert states will have position-specific observation or emission distributions. Finally, each arrow will have associated transition probabilities, with the expectation being that the horizontal transition probabilities are typically near unity. This the chain proceeds from left to right, and if it remains within match states, its output will be an amino acid sequence of length 4. Deviation to the insert or delete states will modify the output accordingly. The similarity with a direct product of a sequence of position-specific distributions should be unmistakeable. The profile HMMs in use now have considerably more features, while sharing the basic M, I and D architecture.

Why was the introduction of the HMM formalism such an advance? The answer is simple: it permitted the construction and application of profiles to be conducted entirely within a formal statistical framework, and that really helped. Instances of the motif embodied in an HMM could be identified by calculating $pr(sequence|M)/pr(sequence|B)$ as was done with profiles, using the algorithm for problem a) in X above. Instances of the motif could be aligned to the HMM by calculating the most probable state sequence giving rise to the motif sequence, in essence finding the most probable sequence of matches, insertions, deletions which align the given sequence to the others which gave rise to the HMM, cf. problem b) above. And finally, the parameters in the HMMs could be estimated from data comprising known instances of the motif by using maximum likelihood, an important step for many reasons, one being that it put insert and delete scores on precisely the same footing as match and mismatch scores. Although the estimation of HMM parameters is easiest if the example sequences are properly aligned, the EM algorithm (problem c) above) does not require aligned sequences.

In the years since the introduction of profile HMMs, they have been become the standard approach to representing motifs and protein domains. The database Pfam (`http://pfam.wustl.edu`) now has 3,849 hidden Markov models (May 2002) representing recognized protein or DNA domains or motifs. Profile HMMs have essentially replaced the use of regular expressions and the original profiles for searching other databases to find novel instances of a motif, for finding a motif or domain match to an input sequence, and for aligning a motif or domain to a an existing family. There is considerable evidence that the HMM-based searches are more powerful than the older profile based ones, though they are slower computationally, and at times that is an important consideration.

7. Finding genes in DNA sequence

Identifying genes in DNA sequence is one of the most challenging, interesting and important problems in bioinformatics today. With so many genomes being sequenced so rapidly, and the experimental verification of genes lagging far behind, it is necessary to rely on computationally derived genes in order to make immediate use of the sequence.

What is a gene? Most readers will have heard of the famous *central dogma* of molecular biology, in which the hereditary material of an organism resides in its genome, usually DNA, and where genes are expressed in a two-stage process: first DNA is *transcribed* into a messenger RNA (mRNA) sequence, and later a processed form of this sequence is *translated* into an amino acid sequence, i.e. a protein. In general the transcribed sequence is longer than the translated portion: parts called introns (intervening sequence) are removed, leaving exons (expressed sequence), of which only some are expressed, while the rest remain untranslated. The translated sequence comes in triples called codons, beginning and ending with a unique start (ATG) and one of three stop (TAA, TAG, TGA) codons. There are also characteristic intron-exon boundaries called splice donor and acceptor sites, and a variety of other motifs: promoters, transcription start sites, polyA sites, branching sites, and so on.

All of the foregoing have statistical characterizations, and in principle they can all help identify genes in long otherwise unannotated DNA sequence segments. To get an idea of the magnitude of the task with the human genome, consider the following facts about human gene sequences [5]: the coding regions comprise about 1.5% of the entire genome; the average gene length is about 27,000 bp (base pair); the average total coding region is 1,340 bp; and the average intron length is about 3,300 bp. Further, only about 8% of genes have a single exon. We see that the information in human genes is very dispersed along the genome, and that in general the parts of primary interest, the coding exons, are a relatively small fraction of the gene, on average about $\frac{1}{20}$.

8. Generalized HMMs for finding genes

The HMMs which are effective in finding genes are the generalized HMMs (GHMMs) described in section 5. above. Space does not permit our giving an adequate description here, so we simply outline the architecture of Genscan [1] one of the most widely used human genefinders. States represent the gene features we mentioned above: exon, intron, and of course intergenic region, and a variety of other features (promotor, untranslated region, polyA site, and so on. Output observations embody state-dependent nucleotide composition, dependence, and specific signal features (such as stop codons). In a GHMM the state *duration* needs to be modelled, as well as two other important features of genes in DNA: the *reading frame*, which corresponds to the triples along the mRNA sequence which are sequentially translated, and the *strand*, as DNA is double stranded, and genes can be on either strand, i.e. they can point in either direction. These features can be seen in Figure 4, which was kindly supplied by Lior Pachter.

The output of a GHMM genefinder after processing a genomic segment is broadly similar to that from a profile HMM after processing an amino acid sequence: the most probable state sequence given an observation sequence is a best gene annotation of that sequence, and a variety of probabilities can be calculated to indicate the support in the observation sequence for various specific gene features.

9. Comparative sequence analysis using HMMs

The large number of sequenced genomes now available, and the observation that functionally important regions are evolutionarily conserved, has led to efforts to incorporate conservation into the models and methods of biological sequence analysis. Pair HMMs were introduced in [2] as a way of including alignment problems under the HMM framework, and recently [4] they were combined with GHMMs (forming GPHMMs) to carry out alignment and genefinding with homologous segments of the mouse and human genomes. Use of the program SLAM on the whole

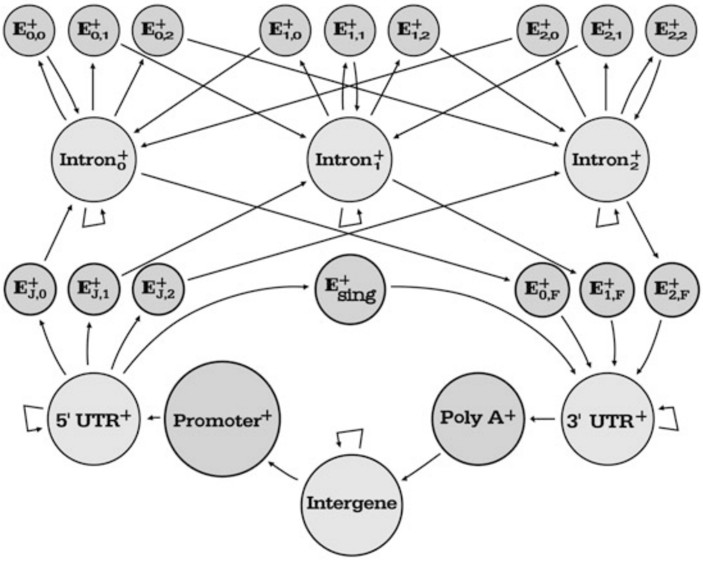


Figure 4: Forward half of the Genscan GHMM state space

mouse genome (`http://bio.math.berkeley.edu/slam/mouse/`) demonstrated the value of GPHMMs in this context.

10. Challenges in biological sequence analysis

The first challenge is to understand the biology well enough to begin biological sequence analysis. This part will frequently involve collaborations with biologists. With HMMs, GHMMs and GPHMMs, designing the underlying architecture, and carrying out the modelling for the components parts, e.g. for splice sites in genefinding GHMM is perhaps the next major challenge. Undoubtedly the hardest and most important task of all is the implementation: coding up the algorithms and making it all work with error-prone and incomplete sequence data. Finally, it is usually a real challenge to find good data sets for calibrating and evaluating the algorithms, and for carrying out studies of competing algorithms.

For a recent example of this process, which is a model of its kind, see [3]. There an HMM is presented for the so-called σ^A recognition sites, which involve two DNA motifs separated by a variable number of base pairs. In addition to the examples mentioned so far, there are many more HMMs in the bioinformatics literature, see p. 79 of [2] for ones published before 1998.

11. Closing remarks

T. P. Speed

In this short survey of biological sequence analysis, I have simply touched on some of the major ideas. A much more comprehensive treatment of material covered here can be found in the book [2], whose title not coincidentally is the same as that of this paper. Many important ideas from biological sequence analysis have not been mentioned here, including molecular evolution and phylogenetic inference, and the use of stochastic context-free grammars, a form of generalization of HMMs suited to the analysis of RNA sequence data.

At this Congress I have talked (and am now writing) on the research of others, in an area in which my own contributions have been negligible. I chose to do so upon being honoured by the invitation to speak at this Congress because I believe this topic – HMMs – to be one of the great success stories of applying mathematics to bioinformatics. In my view it is the one most worthy of a wider mathematical audience. I hope that the fact that there are many others better suited than me to speak on this topic will not prevent readers from appreciating it and following it up through the bibliography.

I owe what understanding I have of this field to collaborations and discussions with a number of people, and I would like to acknowledge them here. Firstly, Tony Wirth, Simon Cawley and Mauro Delorenzi, with whom I have worked on HHMMs. Next, it has been an honour and pleasure to observe from close by the development of SLAM, by Simon Cawley, Lior Pachter and Marina Alexandersson. Finally I'd like to thank Xiaoyue Zhou and Ken Simpson for their kind help to me when I was preparing my talk and this paper.

References

[1] C. Burge & S. Karlin, Prediction of complete gene structures in human genomic DNA, *J. Mol. Biol.* 268 (1997) 78–94.

[2] R. Durbin, S. Eddy, A. Krogh & G. Mitchison, *Biological Sequence Analysis. Probabilistic models of proteins and nucleic acids*, Cambridge University Press, 1998.

[3] H. Jarmer, T. S. Larsen, A. Krogh, H. H. Saxild, S. Brunak & S. Knudsen, Sigma A recognition sites in the *Bacillus subtilis* genome, *Microbiology* 147 (2001), 2417–2424.

[4] L. Pachter, M. Alexandersson & S. Cawley, Applications of generalized pair hidden Markov models to alignment and gene finding problems, *J. Comp. Biol.* 9 (2002), 389–399.

[5] The Genome Sequencing Consortium, Initial sequencing and analysis of the human genome, *Nature* 409 (2001), 860–921.