ELSEVIER

Computers and Chemistry 24 (2000) 43–55

Computers & Chemistry

www.elsevier.com/locate/compchem

# Sequence complexity for biological sequence analysis

L. Allison [a,*], L. Stern [b], T. Edgoose [a], T.I. Dix [a]

[a] *School of Computer Science and Software Engineering, Monash University, Melbourne, 3168 Australia*
[b] *Department of Computer Science and Software Engineering, The University of Melbourne, Melbourne, 3052 Australia*

Received 7 August 1998; accepted 18 February 1999

**Abstract**

A new statistical model for DNA considers a sequence to be a mixture of regions with little structure and regions that are approximate repeats of other subsequences, i.e. instances of repeats do not need to match each other exactly. Both forward- and reverse-complementary repeats are allowed. The model has a small number of parameters which are fitted to the data. In general there are many explanations for a given sequence and how to compute the total probability of the data given the model is shown. Computer algorithms are described for these tasks. The model can be used to compute the information content of a sequence, either in total or base by base. This amounts to looking at sequences from a data-compression point of view and it is argued that this is a good way to tackle intelligent sequence analysis in general. 

*Keywords:* Algorithm; DNA; Complexity; Entropy; Pattern discovery; Sequence analysis

## 1. Introduction

This paper describes a new statistical model for DNA sequences which considers a sequence to be a mixture of two kinds of region. Firstly there are regions with little or no structure which are generated in the 'base state' in the model. Secondly there are regions which are approximate repeats of other substrings of the sequence, as controlled by a small number of 'repeat states' in the model. Repeats need only match each other approximately and can occur in both the forward and reverse directions in the complementary strand in the case of DNA. The model gives good results when used for computing the information content of DNA, giving better data-compression than other models. It has a small number of parameters which are fitted to the data and which can be related to biological processes. An $O(n^2)$ computer algorithm is described and visualizations based on its results show patterns and structure in sequences. A faster approximation algorithm is available for long sequences.

More generally, the paper describes a family of models for DNA sequences and explores a small number of instances of models from this family. It argues that data-compression is a good viewpoint for studying the intelligent analysis of sequences, and that compression provides a natural criterion for comparing competing hypotheses and a natural significance test for the acceptability of hypotheses. There has been a certain amount of interest in the compressibility of biological sequences, not for the most part to save data transmission time or computer storage space as such, but rather to understand sequence structure. Agarwal and States (1994), Grumbach and Tahi (1994) and others have recognized the importance of compression to pattern discovery in biological sequences.

Using compression as the criterion for what is variously known as inductive inference or machine learning dates back to the work of Solomonoff (1964), Kolmogorov (1965), Chaitin (1966). It is believed that

* Corresponding author. Tel.: +61-3-9905-5200; fax: +61-3-9905-5146; http://www.cs.monash.edu.au/~lloyd/tildeStrings/.

0097-8485/00/$ - see front matter 
PII: S0097-8485(99)00046-7

Wallace and Boulton (1968) gave the first major application of this idea in the form of a computer program for what is variously called classification, clustering, mixture modeling or numerical taxonomy. For a hypothesis, H, and data, D, Bayes theorem (Bayes, 1763) gives the following relationship between the probabilities:

$$P(\mathrm{H\&D}) = P(\mathrm{H})\cdot P(\mathrm{D} \mid \mathrm{H}) = P(\mathrm{D})\cdot P(\mathrm{H} \mid \mathrm{D}),$$

which rearranges to give:

$$P(\mathrm{H} \mid \mathrm{D}) = P(\mathrm{H})\cdot P(\mathrm{D} \mid \mathrm{H})/P(\mathrm{D}).$$

Intuitively, an efficient code will allocate its shortest codewords to events of higher probability. Shannon's theory of communication (Shannon, 1949) shows that, in a message using an optimal code, an event, E, of probability $P(\mathrm{E})$, has a code-word of length $-\log_2(P(\mathrm{E}))$ bits. Writing msgLen(E) for $-\log_2(P(\mathrm{E}))$ we have:

$$\mathrm{msgLen(H\&D)} = \mathrm{msgLen(H)} + \mathrm{msgLen(D} \mid \mathrm{H}).$$

If one imagines a transmitter communicating some data D in a message to a distant receiver using an optimal code, the above equation gives the length of a two-part message that first specifies a hypothesis H and then specifies the data given H. It can be seen that the most interesting hypothesis with the greatest posterior probability, $P(\mathrm{H} \mid \mathrm{D})$, is the one that minimizes the length of such a two-part message. In data-compression terms, the first part, msgLen(H), can be thought of as the header of the message (or compressed file).

Interestingly, common file-compression programs perform poorly on DNA sequences, often failing to get below the level of two bits per base that the simplest uniform code achieves, i.e. failing to compress the DNA at all. Milosavljevic and Jurka (1993) applied a Lempel–Ziv (LZ) type of model to DNA and Powell et al. (1998b) examined the compression that could be extracted from variations on that work. Wootton (1997) defined the notion of the 'compositional complexity' of a sequence as the complexity of the multi-state distribution (Boulton and Wallace 1969) of characters in a sliding window. He stopped short of providing a figure for the total information content of a complete sequence, the application being to mask-out regions of low complexity because they bias searches of genomic databases. In passing, an alternative method is not to mask such regions out but rather to give them their appropriate (low) weight during matching, see Powell et al. (1998a). Rivals and Dauchet (1997) presented a DNA compression program which can join exact repeats that are close together into clusters or runs, hence it can make some allowance for differences within (approximate) repeats. Loewenstern and Yianilos (1997) described a modification of popular file-compression methods to cater for the special properties of DNA sequences: their program considers short previously seen 'contexts' that match the most recent bases of the sequence. These matches need not be exact and can contain mismatches, but not insertions or deletions. Predictions of the next base by the selected contexts are blended by several dozen 'weights' which are fitted to the data but do not have any direct biological interpretation. Their results contained the best overall compression on a range of DNA at that time. The new model of sequences (Allison et al., 1998) directly describes the ability of DNA to duplicate substrings, either in the forward direction or in the reverse direction from the complementary strand. Once a sequence contains two or more copies of a substring, the copies can diverge by the usual processes of mutation and this is also modeled. The new model gives improved compression figures for DNA. It has a small number of parameters, which are fitted to the data, and have a biological interpretation. The associated computer algorithm take $O(n^2)$ time for sequences of length $n$. A faster approximation algorithm is available for long sequences.

## 2. Compression

There are generally two aspects to an inductive inference problem: (i) assigning a cost (or conversely a score) to a hypothesis so that one can compare two or more hypotheses fairly, and (ii) searching for the best hypothesis, or at least a good one, under this criterion. Recalling from Section 1 that the hypothesis with the greatest posterior probability, $P(\mathrm{H} \mid \mathrm{D})$, minimizes the length of a two-part message, H followed by $\mathrm{D} \mid \mathrm{H}$, we argue that message length is therefore a good cost criterion for use in inference. It can be objected that such considerations simply replace probabilities with their negative logarithms, but using this framework has several practical advantages:

- It keeps us 'honest' in that any information that has not been agreed to beforehand must be properly costed in msgLen(H): The transmitter and receiver can cooperate to design a code before any data are transmitted but the code can only be optimal 'in expectation'. They are separated before the particular data are given to the transmitter who must send them to the receiver.
- This matter includes stating any parameters that have continuous values to optimum accuracy and describing any variable parts of a hypothesis.
- All the heuristics, algorithms and techniques of coding theory can be used to find the length of a 'reasonable' code if probabilities cannot be calculated exactly; a sub-optimal code can never be shorter than an optimal one so this is safe and conservative, in that it cannot make a hypothesis appear more attractive than it should be.

- There is frequently a natural null-hypothesis, in the case of DNA costing 2 bits per base, and any hypothesis that cannot deliver better compression than this is not acceptable.

The aim here is to use compression as a criterion for evaluating models, and we do not usually carry out the actual compression step, rather calculating what the compressed lengths would be. DNA is not very compressible in any case, it being very difficult to compress 'typical DNA' below 1.8 bits per character, if it makes sense to speak of typical DNA. Any savings in communication costs or disc space would therefore be small, the exception being if we had multiple similar sequences to deal with (Allison and Yee 1990). The trend in data-compression is to separate compression into prediction and coding: a predictor makes predictions to the (arithmetic) coder at the transmitter end and a similar predictor makes predictions to the decoder at the receiver end. It is considered that data modeling is the hard part of data-compression and that the best model leads to the greatest compression. Arithmetic coders (Langdon 1984) are quite capable of encoding an event (in a sequence) in a non-integer number of bits and of approaching the calculated lengths arbitrarily closely.

Over-fitting is a well-known problem in inference: a complex model will often fit data better than a simpler model and the problem is to decide if the added complexity is justified or if it is just fitting noise in the data. Including the term msgLen(H) in the total complexity helps prevent over-fitting by forcing a complex model to pay for its added complications. The message length of the model is the $-\log_2$ of the prior probability of the hypothesis, $P(\mathrm{H})$, and is a characteristic of minimum message length (MML) theory when used for inference purposes (Wallace and Freeman 1987). If there is problem-specific prior knowledge, it should be used, and costed appropriately. In contrast, Rissanen's minimum description length (MDL) (Rissanen, 1987), and stochastic complexity methodologies argue for universal priors, discarding prior knowledge. Note that if inference is not the aim then a one-part message, its length derived by integrating over all possible H, will give slightly better compression than a two-part message. For example, the improvement is a fraction of a bit in the case of the multi-state distribution discussed in the next section.

Table 1 shows results from a sequence of 100 random DNA bases generated from a uniform model (no structure) and analyzed under competing models of varying complexity. The uniform model itself requires 200 bits to transmit the data. The zero-order Markov model finds a chance bias in the frequency of the bases, as revealed by $(\mathrm{D} \mid \mathrm{M})$ costing 197 bits, but is not significant, i.e. does not pay for the model's parameters (H). The first-order Markov model and the new model of approximate repeats (described later) also find some insignificant, chance patterns. The conclusion is that the uniform model best describes this sequence. In contrast, Table 2 shows a similar analysis for a low information content sequence used by Milosavljevic and Jurka (1993) for illustrative purposes. Here the more complex models do find structures which are significant, i.e. they pay for the costs of their parameters and result in compression better than 2 bits per character. The new model performed best on this sequence. Note that a zero-order Markov model for DNA consists of a four-state distribution, a first-order model consists of four such distributions, and the new approximate repeats model embodies several multi-state distributions.

## 3. Multi-state distribution

The multi-state distribution was analyzed by Boulton and Wallace (1969). This distribution is very useful in sequence analysis because it applies to finite alphabets such as DNA {A,C,G,T}, the protein alphabet of 20 amino acids, and to other situations containing discrete choices, e.g. hidden Markov models, probabilistic finite state machines and so on. Its main points are summarized below because it is not widely known and because it is important to the models discussed in later sections. A random variable can take any one of $M$ values. A uniform prior is assumed for the probabilities of the $M$ values. The random variable is sampled $N$ times, giving values v1, v2, …v$N$. There are at least three 'obvious' ways to encode these data:

Table 1
Analysis of 100 random bases under 4 models.

| | Bits | | | |
|---|---|---|---|---|
| | Hypoth | Data | Total | Bits/char |
| Uniform model | 0.0 | 200.0 | 200.0 | 2.00 |
| Order-0 Markov model | 8.2 | 197.4 | 205.6 | 2.05 |
| Order-1 Markov model | 21.1 | 191.0 | 212.1 | 2.12 |
| Approximate repeat model | 31.6 | 191.1 | 222.7 | 2.22 |

Table 2
Analysis of a low information content sequence

| | Bits | | | | |
|---|---|---|---|---|---|
| | Hypoth | Data | Total | Bits/char | |
| *(a) Analysis* | | | | | |
| Uniform model | 0.0 | 256.0 | 256.0 | 2.00 | |
| Order-0 Markov model | 10.0 | 211.0 | 221.0 | 1.72 | |
| Order-1 Markov model | 30.0 | 152.1 | 182.2 | 1.42 | |
| Approximate repeat model | 49.5 | 128.5 | 178.0 | 1.39 | |
| *(b) Sequence used* | | | | | |
| 1 | TGATAGGTGA | TAGATAGATT | GATAGATGAT | AGAAGATTGA | TAGATGATAG |
| 51 | ATACATAGGT | GATAGTAGAT | GTAAGATGAT | AGATGATAGA | TAGATAGATG |
| 101 | ATAGACAGAT | TGATAGATGA | TAGAGAGA | 128 bp | |

1. Fixed code: state the probabilities of the $M$ possible values, $P1, P2, \ldots, PM$, and base a code on these probabilities. coding v$i$ in $-\log_2(P(\text{v}i))$ bits. The transmitter can compute the probabilities by examination of the data and must transmit the probabilities, at some finite accuracy, to the receiver before the receiver can decode the data.
2. Adaptive code: initialize $M$ counters c1, c2, …, c$M$ to 1, say. At step $i$, encode v$i$ in $-\log_2(\text{c}[\text{v}i]/(i-1+M))$ bits and then increment c[v$i$]. The code thus adapts in the light of successive observations v$i$. Both transmitter and receiver can compute this adaptive code. The counters do not start at zero or it would be impossible to encode the first instance of a value.
3. Factorial method: first, state the number of times each possible value actually occurs in v1, … v$N$, i.e. specify an $M$-way partition of $N$. Second, state the particular combination, i.e. order in which the values occur. All such combinations are equally likely.

Boulton and Wallace (1969) proved that methods (2) and (3) give exactly the same message lengths and that method (1) requires only $M(\ln(\pi/6)+1)/2$ nits (they worked in natural bits) more, assuming $N \gg M$. It is satisfying that these lengths are (nearly) equal because otherwise it would make little sense to talk about the information content of the data. The slight extra cost of method (1) is explained by its being the only method to make an explicit inference of a hypothesis, i.e. a statement of the probability values $P1, \ldots, PM$. The values stated are close to the maximum likelihood estimate but are given to an optimum and finite degree of accuracy. Note that even values far from the maximum likelihood estimate for $P1, \ldots, PM$ could in principle be used with method (1) and the data could still be transmitted, albeit inefficiently. The small, but non-zero, probability of such possibilities accounts for the slight excess cost of (1) over (2) and (3). Alternatively, integrating over all possible values for $P1, \ldots, PM$ would lead to a one-part message with the same message length as (2) and (3) but then no inference would have been made.

Wootton (1997) defined 'compositional complexity' as the information content, under the multi-state distribution, within a sliding window along a sequence. The sliding window effectively performs a moving average on the complexity under the multi-state distribution. One application is to mask-out low-information content regions of biological sequences so that they do not bias searches of genomic databases.

The multi-state distribution also applies to Markov models of order $k$ because the events that follow each 'context' of length $k$ form a multi-state distribution. A zero-order Markov model over protein is specified by a 20-state distribution. A first-order Markov model over DNA is specified by a quartet of four-state distributions etc. Probabilistic finite-state machines and other hidden Markov models have a multi-state distribution over the transitions out of each state.

## 4. Repeats

The previous sections have argued that to analyze DNA (or other) sequences requires a good model of sequences, that the best model will give the greatest compression, i.e. shortest two-part message, and that problem-specific prior knowledge should be used, carefully. Computer algorithms for sequence analysis should also be reasonably efficient, although there is less pressure for them to be as efficient as typical file-compression programs which must run in near linear time with a small constant of proportionality. We now start to consider models that might fulfill these aims. This section recalls the LZ model of sequences, its properties and its relation to biological sequences. The following section describes a new model of DNA inspired by it.

The LZ model (Lempel and Ziv, 1976) has inspired many file-compression programs. It considers a sequence to be a mixture of random characters and repeated substrings. For example, ATACGTGCACGTTA can be coded as ATACGTGC(3,4)TA where (3,4) denotes 'repeat 4 characters starting from position 3'. To use the LZ model to encode sequences, there must be a codeword for 'repeat', in addition to codewords for the characters. There will also be codewords for the start and the length of a repeat, and their lengths will depend on the probability distributions of start positions and lengths. A uniform distribution is the simplest model for start positions and gives a code length of $\log_2(x-1)$ for a repeat into target position $x$, there being $x-1$ alternatives for the source location. One can easily use other distributions, e.g. giving a bias to close repeats. The form of the distribution on the lengths of repeats has implications for the speed of the algorithms and is discussed later. Parameters of the model, such as the probability of repeats, might be fixed or might be fitted to the data, in which case they must be included in the total cost. Note that if there are no significant repeats, only chance ones, the model will fail to compress sequences, on average, because the mere possibility of stating 'repeat', i.e. a finite probability of a repeat, reduces the combined probabilities of the characters, and their codewords are correspondingly longer.

There are many ways to encode a sequence under the LZ model and one can search for a single optimal explanation. But imagine that there were two optimal ways, each giving probability $P$ for the data. These two ways are two exclusive hypotheses for generating the data under the model; if one is 'true' the other cannot be. This means their probabilities can be added, giving probability $2P$ for the data, i.e. saving one bit in message length when the $-\log_2$ is taken. There are generally a great many sub-optimal ways to encode the data. Even if each of these gives the data a probability much less than the optimal probability $P$, their sheer number may mean that collectively they contribute a great deal towards $P(\mathrm{D} \mid \mathrm{LZ})$. It is quite possible to devise a code that realizes this saving, e.g. see Wallace and Freeman (1987). Now, it is a legitimate question to ask what is a single optimal explanation of a sequence under the model, and this is what Milosavljevic and Jurka (1993) did (while declining to give a figure for the compression). However, the answer to this question becomes a 'nuisance parameter', and may lead to bias, if the objective is more general, e.g. to calculate the probability of the data under the model or to estimate model parameters. It has a characteristic typical of nuisance parameters in that the number of choices in an explanation grows in proportion to the length of the data. If an optimal explanation really is what is wanted, it is not a nuisance parameter, of course.

Ziv and Lempel (1977) showed that their model asymptotically achieves the same compression as a great variety of other models of sequences. It can therefore claim to be a good model to use when the true nature of the source is unknown. Unfortunately the convergence is slow. Nevertheless this model has inspired a great many file-compression programs. Research has tended to concentrate on clever algorithms and data structures (Cleary and Teahan 1997), such as hash-tables and suffix-trees of 'contexts', to make the programs fast, as this is crucial in the areas of data communications and file-compression.

As noted before, typical file-compression programs do not compress DNA well. The reasons are probably the small alphabet, the lack of any obvious 'punctuation', and the subtle (weak) nature of any pattern and structure that is present. The LZ model's repeats do roughly correspond to the observed duplication of segments of DNA, but the model's repeats must be exact while DNA's can contain mutations, gaps, and rearrangements. A number of workers have tried to address this discrepancy. Rivals and Dauchet (1997) searched for exact repeats but employed a heuristic to join neighboring repeats together. Loewenstern and Yianilos (1997) extended a file-compression algorithm to consider approximate matches on past contexts. These approximate matches can contain mismatches but not insertions or deletions. Several dozen weights blend the predictions from the approximate matches. The present work explicitly models the duplication and subsequent mutation of sections of a sequence.

## 5. Approximate repeats

The new model considers a sequence to be a mixture of random characters and repeated substrings in either the forward- or reverse-complementary senses; instances of repeats may differ by change, insertion and deletion. Fig. 1 shows the model's generating machine which can be used to generate random sequences according to the model. Starting in the base state, B, the machine can generate a character and return to the base state. Alternatively, it can 'start' a repeat (specifying a source origin) and jump to state R. Presuming repeats are common, it is likely to 'copy' a character and jump to state R2 where it can 'continue' (state R3) to 'copy' more characters and eventually 'end' the repeat returning to state B. The possibility of changes, insertions and deletions in a repeat allow instances to differ. In essence, states R, R2 and R3 embody a simple mutation machine, as can be used in the sequence alignment problem (Allison et al., 1992), here used to allow local alignments of the sequence with itself. For the analysis of DNA, approximate reverse complementary repeats are allowed by a further set of states R′, R2′ and R3′

and corresponding operations, not shown. Finally, the base state of the machine can be replaced by some other sub-model. A first-order Markov model works well here for naturally occurring DNA, giving a small but significant overall improvement, even when its extra parameters are considered.

The precise architecture of the machine and the organization of states R, R2 and R3 is arbitrary to a certain extent. The current design prevents invisible repeats, i.e. where all characters are deleted, and this simplifies the algorithm design. But it is quite possible that some variation on the architecture might perform slightly better and a few variations have already been investigated. The important points are that the current model is simple, and that its complexity is determined by multi-state distributions (see Section 3) on the transitions out of each state and by the probability distribution on the source locations of repeats. More generally we advocate the family of such finite-state models. For example, linear costs for gaps (indels) within repeats can be modeled by states and operations for start-insert and continue-insert etc. as in the sequence alignment problem (Allison et al., 1992). One can even envisage a systematic search through simple machines to complex machines.

Note that probabilistic finite state machines, such as that above, are hidden Markov models (HMM) in mathematical terms. However, HMM has largely come to mean a generalization of profiles in molecular biology, see Eddy (1998) for example. The latter usually contains an explicit representation of the characters of a type of sequence. The machine discussed here contains no such representation of characters. Instead it contains general statistics on the relationship between parts of the sequence. The machine could be made to match or recognize a particular family of sequences but only if it were given one or more examples prepended to the data to be searched. To an extent, it models the process by which a sequence could be generated and it is natural to use the term machine for this reason, and also because it is common in compression, makes a distinction with the other kind of HMM and is consistent with earlier work (Allison et al., 1992).

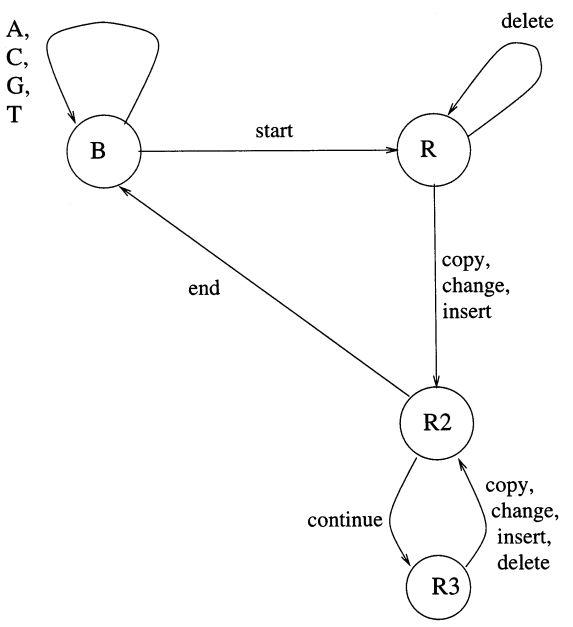


Fig. 1. Generating finite-state machine.

Given probabilities for the machine's operations in its various states, it is possible to generate random sequences from the model. The machine can also be used as the basis of inference algorithms to analyze a given sequence. The repeat graph (Fig. 2) represents all possible sequences of operations the machine might have gone through in generating a particular sequence, here one beginning ACA… A node in the graph represents the machine in some state at some position in the sequence. State R3 has been collapsed into state R2 for simplicity. Note that the graph is acyclic. There are many explanations for the sequence. One possibility is to generate all characters, ACA…, in the machine's base state. Another is to generate AC in the base state and then start a repeat which copies a character, A, and so on. The probability of each such explanation is the product of its individual steps. Any two explanations are exclusive hypotheses for the sequence so it is legitimate to add all of their probabilities together. Doing so gives the total probability of the sequence under the model, $P(\mathrm{D} \mid \mathrm{H})$ where H now represents the machine, because there is no other way in which the machine could generate it. This sum can be calculated in $O(n^2)$ time by an algorithm that scans the graph row by row, there being $O(n^2)$ nodes in the graph. $O(n)$ space is sufficient because a row of the graph can be computed given just the previous row.

The algorithm also calculates the contribution of each of the paths to a node towards the probability of the sequence up to the current position, and uses this information to produce a grey-scale plot which shows the positions of repeated substrings and their fidelity. Fig. 3 shows such a repeat plot for the human beta globin cluster HUMHBB. For short sequences one can perform a second backwards pass through the repeat graph and thus calculate the probability of the true path going through each node. This is analogous to the forward–backward dynamic programming algorithm which yields alignment density plots in sequence alignment (Allison et al., 1992). However, it requires either $O(n^2)$ space or greater time-complexity and is impractical for long sequences. In practice, the plot derived from the forward pass alone gives adequate indication of repeats.

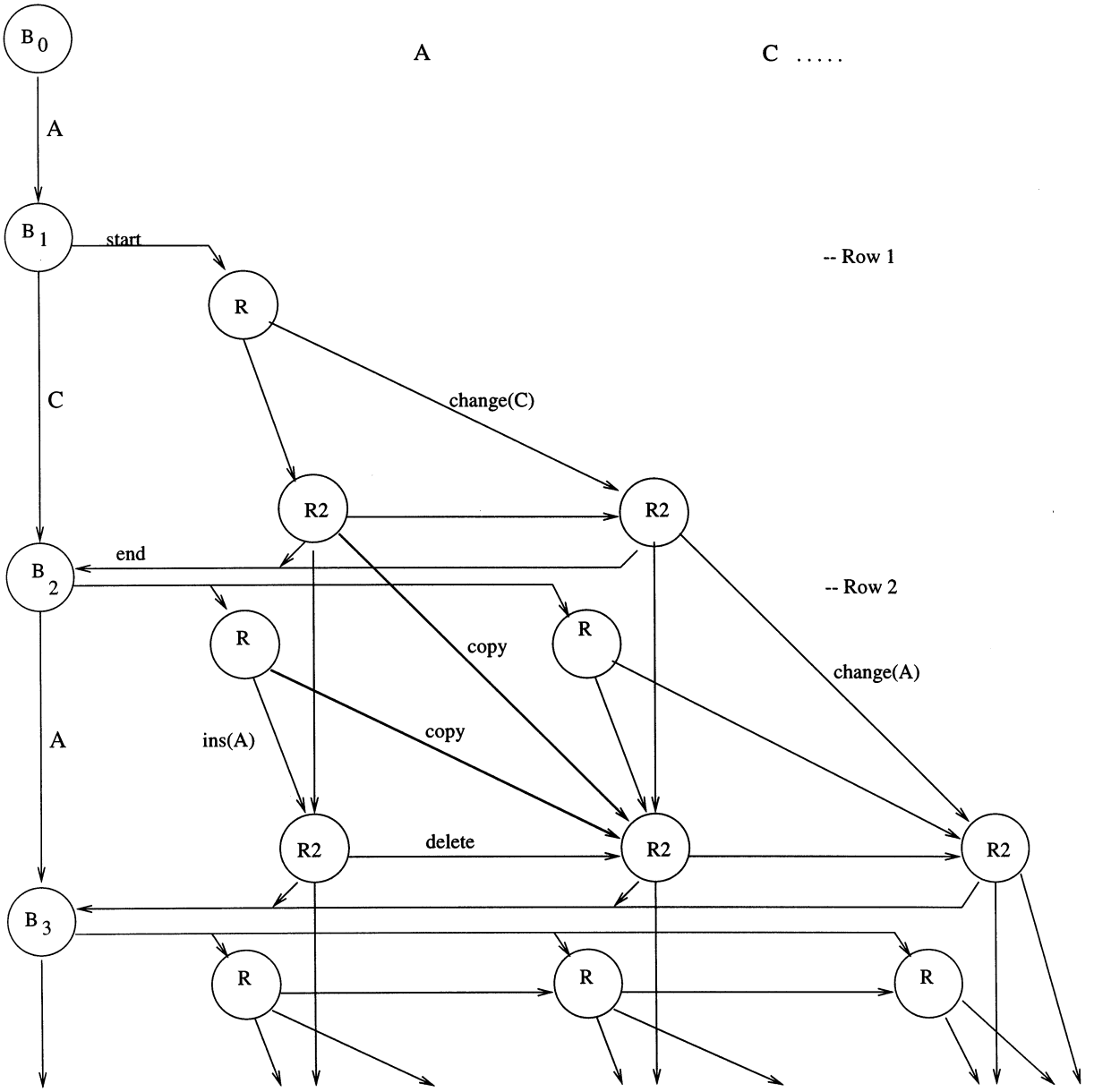


Fig. 2. Repeat graph.

The length of a repeat is coded by stating that it continues base by base until it finally ends. This amounts to a unary code which corresponds to a geometric probability distribution on lengths. It is unlikely that repeat lengths are drawn from a geometric distribution. The issue of repeat lengths is similar to that of indel (gap) costs in sequence alignment (Gotoh 1982). The incremental cost of extending a repeat by one base is a constant, $-\log_2(P(\text{continue}))$. So ignoring the start and any mutations, the overall cost of a repeat is linear in its length. This is what allows the probabilities of all explanations to be summed in $O(n^2)$ time; in effect all of the paths through a node are extended simultaneously. In fact, piece-wise linear costs for repeat lengths have the same property and correspond to a 'mixture' of geometric distributions, e.g. short, probable repeats and long, less probable repeats. They are implemented by adding extra states to the generating machine and still lead to an $O(n^2)$ time algorithm, although one with a larger constant of proportionality. If one wanted a single optimal explanation under the model it would be possible to use other

probability distributions for repeat lengths, provided they give concave (down) cost functions when the negative $\log_2$ is taken, by adapting the technique of Miller and Myers (Miller and Myers, 1988) from alignment. This would give an $O(n^2)$ or $O(n^2 \log n)$ algorithm depending on the properties of the cost function.

## 6. Parameter estimation

The previous discussion of the sequence analysis algorithm assumed that the machine's parameters were known in advance, but usually they are not. Parameters are estimated by an expectation maximization (EM) process (Baum and Eagon 1967; Baum et al., 1970; Dempster et al., 1977). Initial parameter values are assumed and the algorithm makes a pass through the repeat graph. As it does so it computes the frequencies of the machine operations up to each node in the graph. When two or more paths meet, the weighted averages of their frequency counters are formed, weighted by the relative probabilities of the paths. The node representing the final base state gives overall operation frequencies and these yield parameter estimates for the next iteration. The process is stopped when the overall message length improves by less than some small limit; a few iterations are generally sufficient. Convergence is guaranteed; it could be convergence to a local optimum but this is not a problem in practice. (Also note that more complex machines which realize mixtures of different types of repeats effectively accommodate multiple local optima of simpler machines.)

The parameters of the model are $P$(repeat) the probability of starting a repeat, $P$(continue) the probability of continuing a repeat, $P$(end) the probability of ending a repeat, $P$(copy) the probability of a copy, $P$(change) the probability of a change, $P$(insert) the probability of an insertion and $P$(delete) the probability of a deletion within a repeat; the last four sum to one. Test runs were performed by generating data from the model with known parameter settings and then attempting to re-

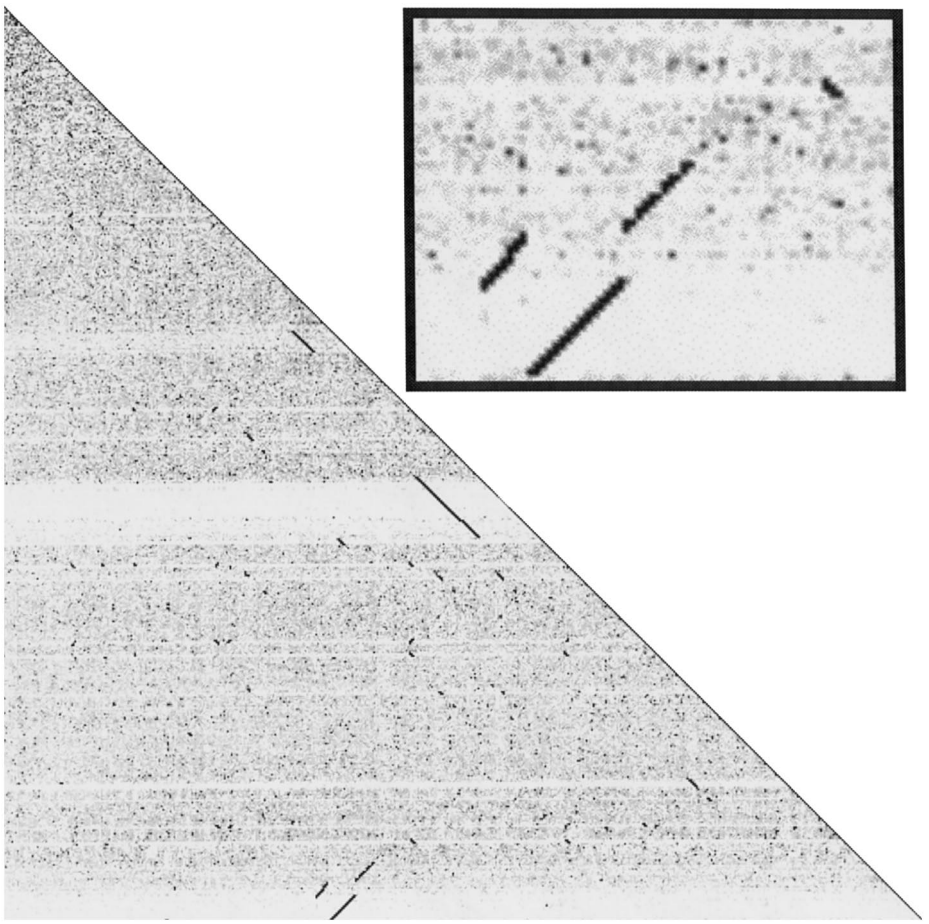

Fig. 3. Grey-scale repeat plot of HUMHBB (73 kb).

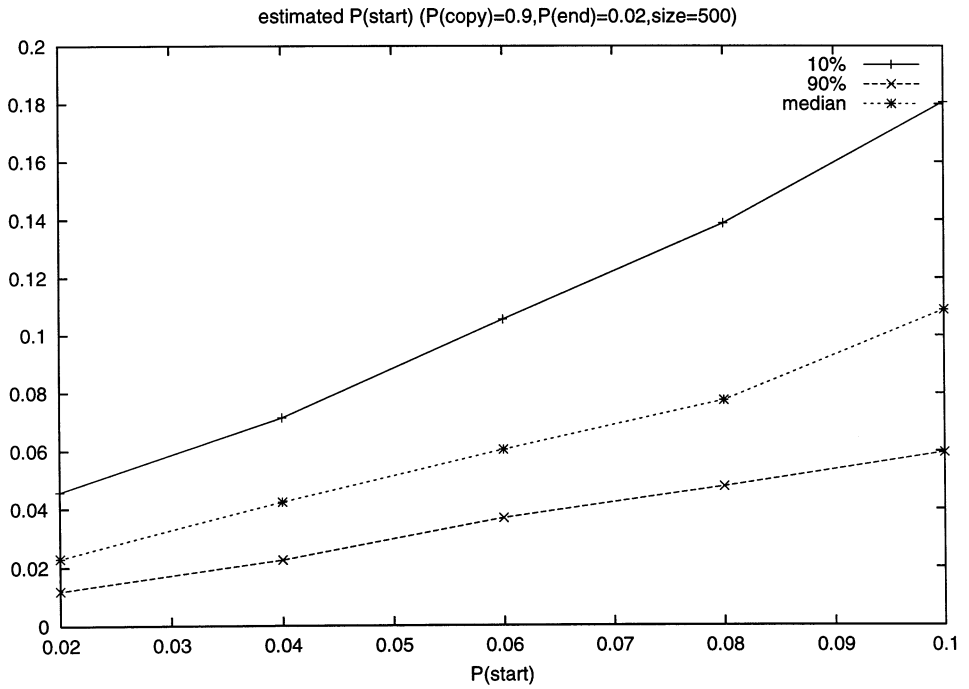


Fig. 4. Real vs. inferred values of $P$(start).

cover these values by the EM process described above. Any inference program must be able to perform well in this situation. One parameter at a time was varied systematically while the others were held constant. We generated and tested 100 sequences of length 500 for each parameter setting. For example, Fig. 4 shows a graph of actual versus inferred values of $P$(start). The median shows good agreement across the likely range of this parameter. Similar tests for other parameters also gave good results.

## 7. DNA

We can look at the compression of a DNA sequence in three different ways. First, we can calculate a number for the overall compression of a DNA sequence under a given model, in bits per character. Second, we can calculate and plot the information content of the sequence under the model, base by base. Finally, we can generate the grey-scale repeat plot mentioned previously, which shows the location of repeats, with the location and contribution of the preceding similar subsequences. The repeat plot looks superficially like the traditional dot-plot, but does not require the user to guess at parameters such as window-size and stringency. The three different representations of the compression of a sequence under the model, yield different information and complement each other.

The single figure for compression is, as mentioned previously, a natural way to compare competing models of the sequence. Table 3 gives the (compressed) message lengths of *Drosophila melanogaster* mastermind cDNA (DMMASTER), human beta globin region (HUMHBB), tobacco chloroplast genome DNA (CHTNTXX), and yeast chromosome III (SCCHRIII) under Biocompress2 (Grumbach and Tahi 1994), CDNAcompress, and our approximate repeat model. As noted in the table, the long sequences were processed by a faster approximation algorithm, which is described in the next section. While giving no detail about the structure of a sequence other than its complexity under models, we believe that if a single figure of merit is needed then the total message length figure is the natural one to use when comparing competing models of sequences, as was argued before.

The Drosophila mastermind protein is repetitive. Not surprisingly its cDNA is also repetitive. Looking at the parameter estimates, the algorithm finds numerous forward repeats ($P$(start) = 0.015) that are short ($P$(end) = 0.06) with few differences ($P$(copy) = 0.94), which is consistent with what is known about this gene. Tobacco chloroplast DNA contains a 25 kb reverse complementary repeat which dominates the results: $P$(start) = 0.00005 and $P$(copy) = 0.9999. Yeast chromosome III was included as an example of a long sequence, and overall is less compressible than the other sequences shown here.

Table 3
Compression of real DNA Sequences [a,b]

| Sequence | Length | Compressed (bits/base) by three methods | | |
|---|---|---|---|---|
| | | Biocompress2 | CDNAcompress | Approx. repeat model |
| DMMASTER | 6.3 K | – | – | 1.853 [c] |
| HUMHBB | 73 K | 1.88 | 1.77 | 1.728 [d] |
| CHNTXX | 155 K | 1.62 | 1.65 | 1.614 [d] |
| SCCHRIII | 315 K | 1.92 | 1.94 | 1.913 [d] |

[a] Sequences are listed using their GenBank identification and are: *Drosophila melanogaster* neurogenic locus mastermind cDNA (DMMASTER), human beta globin gene cluster (HUMHBB), tobacco chloroplast genomic DNA (CHNTXX), and *Saccharomyces cerevisiae* chromosome III (SCCHRIII).
[b] Figures for Biocompress2 and CDNAcompress from Loewenstern and Yianilos (1997).
[c] Full $O(n^2)$ algorithm.
[d] Faster approximation algorithm.

We used the single figure for compressibility to test an extension to the model. As mentioned above, the new model can be extended to allow a mixture of types of repeats. In fact a whole family of such models based on different 'machines' is possible. A model distinguishing two types of forward repeats and two types of reverse complementary repeats, was tried on the human beta globin gene cluster. This model gave a small improvement in compression, from 1.728 bits/base to 1.725 bits/base, even when the costs of stating its extra parameters were accounted for. Low fidelity ($P(\text{copy}) = 0.78$) forward repeats occur most often ($P(\text{start}) = 0.0006$). High fidelity ($P(\text{copy}) = 0.96$) forward repeats occur less often and are shortest on average ($P(\text{end}) = 0.006$). Low fidelity ($P(\text{copy}) = 0.70$) reverse complementary repeats occur rarely ($P(\text{start}) = 0.0002$). Medium fidelity ($P(\text{copy}) = 0.86$) reverse complementary repeats occur rarely and are longest on average ($P(\text{end}) = 0.003$). Point mutations are about ten times more probable than insertions and deletions.

Plotting the information content, base by base, allows the immediate detection of repeats and areas of low information content in the sequence. Fig. 5 shows a moving average of information content over a window of 100 bases for the human beta globin cluster (HUMHBB). This gene cluster contains five transcribed genes and one pseudogene, in the following order: 5′-epsilon-G-gamma-A-gamma-pseudobeta-delta-beta-3′ (Collins and Weissman, 1984). Two large areas of low information content are immediately apparent in the figure and signal large areas of repeated sequence. In the grey-scale repeat plot (Fig. 3), it can be seen that the long repeat starting around position 40 000 originates from the region around positions 34 000–38 000. This corresponds to the known close relationship of the gamma-A gene (39 432–41 000 bp) to the upstream gamma-G gene (34 396–36 087 bp) (Slightom et al., 1980). Also in the grey-scale plot, the long repeat at the 3′ end of the gene can be seen to originate from the region 25 000–29 000, inverted and rearranged, and corresponds to the known copy and rearrangement of an element belonging to the L-1 family of long interspersed repeats (Rogan et al., 1987).

The moving average plot (Fig. 5) also shows several sharp troughs of low information content that are less readily seen on the repeat plot (Fig. 3). Repeated Alu elements bring the information content down to 0.8–1.2 bits per character, seen in the sharp troughs at positions 1987, 5624, 8038, 10 630, 16 910, 17 945, 32 408, 37 343, 50 933, 51 944, 65 531 and 66 794 (Barabelle et al., 1980; Collins and Weissman 1984; Li et al., 1985; Poncz et al., 1983). These regions show up as small diagonals on the repeat plot. When printed on a large scale, it can be seen that some of the Alu elements are more closely related than others, and that some are in an inverted orientation. The first instance of an Alu sequence in HUMHBB is at position 175 (Li et al., 1985), and cannot be seen on the base-by-base plot. It is barely visible on the repeat plot, as a contributor to the second Alu at position 1987. The base by base plot of information content also shows similarities among the various genes in this cluster, particularly at their 5′ ends, detected as dips around 34 000, 45 000, 55 000 and 62 000 bp. While these can also be seen on the repeat plot, they do not stand out as clearly as the longer repeats, so the dip in the information content plot serves as a useful locator.

The base-by-base plot of information content also shows up small repeats, such as the small inverted RY6 repeat at position 9050 and direct repeats at positions 9745 and 12916 (Li et al., 1985), as well as areas of low information content, such as a region of poly(GT) around position 58 000, a region of (ATTTT)*n* around 59 500, and a region of (T(A | G))*n* around position 60 500 (Collins and Weissman 1984). Both of these kinds of repeats decrease the local information content

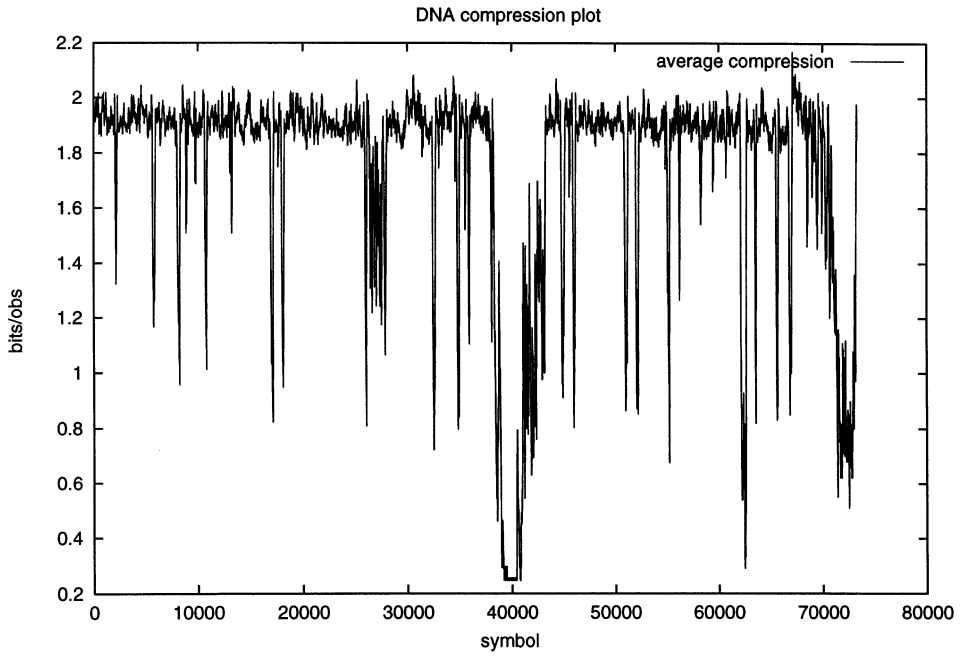


Fig. 5. Information content plot of HUMHBB.

significantly less than the Alu sequences, to only 1.6–1.7 bits per character, and are too small or too distantly related to show clearly on the grey-scale repeat plot of the entire gene cluster.

## 8. Faster approximation algorithm

The inference algorithm described above takes $O(n^2)$ time per EM iteration. This amounts to 30 min per iteration for DMMASTER (6 kb) and 2 days per iteration for HUMHBB (73 kb) on a Silicon Graphics 'Indy' workstation. A faster algorithm is necessary for long sequences and an approximation algorithm was created for this reason. The idea is to investigate only the most important regions of the repeat graph (Fig. 2). Ignoring parts of the repeat graph only results in lost probability from $P(D \mid H)$, and gives a lower bound on the true probability, thus giving an upper bound on message length, and so is a conservative approximation. This is effected by using a hash-table which contains $k$-tuples and their locations in the sequence, where $k$ is a constant, typically in the range 6–14. It is assumed that most approximate repeats will contain at least one exact match on a $k$-tuple, hopefully one near the start of the repeat. Such a match 'turns on' a region of the repeat graph $\pm 5$ cells around the hit. The region remains turned on while it is contributing more than a 'threshold' of $(-\log_2)$ probability to the sequence, currently at least as much as the base state, minus the repeat start-up cost, plus 4 bits. Regions can grow, shrink, merge or be turned off. Adjusting the threshold and the value of $k$ allows the algorithm to process long sequences quickly, at some loss of accuracy. Fig. 6 shows a plot of $-\log_2$ probability, in bits per base, and of running time as $k$ is varied, the data being 6000 bases of HUMHBB from position 23 000 which contains one long approximate repeat. Alternative methods of speeding up the full $O(n^2)$ algorithm are being investigated.

## 9. Related models

The new sequence model explicitly describes approximate repeats. These include both short 'self similar' repeats such as TATAATATTA which might be classed as low-order Markov model patterns and long repeats such as gene duplication events. As presented above, the model does not contain any library of predefined patterns but it is straightforward to prepend one or more library patterns, e.g. a consensus Alu, to a sequence and run the algorithm on the result. This also suggests that two (or more) sequences, S1 and S2, can be compared by running the algorithm on S1, on S2, and on their concatenation, S1++S2 or S2++S1, to give an information measure for $(S1 \mid S2)$ or for $(S2 \mid S1)$: If S1 and S2 are unrelated then S1 gives no information about S2. However, if the compressed length of S1++S2 is less than the compressed length of S1 plus the compressed length of S2 then this shows

that S1 gives significant information about S2. In this case the repeat plot gives a non-order-preserving alignment of S1 and S2. This could be useful for comparing S1 and S2 when the individual sequences are repetitive or have a low information content.

The above idea can be related to Tichy (1984) edit-distance based on 'block-moves', the application being in revision control of documents. The object is to create a 'target sequence', S2, from a 'source sequence', S1. The atomic operation is to copy a block of some length from the source sequence, and the target sequence is built up by a sequence of these block-moves. The objective is to use the minimum number of block-moves. If all block-moves have the same cost (same probability) then the result is a most probable explanation for S2 given S1. This holds if the source addresses of blocks and if block lengths are drawn from uniform distributions. Tichy's model can be seen as a kind of LZ model between two sequences. It can be implemented by an algorithm for LZ compression by concatenating S1 and S2 and restricting repeats to be from S1 into S2. In the new model defined in this paper, the blocks can be approximate copies.

The approximate repeats model can be applied to alphabets other than DNA, compressing text rather well, for example. It can be applied to protein sequences as is, but ideally should have an amino-acid substitution matrix incorporated to deal with mismatches in that case.

## 10. Conclusions

Sequence analysis is carried out for a variety of purposes, e.g. to find common subsequences (low information) or surprising subsequences (high information), to find repetition, signals, motifs, pattern or structure (low information), all against a background of chance matches, natural variation, evolution and mutation. It has been argued that information content, as realized in the compression or message length criterion, captures an important aspect of these intuitions. Only by having a good statistical model of sequences is it possible to quantify 'common' and 'surprising'. Even if this is done subject to a number of simplifying assumptions then at least those assumptions, i.e. the design and parameters of a model, are explicit and are open to challenge and objective improvement. Including the cost of the model itself prevents overfitting and allows simple and complex models to be compared fairly. We suggest that models based on finite-state machines have many practical advantages: their complexity is easy to quantify, the resulting inference algorithms are feasible, i.e. have reasonable complexity, and one can envisage a systematic search through at least the smaller finite-state machines.

We have presented a new model of biological sequences (and more generally a family of such models) which explicitly models the duplication of substrings, in the forward and reverse-complementary senses, and

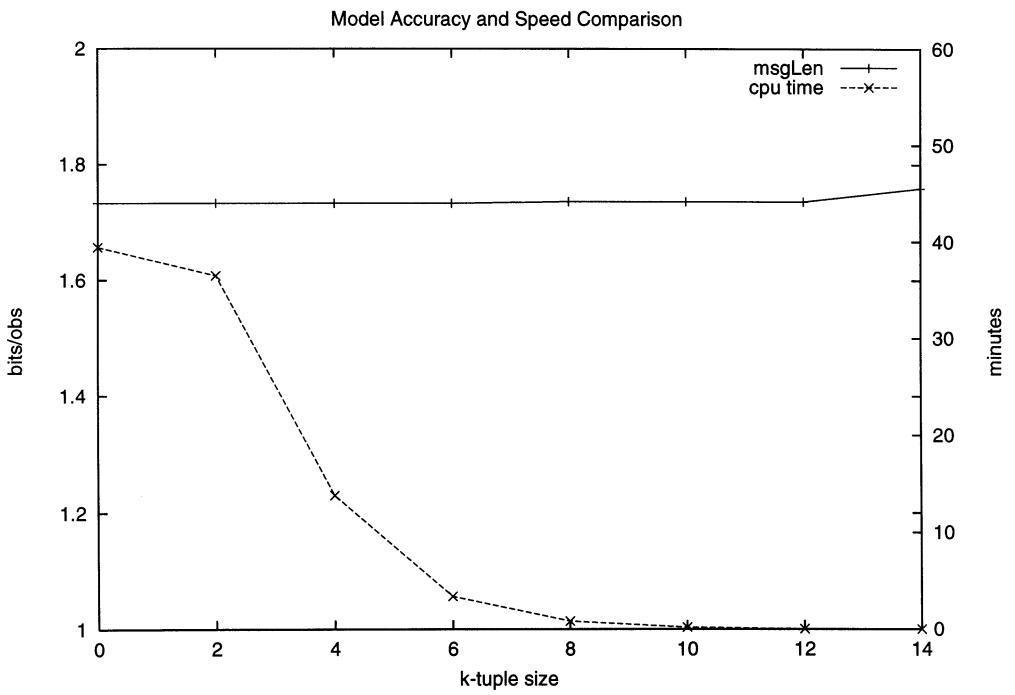


Fig. 6. Model accuracy and speed for the approximation algorithm.

their subsequent mutation. An $O(n^2)$ algorithm calculates the information content of sequences and identifies interesting regions. It is practical for sequences of thousands of characters. A faster approximation algorithm with near-linear running time exists for longer sequences. The algorithms give good compression on real DNA sequences. Their outputs include the base by base information content of a sequence, useful for locating repetitive areas, and a grey-scale repeat plot showing the origin of approximate repeats.

## References


Agarwal, P., States, D.J., 1994. The repeat pattern toolkit (RPT): analyzing the structure and evolution of the *C. elegans* genome, Proceedings of the Second Conference on Intelligent Systems, Mol. Biol., pp. 1–9.

Allison, L., Edgoose, T., Dix, T.I., 1998. Compression of strings with approximate repeats, Intelligent Systems in Molecular Biology, ISMB '98, Montreal, pp. 8–16.

Allison, L., Wallace, C.S., Yee, C.N., 1992. Finite-state models in the alignment of macro-molecules. J. Mol. Evol. 35 (1), 77–89.

Allison, L., Yee, C.N., 1990. Minimum message length encoding and the comparison of macromolecules. Bull. Math. Biol. 52 (3), 431–453.

Barabelle, F.E., Shoulders, C.C., Goodbourn, S., Jeffreys, A., Proudfoot, N.J., 1980. The 5′ flanking region of human epsilon-globin gene. Nucleic Acids Res. 8, 4393–4404.

Baum, L.E., Eagon, J.E., 1967. An inequality with applications to statistical estimation for probabilistic functions of Markov processes and to a model of ecology. Bull. AMS 73, 360–363.

Baum, L.E., Petrie, T., Soules, G., Weiss, N., 1970. A maximization technique occurring in the statistical analysis of probabilistic functions of Markov chains. Ann. Math. Stat. 41, 164–171.

Bayes, T., 1763. An essay towards solving a problem in the doctrine of chances. Phil. Trans. Royal Soc. Lond., 53, 370–418; reprinted in Biometrika 45, 296–315 (1958).

Boulton, D.M., Wallace, C.S., 1969. The information content of a multistate distribution. J. Theor. Biol. 23, 269–278.

Chaitin, G.J., 1966. On the length of programs for computing finite binary sequences. J. Assoc. Comp. Mach. 13 (4), 547–569.

Cleary, J., Teahan, W.J., 1997. Unbounded length contexts for PPM. Comp. J. 40 (2/3), 67–75.

Collins, F.S., Weissman, S.M., 1984. The molecular genetics of hemoglobin. Prog. Nucleic Acids Res. Mol. Biol. 31, 315–462.

Dempster, A.P., Laird, N.M., Rubin, D.B., 1977. Maximum likelihood from incomplete data via the EM algorithm. J. Royal Stat. Soc. B 39, 1–38.

Eddy, S.R., 1998. Profile hidden Markov models (review). Bioinformatics 14 (9), 755–763.

Gotoh, O., 1982. An improved algorithm for matching biological sequences. J. Mol. Biol. 162, 705–708.

Grumbach, S., Tahi, F., 1994. A new challenge for compression algorithms: genetic sequences. Inf. Proc. Manag. 30 (6), 875–886.

Kolmogorov, A.N., 1965. Three approaches to the quantitative definition of information. Probl. Inf. Transm. 1 (1), 1–7.

Langdon, G.G., 1984. An introduction to arithmetic coding. IBM J. Res. Dev. 28 (2), 135–149.

Lempel, A., Ziv, J., 1976. On the complexity of finite sequences. IEEE Trans. Inf. Theory IT-22, 783–795.

Li, Q., Powers, P.A., Smithies, O., 1985. Nucleotide sequence of 16-kilobase pairs of DNA 5′ to the human epsilon-globin gene. J. Biol. Chem. 260, 14901–14910.

Loewenstern, D.M., Yianilos, P.N., 1997. Significantly lower entropy estimates for natural DNA sequences. IEEE Data Compress. Conf. DCC97 C97, 151–160.

Miller, W., Myers, E.W., 1988. Sequence comparison with concave weighting functions. Bull. Math. Biol. 50 (2), 97–120.

Milosavljevic, A., Jurka, J., 1993. Discovering simple DNA sequences by the algorithmic significance method. Comp. Appl. BioSci. 9 (4), 407–411.

Poncz, M., Schwartz, E., Ballantine, M., Surrey, S., 1983. Nucleotide sequence analysis of the delta-beta globin gene region in humans. J. Biol. Chem. 258, 11599–11609.

Powell, D.R., Allison, L., Dix, T.I, Dowe, D.L., 1998a. Alignment of low information sequences, Australian Computer Science Theory Symposium, CATS '98, Perth, Springer Verlag, ISBN 981-3083-92-1, pp. 215–230.

Powell, D.R., Dowe, D.L., Allison, L., Dix, T.I. 1998b. Discovering simple DNA sequences by compression, Pacific Symposium on Biocomputing, Hawaii, pp. 597–608.

Rissanen, J. 1987. Stochastic complexity. J. Royal Stat. Soc. B 49(3) 223–239 and 252–265.

Rivals, E., Dauchet, M. 1997. Fast discerning repeats in DNA sequences with a compression algorithm, Proceedings of the Genome Informatics Workshop, Tokyo, pp. 215–226.

Rogan, P.K., Pan, J., Weissman, S.M., 1987. L1 repeat elements in the human epsilon-G-gamma-globin gene intergenic region: sequence analysis and concerted evolution within this family. Mol. Biol. Evol. 4, 327–342.

Shannon, C.E., 1949. The Mathematical Theory of Communication. U. Illinois Press, USA.

Slightom, J.L., Blechl, A.E., Smithies, O., 1980. Human fetal G-gamma- and A-gamma globin genes. Cell 21, 627–638.

Solomonoff, R., 1964. A formal theory of inductive inference, I and II. Inf. Control 7, 1–22; 224–254.

Tichy, W.F., 1984. The string-to-string correction problem with block moves. ACM Trans. Comp. Sys. 2 (4), 309–321.

Wallace, C.S., Boulton, D.M., 1968. An information measure for classification. Comp. J. 11 (2), 185–194.

Wallace, C.S., Freeman, P.R., 1987. Estimation and inference by compact coding. J. Royal Stat. Soc. Series B. 49 (3), 240–265.

Wootton, J.C., 1997. Simple sequences of protein and DNA, In: Bishop, M.J., Rawlings, C.J. (Eds.), DNA and Protein Sequence Analysis, A Practical Approach, IRL Press, pp. 169–183.

Ziv, J., Lempel, A., 1977. A universal algorithm for sequential data-compression. IEEE Trans. Inf. Theory IT-23, 337–343.