# A Genome Alignment Algorithm Based on Compression

Minh Duc Cao*[1], Trevor I Dix[1,2]and Lloyd Allison[1]

[1] Clayton School of Information Technology
Monash University
Clayton 3800, Australia
[2] Faculty of Information & Communication Technologies,
Swinburne University of Technology,
Hawthorn 3122, Australia

Email: Minh Duc Cao*- minhduc@infotech.monash.edu.au; Trevor I Dix - trevor@infotech.monash.edu.au; Lloyd Allison - lloyd@infotech.monash.edu.au;

*Corresponding author

**Abstract** Traditional genome alignment methods based on dynamic programming are often a. computational expensive, b. unable to compare the genomes of distant species and c. unable to deal with low information regions. We are presenting an information-theoretic approach for pairwise genome local alignment. Our method, the expert model aligner, the XMAligner, relies on the expert model compression algorithm. To align two sequences, XMAligner first compresses one sequence to measure the information content at each position in the sequence. Then the sequence is compressed again but this time with the background knowledge from the other sequence to obtain the conditional information content. The information content and the conditional information content from the two compressions are examined. Similar regions in the compressed sequence should have the conditional information content lower than the individual information content. The method is applied to align the genomes of *Plasmodium falciparum* and *Plasmodium knowlesi* against other three *Plasmodium* genomes with different levels of diversity. Despite the differences in nucleotide composition of the reference sequences, the conserved regions found by XMAligner in three alignments are relatively consistent. A strong correlation was found between the similar regions detected by the XMAligner and the hypothetical annotation of *Plasmodium* species. The alignment results can be integrated into the DNAPlatform for visualisation.

## 1 Background

Advances in sequencing technology allow the high throughput production of biological sequences extracted in sequencing laboratories around the world. The exponential increase of biological data extracted recently has led to the development of bioinformatics applications which attempt to solve biological problems by applying techniques from other disciplines such as statistics and computer science. One of the most fundamental objectives of bioinformatics is the discovery of important patterns from large biological databases. This task can be quite challenging as conventional information extraction methods can be overwhelmed by volume and misled by statistical biases. It is important to develop new and novel tools for analysing such data. Such tools need to be time efficient as biological sequences can be very long.

The most important tool for sequence analysis is sequence alignment which attempts to arrange biological sequences of DNA, RNA or protein to identify regions of similarity. The similarities between sequences could provide clues to discover the relationship between species, annotate new sequences or compare an unknown sequence against existing sequences in a large database. There are two types of sequence alignment, namely *global alignment* and *lo-*

*cal alignment.* Global alignment attempts to match the entire sequence from one end to the other of each sequence. This technique is suitable for comparing two short sequences which are expected to have similar functions such as two proteins or two genes. On the other hand, when analysing long sequences such as chromosomes or genomes, local alignment which finds only similar regions between two sequences is more suitable.

Most alignment methods are based on dynamic programming paradigm. Needleman-Wunsch [1] and Smith-Waterman [2] are two examples of early global alignment and local alignment respectively. The quadratic time complexity of dynamic programming was feasible in the early days of bioinformatics when sequences to be analysed were relatively short. However, in the last decade, as more and more long sequences available, it becomes impossible to use the dynamic programming algorithms for analysing chromosomes and genomes which can be hundreds of millions of bases long.

Traditional alignment methods also rely heavily on a substitution matrix which is selected empirically or based on some assumptions about the distance between two species. Using a generic substitution matrix may be justified for protein alignment as different amino acids have different properties. However, for DNA, more than one codon can code for an amino acid. Different strains show different preferences for a codon that encodes a given amino acid [3]. It is therefore, sometimes very hard to find a suitable substitution matrix for alignment, especially when the sequences being analysed are unknown.

Other problems associated with genome alignment are rearrangements and low information regions. Genomes often contain a great deal of repetitive and low information regions. It is estimated that the human genome contains more than 50% of repeat DNA and about 30000 CpG islands which are genomic regions that contain a high frequency of CG [4]. Such sequences of biased composition and low information could cause false positives to alignment algorithms.

In this paper, we present XMAligner, a novel method for genomic alignment based on information theory. First, the information content for each position in a sequence is measured by the expert model compression algorithm [5]. Next the sequence is compressed again with the background knowledge from the other sequence. The resulting conditional information content should be lower in related parts of the two sequences. This method does not require masking out low information areas; hence no information is lost. It can adaptively build a substitution matrix by gathering statistics for the two sequences being analysed. The method is shown to be practical and can handle sequences of hundreds of millions of bases.

## 1.1 Related work

Since alignment is the most basic tool for sequence analysis, much research has been done in this field. The dynamic programming inspired alignment algorithms [1] and [2] were developed in 1970 and 1981 respectively. These methods attempt to match all possible pairs the two sequences by using a scoring scheme and find the optimal alignment which has the best matching score. They have been used intensively, primarily for comparing proteins or DNA sequences of a single gene.

These alignment approaches require run-time and space complexity of $O(mn)$ for aligning two sequences of lengths $m$ and $n$ and therefore are less attractive for long sequences. They became infeasible for many applications in early 1990s which required matching a sequence with a relatively large database of known sequences. To trade the sensitivity for running time, heuristic search methods are used. Instead of comparing every single base of the two sequences, FASTA [6] and BLAST [7], the two most popular database search tools, search for seeds of $k$ consecutive matches. Seeds are then extended to include substitutions and gaps by dynamic programming approach to form similar regions.

Since 1995 when the first genome was sequenced [8], there has been much research on tools that are capable of comparing genomes. Most alignment methods rely on the ideas of FASTA and BLAST; they use different variations for finding seeds and extending seeds to find conserved regions. A hash table is used in SAHA [9] for locating seeds which are matched k-tuples. The seeds are then sorted and linked together. Gapped BLAST [10] and BLASTZ [11] find seeds of short near exact matches. Seeds are extended first without allowing gaps. Each gap-free match that is longer than a certain threshold is then extended by the dynamic programming procedure that permits gaps. The BLAST Like Alignment Tool, BLAT, [12] works in a similar way to BLAST to find seeds, and clumps similar regions together to form larger regions.

A number of genome global alignment tools make use of suffix tree [13] to find seeds instead of using hash table that can only find fixed length k-mer matches. MUMmer [14,15] represents a sequence by a suffix tree and finds the maximum unique matches (MUMs) of the two genomes. The MUMs are then clustered based on size, gap and distance and the gaps between clusters are closed using a modified Smith-Waterman algorithm. Similarly, AVID [16] also locates exact matches as seeds using suffix trees. Short matches are considered not biologically significant and are filtered out while longer matches are used as non-overlapping, non-crossing anchors. AVID recursively aligns regions between anchors by dynamic programming to perform global alignment. The Multiple Genome Aligner, MGA [17], extends the idea to perform multiple alignment by detecting all *maximal multiple exact matches* (multiMEM) longer than a threshold.

One problem in genomic alignment research is that genomes contain a great deal of low information areas such as repeats and skewed composition of bases. Alignment tools based on string matching and dynamic programming are prone to false positives in these areas. The common technique to address this problem in [9,11,12,14–17] is to *mask out* low information content areas, but this may miss out some important patterns. Some genes, for example, are copied abundantly back to the genome to maximise their inclusive fitness. Masking out low information areas also gives rise the issue "how low is low" [18].

Dynamic programming alignment algorithms rely on a scoring scheme of mismatches and gaps to extend seeds. This includes a substitution matrix and gaps scores. A substitution matrix is drawn from some assumptions about the sequences being analysed such as the rate of mutation (PAM [19]) or the minimum percentage identity of the sequences (BLOSUM [20]). While much research has been done to find substitution matrices for protein alignment, little attention has been paid to DNA substitution matrix. Since more than one codon can code for the same amino acid, and different species have different preferences for nucleotide composition, it is more difficult to anticipate the mutation rates in DNA sequences. For that reason, genomic alignment tools often use some ad-hoc substitution matrices and gap scores. Furthermore, DNA sequences tend to be more divergent than proteins and therefore these methods often fail to align more distant genomes.

We take an information theory [21] approach to sequence analysis. Similar to Powell et al. [18], our work is based on the premise that if two sequences are related, one sequence must tell something useful about the other [18]. The information content of a sequence can be measured by lossless compression. By examining information content sequences [22] produced with and without a background sequence, we can identify the similar regions of the two sequences. The biological related compression *expert model* [5] in our earlier work provides attractive features for genome comparison. It performs among the best biological compression algorithms in the literature in practical time and space requirement. More importantly, it can measure the information content of each individual nucleotide in a sequence.

*Plasmodium falciparum* and *Plasmodium vivax* cause the two most dangerous malaria parasites in the world and they together account for as many as three million deaths a year. *P. falciparum*, the deadliest malaria, is prevalent in sub-Saharan Africa while *P. vivax*, the most common malaria and is found mainly in Asia and Latin America. Though the two species are both malaria parasites and cause similar symptoms, their genomes are greatly different. The *P. falciparum* genome consists of 80.6% AT while the AT composition in the *P. vivax* genome is 62.4% [23]. Even in coding regions, the AT ratios in *P. falciparum* and *P. vivax* are 76.22% and 53.70% respectively though the proteins of the two species are relatively similar. A distinct advantage of our algorithm is that it is able to compare sequences with such biased composition.

## 2 Method

Information theory directly relates entropy to the transmission of a sequence under a statistical model of compression. Suppose a sequence $X$ is to be transmitted over a reliable channel. The sender first compresses $X$ using a compression model and transmits the encoded message to the receiver, who decodes the compressed stream using the same model to recover the original sequence $X$. The *information content* $\mathcal{I}_X$ of sequence $X$ is the amount of information actually transmitted, i.e. the length of the encoded message.

Suppose a sequence $Y$ related to $X$ is available to both parties, the sender needs to transmit only the information of $X$ that is not contained in $Y$.

Since the receiver also knows $Y$, $X$ can be recovered correctly. The amount of information actually transmitted in this case is called *conditional information content* of $X$ given $Y$, denoted as $\mathcal{I}_{X|Y}$, which is expected to be lower than the information content of $X$, i.e. $\mathcal{I}_{X|Y} < \mathcal{I}_X$. The more related the two sequences are, the more information the two sequences share, the shorter message is transmitted. The *mutual information* of $X$ and $Y$ is defined as the difference between the information content and the conditional information content: $\mathcal{I}_{X;Y} = \mathcal{I}_X - \mathcal{I}_{X|Y}$.

If one can find the references to the shared information between two sequences, one can compute the optimal alignment of the two sequences and vice verse. For genome analysis, a local alignment is applicable since only conserved areas show the similarities of the two sequences. The more similarities the alignment can find, the more shared information one can find. We define the optimal alignment as the one that produces the maximum mutual information content. The alignment of two sequences is in fact the best compression of one sequence given the background knowledge of the other. The method of compression is largely based on our early work, the expert model (XM) [5].

## 2.1 The Expert Model

As a statistical compression method, the XM algorithm compresses each symbol of a sequence $X$ by forming the probability distribution for the symbol and then using a primary compression scheme to code it. The information content symbol $x_i$ is computed as the negative log of the probability of the symbol:

$$\mathcal{I}(i) = -logPr(x_i) \tag{1}$$

The probability distribution at a position is based on symbols seen previously. Correspondingly, the decoder, also having seen all previous decoded symbols, is able to compute the identical probability distribution and can recover the symbol at the position.

In order to form the probability distribution of a symbol, the algorithm maintains a set of experts, whose predictions of the symbol are combined into a single probability distribution. An expert is any entity that can provide a probability distribution at a position. Expert opinions about a symbol are blended to give a combined prediction for the symbol.

The statistics of symbols may change over the sequence. One expert may perform well on some region, but could give bad advice on others. A symbol is likely to have similar statistical properties to the context surrounding, particularly the context preceding the symbol. The reliability of an expert is evaluated from its recent predictions. A reliable expert has high weight for combination while an unreliable one has little influence on the final prediction or may be ignored.

## 2.2 Type of Experts

An expert can be anything that provides a reasonably good probability distribution for a position in the sequence. A simple expert can be a Markov model (*Markov expert*). An order-k Markov expert gives the probability of a symbol in a position given $k$ preceding symbols. Initially, the Markov expert does not have any prior knowledge of the sequence and thus gives a uniform distribution to a symbol. The probability distribution adapts as the encoding proceeds. Essentially, the Markov expert provides the background statistical distribution of symbols over the sequence. Here we use an order-2 Markov expert for DNA, and order-1 for protein.

Different regions of a DNA sequence may have differing functions and thus may have different symbol distributions. Another type of expert is the *context Markov expert*, whose probability distribution is not based on the entire history of the sequence but on a limited preceding context. In other words, the context Markov expert bases its prediction on the local statistics. The Markov expert and context Markov expert are employed to compress a sequence to compute its information content.

To compress a sequence $X$ on the background knowledge from some sequence $Y$, a new type of expert is employed. XM employs a *align expert* that considers the next symbol in $X$ to be part of a conserved region and aligned with a certain symbol in the background sequence. After each prediction, the expert has a chance to review before moving on to the next symbol. It therefore, can use an adaptive code [24], over some recent history, to predict the probability of the next symbol. If the align expert believes that the current symbol $x_i$ is aligned with the symbol $y_j$, it gives a probability to its predicted symbol of:

$$p = \frac{C_{xi|yj} + 1}{C_{yj} + 4} \tag{2}$$

where $C_{yj}$ is the number of symbol $y_j$ the align expert has seen in $Y$ and $C_{xi|yj}$ is the number of $x_i$ that aligns to $y_j$. An align expert can proceed forward and backward to handle copy and reverse complement.

## 2.3 Proposing of Experts

At a position on the sequence, there could be $2|Y|$ possible align experts. This is too many to combine efficiently and anyway, most would be ignored. To be efficient, the algorithm must use at most a small number of align experts at any one time. The limit of experts at a given time is a parameter of the algorithm.

Proposing the best possible experts is similar to seeding in other alignment methods. There are several techniques the XMAligner uses to nominate experts. The first technique is using a hash table of hash size $k$. The hash table suggests every k-mers match as an align expert. In the second technique, our algorithm nominates the experts that have the longest match to symbols preceding the current position. It first builds the suffix array of the background sequence to suggest such experts quickly. We choose the *divsufsort* algorithm [25] for constructing the suffix array due to its fast speed and effective memory requirement.

There are two groups of nucleotides namely purine (C and T) and pyrimidine (A and G). The chemical structures of two nucleotides in a group are more similar than that from the other group. Because of this, the substitutions changing nucleotides in a group (transitions) are more common than substitutions that change the group of nucleotides (transversions). In order to permit mismatches in seeds, the XMAligner provides an option to match nucleotides to that from the same group. From our experience, performance of matching the group is significantly better than exact matching.

## 2.4 Combining Expert Predictions

The core part of the XM algorithm is the evaluation and combination of expert predictions. Suppose a panel of experts $E$ is available to the encoder. Expert $\theta_k$ gives the prediction $P(x_{n+1}|\theta_k, x_{1..n})$ of symbol $x_{n+1}$ based on its observations of preceding $n$ symbols. A sensible way to combine experts' predictions is based on Bayesian averaging:

$$\begin{aligned} P(x_{n+1}|x_{1..n}) &= \sum_{k\in E} P(x_{n+1}|\theta_k, x_{1..n}) w_{\theta_k,n} \\ &= \sum_{k\in E} P(x_{n+1}|\theta_k, x_{1..n}) P(\theta_k|x_{1..n}) \end{aligned} \tag{3}$$

In other words, the weight $w_{\theta_k,n}$ of expert $\theta_k$ for encoding $x_{n+1}$ is the posterior probability $P(\theta_k|x_{1..n})$ of $\theta_k$ after encoding $n$ symbols. $w_{\theta_k,n}$ can be estimated by Bayes's theorem:

$$\begin{aligned} w_{\theta_k,n} &= P(\theta_k|x_{1..n}) \\ &= \frac{\prod_{i=1}^{n} P(x_i|\theta_k, x_{1..i-1}) P(\theta_k)}{\prod_{i=1}^{n} P(x_i|x_{1..i-1})} \end{aligned} \tag{4}$$

Normalising equation 4 by a common factor $M$ we have:

$$w_{\theta_k,n} = \frac{1}{M} \prod_{i=1}^{n} P(x_i|\theta_k, x_{1..i-1}) P(\theta_k) \tag{5}$$

The normalisation factor $M$, in fact does not matter as equation 3 could be again normalised to have $\sum P(x_{n+1}|x_{1..n}) = 1$. Take the negative log of equation 5 and ignore the constant term:

$$\begin{aligned} &-log_2(w_{\theta_k,n}) \sim \\ &\quad -\sum_{i=1}^{n} log_2 P(x_i|\theta_k, x_{1..i-1}) - log_2 P(\theta_k) \end{aligned} \tag{6}$$

Since $-log_2 P(x_i|\theta_k, x_{1..i-1})$ is the cost of encoding symbol $x_i$ by expert $\theta_k$, the right hand side of equation 6 is the length of encoding of subsequence $x_{1..n}$ by expert $\theta_k$. As we want to evaluate experts on a recent history of size $w$, only the message length of encoding symbols $x_{n-w+1..n}$ is used to determine the weights of experts. We rewrite equation 6 as

$$w_{\theta_k,n} \propto 2^{MsgLen(x_{n-w+1..n}|\theta_k) - log_2 P(\theta_k)} \tag{7}$$

Suppose there are three hypotheses about how a symbol in sequence $X$ is generated: by the distribution of the species genome; by the distribution of the current subsequence; or by copying from sequence $Y$. We therefore entertain three experts for these hypotheses: (i) a Markov expert for the species genome distribution, (ii) a context Markov expert for the local distribution, and (iii) a repeat expert, which is the combination of any available align experts, for the third hypothesis. The experts' predictions are blended as in equations 3 and 7.

If a symbol is part of a conserved region, the align expert of that region must predict significantly better than a general prediction such as that from the Markov expert. We therefore define a *listen threshold*, $T$, to determine the reliability of an align expert. An align expert is considered reliable if the length of its encoding of the last $w$ symbols is smaller than $C_{mk} - T$ bits where $C_{mk}$ is the length of encoding by the Markov expert. $T$ is a parameter of the algorithm.

An align expert is expected to be involved in prediction on a conserved region. Beyond the region, its prediction becomes random and therefore its performance gets worse. If the align expert performance falls below the threshold, the expert is discarded to make way for others.

### 2.5 Identifying Similar Regions

The main idea for our algorithm is that if two sequences are related, one would tell something new and useful about the other, that would not be known otherwise. If a region $R_x$ in sequence $X$ has some biological relationship with some region $R_y$ in sequence $Y$, the similarity between $R_x$ and $R_y$ should be better than random. In other words, the align expert based on $R_y$ should perform better on $R_x$ than the Markov experts whose prediction bases purely on the statistics of sequence $X$.

As the background sequence $Y$ could be very long and there could be moderately long matches just by chance. If an align expert were based on a random match, it would predict well during the match; but subsequent predictions would be bad. If the expert is consulted, the conditional information content would not be lower than the information content produced by the Markov experts. We therefore, consider a region conserved if there is an align expert whose predictions during its lifetime, if combined with the Markov experts predictions, would result in a lower conditional information content. The amount of shared information, measured in bits, indicates the similarity of the two regions. The more information shared, the more similar they are.

With reference to the traditional dynamic programming matrix, an align expert proceeds diagonally and thus could only find gap-free similar regions. However, there can be more than one align expert employed at any time. If there are gaps in the conserved regions, some neighbouring experts would be proposed. Therefore, the XMAligner can handle gaps implicitly without any assumptions about gap scores.

## 3 Experiment Results

We applied the XMAligner to study the genomes of four *Plasmodium* species, namely *P. falciparum, P. knowlesi, P. vivax* and *P. yoelii*. The genomes are obtained from PlasmoDB release 5.4 (*http://www.plasmodb.org/common/downloads/release-5.4/*). Of the four *Plasmodium* species, *P. falciparum* and *P. vivax* are malaria parasites on human while *P. knowlesi* and *P. yoelii* cause malaria in monkey and rodent respectively. The nucleotide compositions in these species' genomes are very different. The A+T content in the genome of *P. falciparum* is as high as 80% and the in coding regions is 76.22% while the A+T content in the *P. vivax* genome and *P. vivax* coding regions is 57.71% and 53.70% respectively. The characteristics of these genomes are presented in table 1. The four genomes have been annotated for genes but only the genomes of *P. falciparum* and *P. knowlesi* have been assembled.

It is normally hard to compare the quality of alignment because we do not know what should be matched and what should not be matched. We can assume that coding areas are more likely to conserved so they are expected to be matched. We therefore evaluated the performance of the XMAligner by comparing the regions detected by XMAligner to the exons annotated in the PlasmoDB. We compare the accuracy of alignment at nucleotide and exon levels.

At the nucleotide level, we define true positives ($TP$) as the number of coding nucleotides that are correctly predicted as coding, true negatives ($TN$) as the number of non-coding nucleotides that are correctly predicted as non-coding, false positives ($FP$) as the number of non-coding DNA what are predicted as coding and false negatives ($NF$) as the number of coding nucleotides that are incorrectly predicted as non-coding. At this level, we evaluated the alignment with the sensitivity ($Sn_n$) and specificity ($Sp_n$) measures as defined by [26]:

$$Sn_n = \frac{TP}{TP + FN} \tag{8}$$

| Species | Host | Genome Size (Mb) | %(AT) in Genome AT | %(AT) in CDS |
|---|---|---|---|---|
| *P.falciparum* | Human | 23.2 | 80.63% | 76.22% |
| *P.vivax* | Human | 26.9 | 57.71% | 53.70% |
| *P.knowlesi* | Monkey | 23.4 | 60.79% | 69.77% |
| *P.yoelii* | Rodent | 20.1 | 77.36% | 75.22% |

Table 1: *Plasmodium* genomes characteristics.

| Algorithm | Measurement | Pf/Pk | Pf/Pv | Pf/Pk | Pf/ALL | Pk/Pf | Pk/Pv | Pk/Py | Pk/ALL |
|---|---|---|---|---|---|---|---|---|---|
| nucmer | $Sn_n$ | 0.36 | 0.24 | 0.77 | 0.97 | 0.26 | 9.66 | 0.18 | 9.71 |
| | $Sp_n$ | 47.26 | 41.43 | 64.29 | 58.55 | 30.51 | 73.13 | 31.60 | 72.41 |
| | $Sn_e$ | 2.03 | 1.91 | 5.76 | 6.78 | 1.43 | 32.72 | 1.26 | 32.92 |
| | $Sp_e$ | 49.06 | 45.05 | 69.31 | 64.02 | 33.85 | 75.07 | 36.64 | 74.17 |
| promer | $Sn_n$ | 14.07 | 13.56 | 14.25 | 11.06 | 15.24 | 54.47 | 13.73 | 52.86 |
| | $Sp_n$ | 75.80 | 76.41 | 75.88 | 73.18 | 71.72 | 65.72 | 72.09 | 65.93 |
| | $Sn_e$ | 40.67 | 39.34 | 40.72 | 37.13 | 40.29 | 85.51 | 36.92 | 84.49 |
| | $Sp_e$ | 76.83 | 77.18 | 77.13 | 73.04 | 73.01 | 69.00 | 73.73 | 69.21 |
| XMAligner | $Sn_n$ | 42.61 | 39.84 | 51.43 | 52.08 | 45.82 | 90.12 | 43.13 | 89.17 |
| | $Sp_n$ | 89.85 | 91.43 | 88.68 | 89.35 | 80.26 | 64.64 | 83.92 | 65.87 |
| | $Sn_e$ | 69.95 | 67.64 | 76.66 | 76.62 | 70.25 | 95.57 | 65.32 | 95.90 |
| | $Sp_e$ | 87.52 | 88.08 | 90.01 | 90.49 | 48.82 | 72.19 | 57.29 | 70.43 |

Table 2: Sensitivity and Specificity of coding regions detection by promer, nucmer and XMAligner (in percentage)

$$Sp_n = \frac{TP}{TP + FP} \tag{9}$$

It is worth noting that the specificity defined in equation 9 is not the traditional specificity which is defined as

$$Sp = \frac{TN}{TN + FP} \tag{10}$$

As argued by [26], because the frequency of non-coding nucleotides tends to be much higher than that of coding nucleotides, $TN$ is much larger than $FP$ and thus the traditional specificity in equation 10 produces very large non-informative values. Therefore, the specificity in equation 9, which is usually referred to as precision in statistics, is more suitable in evaluation of gene finding programs.

At the exon level, we measured the proportion of exons detected by the alignment programs. As a alignment program can find only similar regions, and similar regions may not make up the whole exon, we assume that an exon that is correctly detected if some parts of the exon are detected by the alignment program. Likewise, a detected region is correctly predicted if it overlaps with some exon. As in [26], we define the sensitivity as the proportion of actual exons that are correctly detected and the specificity as the proportion of detected regions that are in some coding area. Formally they are:

$$Sn_e = \frac{Number\ of\ Detected\ Exons}{Number\ of\ Actual\ Exons} \tag{11}$$

$$Sp_e = \frac{Number\ of\ Correct\ Regions}{Number\ of\ Regions} \tag{12}$$

We aligned each of the assembled genomes, *P. falciparum* and *P. knowlesi* against each of three other genomes and against the concatenation of the other three genomes. The similar regions detected during alignment were compared with the latest annotation (version 5.4) from PlasmoDB. We compared our results with that from MUMmer [15], one of the best performance genome alignment programs in the literature. The MUMmer package provides two programs, namely *nucmer* which attempts to align the sequences at the nucleotide level, and *promer* which translates two sequences to protein and aligns at the protein level. The promer is generally used when the sequences are relatively divergent which the nucmer is unable to handle. We compared our results with both programs.

The alignment of these genomes was carried out on a desktop with duo core 2.66Ghz CPU and 3GB of memory. We used a hash table of hash key 20 for proposing experts in XMAligner, and used the default parameters for nucmer and promer. The alignment of one genome against another genome by XMAligner took about 40 minutes, and that of one genome against the other three took about 1 hour. The running time of promer was shorter, about 4 to 5 minutes for alignment one genome against another, and 20 minutes to align one genome to the three other genomes. Nucmer is even faster, it needed only 1 minute for pairwise alignment and 4 minutes for aligning one against three genomes.

The sensitivity and specificity of the three programs are shown in table 2. A column with the header X/Y shows the performance of the three algorithms when aligning the genome of X against the genome of Y and a column with header X/ALL shows that of aligning the genome X against the other three genomes. The performance of each program is shown in four rows for sensitivity at nucleotide level ($Sn_n$), specificity at nucleotide level ($Sp_n$), sensitivity at exon level ($Sn_e$) and specificity at exon level ($Sp_n$). All values are in percentage.

Among the three programs, the XMAligner is closest to the annotations from PlasmoDB we obtained. Its sensitivity and specificity at both nucleotide and exon levels are much higher than that from nucmer and promer, except that promer produced higher specificity at the exon level when aligning the *P. knowlesi* genome against the *P. falciparum* genome and *P. yoelii* at the expense of much lower sensitivity.

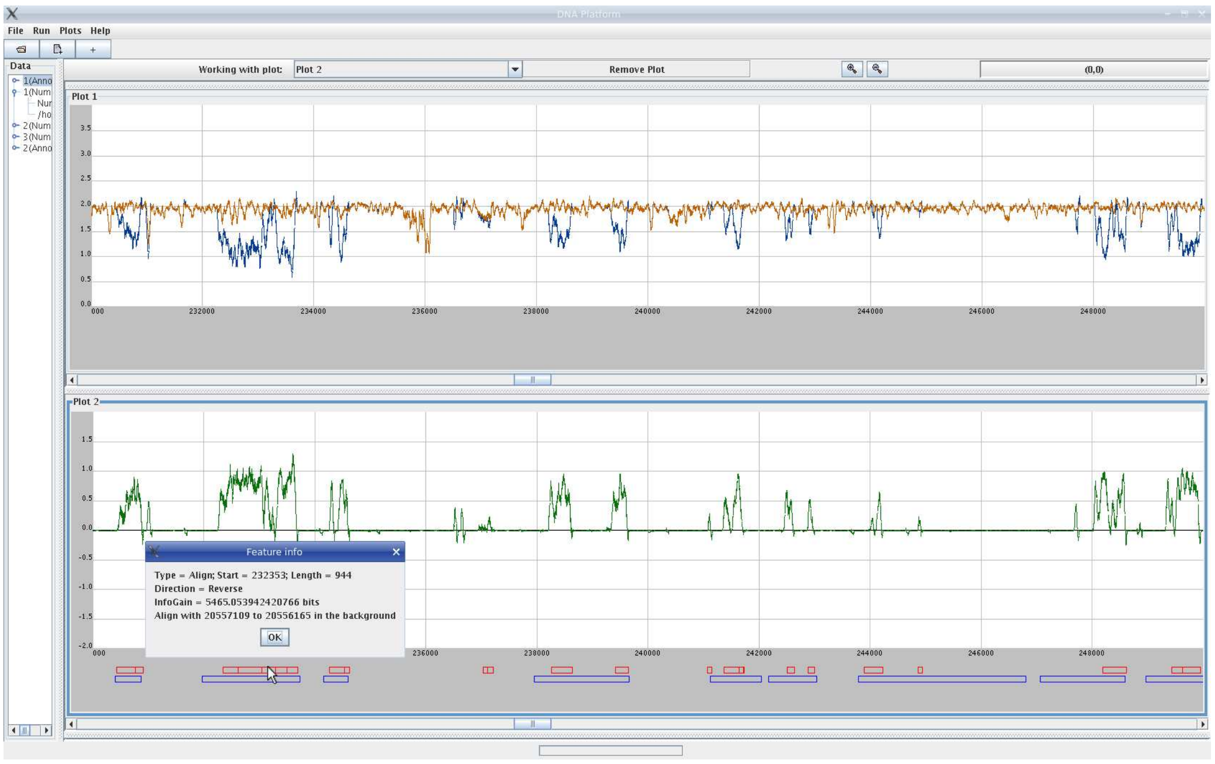
The performance of nucmer was very poor with the sensitivities are less than 1% at the nucleotide level, and less than 7% at the exon level in most alignments. The results suggest that the dynamic programming approach is unable to deal with comparing relatively distant sequences. The promer program aligns sequences at protein level but is only able to achieve around 15% sensitivity at the nucleotide level except for the alignment of the *P. knowlesi* genome against the *P. vivax* and against the other three genomes.

The output of the XMAligner can be integrated into the DNAPlatform [27] for visualisation. We applied the XMAligner to perform alignment contig ctg6843 of the *P. vivax* genome against the *P. falciparum* genome, and compare the alignment with the PlasmoDB annotation in the DNAPlatform viewer. Contig ctg6843 is 589976 bp long. Compressing the sequence by the Markov experts yields 1.91 bits per symbol or the total information content of the sequence is 1126854.16 bits. Compressing the sequence on the background of the *P. falciparum* genome produces 1.85 bits per symbol or 1091455.60 bits. In other words, the share information of the contig ctg6843 given the genome of *P. falciparum* is $1126854.16 - 1091455.60 = 35398.56$ bits. The resulting information content and the similar regions can be viewed from the viewer. Figure 3 shows the viewing of the alignment. Both the information content of contig ctg6843 and the conditional content of that given the genome of *P. falciparum* are displayed in the top plot. The bottom plots shows the shared information of the two sequences. Users can zoom in and zoom out to view particular areas of interest. The viewer tool is also able to read and display annotations. Two lines of boxes near the bottom of the viewer present the annotations from PlasmoDB and that produced by the XMAligner.

During our experiment, we noticed an area in contig ctg6843 very similar to a coding region in the genome of *P. falciparum*, but it is not yet annotated. The region starts at position 491038 and is about 15000 bp long and the counterpart from the *P. falciparum* genome starts at position 6971447. We tracked down and found that the region in the *P. falciparum* genome is a cluster of three genes *MAL7P1.203*, *MAL7P1.320* and *MAL7P1.204*. Figure 3 shows the region in contig ctg6843 in the viewer. The region is thought to be a synteny region conserved across malaria species, and contain some genes [28].

## 4 Conclusions

We have presented XMAligner, a genome local alignment algorithm, based on information theory. Unlike traditional alignment algorithms which depend on dynamic programming, our algorithm finds the similarities if there is some shared information between two regions. It therefore is able to align sequences relatively distant and with different compositions. We have shown that the XMAligner is able to align genome sequences with different nucleotide compositions where a traditional alignment tool is



Type = Align; Start = 232353; Length = 944
Direction = Reverse
InfoGain = 5465.053942420766 bits
Align with 20557109 to 20556165 in the background

**Figure 1: Visualisation of aligning contig ctg6843 from the *P.vivax* genome against the *P. falciparum* genome.**
Figure 3 shows the view from position 230000 to 250000 of contig ctg6843 from the P.vivax genome. In the top plot, the purple graph represents the information content of contig ctg6843 and the blue graph represents the conditional information content of the contig given the *P. falciparum* genome. In the bottom plot, the green graph shows the mutual information content of the two sequences. The blue boxes are the exons annotated obtained from PlasmoDB and red boxes are the similar regions detected by XMAligner. The dialogue box shows the properties of a region when it is clicked.

unable to perform. The output from the XMAligner can be integrated into a visualisation tool to aid the analysis of sequences.

## Acknowledgements

The authors would like to thank Robert Huestis for drawing our attention to malaria and for many insightful discussions on the genomes of *Plasmodium*.

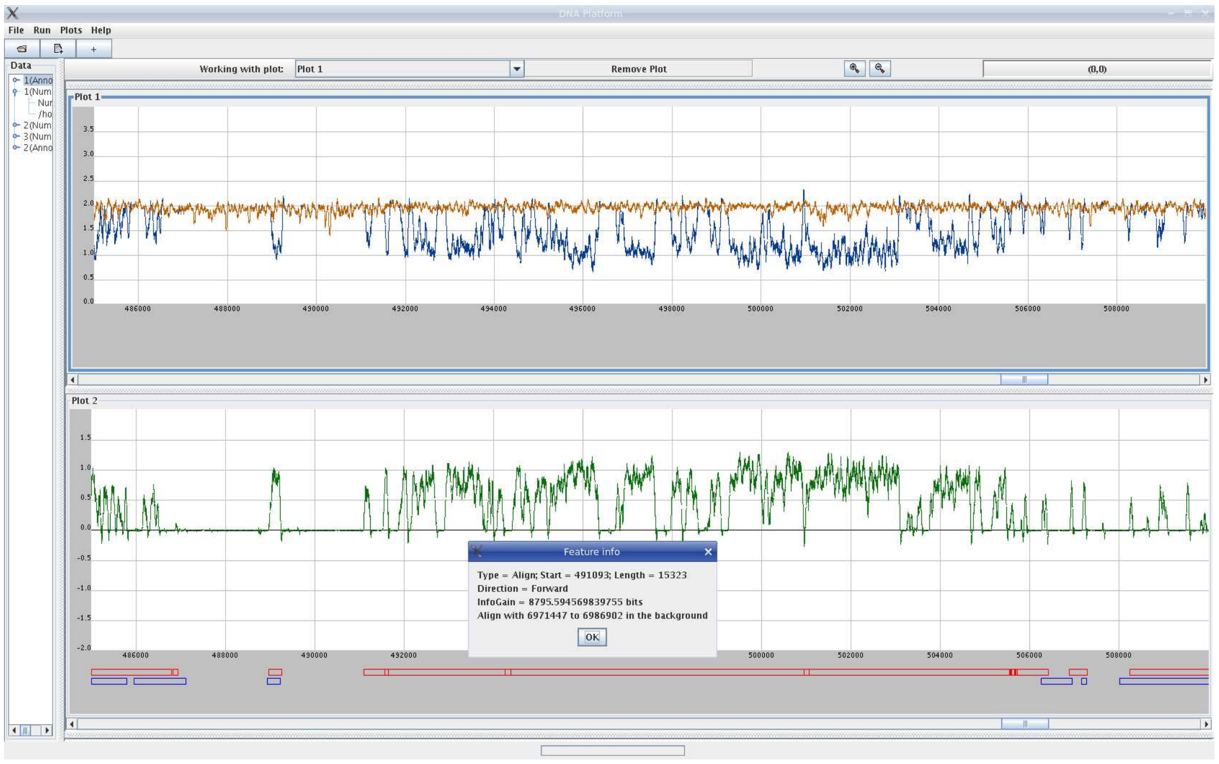
## References

1. Needleman SB, Wunsch CD: **A General Method Applicable to the Search for Similarities in the Amino Acid Sequences of Two Proteins**. *Journal of Molecular Biology* 1970, **48**:443–453.

2. Smith TF, Waterman MS: **Identification of Common Molecular Subsequences**. *Journal of Molecular Biology* 1981, **147**:195–147.

3. Comeron JM, Aguade M: **An Evaluation of Measures of Synonymous Codon Usage Bias**. *Journal of Molecular Evolution* 1998, **47**(3):268–274.

4. Lander ES, Linton LM, Birren B, Nusbaum C, Zody MC, Baldwin J, Devon K: **Initial Sequencing and Analysis of the Human Genome**. *Nature* 2001, **409**:860–921.



Type = Align; Start = 491093; Length = 15323
Direction = Forward
InfoGain = 8795.594569839755 bits
Align with 6971447 to 6986902 in the background

**Figure 2: A long similar region detected by XMAligner but not annotated**
XMAligner finds a region in contig ctg6843 of the *P. vivax* genome, starting from position 491038 which is similar to a region on the P. falciparum starting from position 6971447. The region is more than 15000 base long and the similarity is 8795 bits.

5. Cao MD, Dix TI, Allison L, Mears C: **A Simple Statistical Algorithm for Biological Sequence Compression**. *Data Compression Conference* 2007, :43–52.

6. Pearson WR, Lipman DJ: **Improved tools for biological sequence comparison.** *Proc Natl Acad Sci* 1988, **85**(8):2444–2448.

7. Altschul SF, Gish W, Miller W, Myers E, Lipman D: **Basic local alignment search tool.** *J Mol Biol* 1990, **215**:403–410.

8. Fleischmann R, Adams M, White O, Clayton R, Kirkness E, Kerlavage A, Bult C, Tomb J, Dougherty B, Merrick J, al e: **Whole-genome random sequencing and assembly of Haemophilus influenzae Rd**. *Science* 1995, **269**(5223):496–512.

9. Ning Z, Cox AJ, Mullikin JC: **SSAHA: A Fast Search Method for Large DNA Databases**. *Genome Res.* 2001, **11**(10):1725–1729.

10. Altschul SF, Madden T, Schaffer A, Zhang J, Zhang Z, Miller W, Lipman D: **Gapped BLAST and PSI-BLAST: a new generation of protein database search programs**. *Nucl. Acids Res.* 1997, **25**(17):3389–3402.

11. Schwartz S, Kent WJ, Smit A, Zhang Z, Baertsch R, Hardison RC, Haussler D, Miller W: **Human-Mouse Alignments with BLASTZ**. *Genome Res.* 2003, **13**:103–107.

12. Kent WJ: **BLAT - The BLAST-Like Alignment Tool**. *Genome Res.* 2002, **12**(4):656–664.

13. Ukkonen E: **On-Line Construction of Suffix Trees**. *Algorithmica* 1995, **14**(3):249–260.

14. Delcher AL, Phillippy A, Carlton J, Salzberg SL: **Fast Algorithms for Large-scale Genome Alignment and Comparison**. *Nucl. Acids Res.* 2002, **30**(11):2478–2483.

15. Kurtz S, Phillippy A, Delcher AL, Smoot M, Shumway M, Antonescu C, Salzberg S: **Versatile and open software for comparing large genomes**. *Genome Biology* 2004, **5**(2).

16. Bray N, Dubchak I, Pachter L: **AVID: A Global Alignment Program**. *Genome Research* 2003, **13**:97–102.

17. Höhl M, Kurtz S, Ohlebusch E: **Efficient Multiple Genome Alignment**. *Bioinformatics* 2002, **18**(Suppl. 1):S312–S320.

18. Powell DR, Allison L, Dix TI: **Modelling-Alignment for Non-random Sequences.** In *AI 2004: Advances in Artificial Intelligence* 2004:203–214.

19. Dayhoff MO, Schwartz RM, Orcutt BC: *A model for evolutionary change in proteins*, Atlas of Protein Sequence and Structure 1978. **5**:345–352.

20. Henikoff S, Henikoff JG: **Amino acid substitution matrices from protein blocks.** *Proc Natl Acad Sci* 1992, **89**(22):10915–10919.

21. Shannon CE: **A Mathematical Theory of Communication**. *The Bell System Technical Journal* 1948, **27**:379–423.

22. Dix T, Powell D, Allison L, Bernal J, Jaeger S, Stern L: **Comparative analysis of long DNA sequences by per element information content using different contexts**. *BMC Bioinformatics* 2007, **8**(Suppl 2):S10.

23. Carlton J, Silva J, Hall N: **The Genome of Model Malaria Parasites and Comparative Genomics**. *Curr. Issues Mol. Biol.* 2005, **7**:23–38.

24. Boulton DM, Wallace CS: **The Information Content of a Multistate Distribution**. *Journal of Theoretical Biology* 1969, **23**(2):269–278.

25. Mori Y: **A lightweight suffix-sorting library** 2008, [http://homepage3.nifty.com/wpage/software/libdivsufsort.html].

26. Burset M, Guigó R: **Evaluation of Gene Structure Prediction Programs**. *Genomics* 1996, **34**(3):353–367.

27. Dix TI, Powell DR, Allison L, Jaeger S, Bernal J, Stern L: **Exploring long DNA sequences by information content**. *Probabilistic Modeling and Machine Learning in Structural and Systems Biology, Workshop Proc* 2006, :97–102.

28. Huestis R: **Personal Communication**, 2008.