Refining Probability Motifs for the Discovery of Existing Patterns of DNA Bachelor Project

Susan Laraghy 0584622, Leiden University
Supervisors: Hendrik-Jan Hoogeboom and Walter Kosters (LIACS),
Kai Ye (Bioinformatics)

June 20, 2008

Abstract

The aim of this project was to build a probability motif refining program. In the past this process has been both too computationally demanding and time consuming to be a feasible tool in the world of Bioinformatics. The notion is to take a file of DNA sequences and containing hidden motifs and apply a set of given position specific weight matrices to these sequences in order to discover the instances that resemble the motif sequences. Based on these found instances, the position weight matrices can therefore be adjusted and the process iterated. Various approaches were undertaken in an attempt to find the most efficient method.

1 Introduction

Existing sequence patterns of DNA are called "motifs" and have a certain biological significance. These motifs may occur in various positions, whether it be within a sort of genome, or genomes of various sorts but similar genes. By far the most common representation of DNA motifs is the Position Weight Matrix (PWM), also known as a Position-Specific Scoring Matrix (PSSM). These are frequency matrices built upon occurrences of resembling motifs within a sequence. For each position of the motif the matrix contains the probability of each of the nucleotides A, C, G and T occurring at that position. Using the PWM, the genome can be searched for occurrences of the motif. Based upon found instances that resemble the given motifs above a

certain threshold, these position weight matrices can be adjusted. This process can then be iterated to optimise the Position Weight Matrices.

2 Background

Deoxyribonucleic acid (DNA) is a molecule that contains the genetic instructions used in the development and functioning of all known living organisms and some viruses. It is composed from four nucleotides (sometimes referred to as bases), i.e., adenine, guanine, cytosine and thymine. The four nucleotides are given one letter abbreviations as shorthand for the four bases, A, C, G, and T respectively.
DNA is normally double stranded which is simply two chains of single-stranded DNA, positioned so their "bases" can interact with each other forming a helical spiral (see Figure 1). The nucleotides "pair up" with bases on the opposite strand, so that a type 'A' nucleotide is always opposite a type 'T', and 'G' is opposite 'C'. The attraction between the paired nucleotides is fairly weak, but when there is a whole string of them, it adds up to enough strength to hold the strands together. Importantly, the two strands travel in opposite directions; hence the structure is said to be "anti-parallel".

In double-stranded DNA, only one strand codes for the RNA that is translated into protein. This DNA strand is referred to as the antisense strand [1]. The strand that does not code for RNA is called the sense strand. Another way of defining antisense DNA is that it is the strand of DNA that carries the information necessary to make proteins by binding to a corresponding messenger RNA. Although these strands are exact mirror images of one another, only the antisense strand contains the information for making proteins. The sense strand does not [2].

DNA motifs may also express ambiguity of the nucleotides at any given position within the motif [3]. By extending the DNA alphabet from 4 letters to 15 letters (Table 1), motifs such as KGTTGCTWRGCAACM can be expressed. A graphical method exists for displaying such patterns as seen in Figure 2. The characters representing the sequence are stacked on top of each other for each position in the motif. The height of each letter is made proportional to its frequency, and the letters are sorted so the most common one is on top. These graphical representations also contain more information relevant to the motif [4], i.e., the relative probabilities. Therefore it can be

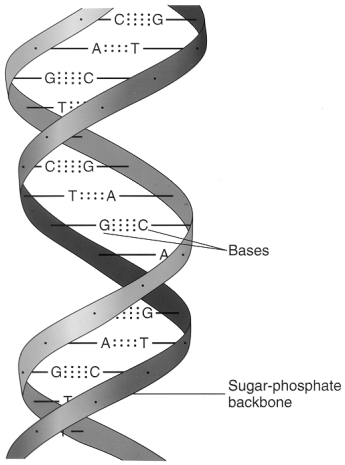

Figure 1: Double stranded DNA helix.

Symbol	**Nucleotides**
R	G or A
W	A or T
Y	C or T
M	A or C
K	G or T
S	G or C
H	Not G (A or C or T)
B	Not A (C or G or T)
V	Not T (A or C or G)
D	Not C (A or G or T)
N	Any nucleotide (A, T, C or G)

Table 1: Nucleotides Ambiguity Codes

Figure 2: Sequence logo for the motif KGTTGCTWRGCAACM

seen in this figure that in the first position there mostly exists a chance of a G but also a T now expressed as a K.

3 Theory

In this section, *Position Weight Matrices* are introduced and explained, as well as the method of generating them and calculating a score for a given string based upon the PWM.

A *Position Weight Matrix* (hereafter referred to as PWM) is a motif descriptor. It attempts to capture the intrinsic variability characteristic of sequential patterns [5]. As described in detail by Stormo [6], in a PWM, the motif is of fixed size. It is generated by recording the occurrences of each given nucleotide at each given position given a set of strings (see Figure 3). This illustrates the generation of a PWM with absolute values, whereby from a set S of n aligned sequences of length $\ell, s_1, \ldots, s_n$, where $s_k = s_{k_1} \ldots s_{k_l}$ (the s_{k_ℓ} being one of A, C, G, T in the case of DNA sequences) a $4 \times \ell$ Position Weight Matrix, M is defined as

$$M_{ij} = \sum_{k=1}^{n} I(i, s_{k_j}) \qquad \begin{array}{l} i = A, C, G, T \\ j = 1, \ldots, \ell \end{array}$$

where $I(i,q) = \begin{cases} 1 & \text{if } i = q \\ 0 & \text{otherwise} \end{cases}$

```
   G  A  G  G  T  A  A  A  C
   T  C  C  G  T  A  A  G  T
   C  A  G  G  T  T  G  G  A
   A  C  A  G  T  C  A  G  T
   T  A  G  G  T  C  A  T  T
   T  A  G  G  T  A  C  T  G
   A  T  G  G  T  A  A  C  T
   C  A  G  G  T  A  T  A  C
   T  G  T  G  T  G  A  G  T
   A  A  G  G  T  A  A  G  T

--|---------------------------
A |  3  6  1  0  0  6  7  2  1
C |  2  2  1  0  0  2  1  1  2
G |  1  1  7 10  0  1  1  5  1
T |  4  1  1  0 10  1  1  2  6
```

Figure 3: Manner in which a PWM is generated using absolute frequencies.

Position Weight Matrices however are normally expressed as relative frequencies whereby the probabilities for each position total one (see Figure 4). Here the matrix is now transposed such that the columns now represent the nucleotides A, C, G, T respectively, and each position in the motif (a row in the matrix) has four associated probabilities: the probability of an A, G, C and T at that position. The positions are assumed to be independent, and therefore a score can be calculated for a particular string simply by multiplying the probabilities of each nucleotide in the string at each position. So given a sequence of length ℓ, the product of the coefficients from such a matrix corresponding to each nucleotide in each position of the sequence is the probability the string is "generated" by the matrix — a measure that indicates the similarity of the string and the motif represented by the PWM. For example, using the Position Weight Matrix in Figure 4, the probability of finding the sequence GGATGCTGAGCTAGT would be $0.407 \times 0.976 \times 0.006 \times 0.981 \times 0.897 \times 0.982 \times 0.703 \times 0.015 \times 0.506 \times 0.982 \times 0.792 \times 0.010 \times 0.954 \times 0.006 \times 0.190 \approx 9.30^{-8}$

Typically however, the coefficients in a Position Weight Matrix are directly computed as log-likelihood values (i.e., the logarithm base 10 of the original frequencies) Then, given a sequence of length ℓ the log-likelihood ratio can be computed by summing the coefficients of the log-likelihood matrix corresponding to each nucleotide in each position on the sequence. Therefore,

```
>LM3 15 KGTTGCTWRGCAACM

  A      C      G      T
0.169  0.033  0.407  0.392  K
0.009  0.006  0.976  0.009  G
0.006  0.022  0.007  0.965  T
0.007  0.006  0.006  0.981  T
0.023  0.018  0.897  0.062  G
0.006  0.982  0.006  0.006  C
0.012  0.279  0.006  0.703  T
0.566  0.006  0.015  0.413  W
0.506  0.006  0.466  0.022  R
0.006  0.006  0.982  0.006  G
0.119  0.792  0.030  0.059  C
0.975  0.006  0.009  0.010  A
0.954  0.010  0.030  0.006  A
0.012  0.974  0.006  0.008  C
0.339  0.449  0.023  0.190  M
```

Figure 4: PWM of motif KGTTGCTWRGCAACM of length 15 where each column represents the probability of the nucleotide A, C, G, T respectively, and each row represents a position in the motif.

the score m_s of a matrix M for a given string $s = s_1 \dots s_\ell$ of length ℓ, and s_k being one of A, C, G, T, can formally be expressed as follows:

$$m_s = \sum_{j=1}^{\ell} M_{s_j j}$$

4 Approach

Given a file of DNA sequences containing hidden motifs, and furthermore a set of Position Weight Matrices, for each string s of length ℓ contained within the DNA sequences whereby ℓ is the maximum motif length of the given PWMs, a score is calculated for each given PWM. On the basis of this, the strings that resemble the motif sequences corresponding to the PWMs above a certain threshold are discovered. The string or instance is then assigned to the most appropriate PWM. On the basis of this new found information the PWMs are then adjusted to include these new found instances, and the process is then reiterated.

Three separate but related approaches were taken to accomplish this. In the next section we discuss the standard approach. The two other approaches ('centred-PWM' and 'rank-based selection') differ in details and are presented in subsequent sections.

4.1 Standard Approach

The basic algorithm for this procedure is:

Input: set of PWMs and DNA sequences
Output: set of adjusted PWMs
for $i \leftarrow 1$ **to** n **do**
  foreach *position within the DNA sequence* **do**
    score each PWM;
    normalise the scores;
    select the PWM for assignment via roulette wheel selection;
  end
  remove overlapping instances;
  update the PWMs using new information;
  examine for possible shifting;
end

Pre-processing the DNA Sequences The PROSITE notation used in DNA sequences uses the one-letter code (A, C, G, T for each of the nucleotides) and a concatenation symbol, '-', is used between pattern elements, but it is often dropped between letters of the pattern alphabet. Also the lower case letter 'x' can be used as a pattern element to denote any amino acid. Therefore upon commencement, a sequence of DNA is processed to replace all non-nucleotides ('-' and 'x') with the generic N (representing aNy). During processing, the total number of occurrences of each nucleotide are recorded, and based on this information, the *null* or *background* motif is generated. This motif is added to the PWMs already given and subsequently used when classifying to which PWM a string is assigned. Based on a relatively even probability, any given instance within the sequence will default most often to this motif. The null motif contains the background frequency which is the overall probability for each nucleotide calculated as the individual count for each A, C, G, and T, divided by the length of the sequence.

Specification of the PWMs Furthermore the PWM's are stored as the log_{10} likelihood ratio. This is for two important reasons. This is not only more precise owing to the often minuscule probabilities and precision within the program, but more so for the efficiency of the calculations as addition is a far more efficient calculation than product. The PWMs are varying in length and so the maximum motif length is recorded for future calculations. There is also an option of starting with motifs such as GGATGCTGAGCTAGT instead of a set of PWMs, and building the relative PWMs from these motifs. Here only the nucleotides may be used in defining the motif and not the nucleotide ambiguity codes (such as K, W, R etc). Each of the nucleotides in the motif receives a probability of 0.97 at that position, whereas 0.01 is allocated to each of the other non-occuring nucleotides for each position within the matrix.

Scoring the PWM Now that the maximum motif length n has been determined, every substring s of length n of the DNA sequences are used to score the PWMs. Here the evaluation of a single instances s is considered. As the motifs may have instances in both sense and anti-sense strands, the sequence is traversed in both directions. The instances of non-nucleotides have no significance for the refinement process of PWMs, and so strings that contain an occurrence of an N are ignored. For the rest of the sequence, at each position the score is calculated for each of the PWMs. To allow for the difference of motif lengths of PWMs when scoring, initially the score was divided by the motif length of the PWM to give an average score for the PWM. However this refinement process is dealing with probabilities, shorter motifs are easier to match than their longer counterparts and therefore have a higher probability. Thus this method was not found to be impartial and afforded preference to the shorter motifs. To avoid this the score for each PWM was supplemented by adding the log_{10} likelihood values of the background motif for the length of the maximum motif length. Furthermore as there are no non-nucleotides to consider, the score m_s for a sequence $s_1 s_2 \dots s_n$ is now calculated as follows:

$$m_s = \sum_{j=1}^{\ell} M_{s_j j} + \sum_{j=\ell+1}^{n} M^0_{s_j j}$$

where n refers to the maximum motif length, and M^0 to the null or background motif.

Normalising the Scores Many of these scores are minuscule and are therefore considered to be inconsequential. A minimum threshold is introduced whereby values for PWMs under this threshold default to a score of zero. This threshold is currently set at 10^{-14} (which is based on the average motif length of the given set of PWMs i.e., for the given set used in implementation, the average motif length was 18), however the setting of this variable remains an area of experimentation to discover any possible impact. The remaining scores of candidate PWMs are then normalised using the following formula:

$$N = \frac{v - v^{min}}{v^{max} - v^{min}}$$

where N is the normalised value, v corresponds to the score, and v^{min} and v^{max} correspond the the minimum and maximum scores of all the given PWMs respectively.

Assigning Instances to a PWM A technique known as the roulette-wheel selection is then used to determine to which PWM to assign this instance. A random number between zero and one is generated, and the instance is then assigned to the PWM with the score that corresponds to this random number. This method guaranties that more likely PWMs have more chance of being chosen.

The general idea of this approach is to calculate the probability based on the information from the instances peer positions but not the position itself. Therefore during the calculation of the score of position i to PWM j, if position i already contributes to the PWM j, its contribution is removed from the PWM for this particular calculation. Otherwise the position is recorded, and the number of found instances for that particular PWM is incremented.

As the original sequence is traversed in two directions, a record of assigned PWMs equal to twice the length of the original sequence now exists. However within that sequence overlapping instances may occur and interfere with each other (see Figure 5). These may also be instances of the same motif. For this reason the assigned PWMs (excluding the null motif) are examined to detect overlaps within the motif length of the given PWM. Once detected, one is chosen for removal by the same method as in selections. The scores for theses two assigned PWMs are once again normalised, a random number

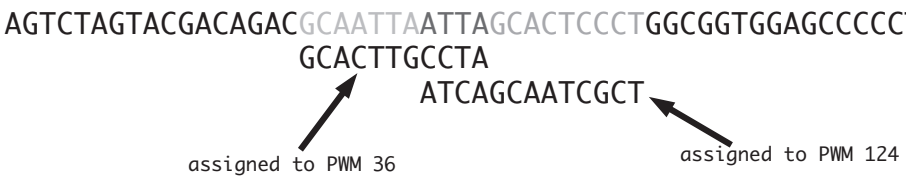

Figure 5: The existence of overlapping instances.

selected and the corresponding PWM is then removed. This process is then iterated for the length of the string to remove any further overlaps.

Updating the PWMs The PWMs are updated at the end of each scan of the sequence. In practice, prior information is included as such:

$$updating\ rate = \frac{\begin{array}{c}(the\ number\ of\ found\ instances\ of\ the\ current\ PWM \\ +\ prior\ pseudo\ counts\ set)\end{array}}{\begin{array}{c}(the\ total\ number\ of\ instances\ in\ the\ sequence \\ +\ the\ total\ number\ of\ pseudo\ counts\ per\ motif)\end{array}}$$

To accomplish this a separate set of matrices (which are used to record the found instances) are initialised with a pseudo count of twenty per motif. The pseudo count may be initialised as any given number, but from previous experimentation this gives a reasonable result. The pseudo counts are then further distributed across the nucleotides using the probability in the original PWM as such:

$$S_{ij} = \lfloor p * 10^{M_{ij}} + 0.5 \rfloor$$

whereby S is the set of scoring matrix, and p refers to the pseudo count. To prevent null values, positions in the scoring matrix with a value of zero are incremented to one, and the nucleotide with the maximum value on that position is decremented.

As instances are assigned to each PWM, the relevant values in this separate set of scoring matrices are incremented. During updating these scoring matrices (which were initialised with the prior information and therefore still contain such information) are then converted into the adjusted PWM by dividing the each value in the scoring array by the number of found instances

for that particular PWM.

Examining for Shifting The PWMs given by the user may capture just part of the motif and perhaps the whole PWM needs to be shifted one direction or another to include other nucleotides before or after the original PWM that may have a stronger presence in the genome. Therefore the next step is to examine each of the PWMs for possible shifting. Perhaps the original motif is not as strong as the position adjacent within the sequence and so the genome around the found instances for a PWM are examined to determine if there are more conserved positions to include. By comparing the entropy values on the left of all found instances of motifs combined with the entropy for the beginning of the motif, to the combined entropies of the end of the motif and those on the right of all found instances, the decision can be made to shift the PWM and to which direction. Entropy values for the beginning and end of the motif are calculated as such:

$$E = \sum_{j=1}^{4} \frac{S_{ij}}{I} * (log_{10} S_{ij} - log_{10} I) + (g * \frac{n}{I} * (-log_{10} I))$$

where E is the entropy value, S the scoring matrix, I the number of instances found for this particular PWM (including pseudo counts), n is the so called N value and g the gap penalty for the entropy. These last two values are currently set as 1, however this may be an area of further experimentation.

The left and right entropies are calculated based on all found instances in the genome of that particular PWM. The nucleotides to the left and right of all found instances for the PWM are recorded as set R and then an entropy value determined in using the almost the same formula as above but now using these new found nucleotide values:

$$E = \sum_{j=1}^{4} \frac{R_j}{I} * (log_{10} R_j - log_{10} I) + (g * \frac{n}{I} * (-log_{10} I))$$

where n (N value) now equals the pseudo count minus the number of nucleotides ($n = p - 4$).

If the combined entropies of the left and beginning are smaller than those of combined right and end, then the left is more conserved than the right, and

if the left entropy is also smaller than the end, then the PWM is shifted one position to the left and the information gained about the the nucleotides at these positions during the calculation of the entropy is then used to generate the prior information for that new position of the PWM as well as the new values within the PWM itself. If however the right and end entropies are smaller than those of the of the left and beginning, then the right is more conserved, and if the right is smaller than the start then using the same process the PWM is then shifted to the right.

This completes the scan and the new PWMs are then written to a log file. This whole process from examining each position within the sequence, scoring each PWM, normalising the scores, selecting the PWM for assignment via roulette wheel selection, removing overlapping instances, updating the PWMs with new information, and examining for possible shifting of the PWM is then iterated x times, determined by the user.

4.2 Centred PWM Approach

The second approach generally follows the first, however instead of only adjusting the PWM to the length of the motif, and then later examining the PWM for possible shifting, this approach immediately examines n characters either side of the PWM. To accomplish this, either side of the PWM is filled with the background/null motif so that the length examined is now 40 (see Figure 6 on page 13). In this way, a wider margin either side of the possible motif is immediately examined and updated, rather than examining only the one nucleotide either side as in the standard version. In the score m_s (see page 8) the positions $\ell + 1 \ldots n$ are also set to the background, however they are not adjusted after matching and the background information is purely used for scoring purposes only. Here in the centred approach, they are adjusted and this in turn removes the need to test the entropy values for shifting, and allows the ability to examine further than one position beyond the motif per scan of the database.
The rest of this approach follows the standard approach.

4.3 Rank-based Selection Approach

The third and final approach uses rank-based selection instead of the roulette-wheel selection method used in both the standard and the centred-PWM aproaches. In this approach the set of candidate PWMs (again above a

```
Original Motif: TTGACCTTTAAAGCW

row  1    0.302830   0.248525   0.228223   0.220422
row  2    0.302830   0.248525   0.228223   0.220422
row  3    0.302830   0.248525   0.228223   0.220422
row  4    0.302830   0.248525   0.228223   0.220422
row  5    0.302830   0.248525   0.228223   0.220422
row  6    0.302830   0.248525   0.228223   0.220422
row  7    0.302830   0.248525   0.228223   0.220422
row  8    0.302830   0.248525   0.228223   0.220422
row  9    0.302830   0.248525   0.228223   0.220422
row 10    0.302830   0.248525   0.228223   0.220422
row 11    0.302830   0.248525   0.228223   0.220422
row 12    0.302830   0.248525   0.228223   0.220422
row 13    0.071800   0.111400   0.081700   0.735100    T
row 14    0.022300   0.002500   0.012400   0.962900    T
row 15    0.012400   0.012400   0.953000   0.022300    G
row 16    0.923300   0.052000   0.012400   0.012400    A
row 17    0.042100   0.903500   0.012400   0.042100    C
row 18    0.042100   0.844100   0.022300   0.091600    C
row 19    0.002500   0.002500   0.002500   0.992600    T
row 20    0.012400   0.012400   0.012400   0.962900    T
row 21    0.052000   0.022300   0.012400   0.913400    T
row 22    0.913400   0.061900   0.012400   0.012400    A
row 23    0.982700   0.012400   0.002500   0.002500    A
row 24    0.972800   0.002500   0.022300   0.002500    A
row 25    0.061900   0.012400   0.863900   0.061900    G
row 26    0.240100   0.616300   0.091600   0.052000    C
row 27    0.240100   0.210400   0.081700   0.467800    W
row 28    0.302830   0.248525   0.228223   0.220422
row 29    0.302830   0.248525   0.228223   0.220422
row 30    0.302830   0.248525   0.228223   0.220422
row 31    0.302830   0.248525   0.228223   0.220422
row 32    0.302830   0.248525   0.228223   0.220422
row 33    0.302830   0.248525   0.228223   0.220422
row 34    0.302830   0.248525   0.228223   0.220422
row 35    0.302830   0.248525   0.228223   0.220422
row 36    0.302830   0.248525   0.228223   0.220422
row 37    0.302830   0.248525   0.228223   0.220422
row 38    0.302830   0.248525   0.228223   0.220422
row 39    0.302830   0.248525   0.228223   0.220422
row 40    0.302830   0.248525   0.228223   0.220422
```

Figure 6: PWM of motif TTGACCTTTAAAGCW of length 15 where the original PWM has been centred within the background motif

given threshold) is sorted according to their score for lowest to highest. A percentage is then assigned to each candidate, defined as follows:

$$p = \frac{2 * r}{n(n+1)}$$

whereby p is the percentage assigned to the PWM, r the rank or position in the sorted list of candidate PWMs, and n the total number of PWMs with a score higher than or equal to the minimum threshold.

The ability of recording prior found instances in the sequence greatly increases the complexity of this method as the list of all assigned PWMs as well as their percentages must be recorded for each position within the sequence. As the extra calculations needed per scan of the sequence, as well as the extra memory required would cause this method to be too slow to be a feasible tool, this approach was then further developed without the use of prior information except that of the pseudo counts for initiation purposes, and is therefore also based on the instances positions themselves and not on their peer positions. If a certain position recorded an instance with different PWMs than the previous scan, their previous effect was therefore not removed and the PWMs were updated with the new given information. Overlapping instances were also not removed for the same reason. The shifting was still measured and implemented, and the rest of this method follows the standard approach.

5 Implementation

The probability motif refining software program was implemented in C++, although not using object-oriented programing. For ease of traversal the PWMs were entered into the program and stored as a three-dimensional matrix. As speed of the finished product was an issue, various optimisations were undertaken. For efficiency the DNA sequence was entered line-by-line into a buffer to be processed (non-nucleotides replaced by an N etc as previously discussed in Section 4.1) before concatenating with the previous line to form an extensive string. This string is traversed in both directions as discussed earlier, and to prevent constant testing for the traversal direction during the course of the program and therefore impeding the speed and efficiency, two while loops were created within each scan (one for each direction) and two copies of the necessary functions were created each traversing the string in opposite directions and called upon according to the traversal

direction.

The scores per PWM, the number of recorded instances per PWM, and whether the PWM had been changed for each scan were all stored as simple arrays of length of the number of PWMs; the previously assigned PWM for each position as well as the assigned score (needed when removing the overlap) were stored as vectors, while the positions in the string of the instances found per PWM were stored as vectors of vectors. This allowed for growth and flexibility where necessary, without a waste of memory. As refining probability motifs is extremely calculation intensive, the program as such was also optimised for overall speed by removing as many tests as possible from the while loops whereby increasing the lines of code.
Threshold level, number of scans, as well as input, output, and log files are all variable via a user input menu.

6 Results

The following results are based on a given set of Positional Weight Matrices for 233 discovered motifs [8]. The motifs of these PWMs vary in length from between 12 and 25 nucleotides long with an average length of 18. The DNA sequences are from the human genome [9], and for testing purposes shorter sequences from the genome was used.

6.1 Comparison of Methods

As in the past refining probability has been both too computationally and time consuming to be a feasible tool in the world of Bioinformatics, speed and efficiency were important features to test amongst the three approaches (standard, centred, and rank-based). In Figure 7, the speed of three methods are compared over a variety of sizes of DNA sequences (10Kb, 100Kb, and 1Mb). Although the rank-based was faster overall, it misses the vital prior information on which this idea is based and therefore the results from this method cannot be deemed as reliable.
Of the two reliable methods (standard and centred) it can be clearly seen therefore that the standard was the most efficient.

6.2 Results of Standard approach

Using the standard approach, after 20 scans the original PWMs have been transformed such as can be seen in the PWM227 in Figure 8. Here the first

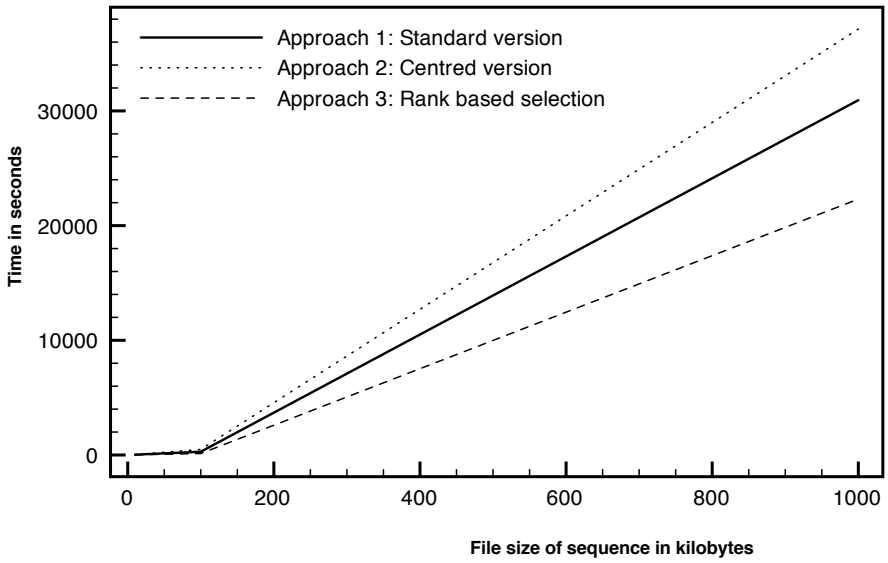

Figure 7: Comparison between methods based on speeds. Lower better.

A and the second last A (rows 2 and 19 respectively) still have very strong probabilities, and the C in row 11 may now be considered to be an M (A or a C) whereas for the most nucleotides within the motif, their probabilities have been somewhat diluted. This behaviour of mostly diluting the PWM was seen across the board amongst all PWMs. No shifting of the motifs was seen across any of the PWMs during any of the testing during this approach.

The process as a whole is very calculation intensive with 20 scans of a 1Mb DNA sequence taking approximately 8 hours. A previously developed program with a maximum allowance of processing 127 motifs (including the null motif) yielded 81 seconds over 20 scans of a 10Kb DNA sequence, however this version succeeds in 17.5 seconds with the same parameters, and completes the full 234 PWMs in just 30 seconds. This is due in part to the fact that the previous version was updating the PWMs after every found instance during the scan, instead of waiting to the end of the scan and updating all PWMs together. Other deciding facts were the implemented optimisations as previously discussed.
The program was further optimised by the utilisation of a hardware specific

0.0431	0.0431	0.8707	0.0431	G		0.163636	0.163636	0.454546	0.218182
0.8362	0.0776	0.0431	0.0431	A		0.718182	0.154546	0.036364	0.090909
0.8362	0.0776	0.0431	0.0431	A		0.545455	0.209091	0.063636	0.181818
0.0086	0.0086	0.0086	0.9741	T		0.154546	0.109091	0.181818	0.554546
0.0431	0.0776	0.0086	0.8707	T		0.081818	0.200000	0.190909	0.527273
0.0431	0.0431	0.0086	0.9052	T		0.145455	0.154546	0.109091	0.590909
0.7672	0.0086	0.0776	0.1466	A		0.454546	0.072727	0.281818	0.190909
0.0776	0.0431	0.8017	0.0776	G		0.145455	0.227273	0.527273	0.100000
0.0431	0.0086	0.0086	0.9397	T		0.272727	0.081818	0.172727	0.472727
0.0431	0.1121	0.8017	0.0431	G		0.227273	0.127273	0.363636	0.281818
0.0776	0.8362	0.0776	0.0086	C		0.345455	0.409091	0.118182	0.127273
0.0776	0.1121	0.0086	0.8017	T		0.200000	0.100000	0.163636	0.536364
0.0776	0.0431	0.1466	0.7328	T		0.136364	0.063636	0.290909	0.509091
0.2155	0.0431	0.6638	0.0776	G		0.218182	0.090909	0.436364	0.254546
0.0086	0.0086	0.1121	0.8707	T		0.190909	0.036364	0.236364	0.536364
0.0086	0.0776	0.8707	0.0431	G		0.036364	0.290909	0.563637	0.109091
0.9052	0.0431	0.0431	0.0086	A		0.563637	0.090909	0.145455	0.200000
0.9052	0.0086	0.0776	0.0086	A		0.563637	0.136364	0.163636	0.136364
0.8707	0.0431	0.0431	0.0431	A		0.854546	0.009091	0.072727	0.063636
0.6983	0.0431	0.2500	0.0086	A		0.518182	0.100000	0.290909	0.090909

Figure 8: The transformation of a PWM after twenty scans.

compiler (namely the Intel C++ Compiler for Mac OS X [7]). This succeeded in a substantial further speed increase of approximately 25%.

6.3 Results of Centred PWM Approach

This approach uses more comparisons per position in the sequence than with the standard version. In the standard version during scoring the sequence is compared to the PWMs and subsequent background PWM for the length of maximum motif length. When adjusting the PWM, only the motif length for that particular PWM is used which varies in length between 12 and 25 with an average length of 18. In this approach however the scoring and adjusting of all PWMs is now using a length of 40 and therefore is more calculation intensive and hence slower than the standard approach (as was discussed in Section 6.1).

Also due to the longer comparisons and therefore lower probabilities that this incurs, the minimum threshold needs to be much lower. After experimentation, a threshold value of 10^{-25} for the centred PWM approach returned approximately the same number of candidate PWMs during the scoring selection as 10^{-14} held for the standard version. This also borders on the limits

of precision and so double precision was introduced. Despite utilising twice as much storage, there was no effect on the efficiency of the program.
Upon examining adjusted PWMs there seemed to be relative little change of any significance amongst the non-original (background) part of the PWM array which is line with the fact that there was no shifting observed in the standard version.

6.4 Results of Rank-based Selection approach

As discussed earlier, this approach does not include prior or peer information and therefore its results are not reliable as it fails to take advantage of vital information. Albeit faster overall, it was actually the slowest method comparatively at this stage of implementation (i.e., when the standard and centred versions were also missing the peer and prior information) due to the extra calculations that must be performed in scoring and adjusting the PWMs. Therefore overall if it had included this information it would have been far slower than both of the other two approaches as this substantially increase the complexity of this method.

The fact that the overlap was not removed seems to have had a significant effect on the shifting of the PWMs. The other approaches recorded no shifting, however in this approach as the overlap was not removed and discovered overlapping instances may have been from the same motif, there was a significant amount of shifting per scan. Examining the total number of shifted motifs within a scan then allowed investigation as to the optimum amount of scanning for a given DNA sequence. As seen in Figure 9 the 20 scans of which was originally thought is indeed a good basis, as this gives allows for a definite peak and then a decided lull in the amount of shifting performed per scan.

7 Conclusion

Despite attempts to find another approach to refine the probability motifs, the best overall approach remained that of the standard. Although the rank-based approach was faster overall, it misses the vital prior information and peer positions on which this idea is based and therefore the results from this method cannot be deemed as reliable. This method could be further developed to include this prior information but the speed of this program would not make it a viable option. The centring of the PWMs surround by background was also not a viable option as this only produced much slower

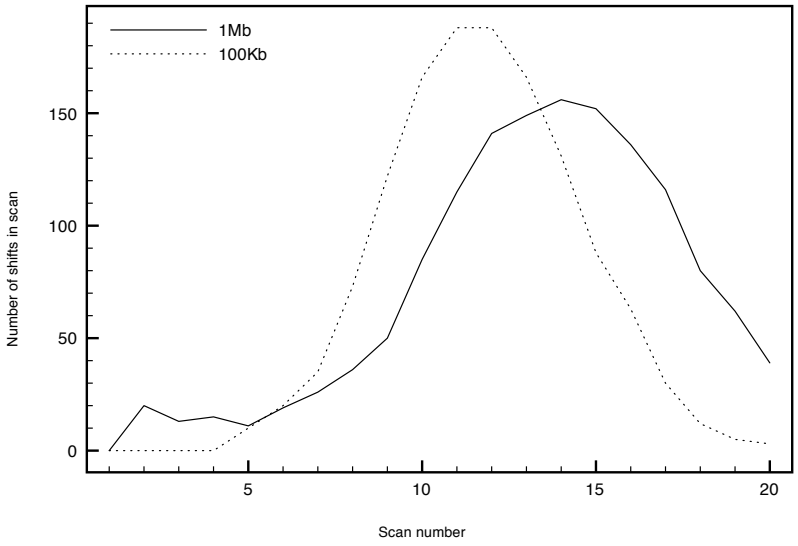

Figure 9: Curve showing the rise and decline in amount of shifting in rank-based selection method over twenty scans.

results without producing any significant results.

By optimising both the code and the compiler, as well as taking the decision to update the PWMs at the end of each scan of the sequence resulted in a much improved and efficient method of refining probability motifs than previously, and therefore a feasible tool in the world of Bioinformatics in the discovery of patterns in DNA.

References

[1] Wikipedia : Sense (Molecular Biology)
`http://en.wikipedia.org/wiki/Sense_(molecular_biology)`
accessed: 6 June 2008

[2] MedicineNet.com
`http://www.medterms.com/script/main/art.asp?articlekey=20468`
accessed: 6 June 2008

[3] Nomenclature for Incompletely Specified Bases in Nucleic Acid Sequences
http://www.chem.qmul.ac.uk/iubmb/misc/naseq.html
accessed: 12 June 2008

[4] Schneider T.D. and Stephens R.M., Sequence logos: a new way to display consensus sequences, Nucleic Acids Research 18 (2006) 6097–6100

[5] Guigo R. (2003), An Introduction to Position Specific Scoring Matrices
http://genome.imim.es/courses/genefinding/T2/MakeProfile.html
accessed: 23 Apr 2008

[6] Stormo G.D., DNA Binding Sites: Representation and Discovery, Bioinformatics 16 (2000) 16–23

[7] Intel C++ Compiler 10.1, Professional and Standard Editions, for Mac OS X
http://www.intel.com/cd/software/products/asmo-na/eng/compilers/266992.htm
accessed: 5 June 2008

[8] Systematic discovery of regulatory motifs in conserved regions of the human genome
http://www.broad.mit.edu/personal/xhx/projects/CNEMOTIF/
accessed: 12 March 2008

[9] Xie X., Mikkelsen T.S., Gnirke A., Lindblad-Toh K., Kellis M., and Lander E., Systematic discovery of regulatory motifs in conserved regions of the human genome, including thousands of CTCF insulator sites, Proceedings of the National Academy of Sciences 104 (2007) 7145–7150