Fast multiple alignment of ungapped DNA sequences using information theory and a relaxation method

Thomas D. Schneider* and David N. Mastronarde†

version = 1.24 of malign.tex 1995 April 19
for Discrete Applied Mathematics, Special Issue on Computational Molecular Biology ‡

ABSTRACT

An information theory based multiple alignment ("Malign") method was used to align the DNA binding sequences of the OxyR and Fis proteins, whose sequence conservation is so spread out that it is difficult to identify the sites. In the algorithm described here, the information content of the sequences is used as a unique global criterion for the quality of the alignment. The algorithm uses look-up tables to avoid recalculating computationally expensive functions such as the logarithm. Because there are no arbitrary constants and because the results are reported in absolute units (bits), the best alignment can be chosen without ambiguity. Starting from randomly selected alignments, a hill-climbing algorithm can track through the immense space of s^n combinations where s is the number of sequences and n is the number of positions possible for each sequence. Instead of producing a single alignment, the algorithm is fast enough that one can afford to use many start points and to classify the solutions. Good convergence is indicated by the presence of a single well-populated solution class having higher information content than other classes. The existence of several distinct classes for the Fis protein indicates that those binding sites have self-similar features.

INTRODUCTION

To study the statistics of bases in binding sites, not only do we need the sequences and an appropriate measure of the property we are interested in, but we also must have the sequences aligned against one another. Much attention has been paid to the alignment of one sequence against another [1] but the alignment of more than two sequences is hindered by the exponential nature of the problem. With a typical binding site having only 10 sequences available, each allowed to move back and forth over 20 positions, we find ourselves exploring a space of 10^{20} possible alignments. With exceptional cases, such as splice junctions [2], one could have 10^{10000} alignments.

It is impractical to thoroughly search this space for "the best alignment", so perhaps we can find a set of reasonable alignments. The algorithm developed in this paper allows one to explore a reasonable portion of the space of possible alignments. The method does not allow for gaps in the sequence because it is designed to align a set of sequences that contain a DNA (or RNA) binding site. It was successfully used to align binding sequences of the OxyR protein [3] and used to determine that an alignment of Fis binding sites was optimal.[1]

METHODS

This method uses a set of nucleic-acid sequences with an initially arbitrary alignment. A window, $\text{window}_{\text{left}}$ to $\text{window}_{\text{right}}$ is chosen relative to a base assigned coordinate 0. For example, the window consisting of positions -10

‡The text of this paper is available from ftp://ftp.ncifcrf.gov/pub/delila/malign.ps

*National Cancer Institute, Frederick Cancer Research and Development Center, Laboratory of Mathematical Biology, P. O. Box B, Frederick, MD 21702-1201. toms@ncifcrf.gov.

†Department of Molecular, Cellular and Developmental Biology, University of Colorado, Boulder, Colorado 80309. mast@beagle.colorado.edu.

[1]The method described in [16] is a descendent of the method described here since Malign originated in 1986.

to +10 is shown by the $*$'s in the alignment below:

```
      -  -                 +   +
      1  1   -         +   1   1
      4  0   5    0    5   0   5
      .............................
    1 cttgatactgtatgagcatacagtataatt
    2 aattatactgtatgctcatacagtatcaag
    3 ccttttgctgtatatactcacagcataact
    4 agttatgctgtgagtatatacagcaaaagg
    5 agcataactgtatatacacccagggggcgg
    6 ccgccccctgggtgtatatacagttatgct
          *********************
```

The algorithm also uses a global measure of how good the sequence alignment is within the window. For this purpose we chose the information measure $R_{sequence}$ [4, 5, 6], which represents the total sequence conservation in the window. Unlike other measures, $R_{sequence}$ has the advantage that the alignment results are consistent with further information analysis [7] and with a two-state thermodynamic model for the binding process [8]. We will describe the algorithm by a series of improvements on a basic method.

Some useful definitions are:

- *step*: to move one sequence left or right.
- *shuffle*: a set of *steps* of one sequence from $\text{window}_{\text{left}}$ to $\text{window}_{\text{right}}$.
- *pass*: a set of *shuffles* over all sequences.
- *run* (or *alignment*): a set of *passes* starting from different initial alignments.

In the simplest algorithm, we perform these operations at each *step*:

1. Tabulate the number of bases $b \in \{A, C, G, T\}$ at each position l within the window. Call this table $n(b, l)$.
2. Determine the number of bases at each position,
$$n(l) = \sum_{b=A}^{T} n(b, l) \tag{1}$$
and then create a table of frequencies:
$$f(b, l) = \frac{n(b, l)}{n(l)}. \tag{2}$$
3. We now evaluate the uncertainty [9, 10, 11] of each base within this window:
$$Hs(l) = -\sum_{b=A}^{T} f(b, l) \log_2 f(b, l) \qquad \text{(bits per base).} \tag{3}$$
4. The information in the sequences is
$$R_{sequence}(l) = 2 - (Hs(l) + e(n(l))) \qquad \text{(bits per base),} \tag{4}$$
where $e(n(l))$ is a small sample correction for $Hs(l)$ [4].
5. Our goal is to maximize the information from the entire window:
$$R_{sequence} = \sum_{l} R_{sequence}(l) \qquad \text{(bits per site).} \tag{5}$$

$R_{sequence}$ is a global measure because it is calculated uniformly from all the sequences at once.

6. In a single *shuffle*, a sequence is moved left by a predetermined parameter "shift left", evaluated, and then moved one position to the right, evaluated, and so forth until it has arrived at "shift right". The position that gives the highest information content, $R_{sequence}$, is chosen as the new alignment for that sequence. Conflicts are resolved by pseudo-random choice.

7. We then perform a series of *passes* through the sequences. A *pass* consists of shuffling the first sequence back and forth to maximize $R_{sequence}$, then the second sequence is *shuffled*, and so forth through the entire set of sequences. The algorithm halts when an entire *pass* has been completed with no change to any alignment, the change in $R_{sequence}$ is less than a given tolerance or, to avoid infinite cycling, when an arbitrary limit of *passes* has been reached.

The algorithm as it stands is slow because each evaluation requires a large amount of tabulation, and the calculation of many additions, divisions, multiplications and logarithms. We now show how the speed of the algorithm can be drastically increased, so that it becomes a practical tool. First, for simplicity, we will assume that $n(l) = n$, a constant. (In our current implementation, the ends of the sequences are not allowed to slide into the window.) Second, any constant quantity, such as the value 2 and $e(n(l))$ in equation (4), can be removed. So, instead of maximizing the information $R_{sequence}$, we minimize the total uncertainty:

$$H = \sum_{l} \sum_{b=A}^{T} -f(b,l) \log_2 f(b,l) \tag{6}$$

(The method is therefore curiously related to maximum entropy procedures. In biological systems the entropy is minimized by evolutionary selective pressure [12].) Third, since there are only n sequences, we can make up a table for values of partial uncertainties

$$flogf(i) = -f \log_2 f \tag{7}$$

for $f = i/n$ over the range $i = 0 \dots n$. Even with many sequences it is not expensive to store this table. $flogf(0) = 0$ since $\lim_{f \to 0} f \log_2 f = 0$. This table is constructed after n has been determined, but before the alignment *passes* are performed, so it eliminates all the divisions, multiplications and logarithms from the main loop. Now only table lookups and additions are needed to do a *shuffle*.

The algorithm can be improved further by reducing the computation at each *step*. At this point to do each *step* we:

1. remove a sequence from the $n(b,l)$ table by subtracting 1 from the appropriate entries. Call the table for the $n-1$ other sequences $n'(b,l)$.

2. add the sequence back to $n'(b,l)$ in all positions determined by "shift left" to "shift right" to regenerate $n(b,l)$.

3. Use the $flogf$ table to find the alignment that gives the minimum H:

$$H = \sum_{l} \sum_{b} flogf(n(b,l)). \tag{8}$$

This algorithm requires changing $n(b,l)$ for every *step* of the *shuffle*. We can avoid this by computing a table of differences of $flogf$ before we begin the *run*:

$$dflogf(i) = \left[-\frac{i+1}{n} \log_2 \frac{i+1}{n} \right] - \left[-\frac{i}{n} \log_2 \frac{i}{n} \right] \tag{9}$$

over the range $i = 0 \dots n-1$. Suppose that a new alignment of the sequence will place base d at position l of the window. Then the uncertainty at position l will be

$$H(l) = flogf(n'(d,l)+1) + \sum_{b, b \neq d} flogf(n'(b,l)). \tag{10}$$

But since

$$dflogf(n'(d,l)) = flogf(n'(d,l)+1) - flogf(n'(d,l)), \tag{11}$$

equation (10) becomes

$$H(l) = dflogf(n'(d,l)) + \sum_b flogf(n'(b,l)) \tag{12}$$

so

$$H = \sum_l dflogf(n'(d,l)) + \sum_l \sum_b flogf(n'(b,l)). \tag{13}$$

The second term is a constant which does not need to be calculated during a *shuffle* because $n'(b,l)$ is a constant that does not change during a *shuffle*. So by precalculating differences of the evaluation function, one can evaluate an aligned set of sequences using only table lookups and a sum. Each sequence is only removed from $n(b,l)$ once and restored in a different position only after the new alignment has been found. (If the sequence is not shifted, then the original $n(b,l)$ need not be changed at all.)

The overall *shuffle* algorithm is now:

1. Remove a sequence from the $n(b,l)$ table to create $n'(b,l)$;
2. For each sequence shift, ($\text{shift} \in \text{shift}_{\text{left}} \ldots \text{shift}_{\text{right}}$) find the sum of the increments in H that would accumulate over the bases in the window:

$$dH = \sum_{l=\text{window}_{\text{left}}}^{\text{window}_{\text{right}}} dflogf[n'(sequence(l + \text{shift}), l)]. \tag{14}$$

3. Choose the alignment shift that gives the smallest dH. Call this dH_{minimum}.
4. Adjust the overall uncertainty H by

$$H_{\text{new}} = H_{\text{old}} + dH_{\text{minimum}} - dH_{\text{old}} \tag{15}$$

   The value of dH_{old} for the alignment without any shift is conveniently found by using an "if shift = 0" statement in the *shuffle* loop that determines dH_{minimum}.

5. Add the sequence back into the $n'(b,l)$ table at the best shift to create the new $n(b,l)$.

An alignment is uniquely identified by the vector consisting of the n shifts. When a new alignment is found at the end of a *run*, its vector is placed into a list of vectors and when an alignment is found that is already in the list, only the total number of times it has appeared needs to be recorded.

The alignment vector $(4, 2, 3)$ is equivalent to $(0, -2, -1)$ because each sequence is shifted by -4, so vectors can be normalized by subtracting the first shuffle element from all elements. (This is not yet implemented in Malign version 2.38.)

The final sets of alignments are sorted by their information contents and reported as vectors. A second program, Malin, allows one to convert various alignments into Delila instructions [13] so that the sequences can be further manipulated and viewed with the sequence logo technique [14].

PROGRAMS

The Malign multiple alignment program was written in Pascal [15] and can be automatically converted to C. It is available as part of the Delila system [13, 17] by anonymous ftp to ftp.ncifcrf.gov in pub/delila/malign.p or via the World Wide Web site http://www-lmmb.ncifcrf.gov/~toms.

RESULTS

We aligned 16 randomly generated sequences that bind to OxyR [3], using a window of 35 bases from -17 to $+17$. The shifting parameters were set to the range -100 to $+100$ so that the Malign program would shift each sequence to its limit (without introducing gaps). For 1000 alignments there is a single well-populated best alignment, separated from all other alignments by more than 3 bits. For 10000 alignments the same well-populated best alignment at $R_{sequence} = 13.9$ bits was found, but three new alignments were found near 13.4 bits. Fig. 1 shows the distribution of $R_{sequence}$ values. (Because alignment vectors are not yet normalized, this is only an approximate distribution. However in the distribution there are only 140 duplicated values for OxyR and none for Fis.)

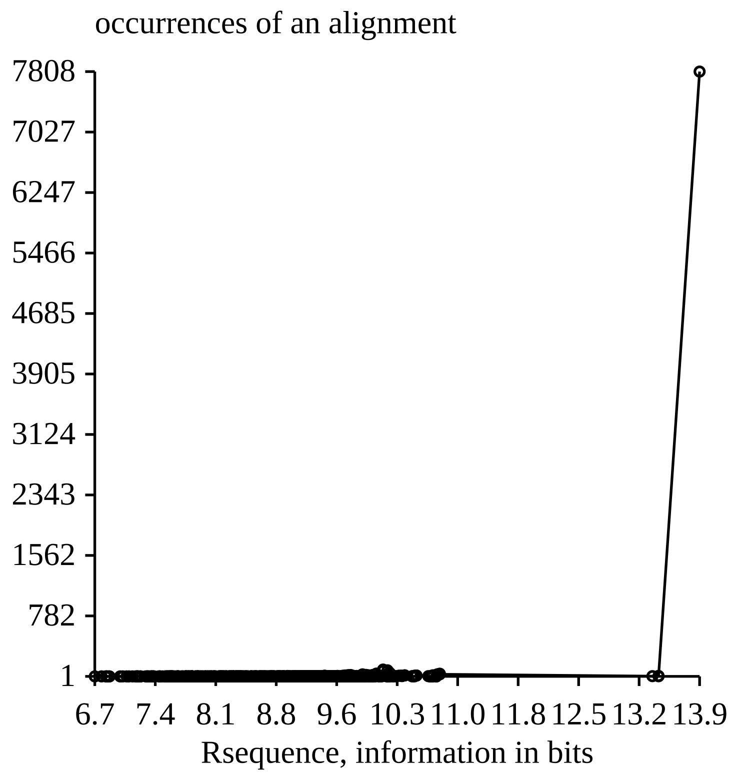

Figure 1: Distribution of 10000 alignments of 16 OxyR binding sites.

We collected 44 binding sites of the Fis protein [18, 19] from *Escherichia coli* and *Salmonella typhimurium* (manuscript in preparation) and we wanted to know whether the sites were aligned correctly. A window of 21 bases, from -10 to $+10$, covers the Fis sites. Since several of the Fis sites are spaced only 7 bases apart, we allowed shifting of each sequence only from -6 to $+6$. The total number of possible alignments in this space is $44^{13} = 2.3 \times 10^{21}$. 1000 alignments took 103 seconds on a Sun SPARCstation 20/61. 10000 alignments took 1023 seconds, and gave almost identical results. Another 10000 alignments starting with a different random number seed took 1014 seconds, and again gave almost identical results. Fig. 2 shows the distribution of $R_{sequence}$ values. There is a well-populated best alignment, a gap of about 1 bit and then a series of worse alignments. The next best alignments are more populated than the best alignment, but this means that they are easier to find, not that they are necessarily better. Despite our naive attempt to prevent the program from finding the nearby sites, the range -6 to $+6$ still allowed Malign to find alignments that include those sites. The striking difference between the distributions for Fis and OxyR can be explained by proposing that Fis sites have a self-similar structure, while OxyR sites do not, in contrast to the previous report [3]. The self-similarity of Fis sites gives spacings of 7 and 11 bases, as will be described elsewhere (P. N. Hengen *et al.*, manuscript in preparation). Inspection of the alignment vectors revealed the nature of the three peaks. The lowest peak represents shifts of ± 6 and ± 5. For example, 1702 alignments occurred with an $R_{sequence}$ of 7.4 bits and the following relative aligned bases:

```
-6  6  6 -6 -5  5  6 -6 -6  6 -6  6  5 -5  3 -3  5 -5  6 -6
-6  6  6 -6 -6  6  6 -6 -2  2 -6  6  6 -6  2 -2 -6  6  2 -2
 6 -6 -5  5  6 -6 -6  6 -6  6 -6  6  5 -5  6 -6  5 -5 -6  6
 5 -5 -6  6 -5  5  6 -6  6 -6 -6  6 -6  6  6 -6  6 -6  6 -6
 6 -6  6 -6  6 -6 -6  6  5 -5  2 -2
```

Many of these combinations would be equivalent to spacings of sites exactly 11 base pairs apart. (There are 92 alignments in the vector because both the sequences and their complements were used. They are listed as pairs of numbers which is why each number is followed by its negative.) The highest peak contained ± 5 and ± 2 (spacing of 7) and the middle peak appears to contain a combination of these.

DISCUSSION

Although the order of the algorithm is proportional to the number of sequences (n), the width of the window ($w = \text{window}_{\text{right}} - \text{window}_{\text{left}} + 1$), and the extent of the *shuffle* ($s = \text{shift}_{\text{right}} - \text{shift}_{\text{left}} + 1$), this is not a major hindrance because the algorithm converges quickly. This allows many of the s^n possible alignments to be tried

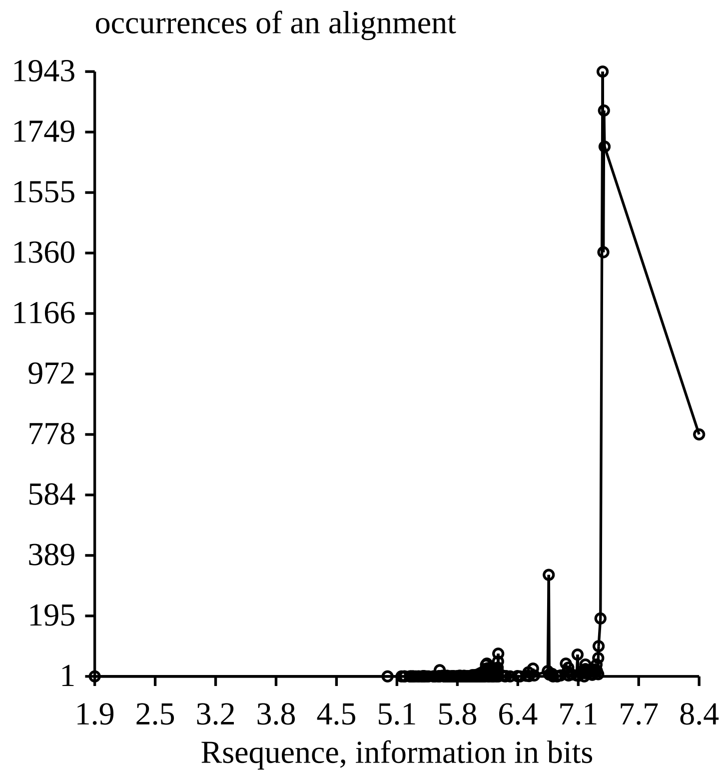

Figure 2: Distribution of 10000 alignments of 44 Fis binding sites.

as starting points, and allows the program to find a distribution of alignments, each a local minimum in the n dimensional alignment space defined by the alignment vectors.

The Malign program has several advantageous features:

1) It is able to try many combinations because it is fast.

2) Confidence in an alignment grows when it is found many times.

3) Although sequence alignments are discrete, they can be almost uniquely identified by their information content since the information measure is continuous.

4) The program is most useful in cases where a clear consensus sequence could not be determined. In retrospect we can understand why this happened for OxyR. The OxyR binding sites are spread out over 4 major and 3 minor grooves of B-form DNA and so have low information content per position on the average (0.4 bits/base), although the total is around 14 bits. This made alignment of synthetic random sequences difficult by hand but straightforward with Malign. Fis binding sites are smaller but their low information content of 8 bits has prevented determination of a consistent consensus [18, 19]. Forming a consensus requires altering the frequencies of bases from low values to zero and from high values to 50 or 100 percent, and this destroys the sensitivity that is maintained by Malign.

5) Because many alignments can be tried, the program provides a sensitive way to detect repetitious structures in a set of sequences.

Each *step* and the sums in the *step* are amenable to parallel processing, as is computation of the precalculated tables.

Multiple alignment with gaps [5, 6] is a difficult problem which we are often able to avoid because DNA-protein contacts are to a first approximation rigid. However cases of flexibility are known, such as the variable distance between procaryotic promoter -35 and -10 regions [20] and a few cases of altered spacing in CRP sites [21]. Ideally we would like to eliminate the arbitrary “gap penalties” used in many methods [1] because unlike the uncertainty, which corresponds to the entropy of the sequences [22], gap penalties have no obvious physical basis and the penalties might vary from position to position in a sequence. To devise a gap-penalty free algorithm we must first determine how to handle the gaps: should they be treated as characters or not? If one treats them as characters then the alignment will expand indefinitely because insertion of columns of blank characters would increase the information content. On the other hand, if one simply accepts the blanks and only calculates on the sequences, then one is reducing the variability of the patterns and so perhaps artificially raising the information content [4]. A simple solution is to calculate the additional uncertainty at each position using gaps and non-gaps as the symbols, since this has the property of not contributing to the total if gaps or non-gaps predominate. This method of counting gaps at each position seems to suggest that gaps could leap from one site to another irrespective of the surrounding sequence, and so it may not be reasonable. Alternatively, one may compute a penalty in bits for each sequence containing a given number of gaps by computing the logarithm of the total possible number of gap and non-gap arrangements, which would be a binomial. The overall penalty could be taken as the average over all sequences in the entire set. This method has the advantage of directly counting the number of ways a sequence recognizing molecule could be stretched to fit the binding site, but it appears to be sensitive to the size of the alignment window. Finally, the gaps could be treated in the computation as a set of bases with equal probabilities (or probabilities from the genome of the organism). It is not yet clear which, if any, method is correct in a philosophical sense. In addition to these subtle issues there are also technical difficulties, one of which is how to introduce the many possible combinations of gaps and non-gaps without requiring impossibly large computations. It is likely that dynamic programming methods could be used for this process.

ACKNOWLEDGEMENTS

We thank R. Michael Stephens, Vishnumohan Jejjala, Elaine Bucheimer, Paul N. Hengen, Peter K. Rogan, and Denise Rubens for useful discussions and comments on the manuscript; Gary D. Stormo for supporting DNM's graduate student rotation; and NIH grant GM28755 for support of TDS.

References

[1] M. Vingron and M. S. Waterman. Sequence alignment and penalty choice: Review of concepts, case studies and implications. *J. Mol. Biol.*, 235:1–12, 1994.

[2] R. M. Stephens and T. D. Schneider. Features of spliceosome evolution and function inferred from an analysis of the information at human splice sites. *J. Mol. Biol.*, 228:1124–1136, 1992.

[3] M. B. Toledano, I. Kullik, F. Trinh, P. T. Baird, T. D. Schneider, and G. Storz. Redox-dependent shift of OxyR-DNA contacts along an extended DNA binding site: A mechanism for differential promoter selection. *Cell*, 78:897–909, 1994.

[4] T. D. Schneider, G. D. Stormo, L. Gold, and A. Ehrenfeucht. Information content of binding sites on nucleotide sequences. *J. Mol. Biol.*, 188:415–431, 1986.

[5] S. C. Chan, A. K. C. Wong, and D. K. Y. Chiu. A survey of multiple sequence comparison methods. *Bull. of Math. Biol.*, 54:563–598, 1992.

[6] A. K. C. Wong, C. Chan, and D. K. Y. Chiu. A multiple sequence comparison method. *Bull. of Math. Biol.*, 55:465–486, 1993.

[7] P. P. Papp, D. K. Chattoraj, and T. D. Schneider. Information analysis of sequences that bind the replication initiator RepA. *J. Mol. Biol.*, 233:219–230, 1993.

[8] T. D. Schneider. Sequence logos, machine/channel capacity, Maxwell's demon, and molecular computers: a review of the theory of molecular machines. *Nanotechnology*, 5:1–18, 1994.

[9] C. E. Shannon. A mathematical theory of communication. *Bell System Tech. J.*, 27:379–423, 623–656, 1948.

[10] C. E. Shannon and W. Weaver. *The Mathematical Theory of Communication*. University of Illinois Press, Urbana, 1949.

[11] N. J. A. Sloane and A. D. Wyner. *Claude Elwood Shannon: Collected Papers*. IEEE Press, Piscataway, NJ, 1993.

[12] T. D. Schneider. Information and entropy of patterns in genetic switches. In G. J. Erickson and C. R. Smith, editors, *Maximum-Entropy and Bayesian Methods in Science and Engineering*, volume 2, pages 147–154, Dordrecht, The Netherlands, 1988. Kluwer Academic Publishers.

[13] T. D. Schneider, G. D. Stormo, J. S. Haemer, and L. Gold. A design for computer nucleic-acid sequence storage, retrieval and manipulation. *Nucl. Acids Res.*, 10:3013–3024, 1982.

[14] T. D. Schneider and R. M. Stephens. Sequence logos: A new way to display consensus sequences. *Nucl. Acids Res.*, 18:6097–6100, 1990.

[15] K. Jensen and N. Wirth. *Pascal User Manual and Report*. Springer-Verlag, New York, 1975.

[16] G. D. Stormo and G. W. Hartzell. Identifying protein-binding sites from unaligned DNA fragments. *Proc. Natl. Acad. Sci. USA*, 86:1183–1187, 1989.

[17] T. D. Schneider, G. D. Stormo, M. A. Yarus, and L. Gold. Delila system tools. *Nucl. Acids Res.*, 12:129–140, 1984.

[18] S. E. Finkel and R. C. Johnson. The Fis protein: it's not just for DNA inversion anymore. *Molec. Microb.*, 6:3257–3265, 1992.

[19] S. E. Finkel and R. C. Johnson. The Fis protein: it's not just for DNA inversion anymore (erratum). *Molec. Microb.*, 6:1023, 1992.

[20] D. K. Hawley and W. R. McClure. Compilation and analysis of *Escherichia coli* promoter DNA sequences. *Nucl. Acids Res.*, 11:2237–2255, 1983.

[21] A. M. Barber and V. B. Zhurkin. CAP binding sites reveal pyrimidine-purine pattern characteristic of DNA bending. *J. Biomol. Struct. Dyn.*, 8:213–232, 1990.

[22] T. D. Schneider. Theory of molecular machines. II. Energy dissipation from molecular machines. *J. Theor. Biol.*, 148:125–137, 1991.