BIOINFORMATICS

Vol. 16 no. 5 2000
Pages 439–450

PatSearch: a pattern matcher software that finds functional elements in nucleotide and protein sequences and assesses their statistical significance

Graziano Pesole[1,*], Sabino Liuni[2] and Mark D'Souza[3]

[1]Dipartimento di Fisiologia e Biochimica Generali, Università di Milano, Via Celoria, 26, 20133 Milano, Italy, [2]Centro di Studio sui Mitocondri e Metabolismo Energetico del Consiglio Nazionale delle Ricerche, Via Orabona, 4, 70126 Bari, Italy and [3]Argonne National Laboratory, Mathematics and Computer Science Division, 9700 South Cass Avenue Argonne, Illinois 60439-4844, USA

Received on September 1, 1999; revised and accepted on November 24, 1999

Abstract

Motivation: *The identification of sequence patterns involved in gene regulation and expression is a major challenge in molecular biology. In this paper we describe a novel algorithm and the software for searching nucleotide and protein sequences for complex nucleotide patterns including potential secondary structure elements, also allowing for mismatches/mispairings below a user-fixed threshold, and assessing the statistical significance of their occurrence through a Markov chain simulation.*

Results: *The application of the proposed algorithm allowed the identification of some functional elements, such as the Iron Responsive Element, the Histone stem-loop structure and the Selenocysteine Insertion Sequence, located in the mRNA untranslated regions of post-transcriptionally regulated genes with the assessment of sensitivity and selectivity of the searching method.*

Availability: *A Web interface is available at: http://bigarea.area.ba.cnr.it:8000/EmbIT/Patsearch.html.*

Contact: *graziano.pesole@unimi.it*

Introduction

A major challenge in molecular biology is the understanding of the regulation of gene expression both in a temporal and spatial framework. Indeed, basic mechanisms of life including cell growth, development and differentiation depend on the differential and regulated expression of specific genes.

Although regulatory elements controlling gene expression are generally embedded in the non-coding part of the genomes since the beginning of the sequencing era, efforts of the researchers were mainly focused on the deciphering of the coding region with the aim of infering the corresponding protein sequence and assessing its biological activity. Consequently, most of the software tools developed so far are devoted to the analysis of protein coding sequences (Burset and Guigo, 1996; Fickett and Hatzigeorgiou, 1997; Snyder and Stormo, 1995) or to the prediction of protein structure (Sternberg *et al.*, 1999).

Since a large part of the genomes, particularly in eukaryotes, does not code for proteins many sequence contigs from whole genome sequencing projects will not provide any useful biological information unless specific software tools are developed allowing *investigators* to fill the gap between data production and interpretation.

Transcriptional or post-transcriptional control of gene expression involves short DNA or RNA tracts respectively interacting with specific binding proteins. The DNA elements controlling transcription such as promoters and enhancers are definitely better characterized and various software tools have been devised for their identification so far (Fickett and Hatzigeorgiou, 1997; Werner, 1999). Contrary to this, RNA elements usually embedded in the 5′- and 3′-untranslated regions (UTR) of mRNA are much less studied and their identification is complicated by the fact that their activity very often derives mainly from the specific secondary structure rather than from the primary nucleotide sequence which instead may be barely conserved.

Among structural elements located in the UTR region of mRNAs whose biological activity has been demonstrated experimentally, there are the Iron Responsive Element (IRE) (Hentze and Kuhn, 1996), the Histone 3′-UTR stem-loop structure (Williams and Marzluff, 1995), and many others which play important roles in the regulation of gene expression.

*To whom correspondence should be addressed.

It is then of utmost importance to develop specific software tools which are able to identify these elements in other sequences, thus greatly contributing to their functional characterization.

We present here the PatSearch algorithm which is able to analyze user submitted sequence collections for the presence of complex patterns including potential secondary structure elements also allowing mismatch and/or mispairing below a user fixed threshold.

The designing of each functional pattern is based on the available experimental data derived from the literature and/or from the scientists involved in its functional characterization. The experimental data considered include expression patterns of genes with recombinant UTRs, site-specific mutagenesis, RNAse protection and chemical probing experiments.

To assess sensitivity and selectivity of the pattern search, a suitable random model is needed which allows us to compute the number of expected pattern hits. The simplest way to accomplish this task is by simulation of natural sequences according to Markov chain models which in most cases reveal a suitable approximation. The comparison, for each pattern, between the number of observed and expected hits according to the simulation procedure, through a measure of statistical significance, will provide an assessment of the probability that a pattern match found in a novel sequence is a good candidate for the functional activity under investigation.

System and methods

The PatSearch program is written in C language and runs under the Unix operating system. It is essentially based on the pattern-matching program 'scan_for_matches' which was written by Ross Overbeek, David Joerg, and Morgan Price in 1993. This version of the program described by D'Souza *et al.* (1997), with some updates, is also available through a Web-based system (http://www.mcs.anl.gov/compbio/PatScan/HTML/patscan.html).

The new version which implements the simulation procedure for assessing the statistical significance of pattern hits is available on the Web at the URL: http://bigarea.area.ba.cnr.it:8000/EmbIT/Patsearch.html.

Implementation

Defining Patterns and Using Them in Search Requests

The PatSearch pattern matcher takes as input a database (or a database subset) available on the server site (EMBL, Genbank, UTRdb and others) or a user defined list of accession numbers indicating the relevant database on the Web submission form.

The users are allowed to choose whether they wish to search for nucleotide (default) or protein sequences, whether they wish to search on both strands of nucleotide sequences, the maximum number of hits reported, and whether overlapping hits should be reported. Sequence data should use the standard codes:

{A,C,D,E,F,G,H,I,K,L,M,N,P,Q,R,S,T,V,W,Y} for amino acids, and {A,C,G,T or U} for nucleotides, where T and U are equivalent. In the sequence data, upper case and lower case are equivalent and ambiguity codes are not recognized and skipped by the pattern matching program.

The PatSearch program locates all sub-sequences from the input sequences that are matched by a specified pattern. The pattern description was inspired by 'regular expression' rules, although both the syntax and the semantics are different, especially for the inclusion of specific operators for finding complementary helices and palindromes. Here, we clarify what we mean by a pattern and how the program locates the sub-sequences matched by it.

A pattern is a sequence of pattern units:

$$\mathrm{p1\ p2\ p3} \ldots \mathrm{p}n.$$

These pattern units are separated by white space (i.e. one or more spaces, tab characters, or end-of-line characters). All the patterns must be named p1, p2, etc. as well as the rules r1, r2, etc. (see below) using lower case letters. For example,

$$\mathrm{GGCC\ 3} \ldots \mathrm{8\ GAACC}$$

is a valid pattern made up of three pattern units. This simple pattern would match any sub-sequence beginning with GGCC, followed by three to eight characters, followed by GAACC.

For example, the pattern would match both

$$\underline{\mathrm{GGCC}}\mathrm{ACG}\underline{\mathrm{GAACC}}$$

and

$$\underline{\mathrm{GGCC}}\mathrm{AAAACG}\underline{\mathrm{GAACC}}.$$

PatSearch patterns should use the standard IUB codes. X is the only ambiguity character allowed for protein sequences. The ambiguity codes supported for nucleotides are M-{A,C}; R-{A,G}; W-{A,T}; S-{C,G}; Y-{C,T}; K-{G,T}; B-{C,G,T}; D-{A,G,T}; H-{A,C,T}; V-{A,C,G}; N-{A,C,G,T}.

Before presenting a detailed description of pattern units and their matches, we need to briefly discuss the ability to reference sections of sequence that have been matched by a previous pattern unit. Consider the following pattern:

$$\mathrm{p1} = 4 \ldots 4\ \mathrm{p1\ p1}.$$

Here the first pattern unit is

$$\mathrm{p1} = 4 \ldots 4.$$

This will match any four-character sequence, and it will allow later pattern units to reference the matched

sub-sequence (as p1). Thus, the pattern will match any 12-character sub-sequence that is made up of three repeats of the same four-character sequence (e.g. ACGTACGTACGT).

Similarly,

$$\text{p1} = 4 \ldots 4 \text{ p1 p2} = 3 \ldots 3 \text{ p1 p2 p1 p1 p2}$$

will match AAAAAAAACCCAAAACCCAAAAAAAACCC as well as ACGTACGTACGACGTACGACGTACGTACG.

The scan of a sequence S begins by setting the current position to 1 (the first character of the sequence to be searched). Then, an attempt is made to match p1 starting at the current position. If the attempt succeeds, then an attempt is made to match the next unit. If it fails, then an attempt is made to find an alternative match for the immediately preceding pattern unit. If this succeeds, then we proceed forward again to the next unit. If it fails, we go back to the preceding unit. This process is called 'backtracking'. If there are no previous units, then the current position is incremented by one, and the process starts again. This process continues until either the current position goes past the end of the sequence or all of the pattern units succeed. On success, PatSearch reports the 'hit'. If the user chooses to detect also overlapping patterns, the current position is set to one character past the start of the match; otherwise, the current position is set just past the hit region. Then the process begins again to find another hit.

In the case of two (or more) internal solutions for a given pattern, only the first match found is reported if the 'overlap' option is not chosen. If this option is chosen, all possible matches will be reported including those with the same ends and with different interior arrangements (for example, if a tRNA can fold into two different structures and a suitable pattern is designed the program will report both).

Pattern units that apply to both protein and nucleotide sequences. Various pattern units apply to both nucleotide and protein sequences.

(a) String Pattern Unit. This is a string of characters that may include ambiguity characters.

EXAMPLES. rGGrGG will match AGGAGG, GGGAGG, AGGGGG, or GGGGGG. C*xx*C*xxx*C will match any eight-character amino acid sub-sequence in which the first, fourth, and last characters are C.

(b) Pattern unit with a [mismatches,deletions,insertions] modifier.
The modifier contains three integers, representing the number of mismatches, deletions, and insertions allowed in the matched sub-sequence.

EXAMPLES. TTTATTT[1,0,0] would match TTTGTTT or any other sub-sequence with a single mismatch.
TTTATTT[0,1,0] would match TTTTTT or any other sub-sequence missing a single character of the pattern unit.
TTTATTT[0,0,1] would match TTTACTTT or any other sub-sequence with an inserted character.
ACGTACGTACGT[1,1,1] would match ACGGTAGGTCGT (very slow). p1[2,0,0] would match a previously matched sub-sequence (recorded as p1) allowing two mismatches.
The use of deletions or insertions considerably slows down matching. Users frequently use loosely defined modifiers, allowing the pattern to match almost any sub-sequence.

(c) Range Pattern Unit. This unit has the form: min...max.

EXAMPLES. 0...1 matches either 0 or 1 characters. 1...200 matches any sub-sequence from 1 to 200 characters.
The range pattern units will first match the minimum number of characters; and the length will be expanded on backtracking if necessary

(d) Either/or pattern unit. This has the form (p1 | p2), which will match either p1 or p2. The alternatives may themselves be complex patterns (made up of more than one pattern unit). Parentheses are required, as well as spaces before and after the vertical bar.

EXAMPLES. (CxxC | CxxM) would match both CAAC and CAAM. ATG (((0...0 | 3...3) | 6...6) | 9...9) ATG would match *ATGATG*, *ATG*CCCATG, *ATG*TTTTTTATG or *ATG*GGGGGGGGGATG where CCC, TTTTTT and GGGGGGGGG are arbitrary sequences.

(e) Start of Sequence. ^ matches only at the start of a sequence (and does not 'consume' characters).

EXAMPLE. ^ ATG matches only an initial ATG.

(f) End of Sequence. $ matches only at the end of a sequence (and does not 'consume' characters)

EXAMPLE. ALV $ matches only a terminal ALV.

(g) Palindrome Pattern Unit. <p1 matches the palindrome of the sub-sequence previously recorded as p1.

EXAMPLE. $\text{p1} = 4 \ldots 4 <$ p1 matches the sub-sequence SAPRRPAS.

Pattern units that apply only to protein sequences. When constructing patterns for protein sequences, it is useful to refer to classes of amino acids. The following two constructs relieve one from the need for ambiguity codes for amino acids.

(a) any(list of characters) matches a single character if it is in the list.

EXAMPLES. any(ILV) matches an I an L or a V any(ILV) 2...3 any(CH) matches IPDH as well as VQMC.

(b) notany(list of characters) matches a single character if it is not in the list.

EXAMPLES. notany(ILV) matches any amino acid other than I, L, or V. any(ILV) 2...3 notany(CH) matches IPDQ as well as VQMD.

Pattern units that apply only to nucleotide sequences. When looking for patterns in nucleotide sequences, it is often necessary to be able to look for regions that 'loop back and bind' a previous region. The most obvious case is that of a hairpin loop. In the simplest case, this pattern search can be done easily by using a pattern unit of the form

$$\sim \text{p1}$$

which matches the reverse complement of the subsequence recorded in p1. Thus, the pattern

$$\text{p1} = 6 \ldots 8\ 3 \ldots 8 \sim \text{p1}$$

can be used to match a hairpin loop (sometimes called a stem-loop) structure in which the stem is six to eight characters in length, and the loop is three to eight characters in length. Depending on the choice of the 'overlapping' option, in the case of two (or more) internal solutions for a given pattern, only the first or all the possible solutions will reported respectively. For example, if the sequence GCGGGCGACCGC is searched using the pattern 'p1 = 3...5 4...6 ~ p1' if the overlap option is not chosen, only the first match found is reported—'GCG GGCGAC CGC'. If the overlap option is chosen, in addition to the first match, the two internal matches are also reported, i.e. 'GCGG GCGA CCGC' and 'CGG GCGA CCG'.

While useful, more capabilities are needed to search for many RNA and DNA structures. We have added a number of features to address this need. Consider the following pattern (which is written on two lines—a line can be broken anywhere that one can put spaces in a pattern):

r1 = {au, ua, gc, cg, gu, ug, ga, ag}
p1 = 2...3 0...4 p2=2...5 1...5 r1 ~ p2 0...4~p1

The 'pattern unit' on the first line does not actually match anything; rather, it defines a 'pairing rule' in which standard pairings are allowed, as well as G–U, U–G, G–A, and A–G. In this format, r1 = {AU, UA, gc, cg} could be used to define the 'standard rule' for pairings. The second line consists of six pattern units, which may be interpreted as follows:

p1 = 2...3	match 2 or 3 characters (call it p1)
0...4	match 0 to 4 characters
p2 = 2...5	match 2 to 5 characters (call it p2)
1...5	match 1 to 5 characters
r1 ~ p2	match the reverse complement of p2, allowing G–U, U–G, G–A, and A–G
0...4	pairs match 0 to 4 characters
~p1	match the reverse complement of p1, allowing only G–C, C–G, A–T, and T–A pairs.

Thus, r1 ~ p2 means 'match the reverse complement of p2 using rule r1'.

Now let us consider the issue of tolerating mismatches and bulges.

One may add a qualifier to the pattern unit that gives the tolerable number of 'mismatches, deletions, and insertions'.

Thus,

$$\text{p1} = 10 \ldots 10\ 3 \ldots 8 \sim \text{p1}[1, 2, 1]$$

means that the third pattern unit must match 10 characters, which are the reverse complement of the 10 characters in p1, allowing one 'mismatch' (a pairing other than G–C, C–G, A–T, or T–A), two deletions (a deletion is a character that occurs in p1, but has been 'deleted' from the string matched by ~p1), and one insertion (an 'insertion' is a character that occurs in the string matched by ~p1, but not in p1). In this case, the pattern would match

ACGTACGTAC GGGGGGGG GCGTTACCT

which is a fairly weak loop.

Weight matrices. A weight matrix can be used as pattern unit. Suppose you want to match a sequence of eight characters. The 'consensus' of these eight characters is GRCACCGS, but the actual 'frequencies of occurrence' are given in the matrix below. Thus, the first character is an A 16% of the time and a G 84% of the time. The second is an A 57% of the time, a C 10% of the time, a G 29% of the time, and a T 4% of the time, and so on as given below.

	C1	C2	C3	C4	C5	C6	C7	C8
A	16	57	0	95	0	18	0	0
C	0	10	80	0	100	60	0	50
G	84	29	0	0	0	20	100	50
T	0	4	20	5	0	2	0	0

The following pattern unit can be used to search for inexact matches related to such a 'weight matrix':

{(16,0,84,0),(57,10,29,4),(0,80,0,20),(95,0,0,5),
(0,100,0,0),(18,60,20,2),(0,0,100,0),(0,50,50,0)} >450.

This pattern unit will attempt to match exactly eight characters. For each character in the matched sub-sequence, the entry for that character in the corresponding 4-tuple is added to an accumulated sum. If the sum is greater than 450, the match succeeds; otherwise it fails. It is also possible to use ranges as in the following example:

600 > {(16,0,84,0),(57,10,29,4),(0,80,0,20),(95,0,0,5),
(0,100,0,0),(18,60,20,2),(0,0,100,0),(0,50,50,0)} >450

When dealing with nucleotide patterns, each weight matrix entry is a 4-tuple, but in a protein sequence each is a 20-tuple (with entries corresponding to the amino acid codes in alphabetic order). It is clear that such matrices are almost impossible to formulate or work with, unless they are automatically generated by a program.

Finally, we note that the crude matrix used above is not really very well formulated. There is a broad literature on the use of weight matrices (see Gelfand, 1995). All we will say here is that it would have been better to convert the entries in negative log values, normalize them, and construct the matrix. It makes more sense to sum the negative logs of the frequencies, rather than the frequencies themselves.

Postprocessing. It is occasionally very convenient to be able to 'reprocess' a section of a sequence that has already been matched. For example, consider a pattern with the form

p1 = 6 . . . 6 GC3 . . . 200 TGCATGCGGC[1, 0, 0] ∼ p1.

This might well match very slowly, given the 3. . . 200 pattern unit. However, the pattern unit TGCATGCGGC[1,0,0] is by far the most discriminating (in that it fails to match in the vast majority of cases). For this reason in PatSearch we allow the use of the following pattern

p1 = 11 . . . 208 TGCATGCGGC[1, 0, 0]
p2 = 6 . . . 6 p1/p2 : (p3 = 6 . . . 6 GC3 . . . 200 ∼ p3$)

and thus a two-pass approach is carried out in which the input sequence is first matched against the most discriminating pattern and then the hits from this pass are processed using the full pattern. The syntax of the last pattern unit is

list : (subpattern)

where list is a list of recorded sections of sequence (in this case, just p1/p2/) and subpattern is a pattern to match against the concatenation of the regions represented by the list of recorded matches. The post-processing is not limited to a single rescan.

Simulation Procedure

The simulation procedure we have devised (described below), allows the user to consider Markov chains of any order for the generation of simulated sequences. For example using a Markov chain of the first order, the actual dinucleotide frequencies are considered for sequence generation, i.e. simulated sequences maintain the dinucleotide frequency of natural sequences. In the case of amino acid patterns the sequences are simulated taking into account only the actual amino acid frequencies of the searched proteins.

Let us consider the collection to be searched for a given pattern which contains N nucleotide sequences of length L_k ($k = 1, \ldots, N$). The simulation procedure, carried out R times for each pattern search, using a Markov chain of order M, is then described as follows:

(1) iterate n from 1 to R;

(2) iterate k from 1 to N;

(3) choose randomly the first w-gram ($w = M + 1$) of the sequence S_k with probability $\pi_{ik} = p_{ik}/(L_k - M)$, where p_{ik} is the observed frequency of the ith w-gram ($i = 1, \ldots, 4^w$) in the sequence k;

(4) Generate nucleotide $w + j$ ($j = 1, L_k - w$) using a M-order Markov chain generator—the key property of a Markov chain generator of order M is that the probability of each symbol in the sequence depends only on the value of the preceding M symbols. In the case of a zero-order Markov chain the nucleotide $w + j$ ($j = 1, L_k - 1$) will be generated randomly as A, C, G or T with a probability defined by their relative frequency in the natural sequences. If $M \geq 1$ the nucleotide $w + j$ will be generated as A, C, G or T with probability respectively proportional to the observed frequency of the four w-grams $x_{j+1}, \ldots, x_{j+w-1}$A, $x_{j+1}, \ldots, x_{j+w-1}$C, $x_{j+1}, \ldots, x_{j+w-1}$G, $x_{j+1}, \ldots, x_{j+w-1}$T as determined by the value of an extracted random number.

The order of the Markov chain is chosen by the user (see Discussion). In the simulation procedure the sequences to be searched for a given pattern are thus shuffled but retain the natural $w$mers composition.

The pattern search can be carried out on a large number (e.g. 100) of simulated sequence datasets thus allowing the calculation of expected hits with the relevant SD values. The statistical significance of the observed hits can be thus easily calculated by comparing observed

and expected values with the usual chi-square statistics [chi-square $= (\text{Obs} - \text{Exp})^2/\text{Exp}$].

Application

We have used the PatSearch matcher to search for specific sequence patterns already known to play some functional role in the regulation of gene expression. In particular, we considered some *cis*-acting elements located in the 5′- or 3′-UTR regions of eukaryotic mRNAs which may play some fundamental role in the post-transcriptional regulation of gene expression.

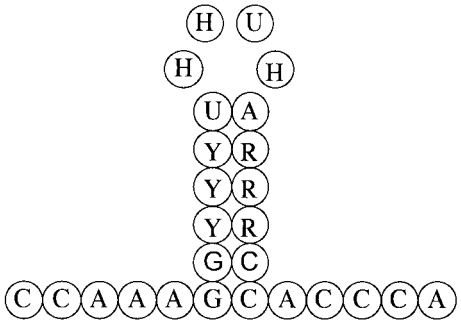
The 5′- or 3′-UTR elements usually correspond to short oligonucleotide tracts, which generally fold into specific secondary structures and are binding sites for various regulatory proteins. The pattern description syntax of the PatSearch program is particularly suitable for modeling the consensus structure of such functional elements.

Among the *cis*-acting oligonucleotide patterns located in the UTR regions of eukaryotic mRNA involved in post-transcriptional regulation of gene expression the histone stem-loop element, the IRE and the SElenoCysteine Insertion Sequence (SECIS) are those more extensively studied and better characterized (Hentze and Kuhn, 1996; Hubert *et al.*, 1996; Williams and Marzluff, 1995). The definition of the sequence patterns specific for each of the above functional elements was based on the extensive comparative analysis of the sequence regions whose biological activity was experimentally demonstrated and on available experimental data obtained by chemical probing or site-specific mutagenesis.

In the following section these functional elements are described in more detail reporting the derived consensus pattern. These functional patterns, with many others which are specific of 5′- and 3′-UTR eukaryotic mRNAs, are collected as entries of the UTRsite database (http://bigarea.area.ba.cnr.it:8000/EmbIT/UTRHome/) where a summary description of their biological activity can be found.

Histone mRNA 3′-UTR stem-loop structure

Metazoan histone 3′-UTR mRNAs, lacking a polyA tail, contain a highly conserved stem-loop structure with a six base stem and a four base loop. Figure 1 shows the derived consensus stem-loop structure and the relevant PatSearch pattern we devised for the histone stem-loop element. In all histone mRNAs analyzed so far no G has been observed in the four base loop. In all metazoan except *Caenorhabolitis elegans*, there are two invariant urydines in the first and third base of the loop. In *C. elegans* the first base of the loop is C. Either 5′ or 3′ flanking sequences are necessary for high affinity binding of SLBP. The 5′ flanking sequence consensus is CCAAA and the 3′ flanking sequence consensus is ACCCA or ACCA with cleavage occurring after the CA.



PatSearch pattern:

r1={au,ua,gc,cg,gu,ug}
0...1 mmmm p1=ggyyy u hhuh a r1~p1 mm 0...3

Fig. 1. Consensus structure devised for the histone mRNA stem-loop element (a). In the corresponding PatSearch pattern (b) only four and two bases respectively, preceding and following the stem-loop structure, are constrained to be A or C (M = A/C in the IUB code).

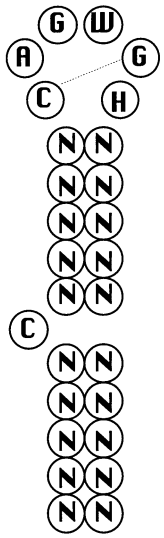
The histone 3′-UTR hairpin structure is peculiar in that the bases of the stem are conserved unlike most functional hairpin motifs where conserved bases are found in single stranded loop regions only. The sequence of the stem and flanking sequences are critical for binding of its interacting stem-loop binding protein (SLBP).

Iron responsive element

The IRE is a particular hairpin structure located in the 5′- or the 3′-UTR of various mRNAs coding for proteins involved in cellular iron metabolism. Iron responsive elements are recognized by transacting proteins known as Iron Regulatory Proteins (IRPs) which control mRNA translation rate and stability. Figure 2 shows the derived IRE consensus structure and the relevant PatSearch pattern. Two alternative IRE consensus have been found both showing a bipartite stem interrupted by a bulged C or by a small internal loop formed by a cytosine nucleotide opposed to a trinucleotide ending with another cytosine. Some evidences also suggest a structured loop with an interaction between nucleotide one and nucleotide five (connected by a dashed line in Figure 2). The lower stem can be of variable length and is AU-rich.

Selenocysteine insertion sequence

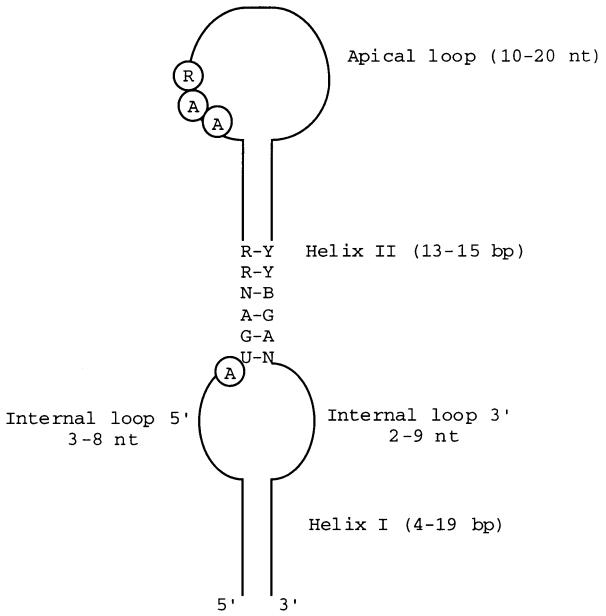
Specific incorporation of selenocysteine in selenoproteins is directed by UGA codons residing within the coding sequence of the corresponding mRNAs. Translation of UGA, usually a termination codon, as selenocysteine



PatSearch pattern:

r1={au,ua,gc,cg,gu,ug}
(p1=2...8 c p2=5...5 CAGWGH r1~p2 r1~p1 |
p3=2...8 nnc p4=5...5 CAGWGH r1~p4 n r1~p3)

Fig. 2. Consensus structure devised for the IRE (a). Two alternative PatSearch patterns (b) are reported and degenerate nucleotides are represented by the IUB code (W = A/U; H = not G; N = any base).



PatSearch pattern:

r1={au,ua,gc,cg,gu,ug}
r2={uu,cu,cc,au,aa,gu,gg}
p1=4...19
p2=2...7 a
uga p3=1...1 p4=rr p5=7...9
p6=0...2 aav p7=5...15
r1~p5[1,1,1] r1~p4 r2~p3 gan
p8=2...9 ~p1[1,0,0]

Fig. 3. Consensus structure devised for the Selenocysteine insertion sequence (a). In the corresponding PatSearch pattern (b) two different pairing rules ($r1$ and $r2$) are used for different helices where mismatches and indels are allowed in some cases.

requires a conserved stem-loop structure called SECIS lying in the 3′-UTR region of selenoprotein mRNAs. The consensus structure of SECIS element determined by comparative analysis of several selenoprotein mRNAs as well as with both RNase and chemical probing is characterized by a hairpin structure composed of two helices of different length separated by an internal loop. In the top helix a quartet of conserved ‘non-Watson–Crick’ base pairs is crucial for functional activity. The derived consensus structure of the SECIS element and the relevant PATSEARCH pattern is shown in Figure 3.

Finding functional elements in mRNA UTR sequences using PatSearch

In order to search for the above described functional elements in the UTR regions we used the PatSearch program with input sequences the entries of the UTRdb database and input patterns those previously described (Figures 1–3).

A non-redundant database, UTRdb collects all UTR sequence regions of eukaryotic mRNAs which is structured in the same taxonomic divisions adopted by the EMBL/Genbank database. The present version of the database (release 12, October 1999) contains more than 85 000 entries and about 30 million nucleotides.

The UTRdb entries found to contain the IRE element in the 5′-UTR and the histone stem-loop or the SECIS elements in the 3′-UTR are listed in Table 1. It is interesting to note that all 30 3′-UTR sequences containing the histone 3′-UTR stem-loop pattern in Figure 1 actually corresponded to histone mRNAs belonging to different species and including both vertebrates and invertebrates. A very high selectivity has also been found for the IRE pattern where 18 out the total 20 matching 5′-UTRs correspond to mRNAs coding for proteins involved in iron metabolism regulation and known to contain the IRE element in their 5′-UTR. A different situation has been observed for the SECIS pattern which seems to be much less selective with only 27 out of the total 77 pattern hits actually corresponding to selenoprotein mRNAs known to contain a SECIS element in their 3′-UTR region—the other genes, some of them unknown, apparently do not code for selenoproteins. Indeed, if the data for the SECIS

element in Table 1 are considered there are many mRNAs of unknown function which could possibly undergo a SECIS-mediated regulation.

The matching patterns found in UTR sequences for which no experimental data are available can be considered either functional candidates or just false positives. This is the case, for example, of the two additional mRNAs found to contain the IRE element in the 5′-UTR (see Table 1) which could be considered as reliable candidates for IRE-mediated regulation.

If we define the true and false positives (t^+ and f^+) as matched functional or non-functional patterns respectively, and true and false negative (t^- and f^-) as unmatched functional or non-functional patterns, we can calculate the sensitivity and selectivity of the method as (Snyder and Stormo, 1995):

$$\text{Sensitivity} = t^+/(t^+ + f^-)$$
$$\text{Selectivity} = t^+/(t^+ + f^+).$$

The sensitivity represents the percentage of genuine functional patterns, in total $t^+ + f^-$, recognized by the pattern matcher. The selectivity represents the percentage of the total positive matches, i.e. $t^+ + f^+$, which actually correspond to genuine functional elements. False negatives can be recognized as UTR sequences which fail to match the derived consensus pattern although their relevant functional activity has been experimentally demonstrated or consistently predicted on a comparative basis. On the contrary, true negatives and false positives can be defined only by experimental check. Therefore, the degree of selectivity, which gives the probability the matched pattern has biological activity, cannot be directly calculated with the formula above.

Our data show that both the histone stem-loop and the IRE derived consensus patterns are able to match all UTRs known to contain either of the two elements, thus suggesting a very high sensitivity level. A good sensitivity has been observed also for the SECIS element as 90% (27/30) of the known elements are predicted by the derived consensus pattern in Figure 3 and only three elements (e.g. Human Gpx3, Tilapia type I iodothyronine deiodinase and Rat type I iodothyronine deiodinase mRNAs) fail to match. Indeed, also in this case total coverage can be obtained just allowing one more mismatch in helix two (see the consensus structure in Figure 3). However, the price to pay for this small increase in sensitivity is a consistent decrease in selectivity as in the resulting two-fold increase of matching UTRs.

Assessing statistical significance and selectivity level by pattern matching simulation

The evaluation of the statistical significance of the observed occurrence of a given pattern can be obtained through the simulation procedure described in the Implementation section. Indeed, if a simulation is carried out which generates a huge number of investigated sequence dataset where simulated sequences retain the nucleotide and/or oligonucleotide composition of the natural sequences, the execution of the pattern searching algorithm using the devised consensus pattern allows us to estimate the average number of matches we may expect just by chance. Assuming the average number of hits represents the expected number of hits under a random model we can easily calculate the statistical significance of the observed hits through the simple chi-square statistics. If we assume that the expected number of pattern hits estimates the number of false positives we can easily assess the selectivity level for each pattern as previously defined.

In Table 2 the number of observed and expected patterns for each of the three functional elements here considered, calculated using a first-order Markov chain in the simulation (see Discussion), with the statistical significance and the selectivity level, are reported. A conservative estimate of the selectivity level can also be obtained as the proportion of UTR sequences known to contain the relevant functional pattern (marked in Table 1) over all matching UTRs. This is likely to underestimate the selectivity level as the biological activity of the unknown matching pattern cannot be excluded without experimental investigation.

According to simulation data, the highest selectivity, about 100%, is observed with the histone stem-loop structure. This means that the probability that a match is not functional (false positive) is negligible. Indeed, all found matches correspond to actual histone mRNAs. Also the IRE pattern was rather selective in 5′-UTRs, with a selectivity level ranging from 84 to 97%. It is interesting to note that both histone stem-loop and IRE patterns were found to be statistically significant only in animal mRNAs and not in plant or fungi mRNAs where these elements are not reported to be functional.

The SECIS pattern presented a very low level of selectivity and resulted in being significantly over-occurring only in rodent and other mammal 3′-UTRs.

Discussion

The enormous flood of sequence data produced by the many sequencing projects now in progress requires the development of suitable bioinformatic tools which may greatly help their functional characterization. Conversely, the gap between data production and their interpretation is doomed to increase ever more, making newly produced genomic sequence data useless over time. Database searching is the most common approach for the characterization of anonymous sequences where the finding of

Table 1. List of mRNA sequences matching the histone stem-loop element in the 3′-UTRs, the IRE in the 5′-UTRs and the SECIS in the 3′-UTRs. For each match the relative position in the UTR region, the UTRdb ID and the EMBL accession numbers are provided. mRNAs known to contain a functional element are also ticked (√)

UTRdb ID	Position	EMBL AC	species	Description	
Histone 3 pattern					
3HSA014005	42…64	Z98744	Human	histone H2A	√
3HSA014006	43…65	Z98744	Human	histone H2B	√
3HSA014008	32…54	Z98744	Human	histone H1.5	√
3HSA014009	35…57	Z98744	Human	histone H3.1	√
3HSA014011	22…44	Z98744	Human	histone H2B	√
3HSA014092	22…44	AL009179	Human	Histone H2B	√
3HSA014093	40…62	AL009179	Human	Histone H2A	√
3HSA014094	27…49	AL009179	Human	Histone H3.1	√
3HSA014418	46…68	L19778	Human	histone H2A.1b	√
3HSA014419	63…85	L19779	Human	histone H2A.2	√
3HSA004810	45…67	X14850	Human	histone H2A.X	√
3HSA010018	39…61	U90551	Human	histone 2A-like	√
3CLO000016	64…86	X80330	Hamster	histone H2a.2	√
3CLO000017	26…48	X80330	Hamster	histone H3.2	√
3MMU005868	54…76	Z30940	Mouse	histone H2A	√
3MMU000265	27…49	Z30939	Mouse	histone H3	√
3MMU002585	42…64	X58069	Mouse	histone H2A.X	√
3MMU002586	23…45	X80328	Mouse	histone H2b	√
3MMU002587	20…42	X80328	Mouse	histone H3	√
3MMU004991	33…55	U62672	Mouse	histone H3.1-D	√
3MMU004994	26…48	U62675	Mouse	histone H3.2-616	√
3MMU004995	34…56	U62675	Mouse	histone H2b-616	√
3MPA000003	28…5	X80324	shrew mouse	histone H3.1	√
3MPA000004	21…43	X80326	shrew mouse	histone H3.2	√
3MPA000005	20…42	X80325	shrew mouse	histone H3.1	√
3RNO001863	23…45	M18046	Rat	histone H2B	√
3TNI000020	24…46	X54078	Nile tilapia	histone H4	√
3XBO000012	24…46	J00985	*Xenopus*	histone H4	√
3EMI000001	22…44	L41834	jackknife clam	nuclear protein	√
3PHE000001	27…49	X54114	*Pycnopodia helianthoides*	histone H3	√
IRE pattern					
5HSA001988	13…35	X60364	Human	5-aminolevulinate synthase	√
5HSA003829	8…30	Y09188	Human	ferritin *L*-chain	√
5HSA003858	35…57	D28463	Human	ferritin heavy chain	√
5HSA013930	34…56	J04755	Human	ferritin H processed pseudogene	√
5MMU001248	15…37	M63244	Mouse	amino levulinate synthase	√
5MMU002159	33…55	M60170	Mouse	ferritin heavy chain	√
5MMU002160	2…24	J04716	Mouse	ferritin light chain	√
5SSC000205	17…39	D15071	Pig	ferritin heavy-chain	√
5OMY000025	6…28	D86626	Rainbow trout	ferritin H-2	√
5RCA000039	28…50	M12120	Bull frog	Ferritin	√
5SSA000004	32…54	S77386	Salmon	ferritin middle subunit	√
5XLA000031	506…528	S64727	*Xenopus*	Ferritin	√
5XLA000423	171…193	X51395	*Xenopus*	Ferritin	√
5GGA000387	229…251	M55644	Chicken	marker protein (Ch21)	
5AAE000011	85…111	L37082	yellow fever mosquito	Ferritin	√
5DME001131	64…92	L27705	Fruit fly	succinate dehydrogenase iron-subunit	√
5DME001318	252…282	U67304	Fruit fly	70 kDa S6 kinase	
5DME001793	151…177	Y15629	Fruit fly	ferritin subunit 1	√
5IRI000001	2…26	AF068224	*Ixodes ricinus*	Ferritin	√
5LST000028	22…44	X56778	great pond snail	Ferritin	√

Table 1. *Continued*

SECIS pattern					
3HSA001292	80…139	U37143	Human	cytochrome P450 monooxygenase CYP2J2	
3HSA002889	168…236	X04076	Human	Catalase	
3HSA003963	312…366	L07077	Human	enyol-CoA: hydratase 3-hydroxyacyl-CoA dehydrogenase	
3HSA004706	212…291	X68314	Human	glutathione peroxidase-GI	√
3HSA004716	216…293	X53463	Human	glutathione peroxidase-like protein	√
3HSA004727	54…117	X71973	Human	phospholipid hydroperoxide glutathione peroxidase GPx4	√
3HSA006271	1431…1494	L06175	Human	P5-1 gene	
3HSA006417	16…86	X64652	Human	MSSP-1 gene	
3HSA006419	16…86	X77494	Human	MSSP-2 gene	
3HSA006578	254…325	D42072	Human	Neurofibromatosis type 1 protein	
3HSA009338	2677…2749	U61167	Human	SH3 domain-containing protein SH3P18	
3HSA010862	541…614	S79854	Human	type 3 iodothyronine deiodinase	√
3HSA011363	744…799	AF029750	Human	Tapasin	
3HSA011596	258…324 + 697…758	Z11793	Human	selenoprotein P	√
3HSA012601	196…254	AB003062	Human	myosin phosphatase targeting/regulatory subunit	
3HSA012772	1796…1853	AB011153	Human	KIAA0581 protein	
3HSA012839	444…506	AB014524	Human	KIAA0624 protein	
3HSA012901	1744…1796	AB014586	Human	KIAA0686 protein	
3HSA014180	111…174	AB002329	Human	KIAA0331 protein	
3HSA015189	2765…2818	AF061189	Human	ectodysplasin-A isoform EDA-A2	
3HSA015329	80…140	AF079529	Human	cAMP-specific phosphodiesterase 8B	
3HSA015816	1546…1624	D26018	Human	KIAA0039 protein	
3HSA015967	972…1038	S48220	Human	type I 5′-iodothyronine deiodinase	√
3SSC000588	48…114	X76009	Pig	phospholipid hydroperoxide glutathione peroxidase	√
3SSC000385	48…115	L12743	Pig	phospholipid hydroperoxide glutathione peroxidase	√
3OCU000306	42…122	X13837	Rabbit	glutathione peroxidase	√
3CFA000025	592…656	U11762	Dog	type I iodothyronine deiodinase	√
3BTA001004	239…302 +635…697	D25220	Bovine	selenoprotein P like protein	√
3BTA000991	613…676	L10325	Bovine	glutathione peroxidase	√
3BTA000614	49…119	X13684	Bovine	glutathione peroxidase	√
3MMU000179	43…112	AF015284	Mouse	selenoprotein W	√
3MMU000508	510…589	U13705	Mouse	glutathione peroxidase (plasma)	√
3MMU001204	727…779	AB000733	Mouse	AF1q protein	
3MMU002046	163…211	D88611	Mouse	mGCMb protein	
3MMU003946	1888…1947	D26047	Mouse	Pig-a protein	
3MMU004110	4959…5016	Z11981	Mouse	Pvt-1 protein	
3MMU004340	271…344 +654…715	X99807	Mouse	selenoprotein P	√
3MMU006629	63…132	AF045768	Mouse	phospholipid hydroperoxide glutathione DE peroxidase (Gpx4)	√
3MMU006811	785…859	AF068865	Mouse	Delta-like 3 (Dll3) gene	
3MMU006926	266…334	U35623	Mouse	EAT/MCL-1 gene	
3MMU007077	917…976	U51126	Mouse	G-protein coupled inwardly rectifying K+ channel	
3MMU007214	492…556	AB013874	Mouse	Low Density Lipoprotein Receptor Related Protein 4	
3MMU007476	3100…3161	AF091047	Mouse	KH domain RNA binding protein QKI-7B	
3MXX000230	1888…1947	S78188	Mouse	Pig-a protein	
3RNO000413	43…113	U25264	Rat	selenoprotein W	√
3RNO001801	49…122	X07365	Rat	glutathione peroxidase	√
3RNO002117	1303…1362	L03294	Rat	lipoprotein lipase	
3RNO002575	59…128	L24896	Rat	phospholipid hydroperoxide glutathione peroxidase	√
3RNO002594	1213…1279	L01507	Rat	PIT-1-beta	
3RNO003011	268…341 +647…708	M63574	Rat	selenoprotein P	√
3RNO003546	101…167	U94330	Rat	osteoprotegerin	
3RNO003908	46…119	X12367	Rat	glutathione peroxidase I	√

Table 1. *Continued*

3RNO003935	741…813	X57999	Rat	type I thyroxine deiodinase	√
3RNO004160	44…115	M21210	Rat	glutathione peroxidase (GSH-PO)	√
3RNO004269	629…691	AF025819	Rat	Rb binding protein	
3RNO004703	275…343	AF072865	Mouse	thioredoxin reductase 2	√
3XLA000901	458…529	L28111	Frog	iodothyronine 5-deiodinase type III	√
3GGA000765	131…198	X06546	Chicken	smooth muscle myosin heavy chain	
3DRE000181	542…605	U62619	Zebrafish	p21 N-ras oncogene	
3DRE000069	320…383	X61389	Zebrafish	Pax[zf-a]	
3DRE000068	319…381	X63183	Zebrafish	pax-6	
3DRE000042	91…150	L05383	Zebrafish	beta-2-microglobulin	
3TAN000010	359…416	Y15794	*Theileria annulata*	spm1 protein	
3SMA000174	26…91	L14329	*Schistosoma mansoni*	glutathione peroxidase	√
3DME001485	503…563	S70118	Fruit fly	ben gene	
3DME001214	627…687	M16599	Fruit fly	src-related gene	
3DME000747	147…218	L39083	Fruit fly	gliotactin	
3TCO000026	447…509	AB017258	*Turbo cornutus*	indoleamine dioxygenase like-myoglobin	
3CVU000001	17…94	U75914	*Caenorhabditis*	lin-28 gene	
3SOL000017	224…285	U34742	*Spinacia*	24 kDa RNA binding protein	
3PVU000075	133…193	U70530	*Phaseolus*	gibberellin 20-oxidase	
3MSA000108	55…121	X58711	*Medicago sativa*	heat shock protein	
3LLA000010	33…100	X54463	*Larix*	ribulose bisphosphate carboxylase	
3LES000079	49…123	M80604	*Lycopersicon*	beta-1,3-glucanase	
3CPE000008	69…144	D55645	*Cucurbita*	catalase	
3BFI000008	246…306	Y13141	*Bromheadia*	extensin	
3ATH002311	208…274	AF057281	*Arabidopsis*	IBC6 gene	

Table 2. Number of observed (first row) and expected (second row) matches found by PatSearch application in the different taxonomic divisions of UTRdb of the derived histone 3′-UTR stem-loop, IRE and SECIS patterns (see description in Figures 1–3). The expected matches represent the average number of hits over 100 simulated datasets. The third and fourth row of each cell report the significance level (*, <5%; **, <1%; ***, <0.1%; NS, not significant) and the selectivity level (SL = 1 − Exp/Obs; ND, not determinable), respectively. The fifth row reports the fraction of hits actually corresponding to known functional elements (i.e. the fraction of marked hits in Table 1 over total hits). The number of searched 5′- and 3′-UTR sequences in the different collections is also shown in parentheses

Functional element	Human Other		Other mammals		Rodent		Other vertebrate		Invertebrate		Fungi		Plant	
	5′-UTR (7651)	3′-UTR (8775)	5′-UTR (2242)	3′-UTR (2969)	5′-UTR (7825)	3′-UTR (8485)	5′-UTR (3238)	3′-UTR (3976)	5′-UTR (4636)	3′-UTR (5785)	5′-UTR (1034)	3′-UTR (1295)	5′-UTR (7221)	3′-UTR (9875)
Histone3		12 0.21 ± 0.61 *** 99% 12/12		0 0.02 ± 0.14 NS ND 0/0		14 0.24 ± 0.53 *** 100% 14/14		2 0.04 ± 0.20 *** 99% 2/2		2 0.04 ± 0.20 *** 100% 2/2		0 0.01 ± 0.10 NS ND 0/0		0 0.04 ± 0.20 NS ND 0/0
IRE	4 0.66 ± 0.82 *** 84% 4/4		1 0.16 ± 0.44 * 84% 1/1		3 0.83 ± 0.98 ** 72% 3/3		6 0.18 ± 0.41 *** 97% 5/6		6 0.17 ± 0.38 *** 97% 5/6		0 0.03 ± 0.17 NS ND 0/0		0 0.21 ± 0.55 NS ND 0/0	
SECIS		24 19.13 ± 5.03 NS 20% 7/24		8 3.23 ± 2.07 * 60% 8/8		28 12.80 ± 3.59 *** 54% 14/28		6 5.68 ± 2.57 NS ND 1/6		7 6.97 ± 3.33 NS 1% 1/7		0 0.86 ± 1.01 NS ND 0/0		8 9.00 ± 3.81 NS ND 0/8

a statistically significant match with a known gene or protein often provides decisive information.

If no homologous sequences are found to match in the database the task of sequence characterization is much more difficult. In this sense the prediction of putative functional elements in the non-coding portion of the mRNAs may provide significant hints. Furthermore, the prediction of such elements in the UTR of mRNAs coding for already characterized products could also provide crucial information on the possible regulatory pathway controlling gene expression thus guiding further experimental investigations. However, the occurrence of false positives cannot be excluded as we do not take into account the possible effect of the bases upstream and downstream of the matching sequence.

The automatic annotation of the huge number of EST

sequences, now providing an almost complete catalogue of the expressed genes in several organisms, is particularly interesting in this context also because they mostly consist of untranslated sequences.

The PatSearch matcher is thus particularly suitable for searching sequence data for the presence of complex oligonucleotide patterns whose structure has been previously derived from experimental characterization of functional elements.

However, we need to assess the probability that the found pattern hits are reliable candidates for the functional activity under investigation and not just false positives produced by the lack of an adequate knowledge of the actual functional pattern. To this end we developed a simulation procedure which can easily allow for calculation of a given pattern of the number of matches we may expect just by chance. Indeed, the simulation procedure rests on the assumption that random generated sequences reproduce the feature of the natural ones.

It is well known that nucleotide sequences are not random as they display preference or avoidance for specific oligonucleotides (e.g. CpG or TpA depletion). For this reason they are better modeled taking into account their oligonucleotide composition and not simply their base composition. In particular, there is some evidence that natural sequences can be adequately represented if their dinucleotide composition is taken into account (Stueckle *et al.*, 1990). Therefore, in the simulation we used a first-order Markov chain sequence generator which retains in the simulated sequences the dinucleotide composition of real sequences known to be strongly deviating from random expectation.

Results shown in Table 2 provide evidence that this is a rather conservative approximation as the simulation procedure predicts a larger number of false positives (i.e. 0.6, 2.2 and 57.7 for histone stem-loop, IRE and SECIS elements respectively) than those we can estimate in real sequence data (i.e. 0, 2 and 40 respectively). In addition, the occurrence of the functional elements considered here to test the method are found significant in most of the expected cases. Namely, the IRE is to be found occurring significantly only in animal 5′-UTRs, where it is known to have functional activity, but not in plant and fungi 5′-UTRs, where it is known that the regulation of the expression of genes involved in iron metabolism is not IRE-mediated. Analogously, the histone stem-loop and the SECIS elements are found significantly over-represented just in the 3′-UTRs, as expected, but not in the 5′-UTRs (data not shown).

The selectivity level, which can be easily calculated by the simulation procedure may thus provide a reliable estimate of the probability that a sequence region matching with a known consensus pattern is a good candidate for the functional activity under exam thus providing useful indications for further investigations.

Acknowledgements

G.P and S.L. were supported by MURST and EU grant ERB-BIO4-CT960030. M.D. was supported by the Office of Biological and Environmental Research, US Department of Energy, under Contract W-31-109-Eng-38. The authors would like to thank Ross Overbeek for his encouragement in this work and one anonymous referee for several useful comments.

References

Burset,M. and Guigo,R. (1996) Evaluation of gene structure prediction programs. *Genomics*, **34**, 353–367.

D'Souza,M., Larsen,N. *et al.* (1997) Searching for patterns in genomic data. *Trends Genet.*, **13**, 497–498.

Fickett,J.W. and Hatzigeorgiou,A.G. (1997) Eukaryotic promoter recognition. *Genome Res.*, **7**, 861–878.

Gelfand,M.S. (1995) FANS-REF: a bibliography on statistics and functional analysis of nucleotide sequences. *Comput. Appl. Biosci.*, **11**, 541.

Hentze,M.W. and Kuhn,L.C. (1996) Molecular control of vertebrate iron metabolism: mRNA-based regulatory circuits operated by iron, nitric oxide, and oxidative stress. *Proc. Natl Acad. Sci. USA*, **93**, 8175–8182.

Hubert,N., Walczak,R. *et al.* (1996) RNAs mediating cotranslational insertion of selenocysteine in eukaryotic selenoproteins. *Biochimie*, **78**, 590–596.

Snyder,E.E. and Stormo,G.D. (1995) Identification of protein coding regions in genomic DNA. *J. Mol. Biol.*, **248**, 1–18.

Sternberg,M.J., Bates,P.A. *et al.* (1999) Progress in protein structure prediction: assessment of CASP3. *Curr. Opin. Struct. Biol.*, **9**, 368–373.

Stueckle,E.E., Emmrich,C. *et al.* (1990) Statistical analysis of nucleotide sequences. *Nucl. Acids Res.*, **18**, 6641–6647.

Werner,T. (1999) Models for prediction and recognition of eukaryotic promoters. *Mamm. Genome*, **10**, 168–175.

Williams,A.S. and Marzluff,W.F. (1995) The sequence of the stem and flanking sequences at the 3′ end of histone mRNA are critical determinants for the binding of the stem-loop binding protein. *Nucl. Acids Res.*, **23**, 654–662.