BIOINFORMATICS

Vol. 19 no. 3 2003, pages 327–335
DOI: 10.1093/bioinformatics/btf868

Towards a computational model for −1 eukaryotic frameshifting sites

Michaël Bekaert[1], Laure Bidou[1], Alain Denise[2,3], Guillemette Duchateau-Nguyen[2], Jean-Paul Forest[3], Christine Froidevaux[3,*], Isabelle Hatin[1], Jean-Pierre Rousset[1] and Michel Termier[2]

[1]Génétique Moléculaire de la Traduction, [2]Bioinformatique des Génomes, Institut de Génétique et Microbiologie (IGM), UMR CNRS 8621 and [3]Laboratoire de Recherche en Informatique (LRI), UMR CNRS 8623, Université Paris-Sud, 91405 Orsay Cedex, France

Received on February 1, 2002; revised on June 6, 2002; accepted on August 27, 2002

ABSTRACT

Motivation: Unconventional decoding events are now well acknowledged, but not yet well formalized. In this study, we present a bioinformatics analysis of eukaryotic −1 frameshifting, in order to model this event.

Results: A consensus model has already been established for −1 frameshifting sites. Our purpose here is to provide new constraints which make the model more precise. We show how a machine learning approach can be used to refine the current model. We identify new properties that may be involved in frameshifting. Each of the properties found was experimentally validated. Initially, we identify features of the overall model that are to be simultaneously satisfied. We then focus on the following two components: the spacer and the slippery sequence. As a main result, we point out that the identity of the primary structure of the so-called spacer is of great importance.

Availability: Sequences of the oligonucleotides in the functional tests are available at http://www.igmors.u-psud.fr/rousset/bioinformatics/

Contact: bekaert@igmors.u-psud.fr; jpforest@lri.fr; chris@lri.fr

INTRODUCTION

The universality of the genetic code is the initial step of automatic determination for hypothetical open reading frames (ORFs) using very simple methods, such as seeking long, terminator-less phases. However, translation machinery appears capable of decoding not only the classical genetic code but also several kinds of signalling patterns embedded in the mRNA (Gesteland *et al.*, 1992).

Three major forms of recoding have been identified: stop codon *readthrough* by the ribosome; ribosomal *frameshifting*, where the ribosome slips either forward or backward; and ribosome *hopping* where dozens of nucleotides on the message can be skipped by the decoding machinery (Gesteland and Atkins, 1996).

In the present work, we focus on −1 frameshifting. Most of these events are found in viruses and transposons, where they serve to produce the replicase domain needed for the life cycle. Enhancing or reducing the effectiveness of the mechanism can dramatically influence virus viability (Dinman *et al.*, 1998). Very few cellular genes using −1 frameshifting are presently known: the *dnaX* gene of *Escherichia coli* (Tsuchihashi and Kornberg, 1990), the cdd gene of *Bacillus subtilis* (Mejlhede *et al.*, 1999) and, more recently, the Edr gene in mice (Shigemoto *et al.*, 2001). To date, there is no general method to identify such genes.

The genetic information carried by genes expressed through a frameshifting event is, by definition, out of frame. Therefore, these genes could be annotated as non-coding (Medigue *et al.*, 1999). The development of molecular approaches has permitted the demonstration that −1 frameshifting is correlated with the presence of specific signals on the coding sequence, which in turn has led to the design of an initial model.

The goal of this research is to establish a program to identify new genes that use −1 frameshifting for their expression. Our present objective is to improve the known computational model of frameshifting sites in eukaryotic viruses. For this purpose, we use a combination of bioinformatics methods, computer science concepts and biological experimentation. This paper presents our approach and our initial results towards the conception of a more refined computational model.

BIOLOGICAL MODEL AND STATE OF THE ART

The current model for eukaryotic frameshift sites consists of two main components: a *slippery site*, which

*To whom correspondence should be addressed.

mechanically promotes frameshifting, and a *stimulatory structure* which probably acts by pausing the ribosome (Jacks and Varmus, 1985; Farabaugh, 1996; Tu *et al.*, 1992; Somogyi *et al.*, 1993; Lopinski *et al.*, 2000; Kontos *et al.*, 2001). These signals are carried by the mRNA and superimposed on the coding sequence. Although the presence of the slippery site is sufficient to induce a low but biologically significant level of frameshifting, that of the stimulatory structure is not (Kollmus *et al.*, 1996). The short sequence between these two components is called the *spacer* and is denoted SP (Figure 1).

1. The *slippery site*, denoted H (for Heptamer), is the place where the ribosome actually shifts. It is an heptameric sequence conforming to the motif X XXY YYZ (spaces display boundaries of codons in frame 0). The special arrangement of the nucleotides in the heptameric site allows the two tRNAs in both frames (0 and −1) to be paired: XXY and XXX for the tRNA in the P-site of the ribosome, YYZ and YYY for the tRNA in the A-site.
2. The *stimulatory structure*, denoted E (for Enhancer), increases the probability of 5′ ribosomal movement. It can be either a single stem-loop or a pseudoknot. The following subsequences can be distinguished:
   (a) **ES1.5**′, first stem, 5′-arm;
   (b) **EL1**, beginning of first loop;
   (c) **ES2.5**′, second stem, 5′-arm;
   (d) **EL1**′, end of first loop;
   (e) **ES1.3**′, first stem, 3′-arm;
   (f) **EL2**, second loop;
   (g) **ES2.3**′, second stem, 3′-arm.

Parts ES2.5′, EL2 and ES2.3′ are present only in the case of a pseudoknot.

For each subsequence of this model, the following features are of interest: (i) the length; (ii) the identity of the base at a given position; (iii) the number of occurrences of a given base; and (iv) the presence of a simple stem-loop or of a pseudoknot.

Experimental analyses have shown the influence of modifying some of these features on frameshifting efficiency. They have demonstrated that the following features are relevant:

(a) For the heptamer, the identity of Y and of Z: in the canonical model, Y is either an A or a U, Z cannot be a G and there is no constraint on X. Some variants are known. In particular, the three X can differ (we will write X1, X2 and X3 when necessary; Brierley *et al.*, 1992).

(b) For the first stem (ES1), the length and the G–C pair number (Brierley *et al.*, 1991).

(c) For the spacer, the length must be taken into account (Napthine *et al.*, 1999).

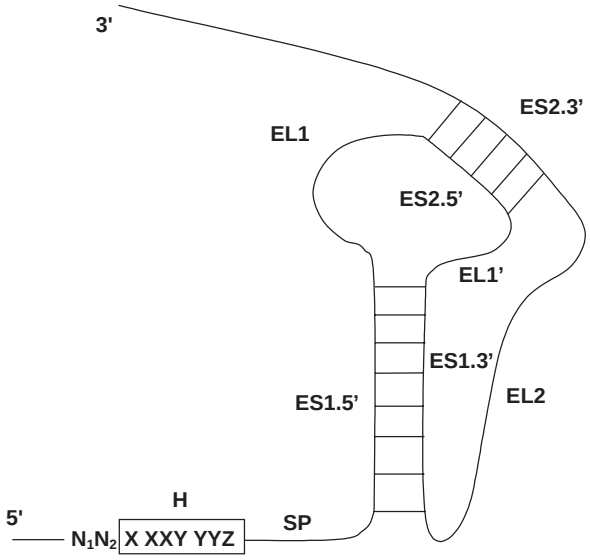

Fig. 1. Labels of subsequences collectively forming the frameshifting signal.

Moreover, pseudoknots are more efficient than stem-loops (Brierley, 1995).

Two attempts to model frameshifting sites were previously made. (Hammell *et al.*, 1999) proposed a model whose main parameters are the structure of H, constraints of lengths, and numbers of pairings in the stems. Liphardt investigated the possibility of using stochastic context-free grammars (SCFG) to model the stimulatory structure (Liphardt, 1999). Both approaches led to models which take into account the main known parameters of the phenomenon. However, the programs based on these models, when applied to entire genomes, found too many false positives. This may be due to two reasons. Firstly, the number of parameters to be considered in each site is large, and therefore difficult to handle 'by hand'. Secondly, the relatively small amount of data (known frameshift sites in wild viruses and mutants) is insufficient to lead to an accurate model of the phenomenon. Nonetheless, these studies constitute a significant step towards modelling frameshifting sites in order to design a prediction tool. In particular, they demonstrate that use of a stochastic model is rather promising, and that filtering constraints on the current model are necessary to increase the efficiency of any search program.

METHODOLOGY

In order to deal with the large number of possibly relevant parameters, we chose to adopt a strategy based on bioinformatics and computational methods. Moreover, since the articles of Hammell *et al.* (1999) and Liphardt,

more sites have been studied by biologists, therefore more data are available.

The general methodology is summarized in the following 'cyclic scheme':

(1) Take a set of sequences that induce frameshifting with a known efficiency level. It will be used as a training set to learn a proper description of the frameshifting event. We can consider this either as a binary event (it occurs or not) or as an event occurring with a given rate. The data representation of the sequences must take into account the consensus organization and the properties known or supposed to be relevant in the frameshifting process.

(2) Refine the model: For this purpose, we use machine learning approaches (supervized or not), associated with classical bioinformatics methods. The aim is to discover new properties shared by frameshifting sites that belong to the same class or have a pre-determined rate.

(3) Test the model: According to the new properties, design a set of sequences that conform to the model. This can be done in two ways: (i) *ab initio* designing new sequences which do not exist in any organism; (ii) using the model to find sequences in genomic databases. Predict their respective classes (or their effective rates) according to the model, and then biologically evaluate their functionality.

(4) Evaluate the model by comparing predictions and experimental results, and modify it if necessary. For instance, some attributes may be added. The cycle can be restarted, with the new sequences added to the learning set. When the model is considered to be sufficiently reliable, it will be used to construct an effective prediction tool.

MATERIALS AND METHODS

Sequences

The sequences under study come from the literature (Brierley *et al.*, 1992; Brierley, 1995; Marczinke *et al.*, 1998; Napthine *et al.*, 1999; ten Dam *et al.*, 1994, 1995; Kim *et al.*, 1999) as well as from electronic resources: Recode (Baranov *et al.*, 2001) and PseudoBase (van Batenburg *et al.*, 2000). A total of 27 wild-type frameshifting sites and 196 mutant sequences were used for computational work. Biological studies were performed on the avian coronavirus, *Infectious bronchitis virus* (IBV) with a minimal pseudoknot (Brierley *et al.*, 1992) (noted IBV.m) and gag/pro frameshifting site of wild type simian retrovirus 1 (ten Dam *et al.*, 1994) noted SRV.wt [†].

Computational methods and learning systems

Each frameshifting site is composed of several subsequences characterized by specific properties. These properties are formalized by attributes that measure some of their different aspects. We consider that an attribute can be of three types: (1) *numerical*: e.g. the G–C pair number in the first stem; (2) *Boolean*: e.g. Y = Z equality in H (possible values: true or false) or (3) *categorical*: e.g. Y identity in H (possible values: A, C, G or U).

We described the sequences with approximately 120 attributes. We then used a machine learning system to identify the relevant attributes and their values such that a given sequence induces efficient frameshifting. However, using so many attributes and relatively few sequences has two limitations: it is computationally expensive and it might decrease the learning performance. In fact, irrelevant attributes deteriorate learning (as expected) but so might redundant ones. To rectify this, we chose a simple decision tree algorithm to perform a selection. We used Weka's implementation (Witten and Frank, 2000) of C4.5 (Quinlan, 1993).

We were interested in learning the frameshifting concept (the target concept). Initially, we simply determined whether the phenomenon would occur or not. We thus considered the binary concept *efficient_frameshifting*. We did not take into account the actual frameshifting rate, but used it to sort the sequences in order to obtain examples (sequences inducing efficient frameshifting) and counter-examples (sequences with low frameshifting efficiency) of the target concept. Since we were interested in defining a 'frontier' between examples and counter-examples, the latter had to be as close as possible to the former.

We know that all sequences do not induce frameshifting in the same way (Giedroc *et al.*, 2000). For example, in some sequences, long stems (hence stable secondary structures) promote efficient frameshifting (Napthine *et al.*, 1999), while in some others bent stems do so (Chen and Tinoco, 1995; Kang *et al.*, 1996; Chen *et al.*, 1996). This can be described with several rules of the form:

R: '*if* condition C1 *and* condition C2 *and* ... *and* condition Ck *then* efficient_frameshifting', in which conditions specify relevant attributes and their values for the concept *efficient_frameshifting* to be satisfied. As a unique rule is not sufficient to cover all the cases, we look for a disjunction of rules of this form (disjunctive learning). Under such rules, efficient frameshifting occurs if at least one of them is satisfied. If a given sequence satisfies the conditions of one rule, it is a good candidate for an actual frameshifting site. Note that we will have only sufficient conditions, not necessary ones, and that a given sequence can satisfy several rules. Moreover, we do not attempt a description of the cases where frameshifting does not occur.

We chose GloBo (Torre, 2000) as a disjunctive learning tool because it performed well on a problem similar to ours (PTE Challenge, Srinivasan *et al.*, 1999). The intuition underlying the GloBo algorithm is as follows: each example is used as a seed to gather the largest possible subset 'around' it without encompassing any

[†] http://www.igmors.u-psud.fr/rousset/bioinformatics/

counter-examples (thus yielding a correct subsets). Once all the subsets have been built (there are as many subsets as examples), a collection of a few subsets is selected such that all examples are covered. We keep only a minimal collection of such sets in order to obtain few rules, which allows for more intelligibility of the target concept. Each subset is associated to a rule. The algorithm is stochastic, which means many correct subsets are tested before the algorithm computes the actual result. In practice, it seems important to cover each known example with at least one rule, even at the expense of covering a few counter-examples (called false positives). Of course, the rules obtained will also be used as predictions (see **Methodology**, step 4) and only experimentation is able to establish the veracity of the prediction.

In order to determine the precise level of frameshifting, we used classical tools that deal with quantitative prediction, such as regression trees (Breiman *et al.*, 1984). They combine both decision tree and regression techniques. Regression trees are like decision trees, except that each leaf is labelled with a number that is the average of the values of the data that reach the leaf and the splitting attributes are chosen to minimize the intrasubset variation in the class values down each branch (Witten and Frank, 2000). The label of each leaf represents the average value of the target concept for the data belonging to it.

Biological materials and methods

We chose the yeast *Saccharomyces cerevisiae* as an experimental system for the following reasons: (i) it is a simple eukaryotic model; (ii) its genome has been fully sequenced (Goffeau *et al.*, 1996); (iii) its translational machinery is able to perform −1 frameshifting (Dinman *et al.*, 1991); and (iv) it is well suited to reverse genetics experiments.

Strain and medium The *S. cerevisiae* FY1679-18B strain (his3-Δ200, trp1-Δ63, ura3-52, leu2-Δ1 MATα; Dujon *et al.*, 1994) was used in this study. Cultures were grown in YNB medium (0.67% Yeast Nitrogen Base, 2% glucose) under standard growth conditions.

Plasmids Constructs of each sequence tested were obtained by insertion of double stranded oligonucleotides into the *MscI* cloning site of the pAC99 plasmid, a centromeric vector carrying the LEU2 selective marker (Figure 2; Bidou *et al.*, 2000). The *MscI* cloning site is present at the junction of a *lacZ-luc* fusion gene.

Enzymatic activities and frameshifting frequency Reporter plasmids were transferred into yeast strains using the lithium acetate method. In each case at least three transformants cultivated in the same conditions were assayed as previously described. Frameshifting frequency expressed as percentage was calculated by dividing the luciferase/β-galactosidase ratio obtained from each test construct by the same ratio obtained with an in-frame control construct (Bidou *et al.*, 2000).

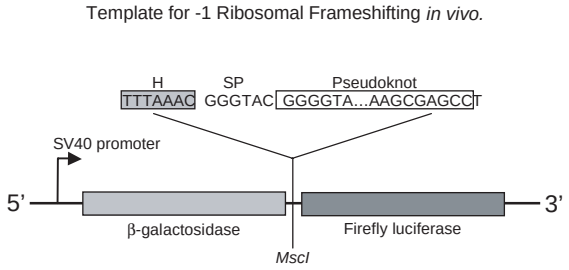

Fig. 2. Template constructed for frameshifting quantification. The tested sequence is localized at the junction between *lacZ/luc* fusion gene. The *in vivo* template is transcribed using SV40 promoter.

RESULTS

Formalizing complete frameshifting sites

We used only pseudo-knotted sites, as they have been more thoroughly studied experimentally than others.

The attributes we used were either new or already known to be relevant. The new attributes we considered are as follows (the others are given in the section **Biological model and state of the art**):

— For the slippery sequence H, the value of each base and the equalities X1 = X2, X1 = X3, X2 = X3, X3 = Y and Y = Z.

— For each other subsequence the nucleotidic composition (number and percentage of each base).

The decision tree method was used on the whole set of sequences. It identified a subset of attributes that are sufficient for correctly classifying almost all sequences. 139 examples and 57 counter-examples were used.

Let |M| be the length of the subsequence M, $|M|_B$ the number of B bases in M and $\%M_B$ the percentage of B in M. Those attributes are (see Figure 1):
Heptamer H: $|H|_A$, $|H|_C$, X1 base, Y base, Z base, X1 = X2, and X3 = Y equalities;
Spacer SP: $|SP|_C$, $|SP|_U$;
First loop EL1 part: |EL1|, $|EL1|_C$, $|EL1|_U$;
First loop EL1′ part: |EL1′|;
Second loop EL2: |EL2|, $|EL2|_A$, $|EL2|_C$, $|EL2|_U$;
First stem ES1: the G-C pair number denoted $|ES1|_{G-C}$, $|ES1.5'|_G$, $\%ES1.5'_G$, $|ES1.3'|_U$;
Second stem ES2: $|ES2.5'|_C$, $|ES2.5'|_G$, $|ES2.5'|_U$, $|ES2.3'|_A$.

This selection step allowed us to reduce the number of attributes from 120 to 25. We added to them a few attributes that seemed to us biologically relevant.

Note that even using the 120 attributes does not allow to correctly classify all the examples and counter-examples. Namely, some examples and counter-examples have the same values for those attributes and cannot be distinguished from one another. In the following, we left the problematic sequences aside. We thus worked with 135 examples and 51 counter-examples.

For our purposes, sequences having a frameshifting rate above 5% were considered to be examples and those having a frameshifting rate below 2% were considered to be counter-examples: sequences whose efficiency is between 2% and 5% were left aside to avoid an overly arbitrary frontier between examples and counter-examples (see **Methodology** step 1). Varying the boundaries did not significantly modify the results.

GloBo ran 25 times using only the subset of attributes selected. Before each run, the training set (70% of data) and the test set (30% remaining) were chosen randomly. Each execution gave between 8 and 13 rules. Some of these rules bore a strong similarity with one another and could even be identical in distinct runs.

We focused on the following two rules (see **Methodology** step 2) because they were almost invariant from one run to another and covered a large amount of examples:

R1: *if* $X1 \neq C$ *and* $X1 \neq U$ *and* $|H|_A \leqslant 5$
and $|ES1|_{G-C} \in [6, 9]$ *and* $|ES2.5'|_U \leqslant 1$
then efficient_frameshifting.

This rule covers about 44% of the examples (training and test set) and no counter-examples.

R2: *if* $Y \neq G$ *and* $Z \neq G$ *and* $|H|_A \leqslant 4$
and $|SP|_C \geqslant 1$
and $\%ES1.5'_G \leqslant 65$ *and* $|ES1|_{G-C} \geqslant 6$
then efficient_frameshifting.

This rule covers about 33% of the examples (training and test set) and no counter-examples.

Note that these rules are not mutually exclusive: some examples are covered by both.

The presence of the attributes dealing with ES1 and ES2 in the rules is not surprising, as they are linked to the stability of the stems and hence to the stability of the pseudoknot. Interestingly, R1 and R2 focus on a rich G–C pair content of ES1. Moreover R2 gives an upper limit on the G content of ES1.5′. The other attributes discriminate between different families of mutants: R1 covers most SRV.wt-like sequences whereas R2 covers most IBV.m-like ones. Together they cover 65% of the examples. We therefore clearly obtain two rules of a disjunctive description of the frameshifting process. Moreover, the conjunctive form of each rule implies that the constraint links the values of several attributes together. Previously the ranges (that is, the sets of relevant values) of the attributes were determined independently.

To test these rules (see **Methodology** step 3), we designed *ab initio* sequences that satisfy them and that remain in the same general context: R1 was thus tested on SRV mutants (see Figure 3a) whereas R2 was tested on IBV mutants (see Figure 3b). We chose to focus on the stems in both series of constructs because they are known to be critical for frameshifting. These two figures give only the elements that were changed in the wild type. ES1 was modified in length for SRV mutants and in composition for IBV mutants. ES2 was also modified in IBV mutants. Given a sequence satisfying a rule, we wanted to verify whether mutants of this sequence that only differ from it in attributes that do not occur in the rule still promote frameshifting (of course these mutants satisfy this rule too).

The results in Figure 3 show that the efficiency level for SRV mutants was relatively stable (between 11% and 18%) and that most mutants were more efficient than the wild type. Concerning IBV mutants, the results show that, although the efficiency level varied from 9% to 25%, all the tested constructs were able to drive a significant frameshifting. In comparison, frameshifting efficiency of defective mutants, IBV pKA9 and IBV pKA96 (Napthine *et al.*, 1999), was 1.2% in our experimental system, similar to that obtained *in vitro* by (Napthine *et al.*, 1999) ($\leqslant$2%). However, important variations were observed between constructs showing the same proportion of G–C versus A–U pairs and differing only by their repartition (compare for example constructs IBV.s1 and s3). This is probably related to the three-dimensional structure of the artificial pseudoknot (Farabaugh, 1996).

Overall, these results demonstrate that frameshifting indeed occurs on constructs that follow one rule, increasing our confidence in the rules (see **Methodology** step 4).

Spacer

A detailed study of the spacer of 27 sites from 24 different viruses (see **Methodology** step 1) has revealed a striking regularity, in spite of the variability of its length[‡].

Notably, it was clear that in efficient −1 frameshifting sites specific nucleotides are found at given positions in the spacer. We verified that similarities are not due to homologies by performing pairwise alignments of nucleic and protein sequences from the 24 different viruses.

We focused on the three first positions of the spacer, where the consensus is particularly clear: the first nucleotide is either a G or a U; the second is either a G or an A, with three known exceptions (including the case of RSV where the spacer is only one nucleotide long) and eventually, at the third position, one finds also a G or an A, with three exceptions. We measured the frameshifting efficiency of a number of spacer mutants, based on the

[‡] http://www.igmors.u-psud.fr/rousset/bioinformatics/

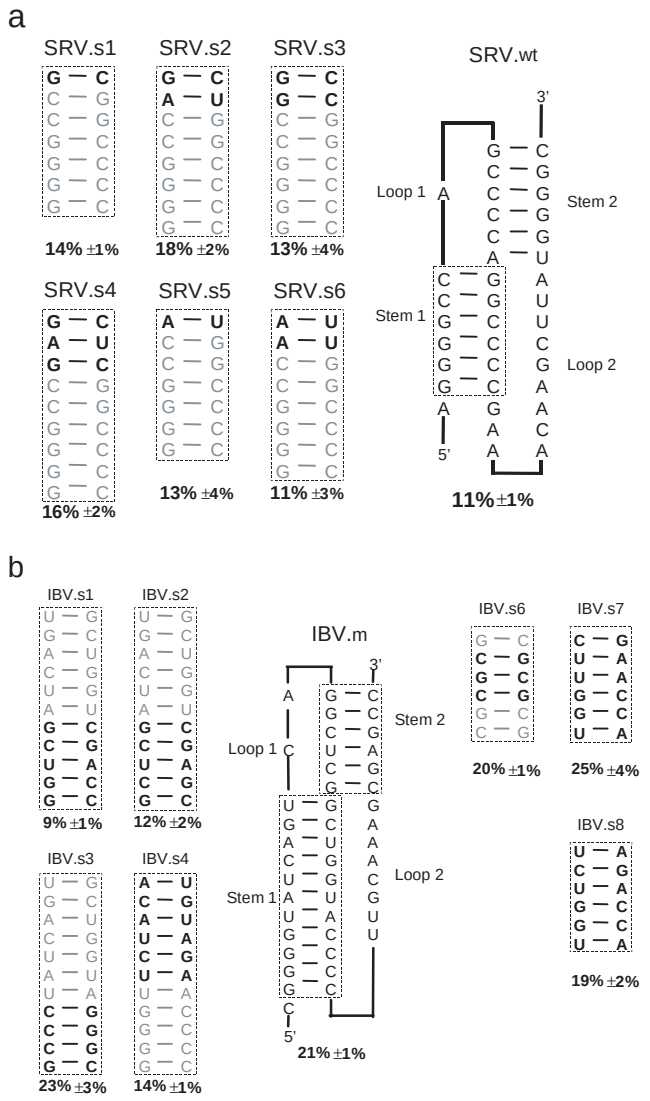

Fig. 3. Secondary-structure of the SRV (a) and IBV (b) pseudoknots. Dashed lines surround the region tested. Modified nucleotides are in bold and the frameshifting rate is indicated below. In the IBV mutants, stem 1 was modified only in IBV.s1 to IBV.s4 and stem 2 was modified only in IBV.s6 to IBV.s8.

wild-type IBV spacer. We systematically modified each of the first three nucleotides in turn. The results are in Table 1.

For each of the three positions, variation of a single base induces significant variations of the frameshifting level. An up to 4-fold difference was observed between constructs, but the extent of the effect seems to be more important for the first two nucleotides than for the third. Although the effect of the spacer sequence is important, the frameshifting efficiency never decreases below the 5% limit.

Table 1. Derived mutations of the three first nucleotides of the spacer (SP). Wild-type spacer is given by IBV.m (GGGUAC)

Construct	Spacer	FS −1 rate
IBV.m	**GGG**UAC	21% ± 1%
IBV.sp1	**A**GGUAC	15% ± 2%
IBV.sp2	**U**GGUAC	25% ± 4%
IBV.sp3	**C**GGUAC	6% ± 1%
IBV.sp4	G**A**GUAC	7% ± 2%
IBV.sp5	G**U**GUAC	19% ± 2%
IBV.sp6	G**C**GUAC	17% ± 1%
IBV.sp7	GG**A**UAC	15% ± 3%
IBV.sp8	GG**U**UAC	19% ± 2%
IBV.sp9	GG**C**UAC	11% ± 1%

Slippery sequence

Brierley *et al.* analyzed a large number of slippery heptamer sequences in the context of the IBV pseudoknot structure, in order to better understand the respective role of these two elements (Brierley *et al.*, 1992). Among the 64 possibilities for the X XXY YYZ heptamer, 44 mutations were created, the remaining 20 were thought to be non-functional.

We analyzed the frameshifting efficiency for the 64 mutants obtained from the wild-type IBV, X, Y and Z being successively replaced by A, C, G and U in the heptamer. We assign a level of 0 to the frameshifting level of the 20 sequences discarded (Brierley *et al.*, 1992). The frameshifting efficiency of the sequences measured below 1 were considered as 0.5. The attributes used were the same as above. First, we used a regression tree technique that (i) chooses relevant attributes to split the set of mutants into classes that share common properties of primary structure and have almost the same efficiency level and (ii) calculates the average efficiency level of each class. Note that the classes are not determined *a priori*. We got the following regression tree rules, where AL denotes the average level and s.d. the standard deviation.

If (Y = C *or* Y = G) *then* AL = 1.1 (s.d. 1.8)
If (Y = A *or* Y = U) *and* (Z = G) *then* AL = 2.68 (s.d. 2.8)
If (Y = A *or* Y = U) *and* (Z = A *or* Z = U *or* Z = C) *and* (X = C) *then* AL = 6.53. (s.d. 4.9)
If (Y = A *or* Y = U) *and* (Z = A *or* Z = U *or* Z = C) *and* (X = A *or* X = G *or* X = U) *then* AL = 22.7 (s.d. 9.7)

The order that the rules and their conditions are written in is important: it gives the attributes in decreasing order according to the minimization of the intrasubset variation. These rules confirm results already established by Brierley for IBV (Brierley *et al.*, 1992):

(1) only triplets YYY of A and of U are functional;

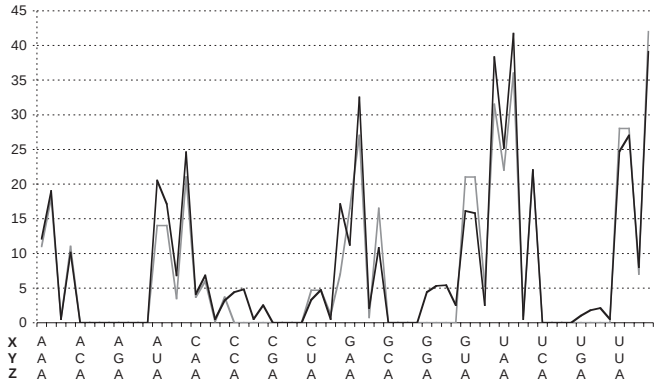

Fig. 4. Frameshifting efficiencies of slippery site mutants in IBV. Comparison between experimental results of Brierley *et al.* (1992) (in black) and the formula above (in grey). The sequences XYC, XYG and XYU lie between XYA and X′Y′A.

(2) the identity of Z in the heptamer is a critical determinant of frameshifting efficiency;

(3) triplets XXX of A, C, G and U are functional but C-triplets are the least slippery.

Moreover, we have a crude approximation of the expected efficiency in each case.

We then used *ad hoc* mathematical tools to show how precisely the frameshifting level depends on the identity of the nucleotides.

Figure 4 presents the frameshifting level for the 64 mutants of the heptamer (they are sorted in lexicographical order). It reveals a strong regularity in the frameshifting levels. Not only can one note a pseudoperiod in the values (as expected given the known rules about $Y = A$ or $Y = U$), but also the main peaks remarkably line up.

The regression equation given below (see the grey graphical on Figure 4) is an approximation function that expresses as closely as possible the 64 sequence levels. (In this equation, expressions in square brackets must be evaluated at 1 if the conditions are satisfied, at 0 otherwise).

$F(XXXYYYZ) = F1xF2xF3$, with

$F1 = (1/3)[X = C] + [X = A] + (3/2)[X = G] + 2[X = U]$

$F2 = [Y = A \text{ or } Y = U]$

$F3 = 11 + 3[Y = U] - 10.5[Z = G] + 7[Y = A][Z = C] + 7[Y = U][Z = U]$.

This equation reveals an unknown characteristic, namely the very interesting role of the XXX triplet expressed in F1. It acts as a multiplicative factor. Thus it appears that the identity of X influences the frameshifting efficiency following the order: $C < A < G < U$.

DISCUSSION

This study validates the methodological approach used to refine the current model. Specifically, our learning method allowed us to identify new features that may be involved in frameshifting and to specify the range of the values of the corresponding attributes. As we saw above (see **Results**), the set of attributes used was not sufficient to distinguish all the examples from the counter-examples. This will lead us to introduce more attributes. Using a disjunctive learning tool such as GloBo led us to two rules which were then verified experimentally. The relevance of their attributes to the mechanism of frameshifting was tested by creating artificial frameshifting sites that follow the rules identified by our bioinformatics approach. The capacity of these sites to induce frameshifting was then assessed *in vivo* in yeast cells. As these two rules do not cover all the examples, other rules should be investigated.

The originality of the obtained rules is derived from their conjunctive form. Namely, each rule provides a set of values that must be assigned to several attributes concerning possibly distinct components of the model.

Our results show that the spacer is involved in the efficiency of frameshifting. Although it has been known for many years that the length of the spacer region is crucial in frameshifting (Brierley, 1995), it has been recently shown that the sequence could also be important in bacterial frameshifting sites (Bertrand *et al.*, 2002). The results presented here demonstrate that this sequence is also important in eukaryotic frameshifting. Different mechanisms could account for this effect: the nucleotides may directly interact with components of the translational machinery (i.e. ribosomal RNA or protein), or indirectly, by codon–anticodon interaction, or through the availability of the corresponding tRNA. In this case, the spacer could not be seen as a sum of individual nucleotides but as a unit. If this is so, the activity of a sequence may not be defined by a single nucleotide but by a suite of nucleotides, and there may be different suites that work well. In order to shed some light on this point, the spacer should be analyzed systematically using a combinatorial approach, similar to a SELEX experiment, that selects the most efficient spacer sequences. Such a procedure, adapted for translational regulation, is available and in use in our laboratory to analyze programmed readthrough (Namy *et al.*, 2001).

Another interesting point is the multiplicative role of the XXX triplet in frameshifting efficiency. This suggests that the core frameshifting signal could in fact be a tetranucleotide and that the XXX triplet is used to modulate the efficiency. It is well known that the bacterial frameshifting mechanism often involves only a tetramer sequence of the form Y YYZ. In eukaryotes also, although simultaneous tandem tRNA slippage is the main mechanism, single slippage has a place, with a low but

significant frequency, at a given site (Jacks *et al.*, 1988; Yelverton *et al.*, 1994).

It is worth noting that this study was only conducted in the IBV context. However, the few experiments that have been done up to now on different kinds of viruses have shown that hierarchies of heptamer efficiency are generally conserved from one virus to another (Farabaugh, 1996). It would be of particular interest to complete the work of Brierley, by testing the remaining 20 heptamers in eukaryotic models and to test the frameshifting efficiency directed by the 64 possible heptamers and the 16 possible tetramers in a bacterial model. These 16 tetramer sequences should also be tested in eukaryotic cells, to assess whether some of them allow efficient frameshifting.

ACKNOWLEDGEMENTS

The authors are deeply grateful to Celine Fabret for critical reading of this manuscript and helpful suggestions. We also thank anonymous referees for pertinent suggestions.

This work was supported by a CNES grant on the Preparatory Program Mars Sample Analysis and by a CNRS INRA - INRIA - INSERM Bioinformatics grant.

REFERENCES

Baranov,P., Gurvich,O., Fayet,O., Prere,M., Miller,W., Gesteland,R., Atkins,J. and Giddings,M. (2001) RECODE: a database of frameshifting, bypassing and codon redefinition utilized for gene expression. *Nucleic Acids Res.*, **29**, 264–267. http://recode.genetics.utah.edu.

Bertrand,C., Prere,M.-F., Gesteland,R., Atkins,J. and Fayet,O. (2002) Influence of the stacking potential of the base 3′ of tandem shift codons on −1 ribosomal frameshifting used for gene expression. *RNA*, **8**, 16–28.

Bidou,L., Stahl,G., Hatin,I., Namy,O., Rousset,J.-P. and Farabaugh,P. (2000) Nonsense-mediated decay mutants do not affect programmed −1 frameshifting. *RNA*, **6**, 952–961.

van Batenburg,F., Gultyaev,A., Pleij,C. and Olienhoek,J. (2000) PseudoBase: a database with RNA pseudoknots. *Nucleic Acids Res.*, **28**, 201–204. http://wwwbio.LeidenUniv.nl/∼Batenburg/PKB.html

Breiman,L., Friedman,J.H., Olshen,R.A. and Stone,C.J. (1984) *Classification and Regression Trees*. Wadsworth, Monterey, CA.

Brierley,I. (1995) Ribosomal frameshifting on viral RNAs. *J. Gen. Virol.*, **76**, 1885–1892.

Brierley,I., Rolley,N.J., Jenner,A.J. and Inglis,S.C. (1991) Mutational analysis of the RNA pseudoknot component of a coronavirus ribosomal frameshifting signal. *J. Mol. Biol.*, **220**, 889–902.

Brierley,I., Jenner,A.J. and Inglis,S.C. (1992) Mutational analysis of the 'slippery-sequence' component of a coronavirus ribosomal frameshifting signal. *J. Mol. Biol.*, **227**, 463–479.

Chen,L.X. and Tinoco,I. (1995) The structure of an RNA pseudoknot that causes efficient frameshifting in mouse mammary tumor virus. *J. Mol. Biol.*, **247**, 963–978.

Chen,X., Kang,H., Shen,L.X., Chamorro,M., Varmus,H.E. and Tinoco,I. (1996) A characteristic bent conformation of RNA pseudoknots promotes −1 frameshifting during translation of retroviral RNA. *J. Mol. Biol.*, **260**, 479–483.

ten Dam,E., Brierley,I., Inglis,S. and Pleij,C. (1994) Identification and analysis of the pseudo-knot containing *gag-pro* ribosomal frameshift signal of simian retrovirus −1. *Nucleic Acids Res.*, **22**, 2304–2310.

ten Dam,E.B., Verlaan,P.W. and Pleij,C.W. (1995) Analysis of the role of the pseudoknot component in the SRV-1 *gag-pro* ribosomal frameshift signal: loop lengths and stability of the stem regions. *RNA*, **1**, 146–154.

Dinman,J.D., Icho,T. and Wickner,R.B. (1991) A −1 ribosomal frameshift in a double-stranded RNA virus of yeast forms a gag-pol fusion domain. *Proc. Natl Acad. Sci. USA*, **88**, 174–178.

Dinman,J., Ruiz-Echevarria,M. and Peltz,S. (1998) Translating old drugs into new treatments: ribosomal frameshifting as a target for antiviral agents. *Trends Biotechnol.*, **16**, 190–196.

Dujon,B., Alexandraki,D., Andre,B., Ansorge,W., Baladron,V., Ballesta,J., Banrevi,A., Bolle,P., Bolotin-Fukuhara,M. Bossier,P. *et al.* (1994) Complete DNA sequence of yeast chromosome XI. *Nature*, **369**, 371–378.

Farabaugh,P. (1996) Programmed translational frameshifting. *Microbiological review*, **60**, 103–134.

Gesteland,R. and Atkins,J. (1996) Recoding: dynamic reprogramming of translation. *Annu. Rev. Biochem.*, **65**, 741–768.

Gesteland,R., Weiss,R. and Atkins,J.F. (1992) Recoding: reprogrammed genetic decoding. *Science*, **257**, 1640–1641.

Giedroc,D.P., Theimer,C.A. and Nixon,P.L. (2000) Structure, stability and function of RNA pseudoknots involved in stimulating ribosomal frameshifting. *J. Mol. Biol.*, **298**, 167–185.

Goffeau,A., Barrell,B., Bussey,H., Davis,R., Dujon,B., Feldmann,H., Galibert,F., Hoheisel,J., Jacq,C. and Johnston,M. *et al.* (1996) Life with 6000 genes. *Science*, **274**, 546, 563–567.

Hammell,A.B., Taylor,R.C., Peltz,S.W. and Dinman,J.D. (1999) Identification of putative programmed −1 ribosomal frameshift signals in large DNA databases. *Genome Res.*, **9**, 417–427.

Jacks,T. and Varmus,H. (1985) Expression of the Rous sarcoma virus *pol* gene by ribosomal frameshifting. *Science*, **230**, 1237–1242.

Jacks,T., Power,M., Masiarz,F., Luciw,P., Barr,P. and Varmus,H. (1988) Characterization of ribosomal frameshifting in HIV-1 gag-pol expression. *Nature*, **331**, 280–283.

Kang,H., Hines,J.V. and Tinoco,I. (1996) Conformation of a non-frameshifting RNA pseudoknot from mouse mammary tumor virus. *J. Mol. Biol.*, **259**, 135–147.

Kim,Y.-G., Su,L., Maas,S., O'Neill,A. and Rich,A. (1999) Specific mutations in a viral RNA pseudoknot drastically change ribosomal frameshifting efficiency. *Proc. Natl Acad. Sci. USA*, **96**, 14234–14239.

Kollmus,H., Hentze,M. and Hauser,H. (1996) Regulated ribosomal frameshifting by an rna-protein interaction. *RNA*, **2**, 316–323.

Kontos,H., Napthine,S. and Brierley,I. (2001) Ribosomal pausing at a frameshifter RNA pseudoknot is sensitive to reading phase but shows little correlation with frameshift efficiency. *Mol. Cell. Biol.*, **21**, 8657–8670.

Liphardt,J. (1999) *The mechanism of −1 ribosomal frameshifting: experimental and theoretical analysis*, PhD thesis, Churchill College, Cambridge.

Lopinski,J., Dinman,J. and Bruenn,J. (2000) Kinetics of ribosomal pausing during programmed −1 translational frameshifting. *Mol. Cell. Biol.*, **20**, 1095–1103.

Marczinke,B., Fisher,R., Vidakovic,M., Bloys,A. and Brierley,I. (1998) Secondary structure and mutational analysis of the ribosomal frameshift signal of Rous sarcoma virus. *J. Mol. Biol.*, **284**, 205–225.

Medigue,C., Rose,M., Viari,A. and Danchin,A. (1999) Detecting and analyzing DNA sequencing errors: toward a higher quality of the *Bacillus subtilis* genome sequence. *Genome Res.*, **9**, 1116–1127.

Mejlhede,N., Atkins,J. and Neuhard,J. (1999) Ribosomal −1 frameshifting during decoding of *Bacillus subtilis* cdd occurs at the sequence CGA AAG. *J. Bacteriol.*, **181**, 2930–2937.

Namy,O., Hatin,I. and Rousset,J.P. (2001) Impact of the six nucleotides downstream of the stop codon on translation termination. *EMBO Rep.*, **2**, 787–793.

Napthine,S., Liphardt,J., Bloys,A., Routledge,S. and Brierley,I. (1999) The role of RNA pseudoknot stem 1 length in the promotion of efficient −1 ribosomal frameshifting. *J. Mol. Biol.*, **288**, 305–320.

Quinlan,J. (1993) *C4.5: Programs for machine learning*. Morgan Kaufmann, San Fransisco.

Shigemoto,K., Brennan,J., Walls,E., Watson,C., Stott,D., Rigby,P. and Reith,A. (2001) Identification and characterisation of a developmentally regulated mammalian gene that utilises −1 programmed ribosomal frameshifting. *Nucleic Acid Res.*, **29**, 4079–4088.

Somogyi,P., Jenner,A.J., Brierley,I. and Inglis,S.C. (1993) Ribosomal pausing during translation of an RNA pseudoknot. *Mol. cell. Biol.*, **13**, 6931–6940.

Srinivasan,A., King,R. and Bristol,D. (1999) An assessment of submissions made to the Predictive Toxicology Evaluation Challenge. In *Proceedings of the Sixteenth International Joint Conference on Artificial Intelligence (IJCAI-99)*, Vol. 1.

Torre,F. (2000) *Intégration des biais de langage à l'algorithme générer-et-tester—Contributions à l'apprentissage disjonctif*, PhD thesis, LRI, Université Paris-Sud, Orsay.

Tsuchihashi,Z. and Kornberg,A. (1990) Translational frameshifting generates the gamma subunit of DNA polymerase III holoenzyme. *Proc. Natl Acad. Sci. USA*, **87**, 2516–2520.

Tu,C., Tzeng,T.-H. and Bruenn,J.A. (1992) Ribosomal movement impeded at a pseudoknot required for frameshifting. *Proc. Natl Acad. Sci. USA*, **89**, 8636–8640.

Witten,I.H. and Frank,E. (2000) *Data mining*. Morgan Kaufmann, San Fransisco, http://www.cs.waikato.ac.nz/ml/weka.

Yelverton,E., Lindsley,D., Yamauchi,P. and Gallant,J. (1994) The function of a ribosomal frameshifting signal from human immunodeficiency virus-1 in *Escherichia coli*. *Mol. Microbiol.*, **11**, 303–313.