# DESIGNING SECONDARY STRUCTURE PROFILES FOR FAST NCRNA IDENTIFICATION

Yanni Sun* and Jeremy Buhler

*Department of Computer Science and Engineering, Washington University, St. Louis, MO 63130, USA*

*Email: (yanni,jbuhler)@cse.wustl.edu*

Detecting non-coding RNAs (ncRNAs) in genomic DNA is an important part of annotation. However, the most widely used tool for modeling ncRNA families, the covariance model (CM), incurs a high computational cost when used for search. This cost can be reduced by using a filter to exclude sequence that is unlikely to contain the ncRNA of interest, applying the CM only where it is likely to match strongly. Despite recent advances, designing an efficient filter that can detect nearly all ncRNA instances while excluding most irrelevant sequences remains challenging.

This work proposes a systematic procedure to convert a CM for an ncRNA family to a *secondary structure profile (SSP)*, which augments a conservation profile with secondary structure information but can still be efficiently scanned against long sequences. We use dynamic programming to estimate an SSP's sensitivity and FP rate, yielding an efficient, fully automated filter design algorithm. Our experiments demonstrate that designed SSP filters can achieve significant speedup over unfiltered CM search while maintaining high sensitivity for various ncRNA families, including those with and without strong sequence conservation. For highly structured ncRNA families, including secondary structure conservation yields better performance than using primary sequence conservation alone.

## 1. INTRODUCTION

Non-coding RNAs (ncRNAs) are transcribed but are not translated into protein. Annotating common ncRNAs, such as tRNAs and microRNAs, as well as non-coding structures like riboswitches in mRNAs, is important because of their functions in many biological processes[1]. The function of an ncRNA is determined not only by its sequence but also by its secondary structure. Exploiting this structural signal can improve ncRNA homology detection[2].

The state-of-the-art method to recognize an ncRNA of known family is to align it to a covariance model (CM). A CM is a stochastic context-free grammar (profile SCFG)[3, 4] that describes an ncRNA family's sequence and secondary structure conservation. Aligning an RNA to a CM uses a probabilistic variant of the well-known CYK parsing algorithm[5]. CM alignment has been implemented in the INFERNAL software suite[2]. In conjunction with a database of CMs, such as Rfam[6], INFERNAL can be used to annotate ncRNAs in genomic DNA.

A major challenge of CM alignment is its high computational cost. Probabilistic CYK is a cubic-time algorithm with a significant constant factor. For example, Weinberg et al. estimated that it would take about 1 CPU year to locate all tRNAs in an 8-Gbase DNA database on a 2.8 GHz Intel Pentium 4 PC[16]. Although CPUs have gotten faster, and INFERNAL implements optimizations designed to lower the cost of CYK, speeding up CM alignment remains a major problem for annotating large sequences or for classifying many sequences into one of the many known ncRNA families.

One approach to accelerate CM alignment is to use a filter to exclude "unpromising" sequences. Sequences that pass the filter have a higher probability of containing the target ncRNA and so are aligned with the full CM. Careful filter design can effectively accelerate pairwise DNA sequence comparison[8–11] as well as alignment of a sequence to a profile hidden Markov model (pHMM)[12, 13].

Several filtering strategies have been proposed to speed up CM search[7, 14–18]. Construction of the Rfam database uses primary sequence comparison with BLAST to exclude sequences unlikely to be part of an ncRNA family[6]. Weinberg and Ruzzo developed a pHMM-based, lossless filtering strategy for arbitrary ncRNA families[7] as well as a faster but lossy strategy that designs a pHMM from a CM for a family[15]. A disadvantage of all these filters is that they forgo the opportunity to exploit RNA structural conservation. Moreover, while pHMMs can be

*Corresponding author.

scanned against a database much more efficiently than CMs, their computational cost remains an issue for large database searches.

Other types of filter exploit RNA structural conservation. Weinberg and Ruzzo used a sub-CM structure [16] to improve the filtering sensitivity for ncRNA families with low sequence conservation. However, the filter design process is expensive (1 to 50 CPU hours per family), and using the resulting filters leads to a slow filtration process. Zhang et al.[17, 20] used the $(k, w)$-stack as the basis for their filter design. It is not clear whether their method can be used to design filters for a large database of ncRNA families because the authors need to choose optimal filters empirically by trying different parameters[20]. In addition, using $(k, w)$-stacks may not be optimal for many families with strong sequence conservation.

Recently, Zhang et al.[18] designed a *chain filter*, based on a collection of short matches to conserved words in a CM, that can sensitively and efficiently identify riboswitch elements. The chain filter does not consider structural conservation either. Moreover, the design of such filters requires specifying score thresholds for matches to the various words in the filter. The procedure for selecting thresholds was not described in Ref. 18, so the design of chain filters appears less than fully automated.

In this work, we describe a robust, efficient approach to fully automated filter design for ncRNA search. We show how to design filters starting from a CM using *secondary structure profiles (SSPs)*, which recognize both primary sequence and secondary structure conservation. The main properties of our filters and filter design program are:

- SSP matching is a simple extension of the standard profile matching algorithm and has linear time complexity;
- Designing SSPs from CMs is efficient;
- SSP-based filters generalize to ncRNA families of all types;
- The match score threshold for an SSP can be automatically computed from its CM, using a practically accurate model-based estimate of its specificity in background DNA sequence.

SSPs were first used in the ERPIN program[19] to characterize RNA signatures. They generalize profiles (a.k.a. position specific score matrices) by incorporating probability distributions for conserved base pairs. The main difference between our data structure and ERPIN's is that our SSP can accommodate gaps inside stacks, such as bulges. Also, we use different methods for SSP design and scanning.

Our method constructs a list of candidate SSPs from a given CM, then uses dynamic programming, first to assign a threshold to each SSP to control its false positive (FP) rate and then to estimate each SSP's sensitivity. The candidate SSP that maximizes sensitivity subject to an upper bound on its FP rate is chosen as the final filter. The sensitivity and FP rate computed via dynamic programming are typically good predictors of a filter's performance on real sequences, so their computation allows us to fully automate selection of SSPs and their associated score thresholds. We extend our filtering strategy to use multiple SSPs to improve the trade-off between sensitivity and FP rate.

Our results demonstrate that automatically designed SSP filters have an average speedup of about 200x over INFERNAL 0.7 without filtration yet detect almost all ($\geq$ 99% of) occurrences of most ncRNA families we tested. For highly structured ncRNA families with limited sequence conservation, such as tRNAs and 5S rRNA, we show that including secondary structure conservation in an SSP yields a better sensitivity/FP rate tradeoff than relying on primary sequence conservation alone.

The remainder of this paper is organized as follows. Section 2 briefly reviews CMs and formally defines SSPs. Section 3 describes how to construct SSPs from a CM and how to evaluate an SSP's performance. In Section 4, we first demonstrate the advantages of using SSPs versus primary conservation profiles on ncRNA families drawn from BRAliBase III[21]. We then measure the sensitivity, FP rate, and speedup obtained for 233 ncRNA families from the Rfam database. We also compare SSPs with other types of filter. Finally, Section 5 concludes and suggests directions for future work.

## 2. CMS AND SSPS

This section briefly reviews CMs and formally defines our secondary structure profiles (SSPs). To distin-

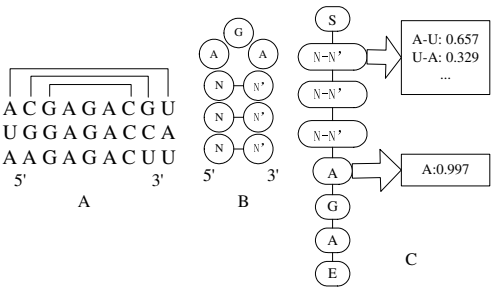


**Fig. 1.** (A) ungapped alignment of RNAs with three conserved base pairings; (B) corresponding secondary structure; (C) CM describing the structure.

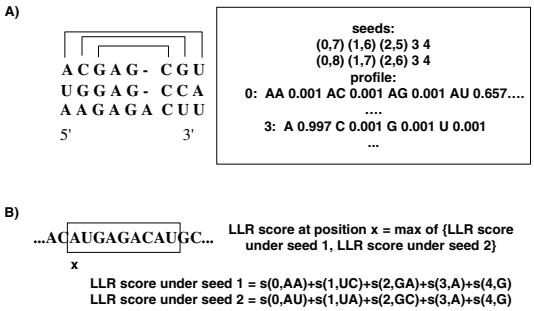


**Fig. 2.** (A) gapped alignment of RNAs with three base pairing interactions, with a corresponding SSP. Two seeds handle the possibility of an insertion after position 4. Column 0 may pair with column 7 or 8, resulting in seed pairs $(0, 7)$ and $(0, 8)$; (B) computation of LLR score at offset $x$ in an input sequence.

guish an SSP, which includes structural information, from a profile that describes only the primary sequence conservation at a series of sequence positions, we call the latter a *regular profile* hereafter.

## 2.1. Covariance models

A CM consists of a series of groups of states, which are associated with base pairs and unpaired bases in an RNA structure.

Figure 1 shows the structure of a simple CM built from an RNA multiple sequence alignment annotated with three base-pair interactions. This CM contains start state S, end state E, three states emitting base pairs, and three states emitting unpaired bases. Each state is associated with an emission distribution; for example, the top paired state emits A-U and U-A pairs only. States are connected by transitions with associated probabilities. All transitions have probability 1 in the example, but insertions and deletions in a structure can be modeled by states with lower-probability in-transitions. The key observations for our work are that (1) the emitting states of a CM encode both primary sequence conservation and the locations of paired bases, and (2) the transition probabilities between these states encode how often a given state is present in a randomly chosen path through the CM.

More detailed descriptions of CMs and probabilistic CYK can be found in Ref. 4.

## 2.2. Secondary structure profiles

SSPs augment regular profiles by characterizing base pair frequencies in an RNA structure. Hence, unlike a regular profile, we must tell an SSP which pairs of bases it inspects are expected to be complementary. Figure 2(A) shows an example of an SSP.

An SSP $\mathcal{P}$ consists of two components. The first component contains one or more *seeds* that designate paired and unpaired base positions. A seed $\pi$ of length $\ell$ is an ordered list of $\ell$ single or paired values. A single value $\pi_i$ denotes that the $i$th base relative to the start of the SSP is unpaired, while a pair of values $(\pi_i^1, \pi_i^2)$, with $\pi_i^1 < \pi_i^2$, indicates that positions $\pi_i^1$ and $\pi_i^2$ are paired. To describe common variations in the locations of paired bases caused by insertion and deletion, an SSP may include multiple seeds. Note that the set of positions described by a seed need not be contiguous in the sequence.

The second component of an SSP describes emission distributions for bases or base pairs in the alignment. For example, the probability of A-U at offsets specified by the first element of both seeds is $\mathcal{P}_{0,AU} = 0.657$. Note that all seeds have the same length as the number of rows in the profile, since each SSP has only one profile.

During search, an SSP $\mathcal{P}$ is aligned with a sequence $S$ at each possible offset $0 \leq x < |S|$. The hypothesis that the bases of $S$ matched by some seed $\pi$ at offset $x$ are generated from the emission distributions of $\mathcal{P}$ is compared to the null hypothesis that the positions come from a background model $\mathcal{P}^0$, using a log likelihood ratio (LLR). Starting at any offset $x$ in $S$, we extract bases of $S$ using positions specified in a seed $\pi$. Define the concatenation of those bases as $\text{substr}(S, x, \pi)$. For example, the substring starting at $x$ under the first

seed in Figure 2(B) is AUGAGACA. Then the LLR score for any substring starting at $x$ under a seed $\pi$ is $\mathrm{LLR}(S, x, \pi) = \log \frac{\Pr(\mathrm{substr}(S,x,\pi)|\mathcal{P})}{\Pr(\mathrm{substr}(S,x,\pi)|\mathcal{P}^0)}$. The background model $\mathcal{P}^0$ has the same length as $\mathcal{P}$, and each base's frequency is that observed in the database as a whole. A base pair's occurrence probability under $\mathcal{P}^0$ is the product of two single bases' probabilities. If the LLR score exceeds a threshold $T$ (to be determined), we declare a match to $\mathcal{P}$ at position $x$.

$\mathrm{LLR}(S, x, \pi)$ can be computed as the sum of LLR scores of individual bases or pairs in $\mathrm{substr}(S, x, \pi)$. For bases $a$ and $a'$, let $s(i, aa')$ and $s(i, a)$ be the LLR scores of base pair $aa'$ or base $a$ at the $i$th column of the SSP. For example, $s(0, \text{A-U}) = \log \frac{\mathcal{P}_{0,AU}}{\mathcal{P}^0_{0,AU}}$. Figure 2(B) shows the computation of LLR scores starting at $x$ in $S$ under two input seeds. Considering all the seeds $\pi \in \mathcal{P}$ and all the offsets $0 \leq x < |S|$, we define the final LLR score between S and $\mathcal{P}$ as

$$\mathrm{LLR}(S, \mathcal{P}) = \max_{x,\pi} \mathrm{LLR}(S, x, \pi).$$

## 3. DESIGNING SSPS FROM A CM

This section describes our algorithm to derive SSPs from a CM. We begin by formally defining the problem. For an SSP filter $\mathcal{P}$ and associated threshold $T$ derived from a CM $\mathcal{M}$, $\mathcal{P}$'s *sensitivity* to $\mathcal{M}$ is defined as the probability that $\mathcal{P}$ matches a sequence generated by $\mathcal{M}$: $Pr_{S\sim\mathcal{M}}(\mathrm{LLR}(S, \mathcal{P}) \geq T)$. The *false positive (FP) rate* for $\mathcal{P}$ at threshold $T$ is defined as the probability of seeing an SSP match with score $\geq T$ at a random position in a background sequence. Thus, the FP rate is $Pr_{S\sim\mathcal{P}^0}(\mathrm{LLR}(S, \mathcal{P}) \geq T)$, where background model $\mathcal{P}^0$ is the same as in Section 2.2 and $|\mathcal{P}^0| = |\mathcal{P}|$.

Many SSPs can be constructed from a CM. Our objective is to choose an SSP with high sensitivity and specificity to its CM. We also wish to keep the length of the designed SSP short to maximize the efficiency of scanning it against a large sequence database. The design problem is therefore as follows:

*Given CM $\mathcal{M}$ and null model $\mathcal{P}^0$, construct an SSP $\mathcal{P}$ of length at most $L_{\max}$ and an associated threshold $T$ so as to maximize $\mathcal{P}$'s sensitivity to $\mathcal{M}$ while keeping its FP rate relative to $\mathcal{P}^0$ no larger than a specified value $\tau$.*

The parameters $L_{\max}$ and $\tau$ are supplied by the user along with models $\mathcal{M}$ and $\mathcal{P}^0$, but threshold $T$ is derived automatically for each SSP.

We construct an SSP from a CM in two steps. First, we identify *gapless intervals* in a CM, which are likely to yield SSPs with few seeds, and extract candidate SSPs from each such interval. Secondly, we select a threshold for each candidate SSP to ensure its sensitivity with a bounded false positive rate, then select the best SSP(s) to act as filter for the CM.

### 3.1. Selecting candidate SSPs

Although our SSPs can handle gaps caused by insertions or deletions, variable-length gaps cause seeds to proliferate and so slow down the search process and increase the FP rate. We therefore design SSPs only in *gapless intervals* of the CM, which are regions without either deletions or more than two consecutive insertions. For a given CM, we calculate the length distributions of insertions and deletions starting from each state via dynamic programming. If an insertion state can generate more than two contiguous random bases with high probability, we call it a *split point*. Similarly, if a deletion state can be hit with high probability, it forms a split point. The positions between a pair of split points constitute a gapless interval.

We extract SSPs from a gapless interval as follows. Let $i$ be a position inside the interval. When there is no base pairing, an SSP of length $L \leq L_{\max}$ is constructed starting at $i$ using the emission probabilities of the match states associated with single stranded bases $i$ to $j = i + L - 1$. The corresponding seed is $0 \ldots L - 1$. If positions $x$ and $y$ are paired, with $i \leq x < y \leq j$, then $(x, y)$ forms a base pair in the SSP. That is, we keep only base pairs inside the same gapless interval. Bases that pair with a base outside the interval are treated as single-stranded.

When a base pair to be included in an SSP spans a gap whose length is not fixed, the resulting SSP contains multiple seeds, reflecting the different possible distances between the pair's endpoints. While the number of seeds can be exponential in the length of interval spanned by the SSP, we generate many fewer seeds in practice and could, if needed, arbitrarily limit the number of seeds generated.

## 3.2. Choosing the best SSP

The gapless intervals in a CM may generate a large number of candidate SSPs. For each candidate $\mathcal{P}_i$, we compute a threshold $T_i$ to achieve an FP rate of at most $\tau$, then compute the candidate's sensitivity given this threshold. The candidate SSP with the highest sensitivity is chosen as the final filter. More precisely, we select a threshold $T_i$ for each candidate $\mathcal{P}_i$ (of length $L$) that satisfies the constraint

$$\Pr_{S\sim\mathcal{P}^0 \text{ and } |S|=L}(\mathrm{LLR}\,(S,\mathcal{P}_i)\geq T_i)\leq\tau, \qquad (1)$$

then choose the candidate SSP $\mathcal{P}_i$ and associated $T_i$ that maximize

$$\Pr_{S\sim\mathcal{P}_i}(\mathrm{LLR}\,(S,\mathcal{P}_i)\geq T_i)\,. \qquad (2)$$

We note that, although we wish to judge whether a given SSP $\mathcal{P}_i$ will detect sequences drawn from a CM $\mathcal{M}$, we use the base distribution of the SSP itself, rather than that of the full CM, to estimate its sensitivity. This estimate may be inaccurate in two ways. On one hand, a path sampled from $\mathcal{M}$ might omit the CM states corresponding to the SSP $\mathcal{P}_i$, in which case the corresponding sequence lacks the portion that *should* match the SSP with a high score. On the other hand, $\mathcal{P}_i$ might happen to match some *other* portion of the CM with a high score. In theory, neglecting these two events results in an inaccurate estimate of the match probability.

Empirically, however, we find that the match probability is well approximated even if the above two events are ignored. For 117 ncRNA families chosen at random from Rfam, we compared our simplified sensitivity, computed via Eq. (2), to sensitivity as measured on a large set of Monte Carlo samples from the family's CM. The simplified and Monte Carlo estimates were highly correlated ($R^2 = 0.9901$), as desired. A detailed comparison of the two estimates is given in our supplementary data[a].

### 3.2.1. *Computing sensitivity and FP rate*

In our previous work[13], we developed a dynamic programming algorithm to compute the sensitivity and FP rate for a regular profile constructed from a profile HMM. In this work, we extend that algorithm to apply to an SSP constructed from a CM, which may include secondary structure conservation as well.

Following the definition of sensitivity in Eq. (2), we compute the sensitivity of an SSP $\mathcal{P}$, $\Pr_{S\sim\mathcal{P}}(\mathrm{LLR}\,(S,\mathcal{P})\geq T)$ as follows:

$$\Pr_{S\sim\mathcal{P}}(\mathrm{LLR}\,(S,\mathcal{P})\geq T)=\sum_{\theta=T}^{A^*}\Pr_{S\sim\mathcal{P}}(\mathrm{LLR}\,(S,\mathcal{P})=\theta),$$

where $A^*$ is the highest possible LLR score for a sequence produced by $\mathcal{P}$. Let $\mathcal{P}_{1..j}$ be a sub-SSP consisting of the first $j$ values in a seed and the corresponding emission profile columns (unpaired *or* paired) for $\mathcal{P}$. The sensitivity in Eq. (2) is given by $\sum_{\theta=T}^{A^*}\Pr(|\mathcal{P}|,\theta)$, where $|\mathcal{P}|$ is the SSP's length.

For convenience below, let $\Pr(\ell,\theta)$ denote the probability $\Pr_{S\sim\mathcal{P}_{1..\ell}}(\mathrm{LLR}\,(S,\mathcal{P}_{1..\ell})=\theta)$. Let $\mathcal{P}_{i,a}$ be the emission probability of unpaired base $a$ at column $i$. Similarly, let $\mathcal{P}_{i,a_1a_2}$ be the emission probability of base pair $a_1a_2$ at column $i$. Two dynamic programming cases are needed, depending on whether column $\ell$ describes an unpaired base or a base pair.

When column $\ell$ describes the frequency distribution of an unpaired base,

$$\Pr(\ell,\theta)=\sum_{a\in\Sigma}\mathcal{P}_{\ell,a}\Pr\left(\ell-1,\theta-\log\frac{\mathcal{P}_{\ell,a}}{\mathcal{P}^0_a}\right),$$

where $\mathcal{P}^0_a$ is the probability of the residue $a$ in the background model $\mathcal{P}^0$. When column $\ell$ describes the frequency distribution of a base pair,

$$\Pr(\ell,\theta)=\sum_{a_1a_2\in\Sigma^2}\mathcal{P}_{\ell,a_1a_2}\Pr\left(\ell-1,\theta-\log\frac{\mathcal{P}_{\ell,a_1a_2}}{\mathcal{P}^0_{a_1}\mathcal{P}^0_{a_2}}\right).$$

Initially, for each base $a\in\Sigma$ or $a_1a_2\in\Sigma^2$, $\Pr(1,\log\frac{\mathcal{P}_{1,a}}{\mathcal{P}^0_a})=\mathcal{P}_{1,a}$ if column 1 is created from an unpaired base, or $\Pr(1,\log\frac{\mathcal{P}_{1,a_1a_2}}{\mathcal{P}^0_{a_1}\mathcal{P}^0_{a_2}})=\mathcal{P}_{1,a_1a_2}$ if it is created from base pair.

If we let $S$ be sampled from $\mathcal{P}^0$ rather than from $\mathcal{P}$, the above algorithm can be modified to compute the FP rate against $\mathcal{P}^0$. The FP rate for $\mathcal{P}$ is

$$\Pr_{S\sim\mathcal{P}^0}(\mathrm{LLR}\,(S,\mathcal{P})\geq T)=\sum_{\theta=T}^{A^*}\Pr_{S\sim\mathcal{P}^0}(\mathrm{LLR}\,(S,\mathcal{P})=\theta).$$

For a given FP threshold $\tau$, the score threshold $T$ chosen for $\mathcal{P}$ is computed as

$$\mathrm{T}=\mathrm{argmin}_{T'}\left(\sum_{\theta=T'}^{A^*}\Pr_{S\sim\mathcal{P}^0}(\mathrm{LLR}\,(S,\mathcal{P})=\theta)\leq\tau\right).$$

[a] http://www.cse.wustl.edu/∼yanni/ncRNA

Let $s_{max}$ be the maximum possible LLR score for a single position of the SSP (one base or base pair). Similarly, let $s_{min}$ be the minimum such score. The time complexity of our dynamic programming algorithm is $\Theta(|\Sigma|L^2(s_{max} - s_{min}))$. Because only short intervals (we set $L_{\max} = 25$) are used to produce SSPs, the range of possible scores, and hence the running time, is limited. It typically takes only seconds to compute an SSP's score threshold and sensitivity.

### 3.3. Using SSPs vs. regular profiles

For many ncRNA families, particularly those with high primary sequence conservation, filtering with a regular profile produces fewer false positives than using an SSP. Regular profiles generally look at shorter intervals of the sequence than equally sensitive SSPs because the latter often need to span long loops to "see" significant stems whose two sides are widely separated in the primary sequence. Long loops tend to have variable length, so the SSP needs more distinct seeds to encode the range of possible loop lengths and hence has a higher chance of matching unrelated sequences purely by chance. On the other hand, for some ncRNA families with low primary conservation, the secondary structure encoded by SSPs may be the only available evidence on which to base a filter.

To best exploit both primary and secondary conservation, our filter design procedure selects between an SSP and a regular profile for each RNA family. When designing a filter for a family, we first design a regular profile without secondary structure information. If this regular profile achieves sensitivity $\leq 0.9$ to sequences from the CM according to our dynamic programming estimate, we instead design a full SSP for the family allowing base pairing. This approach applies the extra complexity of secondary structure filtering only where it is clearly needed.

### 3.4. Using multiple SSPs to improve sensitivity

A sensitive SSP is usually constructed from a well-conserved region within a CM. When multiple such regions exist in one CM, we can improve overall search sensitivity by designing a filter that is a union of SSPs from all well-conserved regions. For a query sequence $S$ and a filter $\Phi$ that contains $m$ SSPs $\mathcal{P}_1, ..., \mathcal{P}_m$, $\Phi$ matches $S$ iff at least one component $\mathcal{P}_i \in \Phi$ matches $S$. The total FP rate for $\Phi$ is at worst the sum of rates for its component filters $\mathcal{P}_i$.

Our SSP design algorithm can be extended to multiple SSPs. Instead of choosing the single SSP with the highest sensitivity under a specified FP rate threshold, we choose the top $m$ non-overlapping SSPs by estimated sensitivity. When two SSPs overlap, only the one with higher sensitivity is kept.

## 4. RESULTS

In this section, we first show that SSPs with secondary structure conservation exhibit a better empirical sensitivity/false positive rate tradeoff than regular profiles for detecting structured ncRNA families, such as tRNA and 5S rRNA, in the BRAliBase III benchmark database[21]. We then apply our automated filter design methods to a large number of ncRNA families from the Rfam database and quantify the resulting filters' sensitivity, FP rate, speedup when used in search, and their dependence on secondary structural conservation. We also compare SSPs and other filter types from related work. Finally, we investigate a small set of Rfam families on which our designed filters exhibit low sensitivity.

### 4.1. SSP utility for structured RNAs

To demonstrate and quantify SSPs' ability to exploit secondary structure, we first tested our heuristics on BRAliBase III[21], a database containing 602 5S rRNAs, 1114 tRNAs, and 235 U5 spliceosomal RNAs. BRAliBase III has been used as a benchmark for comparing ncRNA detection tools, including INFERNAL 0.7. We compared SSPs to regular profiles with no secondary structural information. We also tested a restricted form of SSP that was permitted only a single seed and so fixed the separation of all base pairs. Single-seed SSPs were tested to quantify the importance of handling variable-length gaps as part of SSP filter design.

We used the same methods as Ref. 21 to evaluate the sensitivity and FP rate of SSPs. A total of 40 sequence sets were sampled from each of the three ncRNA types; each tRNA set contained 60 sequences, while each rRNA and U5 set contained 20

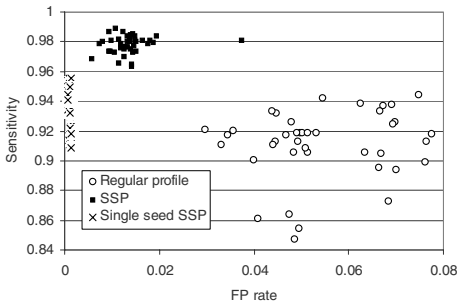


**Fig. 3.** Performance comparison for three types of filter designed for CMs built from tRNAs from BRAliBase III. Each CM was built from 60 sequences with pairwise identities between 0 and 0.6.

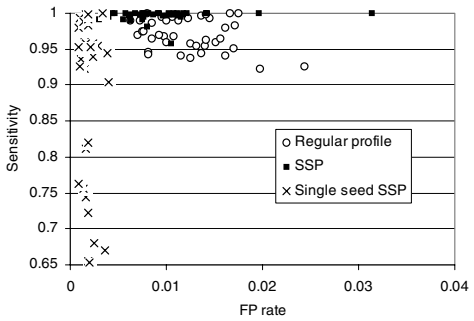


**Fig. 4.** Performance comparison for three types of filter designed for CMs built from 5S rRNAs from BRAliBase III. Each CM was built from 20 sequences with pairwise identities between 0.4 and 0.6.

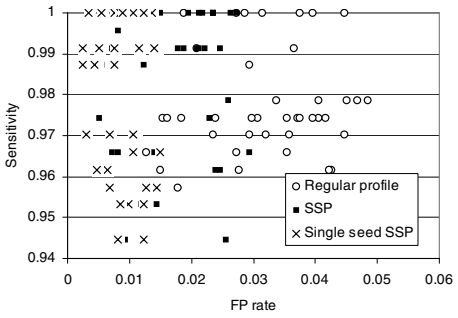


**Fig. 5.** Performance comparison for three types of filter designed for CMs built from U5 RNAs from BRAliBase III. Each CM was built from 20 sequences with pairwise identities between 0.4 and 0.6.

sequences. Sets were chosen so that no two sequences in a set aligned with greater than 60% identity. Each sampled set was used to train a CM. We designed heuristic filters from each CM, then tested the sensitivity of each filter on all sequences of the corresponding type in the database (e.g. 1114 tRNAs for tRNA-derived filters). For a CM $\mathcal{M}$, let $H_{\mathcal{M}}$ be the test set for $\mathcal{M}$, and let $S^{\mathcal{P}}_{H_{\mathcal{M}}}$ be the subset of sequences in $H_{\mathcal{M}}$ which contain a match to filter $\mathcal{P}$. $\mathcal{P}$'s sensitivity is defined as $|S^{\mathcal{P}}_{H_{\mathcal{M}}}|/|H_{\mathcal{M}}|$. FP rate was measured, as in Ref. 21, on a shuffled version of the test set that was ten-fold larger than the original. We note that we tested only the filters, rather than the underlying CMs, because experiments in Ref. 21 showed that CM search is already highly sensitive and specific for this database; hence, few if any true positives from a filter would be discarded by the CM, and nearly all false positives *would* be discarded.

Figures 3, 4, and 5 plot the sensitivities and FP rates of 40 designed regular profiles, SSPs, and single-seed SSPs for tRNA, 5S rRNA, and U5 spliceosomal RNA. Using SSP filters for tRNAs and 5S rRNAs consistently boosted sensitivity compared to regular profiles while reducing FP rate. Improvements for U5 RNAs were more uneven. Using multiple seeds in the SSP consistently improved sensitivity relative to single-seed SSPs, usually from $< 0.95$ to 0.98-0.99, at a cost to FP rate. Overall, incorporating secondary structure in our filter significantly improved its performance on these RNA families.

Variations in improvement observed with SSPs vs. regular profiles across these families can be explained by looking more closely at their conservation. The average sequence lengths for tRNAs, 5S rRNAs, and U5 RNAs are respectively 73, 117, and 119 bases, while the average number of annotated base pairs in their training sets are 21, 18, and 4. SSPs performed best on the tRNAs, which exhibit the highest density of base pairing, and worst on the U5 RNAs, with by far lowest such density.

## 4.2. Evaluation on Rfam database

In order to test our filter design methods on diverse ncRNA families with a wide range of sequence conservation levels, we applied the methods to families from the Rfam ncRNA database[b]. The filters in these tests came from our fully automated design pipeline, including automatic selection between regular profiles and SSPs as described in Section 3.3,

[b]http://www.sanger.ac.uk/Software/Rfam/

and automatic determination of score thresholds for each filter to achieve a uniform target FP rate.

We obtained Rfam release 8.0, which contains 574 non-coding RNA families. For each ncRNA family, Rfam provides a hand-curated seed alignment, as well as a full alignment produced by generating a CM from the seed alignment, then scanning for matches to that CM in EMBL's RFAMSEQ DNA sequence database. We selected for testing those ncRNA families with at least five sequences in the seed alignment (used to train the CM and hence our filters) and ten sequences in the full alignment (used to quantify sensitivity below). These criteria reduced our test set to 233 ncRNA families.

Empirical sensitivity of a filter was measured as the fraction of sequences in the full alignment that it matched. To measure a filter's empirical FP rate, we used the filter to scan 65 Mbases of sequence sampled from RFAMSEQ, using a simple scanning tool written in C++. In actual application, whenever a filter matched a locus in RFAMSEQ, the sequence surrounding that locus would be scanned using the full CM for the family. The filter's FP rate was therefore computed, following Ref. 18, as the ratio of the total length of sequences selected for scanning by the CM to the total length of the database.

More precisely, let $\mathcal{P}$ be the filter designed for CM $\mathcal{M}$. Let the average length of the sequences matched by $\mathcal{M}$ be $L$, and suppose that $\mathcal{P}$ matches the database $D$ at $m$ distinct positions. Then each match to $\mathcal{P}$ results in applying the CM to a region of length $L$ around the match. $\mathcal{P}$'s FP rate vs. data set $D$ was therefore estimated as $(m \times 2 \times L)/|D|$, where $|D|$ is the total length of $D$. For an CM $\mathcal{M}$ in INFERNAL, $L$ is the mean length of a match to $\mathcal{M}$.

Our filter designs used a theoretical FP rate upper bound of $\tau = 5 \times 10^{-6}$ and allowed multiple SSPs or profiles per family. As discussed in Section 3.3, we prefer to use regular profiles to SSPs when our theoretical estimate of sensitivity suggests that a regular profile would detect nearly all instances of an ncRNA family. Of the 233 ncRNA families tested, our methods produced regular profiles with theoretical sensitivity at least 0.9 for 220; for the remaining 13 families, we used SSPs to capture secondary structure information as well.

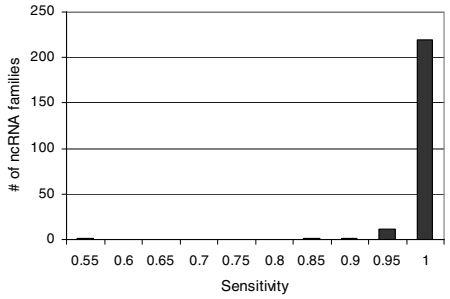


**Fig. 6.** Empirical sensitivities of filters for 233 ncRNA families from Rfam, measured on sequences in each family's full alignment.

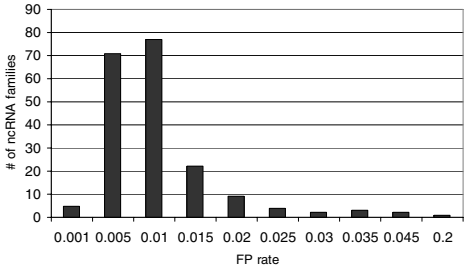


**Fig. 7.** Empirical FP rates of filters with sensitivity $\geq 0.99$ for 196 ncRNA families from Rfam, measured on 65 Mbases of genomic sequence from RFAMSEQ.

### 4.2.1. *Sensitivity and FP rate*

Figure 6 shows the sensitivities for our designed filters. Of the 233 filters designed, 196 (84%) had empirical sensitivity $\geq 0.99$. For these ncRNA families, the average number of sequences in the test set is 321. Only 3 filters (1.3%) had sensitivity less than 0.9. For the 196 filters, Figure 7 shows their empirical false positive rates on our 65-Mbase test database. The average FP rate observed was 0.008.

We note that the observed FP rate is several orders of magnitude greater than our theoretical FP rate $\tau$. This is because $\tau$ measures the filter match probability at a random position in a database. Thus, for a sequence database $D$, the expected number of matches is $|D| \times \tau$, and the empirical FP rate is $(|D| \times \tau \times 2L)/|D| = \tau \times 2L$. $L$ is on the order of hundreds for typical ncRNA-family CMs, so the observed FP rate is expected to be of the order shown in Figure 7 given our choice of $\tau$.

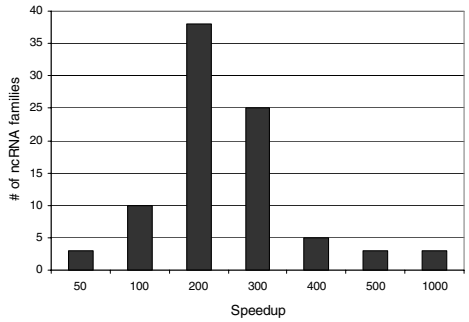


**Fig. 8.** Speedup distribution for 88 randomly chosen ncRNA families using filters composed of at most four profiles or SSPs.

#### 4.2.2. *Acceleration ability*

The key reason to place a filter in front of CM search is the efficiency of filtration compared to CM scanning. The average time (over 233 ncRNA families) to scan a family's CM against one Mbase of genomic DNA using INFERNAL's *cmsearch* tool was about 8701 seconds; in contrast, the average time to scan the same length with one of our filters was only 0.67 seconds. In this section, we first analyze the relationship between FP rate and observed speedup, then show the empirical speedups obtained by our filters.

Let $T_o$ be the time to run `cmsearch -noalign` against a database $D$. Let $T_s$ be the time to scan a filter against database $D$, and let $S_D^P$ be the set of matched substrings output by the filter. We estimate the time $T_f$ to scan a database with filtering enabled as $T_f = T_s + (|S_D^P|/|\mathrm{D}|) \times T_o$, where $|S_D^P|/|\mathrm{D}|$ is the FP rate of the filter. The speedup of filtered over unfiltered search is then $T_o/T_f$. To save time, we estimated the times $T_s$ and $T_o$ for our 65-Mbase RFAMSEQ sample from times measured on a 1-Mbase synthetic DNA sequence, since the cost of the filtration and CM scanning algorithms is insensitive to the actual sequence content. However, we used accurate empirical estimates of $|S_D^P|$ from our FP rate measurements of the previous section.

Figure 8 shows estimated speedups for 88 ncRNA families sampled at random from our set of 233. The average speedup for these families is 222x.

To validate our speedup estimates, we directly measured speedup on our 65-Mbase database for three ncRNA families, whose filters had FP rates ranging from 0.003 to 0.012. Table 1 gives both the estimated and observed speedups for these three families. These observations suggest that our estimates actually *underestimate* the speedup conferred by filtration. The reason is that $(|S_D^P|/|\mathrm{D}|) \times T_o$ empirically overestimates the cost of running *cmsearch* on the sequences emitted by the filter (data not shown). Consequently, the results shown in Figure 8 are conservative estimates of the actual speedups obtained by filtration.

**Table 1.** Estimated vs. observed speedups with filtration for 3 ncRNA families.

| Rfam ID | $T_o$ (s) | $T_f$ (s) | Est. speedup | Obs. speedup |
|---|---|---|---|---|
| RF00476 | 149502 | 1631 | 79 | 91 |
| RF00490 | 131625 | 354 | 281 | 371 |
| RF00167 | 262475 | 1217 | 149 | 215 |

### 4.3. Comparison with other filters

In this section, we compare our filters to two related works on filtered ncRNA search. Our first comparison is to Zhang et al.'s chain filters (CFs)[18], which were tested on a set of twelve riboswitch sub-families. We designed ten sets of regular profiles and two sets of SSPs for these sub-families. The results of our comparison are given in Table 2; the false positive rates shown are measured on the same synthetic data set described in Ref. 18. The average sensitivities observed for CFs and our filters were respectively 0.998 and 0.993, and the corresponding FP rates were 0.0353 and 0.0106. Overall, our automatically designed filters exhibited similar performance to CFs, which in Ref. 18 required manual intervention to choose numerical cut-offs for each filter.

Our second comparison is to the profile HMM-based filters of Ref. 15. The performance of HMM-based filters was tested using *cmsearch* in INFERNAL with option `-hmmfilter`. We compared our methods with HMM-based filters on two datasets: BRAliBase III, and 88 randomly selected ncRNA families from Rfam 8.0. Table 3 presents the median sensitivity and FP rates for the three types of ncRNA families in BRAliBase III. According to these experiments, the sensitivity of the two filter types is comparable, and the FP rate of HMM-based filters is smaller than that of SSP-based filters. However, because searching for HMM matches is much more

**Table 2.** Comparison of SSPs and chain filters (CFs) on 12 riboswitch sub-families.

| Rfam ID | CF sen | CF FP | SSP sen | SSP FP |
|---|---|---|---|---|
| RF00050 | 1 | 0.013 | 0.993 | 0.0034 |
| RF00059 | 1 | 0.063 | 0.994 | 0.02314 |
| RF00080 | 1 | 0.15 | 0.990 | 0.0043 |
| RF00162 | 1 | 0.018 | 1 | 0.0038 |
| RF00167 | 0.99 | 0.038 | 0.991 | 0.0034 |
| RF00168 | 0.99 | 0.015 | 0.986 | 0.0046 |
| RF00174 | 1 | 0.063 | 0.995 | 0.0052 |
| RF00234 | 1 | 0.013 | 1 | 0.0118 |
| RF00379 | 1 | 0.012 | 1 | 0.0024 |
| RF00380 | 1 | 0.012 | 1 | 0.0030 |
| RF00442 | 1 | 0.0017 | 1 | 0.0020 |
| RF00504 | 1 | 0.025 | 0.967 | 0.0598 |

**Table 3.** Comparison of SSPs and HMM filters.

| Name | SSP sensitivity median | HMM sensitivity median | SSP FP rate median | HMM FP rate median |
|---|---|---|---|---|
| tRNA | 0.979 | 0.983 | 0.013 | 0.002 |
| rRNA | 0.998 | 1 | 0.012 | 0.0 |
| U5 | 0.991 | 0.972 | 0.020 | 0.0 |

expensive than searching for profile matches, SSP-based filters yield better speedup. Figure 9 quantifies the advantage of SSP-based filters in the form of box plots describing the distribution of speedups for ncRNA families tRNA, rRNA, and U5 in BRAliBase.

In order to compare SSPs and HMM-based filters in a larger dataset, we then tested the sensitivity, FP rate, and actual search time using these

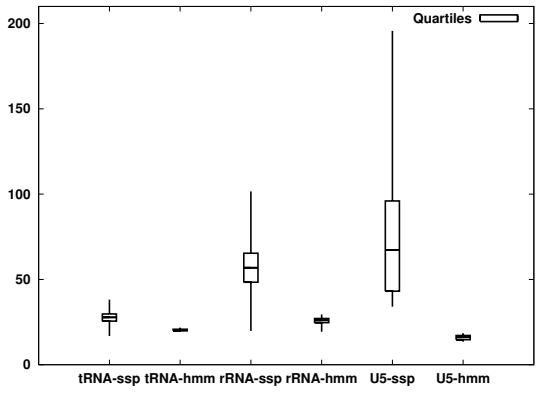


**Fig. 9.** Speedup comparison between HMM- and SSP-based filters for tRNA, rRNA, and U5 in BRAliBase III database. X-axis shows the names of ncRNA families and the used filters. Y-axis measures speedups.

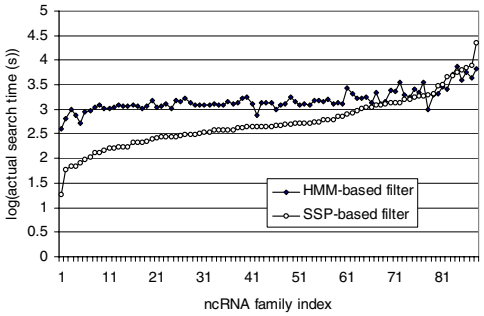


**Fig. 10.** Speed comparison between HMM- and SSP-based filters for 88 ncRNA families from Rfam. Y-axis measures actual ncRNA search time on logarithmic scale.

two types of filters on 88 randomly chosen ncRNA families from Rfam. The experimental setting was as in Section 4.2. The actual search time comparison is summarized in Figure 10. As we can see, INFERNAL runs significantly faster using SSP-based filters than using HMM-based filters for most of the tested ncRNA families. The HMM-based filter proved faster for only 7 out of 88 families, for which the SSP filter exhibited a high FP rate (around 0.02). According to our experimental results on over 200 ncRNA families in Section 4.2, the average FP rate of SSP-based filters is 0.008, which is small enough to ensure a better acceleration ability for a majority of SSP-based filters.

## 4.4. Analysis of low-sensitivity SSPs

For 35 of the 233 Rfam ncRNA families tested, our filters' empirical sensitivity was $< 0.99$. We divide these filters into two groups: those for which our theoretical estimates accurately predicted their low sensitivity (difference from empirical $< 0.05$), and those for which we predicted sensitivity $\geq 0.99$, but the empirical result was $< 0.95$. Our supplementary data gives examples of RNA families in both groups.

All but nine of the 35 "bad" families fall into the first category; while these cases illustrate limitations of our filtering heuristic, we can detect them during design and opt to use a less aggressive filter or no filter at all, depending on the user's tolerance for missed ncRNA occurrences.

For the remaining nine bad families, the high theoretical but low empirical sensitivity of their filters would result in unexpected loss of matches to the

CM. We therefore investigated these failures more closely. Because the CMs used to design our filters are trained only on seed alignments, filter quality depends heavily on whether a family's seed alignment accurately reflects the range of variation in its full alignment. A close look at one bad family, RF00266, reveals that the full alignment contains much shorter sequences than those in the seed alignment, with long deletions that are not described by the CM. As a result, SSPs constructed from the CM do not attempt to avoid these deletions. For three other families, the full alignment has much lower primary conservation than the seed alignment; hence, high predicted sensitivity on the CM's output is misleading as a predictor of empirical sensitivity. For a further three ncRNA families, low empirical sensitivity was an artifact of the family's small test set. For example, the filter for family RF00002 missed only one of 15 sequences in its test set, but this yielded empirical sensitivity of only 0.93.

In the above seven cases, the apparent "badness" appears to be either an artifact of a small test set or a limitation in how representative the seed alignment is of the full family. There are only two cases (RF00008 and RF00013) where we cannot yet explain the discrepancies between the theoretical and experimental sensitivities.

## 5. CONCLUSIONS

Covariance models are a state-of-the-art method to identify and annotate non-coding RNAs. However, their high computational cost impedes their use with large sequence databases. Our automatically designed SSP filters encode both primary sequence and (optionally) secondary structure conservation in an ncRNA family, yet they can scan a large sequence database efficiently. 84% of our designed filters have sensitivity at least 0.99, and their average FP rate is 0.008. Our filters obtain an average speedup of 222x over search using CMs alone on Rfam.

There is considerable room to improve the sensitivity and design efficiency of SSP filters. We plan to study more systematic methods to choose a set of SSPs so as to maximize their combined sensitivity. We also plan to design chain filters using SSPs as components. The lengths of the component SSPs can be shorter than the typical lengths of the filters in this work because all or most must match to yield a chain filter match. We expect that collections of short filters would be most effective for ncRNA families whose alignment contains frequent gaps, preventing the appearance of long gapless intervals.

## Acknowledgments

This work was supported by NSF CAREER award DBI-0237903.

## References

1. S. R. Eddy. Noncoding RNA genes. *Curr. Opin. Genet. Dev.* 1999; **9**:695–9.
2. S. R. Eddy. A memory-efficient dynamic programming algorithm for optimal alignment of a sequence to an RNA secondary structure. *BMC Bioinformatics* 2002; **3**:3–18.
3. S. R. Eddy and R. Durbin. RNA sequence analysis using covariance models. *Nucleic Acids Res.* 1994; **22**:2079–88.
4. R. Durbin, S. R. Eddy, A. Krogh, and G. Mitchison. Biological Sequence Analysis Probabilistic Models of Proteins and Nucleic Acids. UK: Cambridge U. Press, 1998.
5. D. H. Younger. Recognition and parsing of context-free languages in time $n^3$. *Information and Control* 1967;**10**:189–208.
6. S. Griffiths-Jones, S. Moxon, M. Marshall, A. Khanna, S. R. Eddy, and A. Bateman. Rfam: annotating non-coding RNAs in complete genomes. *Nucleic Acids Res.* 2005; *3*3:D121–4.
7. Weinberg Z, Ruzzo WL. Faster genome annotation of non-coding RNA families without loss of accuracy. In *Proc. 8th Ann. Int. Conf. on Res. in Comp. Mol. Bio. (RECOMB '04).* 2004; 243–51.
8. B. Brejova, D. G. Brown, and T. Vinar. Optimal spaced seeds for hidden Markov models, with application to homologous coding regions. *14th Ann. Symp. Combinatorial Pattern Matching.* 2003; 42–54.
9. J. Buhler, U. Keich, and Y. Sun. Designing seeds for similarity search in genomic DNA. *Proc. 7th Ann. Int. Conf. Comp. Mol. Bio.* 2003; 67–75.
10. M. Li, B. Ma, D. Kisman, and J. Tromp. PatternHunter II: highly sensitive and fast homology search. *J. Bioinf. and Comp. Bio.* 2004; **2**:417–39.
11. L. Noe and G. Kucherov. Improved hit criteria for DNA local alignment. *BMC Bioinformatics* 2004; **5**.
12. Portugaly E., Ninio M. HMMERHEAD – accelerating HMM searches on large databases (poster). *Proc. 7th Ann. Int. Conf. Comp. Mol. Bio.* 2003.
13. Sun Y, Buhler J. Designing patterns and profiles for profile HMM search. *IEEE/ACM Trans. Comp. Bio. and Bioinf.* 2008.

14. Lowe T, Eddy S. tRNAscan-SE: a program for improved detection of transfer RNA genes in genomic sequence. *Nucleic Acids Res.* 1997; **25**:955–64.
15. Weinberg Z, Ruzzo WL. Sequence-based heuristics for faster annotation of non-coding RNA families. *Bioinformatics* 2006; **22**:35–9.
16. Weinberg Z, Ruzzo WL. Exploiting conserved structure for faster annotation of non-coding RNAs without loss of accuracy. *Bioinformatics* 2004; **20 suppl. 1**:i334–40.
17. Zhang S, Haas B, Eskin E, Bafna V. Searching genomes for noncoding RNA using FastR. *IEEE/ACM Transactions on Comp. Bio. and Bioinf.* 2005; **2**:366–79.
18. Zhang S, Borovok I, Aharonowitz Y, Sharan R, Bafna V. A sequence-based filtering method for ncRNA identification and its application to searching for riboswitch elements. *Bioinformatics* 2006; **22**:e557-65.
19. Gautheret D, Lambert A. Direct DNA motif definition and identification from multiple sequence alignments using secondary structure profiles. *J. Mol. Bio.* 2001; **313**:1003–11.
20. Bafna V, Zhang S. FastR: fast database search tool for non-Coding RNA. *Proc. 2004 IEEE Comp. Systems Bioinf. Conf.* 2004; 52–61.
21. Freyhult EK, Bollback JB, Gardner PP. Exploring genomic dark matter: a critical assessment of the performance of homology search methods on noncoding RNA. *Genome Res.* 2006; **17**:117–25.