Rfam Documentation

Rfam Team

Apr 01, 2021

About Rfam

Rfam is a collection of non-coding RNA families represented by manually curated sequence alignments, consensus secondary structures, and predicted homologues.

This documentation is maintained by the *Rfam team*. Please report any issues or contribute on GitHub.

CHAPTER 1

Contents

1.1 About Rfam

The Rfam database is a collection of RNA sequence families of structural RNAs including non-coding RNA genes as well as cis-regulatory elements. Each family is represented by a multiple sequence alignment and a covariance model (CM).

You can use the Rfam website to obtain information about an individual family, or browse the families and genome annotations. Alternatively you can download all of the Rfam data from the FTP site. Find out more about the project by exploring the latest *Rfam references*.

For each family Rfam provides:

Summary page Textual background information on the RNA family, which we obtain from the online encyclopedia Wikipedia

Seed alignment A curated alignment containing a small set of representative sequences and a consensus secondary structure annotation

Sequences Information about sequences in the family, including bit score, seed and full alignments, region coordinates, sequence description from the EMBL nucleotide database, and the species name

Secondary structure Secondary structure images, annotated with various measures of sequence and structure conservation

Species Interactive tree graphic displaying species distribution for the full alignment.

Trees Phylogenetic trees are available for the seed and the full alignment

Structures Mappings between PDB structures and Rfam annotations

Database references Links to external databases and references to other data sources

Curation Covariance model files contain information summarising the family, including the author of alignment, references for sources of sequence and/or structure, the number of sequences in each alignment, score thresholds and score distributions

1.2 How Rfam families are built

Note: If you are interested in building new Rfam families or updating the existing ones, take a look at the *Rfam cloud pipeline*.

1.2.1 Seed alignments and secondary structure annotation

Rfam *Seed alignment* is a small, curated set of representative sequences for each family, as opposed to an alignment of all known members. The seed alignment also has as a **secondary structure** annotation, which represents the **conserved** secondary structure for these sequences.

The ideal basis for a new family is an RNA element that:

- has some known functional classification
- is evolutionarily conserved
- has evidence for a secondary structure

In order to build a new family, we must first obtain at least one **experimentally validated example** from the published literature. If any other homologues are identified in the literature, we will add these to the seed. Alternatively, if these are not available, we will try to identify other members either by similarity searching (using *Infernal*) or manual curation.

Where possible we will use a multiple sequence alignment and secondary structure annotation provided in the **literature**. If this is the case, we will cite the source of both the alignment and the secondary structure. You should note that the structure annotations obtained from the literature may be experimentally validated or they may be RNA folding predictions (commonly *MFOLD*). Unfortunately, we do not discriminate between these two cases when we cite the PubMed Identifier (PMID) and you will need to refer to the original publications to clarify.

Alternatively, where this information is not available from the literature, we will generate an alignment and secondary structure prediction using various software, such as *WAR* or *RNAalifold*. This software allows us to cherry pick the best alignment and secondary structure prediction. Historically, the methods used to make these alignments and folding predictions have varied. Author names added to the list indicate that the published or predicted secondary structure has been manually curated in some way. The last author on the list will be the most recent editor of the secondary structure. You can find the method we have used for the seed alignment or the secondary structure annotation in the **SE** and **SS** lines of the *Stockholm format* or in the curation information pages.

From the seed alignment, we use the Infernal software to build a *Covariance model (CM)* for this family. This model is then used to search the *rfamseq* database for other possible homologs.

1.2.2 Expanding the seed (iteration)

If the CM search of *rfamseq* identifies any homologs that we believe would improve the seed, we use the Infernal software (cmalign) to add these sequences to the seed alignment. From the new seed, the CM is re-built and re-searched against *rfamseq*. We refer to this process of expanding the seed using Infernal searching as **iteration**. We continue to iterate the seed until we have good resolution between real and false hits and cannot improve the seed membership further.

Fig. 1: Building an RNA family using Infernal

1.2.3 Important points to remember about seed alignments

- We can only build families using the sequences in *rfamseq*
- We can only build a family where we can identify more than one sequence in *rfamseq*
- Sequences in the seed cannot be manually altered in any way, e.g. no manual excision of introns, no editing of sequencing errors, no marking up modified nucleotides etc
- At least one sequence in the seed will have some experimental evidence of transcription, e.g. northern blot or RT-PCR, and, preferably, some evidence of function
- The secondary structure should ideally have some experimental support (such as structure probing, NMR or crystallography) and/or evidence of evolutionary conservation. We do, however, use and generate predicted structures
- Each seed sequence will be a significant match to the corresponding covariance model. A significant score is generally greater than 20 bits.

1.2.4 Rfam full alignments

The Rfam *Full alignment* contains all of the sequences in *rfamseq* that we can identify as members of the family. The alignment is generated by searching the covariance model for the family against the *rfamseq* database. Matches that score above a *Gathering cutoff* are aligned to the CM to produce the full alignment. All sequences in the seed will also be present in the full alignment.

1.2.5 Wikipedia annotations

In order to provide some background and functional information about a family, we link to a Wikipedia page that provides relevant background information on the family. We have either linked to an existing page or we have created the page ourselves in Wikipedia. As this annotation is hosted by Wikipedia, you are free to edit, correct and otherwise improve this annotation and we would encourage you to do so.

1.2.6 Phylogenetic trees

All our phylogenetic trees are generated using fasttree.

1.3 Glossary

- *Infernal*
- *MFOLD*
- *Pfold*
- *rfamseq*
- *RNAalifold*
- *R-scape*
- *Seed alignment*
- *Sequence region*
- *Stockholm format*
- *Type*
- *WAR*
- *WUSS format*

1.3.1 Clan

An **Rfam clan** is a group of families that either share a common ancestor but are too divergent to be reasonably aligned or a group of families that could be aligned, but have distinct functions. For example, the LSU clan (CL00112) includes 5 families describing different types of large ribosomal subunit RNAs, including bacterial, eukaryotic, and archaeal LSU families.

1.3.2 ClustalW

A general purpose multiple sequence alignment program for DNA (RNA) which we use while building our SEED alignments. See the Clustal web server.

1.3.3 Covariance model (CM)

A secondary structure profile for a RNA structural alignment (also called profile stochastic context-free grammars). Find out more about *Covariance models and stochastic context-free grammars*.

1.3.4 DESC file

Each family is described using in a `DESC` file that includes the information such as family description, database references, RNA type, and publications (see the tRNA DESC file as an example).

1.3.5 Family

A group of RNA sequences which are believed to be evolutionarily related in sequence or secondary structure.

Fig. 2: SAM riboswitch Rfam family

1.3.6 Full alignment

An alignment of the set of related sequences which score higher than the manually set threshold values for the covariance model of a particular Rfam family.

As of Rfam 12.0, we no longer automatically generate full alignments for each Rfam family. You may download the Rfam CM and generate your own alignments using Infernal. For details about generating a full alignment, see the Rfam CPB paper.

1.3.7 Gathering cutoff

The bit score gathering threshold (GA cutoff), set by Rfam curators when building the family. All sequences that score at or above this threshold will be included in the full alignment and are believed to be true homologs to the model. For more information see Nawrocki et al., 2015.

1.3.8 Infernal

Infernal is the core software that enables us to make consensus RNA secondary structure profiles (covariance models (CMs)) for our families. We also use Infernal for searching sequence databases for homologous RNAs. See the Infernal website for more details.

1.3.9 MFOLD

RNA structure prediction algorithm which utilises minimum free energy information. See the MFOLD website.

1.3.10 Pfold

RNA folding software which folds alignments using a Stochastic Context-Free Grammars (SCFG) trained on rRNA alignments. It takes an alignment of RNA sequences as input and predicts a common structure for all sequences. See the Pfold website.

1.3.11 rfamseq

Rfamseq is the underlying nucleotide sequence database on which Rfam is based. Starting with Rfam 13.0, *rfamseq* is based on a collection of complete, non-redundant, and representative genomes maintained by UniProt (find out more in the Rfam 13.0 paper).

rfamseq is usually updated with each major Rfam release, e.g., 12.0 or 13.0. You can find out the information about *rfamseq* currently in use in the README file in the Rfam FTP archive.

1.3.12 RNAalifold

Folds pre-computed alignments using a combination of free-energy and covariation measures. Part of the Vienna package.

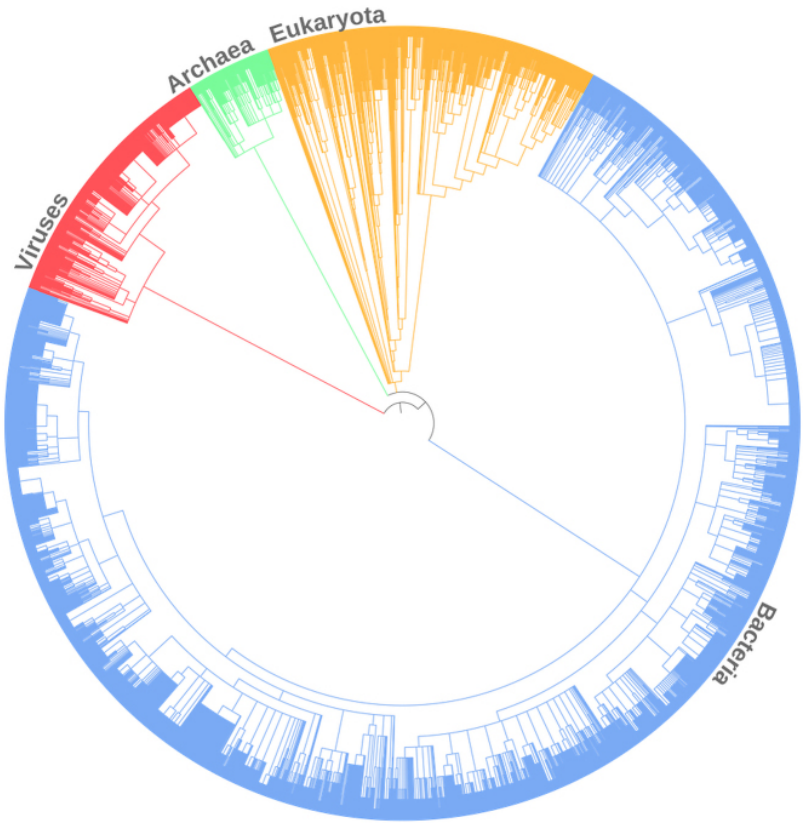


Fig. 3: Taxonomic composition of Rfamseq 13.0

1.3.13 R-scape

R-scape is a method for testing whether **covariation analysis** supports the presence of a conserved RNA secondary structure in a multiple sequence alignment. R-scape is used to create and improve Rfam families, and R-scape visualisations are shown on the secondary structure tab for each family (for example, SAM riboswitch).

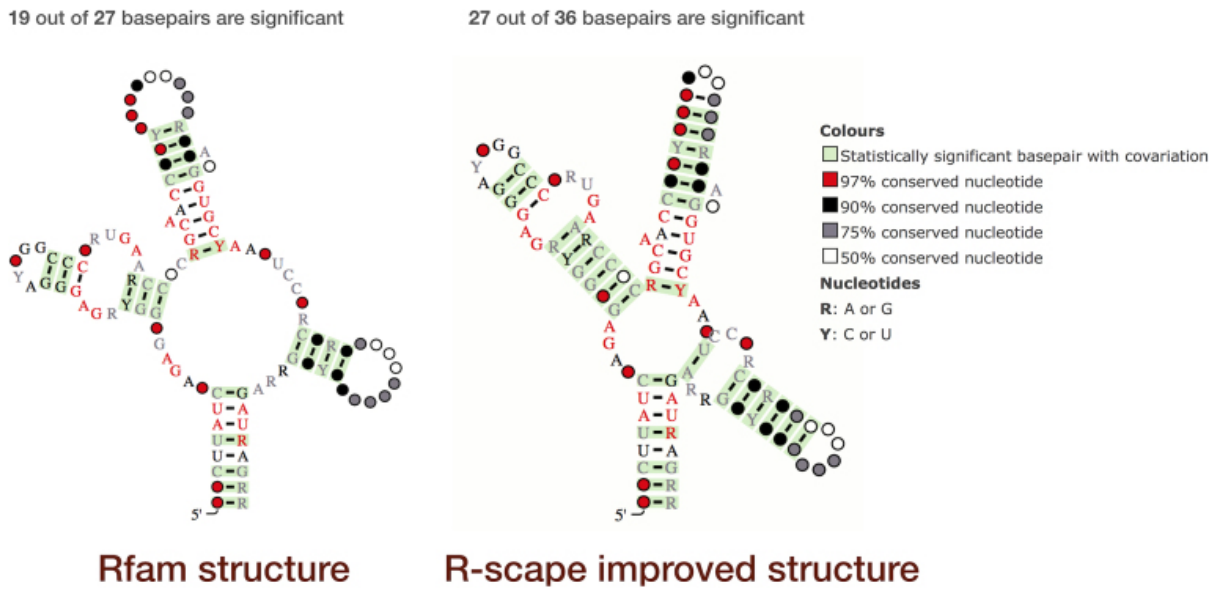


Fig. 4: R-scape visualisation of SAM riboswitch

1.3.14 Seed alignment

A manually curated sample of representative sequences for a family. These sequences are aligned and annotated with a consensus secondary structure. This alignment is used to build the covariance model for the family. See *Seed alignments and secondary structure annotation* for more information.

Fig. 5: An example seed alignment coloured by secondary structure helical regions

1.3.15 Sequence region

A single segment of nucleotide sequence in our alignments. Multiple sequence regions from a single EMBL sequence may be in the same family.

1.3.16 Stockholm format

A multiple sequence alignment format used by Rfam (and Pfam) for the dissemination of protein and RNA sequence alignments. For more information see the Wikipedia article on Stockholm format or the Rfam tRNA alignment.

1.3.17 Type

A simple functional classification used to organise Rfam families into **RNA types**. This ontology does not current directly relate to the ontologies used by other databases. For a full list of RNA types see the *Search by entry type* section.

1.3.18 WAR

A software tool that enables us to simultaneously run several different methods for performing multiple alignment and secondary structure prediction for non-coding RNA sequences. See the WAR website.

1.3.19 WUSS format

The Washington University Secondary Structure (WUSS) format is designed to make it easier to see the secondary structure by eye and follows the following conventions:

Symbol	Meaning
<>	basepairs in simple stem loops
`()`, `[]`, `{}`	basepairs enclosing multifurcations
– (hyphen)	internal loops and bulges
`,` (comma)	single strand between helices
`:` (semicolon)	single stranded residues external to any secondary structure
`.` (period)	insertions relative to the consensus

1.4 Frequently Asked Questions

1.4.1 Documentation

What are "seed" and "full" alignments?

There are two kinds of Rfam multiple sequence alignments:

1. The **seed** alignment is a hand-curated alignment of known members of the family. This alignment may not contain all known members of a family, but rather a representative set. We use the Infernal software to build a covariance model from this alignment.

2. The covariance model is used to search the *rfamseq* sequence database for other family members and build a **full** alignment including all instances of a family. Starting with Rfam 12.0 the full alignments are no longer provided by Rfam, but they can be generated as described in the following protocol.

What do the scores for hits to Rfam models mean?

When you search a sequence against Rfam and obtain a hit to one of our families, we report the start and end coordinates of the matching region, the orientation of the match, and the bit score. The bit score (also known as the log-odds score) is generated by the Infernal software when it tries to match your sequence to the model. In very simple terms, it is a measure of how well your sequence matches the model; the higher the bit score, the better your sequences fits the model.

More specifically the bit score is the $\log_2$ of the probability of the query sequence given the model, over the probability of the sequence given the null model:

In theory this means that positive bits scores are significant but, in practice, more conservative cutoffs are used as the size of the database means we can observe hits with low positive bits scores by chance. (See the Infernal user guide for more information.)

Where does your secondary structure annotation come from?

Ideally, when we build a seed alignment, the initial secondary structure annotation is obtained from the literature. In these cases the secondary structure is usually available only for a few of the member sequences in the seed. Our aim is to generate models that represent conserved secondary structure, so when we begin to expand the membership of the seed to be as representative as possible, we will only retain the secondary structure annotation that is conserved between the majority of sequences. You should also note that the annotations obtained from the literature may be experimentally validated or they may be RNA folding predictions (commonly MFOLD). We do not discriminate between the two and you will need to refer to the original publications to clarify.

In those cases where no secondary structure prediction is available in the literature, but where we have good set of seed sequences, we frequently use a local installation of the WAR software which allows us to cherry pick the best alignment and secondary structure prediction from multiple tools. Historically, the folding prediction tools used has varied and the method used will be indicated.

You can find the alignment and structure source for each family in the curation tab, or in the SE and SS lines in the Stockholm file. Where the source is obtained from the literature, we will provide the PubMed identifier (PMID). You should also note that the seed alignments often get updated between releases and may be manually adjusted by the curator. As a result, attempts to obtain the same structure using the same prediction method, may not return exactly the

same structure as shown on the Rfam SEED alignment. We usually indicate where the a structure has been manually edited.

What is your definition of an RNA family?

We will group sequences into a "family" where we can identify sequence or secondary structure conservation using our covariation models. This is decided when we build our seed alignment and search the CM against rfamseq. From the resulting searches we decide where the cutoff threshold should be.

When we set this cut off threshold, we are essentially deciding that any sequences that score above the threshold are true, homologous members of the family, whilst those below are "chance hits". This discrimination between true and false is usually very clear if we have a representative seed alignment.

Occasionally, for various biological reasons, it can be extremely difficult to get good resolution between true and false predictions. In such case we make an informed decision on where the cutoff should be. As a result, some families may contain false positives (often pseudogenes) or may also lose some true positives below the threshold. In such cases we will have made the best choice we can in order to limit the false positive and loss of true positives. If you have queries about the membership of any of our families, please *Contact us* and we will try to clarify or resolve the problem.

How can I tell which are predicted and which are experimentally confirmed sequences?

Unfortunately, it is not currently possible to do this, since we do not add a source tag to each individual sequence in either our seed or full alignments. All of our families (seed alignments) are based on one or more experimentally validated exemplars of the family, but the majority of the other member sequences are added by homology search and manual curation. We have high confidence in these members of the seed alignment that we use to build the covariance model and computationally predict other possible members in the nucleotide database.

You can study the descriptions of sequences extracted from the EMBL nucleotide database, occasionally this contains useful information about function.

Why is my favourite sequence not in the family?

The most likely reason is that it is not in the EMBL release that rfamseq is based on. With each major release, e.g. 8.0, 9.0, we update the underlying nucleotide database. You can check which version we are currently using here. If, however, your sequence is in the relevant EMBL release but is still absent from a relevant family, it is possible that our model may need to be improved. Please *Contact us* with the relevant information and we will decide whether the sequence should indeed be included and, if so, we will try to improve our model.

Where can I find out more about RNA sequence analysis/covariance models/SCFGs?

The Infernal software package, which is an essential companion to the Rfam database, now has extensive documentation, along with some description of how covariance models work for RNA sequence analysis. Background and theory can also be found in the excellent book Biological Sequence Analysis by Richard Durbin, Sean Eddy, Anders Krogh, and Graeme Mitchison (Cambridge University Press, 1998). For more references see *Citing Rfam*.

1.4.2 Searching

How can I find information about a particular RNA family?

You can do this in several ways. If you already know the Rfam accession or name of the family, you can use the "jump to" boxes on the home page or any tabbed page in the website. Alternatively, if you're not sure of the family accession or correct name and want to try a broad-ranging search, you should use the "keyword" search box in the header of each

page. This search allows the use of ambiguous terms and will search multiple sections of the database for a match to your query term. The results page will give you a list of all the families with matches and you can follow the links to the summary page for each family.

If you're not even sure of your query term and simply want to browse our families, click on the "browse" link in the header of every page. This takes you to an index that lists all Rfam families according to accession and ID and links directly to the summary page for each family.

How can I search my DNA sequence for non-coding RNA genes?

Both our single sequence and batch searches allow you to search a nucleotide sequences against the Rfam model library. Any hits to Rfam families will be returned with start and end coordinates, orientation and a score for each hit.

For short single sequences, our single sequence search tool will return Rfam matches to your sequence interactively. However, if your sequence is longer than 2Kbp, we suggest that you fragment it into smaller, overlapping segments and use the batch search facility. You might find this tool useful for splitting large sequences into fragments.

Finally, if you have a very large number of sequences to search, you may find it most convenient to download and run Rfam locally (see section *Genome annotation* for more information).

1.4.3 Downloading

What do the sequence identifiers in your alignments mean?

The identifier **"AY033236.1/563-353"** means that the EMBL accession is "AY033236", the sequence version is "1" (optional), the start coordinate is "563" and the end coordinate is "353", the strand is given by the order of the coordinates, in this case it is negative.

How can I view or download a family alignment?

From the family summary page, go to the "Alignments" tab on the left side panel. The alignments tab will give you multiple drop down options on how to either view or download the seed sequences for this family, in an aligned or fasta format. The formatting options allow you to select which type of format you would prefer.

If the alignment is very large the formatting tool may not be suitable and you may prefer to use the preformatted alignment in Stockholm format. A number of Stockholm alignment re-formatters and viewers exist, such as the sreformat program from the HMMer package and the RALEE major mode for Emacs. You can read more about Stockholm format on Wikipedia.

As of release 12.0, we no longer provide full alignments for automatic download. You can generate them using the Sunbursts feature for sequences of your choice (for families with full alignments containing less than 1000 sequences), or generate them yourself by downloading the covariance model and using the Infernal suite of software.

If you are interested retrieving alignments for multiple families, you can download all our seed alignments in Stockholm format flat-files, and the covariance models used to generate them, from our ftp site.

How can I download a subset of sequences from a family?

Unfortunately, this has not been implemented yet. There are plans in place to modify the underlying Rfam database to allow this.

How can I download all Rfam sequences for my favourite species?

Unfortunately, this has not been implemented yet. Please *Contact us* if you need help.

The "Taxonomy" tab on the search page will allow you to perform taxonomic queries. In fact, this function also allows you to search with queries from internal nodes of the NCBI taxonomic tree. However, the results are only returned on the family level, not the sequence level.

1.4.4 Rfam and Infernal

How do I filter Infernal output by Rfam family type?

Sometimes it is useful to filter Infernal output based on Rfam family type, for example, if you are only interested in rRNA families.

1. Get a list of Rfam families for each RNA type (see *Search by entry type*).

   For example, selecting the **rRNA** checkbox gives the following list:

```
RF00001     5S_rRNA Gene; rRNA
RF00002     5_8S_rRNA       Gene; rRNA
RF00177     SSU_rRNA_bacteria       Gene; rRNA
RF01118     PK-G12rRNA      Gene; rRNA
RF01959     SSU_rRNA_archaea        Gene; rRNA
RF01960     SSU_rRNA_eukarya        Gene; rRNA
RF02540     LSU_rRNA_archaea        Gene; rRNA
RF02541     LSU_rRNA_bacteria       Gene; rRNA
RF02542     SSU_rRNA_microsporidia  Gene; rRNA
RF02543     LSU_rRNA_eukarya        Gene; rRNA
RF02545     SSU_trypano_mito        Gene; rRNA
RF02546     LSU_trypano_mito        Gene; rRNA
RF02547     mtPerm-5S       Gene; rRNA
RF02554     ppoRNA  Gene; rRNA
RF02555     hveRNA  Gene; rRNA
```

2. Create a file on your computer called `rfam-ids.txt` with a list of Rfam ids:

```
RF00001
RF00002
RF00177
RF01118
RF01959
RF01960
RF02540
RF02541
RF02542
RF02543
RF02545
RF02546
RF02547
RF02554
RF02555
```

Tip: If you would like to download the list of RNA families and types as text, click **Show the unformatted list** at the bottom of the search results page. Then copy and paste into an editor and save the file for

example as `rfam-types.txt`. You can then create the `rfam-ids.txt` file with the command `cat rfam-types.txt | awk '{ print $1 }' > rfam-ids.txt`.

3. Use the grep command to filter Infernal results.

   For instance, given an Infernal *tblout* file `results.tblout` (example file), run this command:

   ```
   grep -f rfam-ids.txt results.tblout
   ```

   It will print only the lines from `results.tblout` that contain Rfam ids specified in `rfam-ids.txt`.

   Alternatively, if you want to **exclude** some families from your analysis, you can use the following command:

   ```
   grep -v -f rfam-ids.txt results.tblout
   ```

   This will print only the lines that **do not** contain Rfam ids listed in `rfam-ids.txt`.

You can use this procedure to filter Infernal results by **any** set of Rfam families. For example, you can get a list of Rfam families using *Taxonomy search* and get Infernal search results from families found in a specific taxonomic group.

1.4.5 Other

I would like to submit a family

Great! We are very keen for the community to help keep us updated on new families. Ideally, a new family for Rfam should contain elements (RNA sequences) that have some known functional classification, are evolutionarily conserved and have evidence for a secondary structure. The families should not solely be based on prediction only, e.g. RNAz, EvoFold, or QRNA predictions, nor solely on transcriptomic data, e.g. tiling array or deep sequencing. For more detailed information on how to submit a family, please read the rest of the Rfam documentation but, if you have any queries, please do *Contact us*.

If your family is sufficiently interesting, or if you have several of them, you may be interested in publishing your family in the RNA families track that is available through the RNA Biology journal.

How can I edit a SEED alignment?

We do not currently provide public access to edit our alignments. This is advantageous in that it maintains our standard of alignments and structures, but, if you feel our seed alignment/structure annotations can and should be improved, please *Contact us*, preferably supplying us with a new alignment, in Stockholm format, and we will do our best to incorporate the improvements.

1.5 Rfam team

The Rfam database is curated and maintained at the European Bioinformatics Institute in Cambridge, UK.

1.5.1 European Bioinformatics Institute

- Nancy Ontiveros-Palacios - Biocurator
- Alex Bateman - Senior Team Leader

- Anton I. Petrov - Rfam Project Leader

1.5.2 Collaborators

- Sean Eddy (Harvard University) - founding developer and author of Infernal software
- Eric Nawrocki (NCBI) - developer of the Infernal software
- Elena Rivas (Harvard University) - developer of the R-scape software
- Manja Marz and Kevin Lamkiewicz (Friedrich Schiller University Jena) - collaborators on the Rfam Viruses project
- Sam Griffiths-Jones (University of Manchester) - *founding Rfam project leader* and a collaborator on the microRNA project
- Daniel Gautheret and Claire Toffano-Nioche (University of Paris-Saclay) - collaborators on the Rfam Cloud project
- Zasha Weinberg (University of Leipzig) - developer of the ZWD database

1.5.3 Previous contributors

- Paul Gardner - *former Rfam project leader*
- Sarah Burge - *former Rfam project leader*
- Rob Finn - *Group Team Leader*
- Ioanna Kalvari - *Software developer*
- Joanna Argasinska - *Biocurator*
- Ruth Eberhardt
- Evan Floden
- John Tate
- Jennifer Daub
- Ben Moore
- Mhairi Marshall
- Simon Moxon
- Adam Wilkinson
- William Mifsud
- Enrico Marantidis
- Diana Kolbe

Rfam is a collaborative venture and we hope to be able to interact with as many people as possible to provide a quality database. Please *Contact us* for information.

1.6 Contact us

You can find the email address for the Rfam helpdesk at the bottom of every page on the Rfam website. We use a request tracking system to monitor emails to Rfam, so you should receive an automated response to your email, letting you know that the system has logged your mail and notified us of its arrival.

1.6.1 Xfam blog

The Rfam group contributes to the Xfam blog. The blog is used to announce releases, new features and important changes to Rfam, as well as for posts discussing general issues surrounding the Rfam resource. You can see blog posts that are specific to Rfam here.

1.6.2 RSS feed

You can keep in touch with the latest goings by subscribing to the RSS feed from the Xfam blog.

1.6.3 Twitter

You can follow the RfamDB team at EMBL-EBI.

1.6.4 Submit an alignment

We're happy to receive updated or improved alignments for new or existing families. Submit your alignment and we'll take a look.

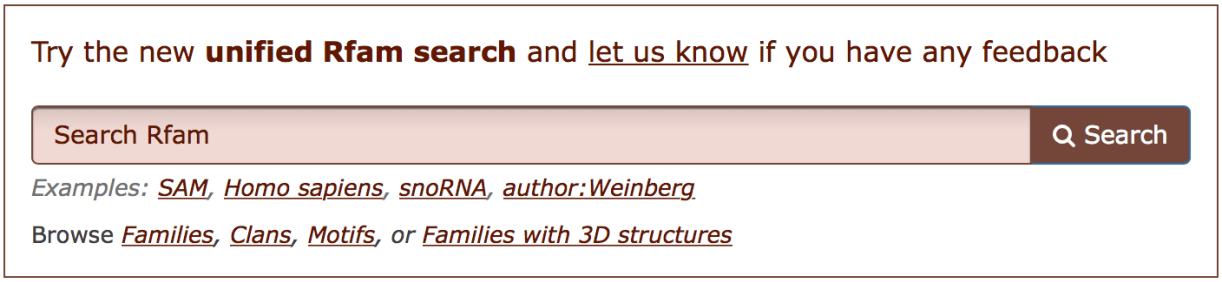
1.7 Searching Rfam

In addition to the quick links on the home page, every page in the Rfam site includes a Search link in the page header, which you can use to access all of the search methods that we offer:

- *"Jump to" search*

Hint: For more details about searching Rfam, please see our paper in Current Protocols in Bioinformatics.

1.7.1 Unified text search



The new text search, available on the Rfam homepage or at the top of any Rfam page, will soon replace older search options, such as *Keyword search*, *Taxonomy search*, *browsing* entries by type, and *Jump To* navigation.

Using the new search one can:

- explore Rfam by category using **facets**
- **sort** results
- **bookmark and share** search URLs

Examples:

- families,
- clans,
- motifs,
- families with 3D structures
- snoRNA families that match human sequences

The new search is a **full replacement** for most of the old search functionality except for taxonomy as the new search can currently find only species but not higher taxa (for example, one can search for *Homo sapiens* but not *Mammals*). Stay tuned for future updates and use the old Taxonomy search in the meantime.

Text search API

Text search is powered by the EBI search which supports a REST API that can be used to access the Rfam data programmatically in addition to the *Rfam API*.

Here is an example query that retrieves riboswitch families as well as their descriptions and the number of sequences in seed alignments:

```
https://www.ebi.ac.uk/ebisearch/ws/rest/rfam?query=riboswitch&format=json&fields=num_
↪seed,description
```

Here is full list of fields that can be retrieved using the text search API.

1.7.2 Sequence search

Searching a nucleotide sequence (DNA or RNA) against the Rfam library of covariance models will identify any regions in your sequence we would classify as belonging to one our RNA families.

There are two ways of running sequence searches:

1. Rfam website
2. EBI cmscan service

Both websites use the same version of Infernal and the same set of Rfam covariance models.

Single sequence search

If your sequence is found in a genome on which *rfamseq* is based, your sequence will already be searched and annotated. You can paste the Genbank/EMBL accession in the text search or the "look up sequence box" on the sequence search page or a "jump to" box. The accession version number is not required.

Medium scale batch searches (less than 1,000 sequences)

If you have multiple nucleotide sequences to search, you can use our batch upload facility to upload a file of your sequences in FASTA format. Information on the format for this file can be found under the more link here. We will search your sequences against the Rfam library of covariance models and email the results back to you, usually within 48 hours. We request that you search a maximum of 1000 sequences in each file. Each sequence may be up to 200kb in length.

Large scale batch searches (more than 1,000 sequences)

If you have a large number of nucleotide searches, it may be more convenient to run Infernal searches locally (see section *Genome annotation*).

1.7.3 Other ways to search Rfam

Keyword search

> **Warning:** The old keyword search will soon be replaced by the Unified text search.

Each page in the Rfam site contains a keyword search box in the header. This is the broadest text search we offer and you can use this to find all Rfam families that match a particular keyword. The search will try to match your query term against textual information from several different sections of the Rfam database:

- text fields for Rfam families, such as family descriptions and identifiers
- Rfam associated Wikipedia entries

- literature reference titles and authors
- PDB structures

Your keyword should be a simple text string (letters and numbers), but underscores, hyphens, periods and spaces are also accepted. Wildcard terms are not necessary, since the search system will add wildcards to the end of your search terms. If in doubt, use the shortest text string you can and you will receive the widest set of possible matches. You can then sort the results and refine your search if needed.

Do remember that the keyword search tries to match against all of the sections of the database, including the Wikipedia article, so if your term is mentioned in the family description text, you will also get a match.

If you search with two terms at once your will only receive a result if a match is found for both terms.

Search results page

Your query term is reported and, if the term you used exactly matched a family ID or accession, this is also reported. This text is followed by a small table that provides a summary showing in which section of the database your query string was found.

The larger table that follows provides links to the families that have a match to your query in at least one section of the database. Each matching family is listed only once, though it may have matches in more than one section of the database. For each family with a match we report:

- accession (linked to the the family page)
- identifier (linked to the family page)
- family description line
- between one and four columns that specify in which of the sections of the database the match was found

If your query term does not match any data in the database, you will be taken to a 'no results' page which will offer you tips on how to refine your search.

Search by entry type

> **Warning:** Entry type search will soon be replaced by the Unified text search.

You can search by entry type to view or download a list of families by type.

Here is a list of Rfam ncRNA types:

- Cis-reg;
  - Cis-reg; IRES;
  - Cis-reg; frameshift_element;
  - Cis-reg; leader;
  - Cis-reg; riboswitch;
  - Cis-reg; thermoregulator;
- Gene;
  - Gene; CRISPR;
  - Gene; antisense;

  - Gene; miRNA;
  - Gene; rRNA;
  - Gene; ribozyme;
  - Gene; sRNA;
  - Gene; snRNA;
  - Gene; snRNA; snoRNA; CD-box;
  - Gene; snRNA; snoRNA; HACA-box;
  - Gene; snRNA; snoRNA; scaRNA;
  - Gene; snRNA; splicing;
  - Gene; tRNA;
- Intron;

Tip: If you would like to download results as text, click **Show the unformatted list** at the bottom of the search results page.

Taxonomy search

Warning: Taxonomy seach search will soon be replaced by the Unified text search.

This is one of the more interesting and powerful ways to search Rfam. Using the taxonomy search form, you can identify families that are specific to a given taxonomic level or those found in a given set of taxonomic levels. You can also limit your queries to those families which are found only in a single species or taxonomic level. Please read the information under the "More…" link on the taxonomy search page for details on how to use this search.

Exploring families by name

The Browse link at the top of each page will take you to an index page, from which you can browse all Rfam families by their family names (otherwise known as the Rfam IDs). These are the familiar names for the RNA, such as "tRNA" or "Hammerhead_1". The families are organised alphabetically and you can use the ranges (A-F, G-L etc) to take you to the appropriate place in the list. Families where the name begins with a number (e.g. "6S", "7SK") can be found under the 0-9 index.

"Jump to" search

Warning: "Jump to" search will soon be replaced by the Unified text search.

Many pages in the site include a small search box, entitled "Jump to…". The "Jump to…" box allows you to go immediately to the page for any entry in the Rfam site. This is primarily useful when you know the family or the sequence accession you are interested in.

The "Jump to…" search understands Genbank/EMBL accessions, Rfam family accessions and identifiers for most types of entry. For example, to find a particular family, you can enter either an Rfam family accession, e.g. **RF00198**, or, if you find it easier to remember, a family ID, such as **SL1**. This will take you to the main entry for this family. Note that the search is case insensitive. Searches for family identifiers such as 'RNase' or 'mrp' will be too ambiguous and you will get an error "Couldn't guess entry". In this case you need to specify the the full family name, e.g. RNase_mrp'. If you want to search with an ambiguous family identifier use the keyword search instead.

Alternatively, if you are interested in the annotations to a particular sequence or genome you can use the Genbank/EMBL accession, e.g. **AE017225** and you will be taken to a list of the relevant Rfam family annotations to this sequence. This also works for EMBL CON files, e.g. **CM000428**.

The order in which the search tries to match your query term against the various types of ID and accession in the database is:

- Rfam accession, e.g. **RF00198**
- Rfam identifier, e.g. **SL1**
- Genome Genbank/EMBL accession, e.g. **AE017225**
- Sequence Genbank/EMBL accession e.g. **AF325543**

If all of the guesses fail, you'll see an error message saying "Entry not found".

1.8 Rfam FTP Site

The following list describes a few of the important files in the Rfam FTP site. Some of these files may be very large (of the order of several hundred megabytes). Please check the sizes before trying to download them over a slow connection.

1.8.1 Documentation

README Release Notes

COPYING Public Domain Information for Rfam

USERMAN A description of the Rfam flatfile formats

1.8.2 Sequences, Alignments, Models and Trees

Rfam.tar.gz Rfam covariance models in ascii INFERNAL format

Rfam.seed.gz Annotated seed alignments in STOCKHOLM format

Rfam.seed_tree.tar.gz Annotated tree files for each seed alignment

Rfam.full_region.gz List of sequence regions making up the full family membership for each family

fasta_files Directory containing the sequences for all significant hits per family

1.8.3 Rfam database dumps

database_files Directory containing MYSQL dump of the the Rfam database data, tables and mysql database schema

Hint: For direct access to the database please visit *Public MySQL Database*

1.9 Rfam API

Most data in Rfam can be accessed programmatically using a RESTful API allowing for integration with other resources.

Hint: You can also access the data using a *Public MySQL Database* that contains the latest Rfam release.

1.9.1 Data access

The data can be accessed in several formats which can be specified in the URL:

- HTML http://rfam.org/family/RF00360
- JSON http://rfam.org/family/RF00360?content-type=application/json
- XML http://rfam.org/family/RF00360?content-type=text/xml

Using *curl*

Here is how to retrieve an XML description of an Rfam family using curl:

```
curl http://rfam.org/family/snoZ107_R87?content-type=text%2Fxml
```

Output:

```
<?xml version="1.0" encoding="UTF-8"?>
<!-- information on Rfam family RF00360 (snoZ107_R87), generated: 12:57:01 31-Oct-
↪2016 -->
<rfam xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
      xmlns="http://rfam.sanger.ac.uk/"
      xsi:schemaLocation="http://rfam.sanger.ac.uk/
                          http://rfam.sanger.ac.uk/static/documents/schemas/entry_xml.
↪xsd"
      release="12.1"
      release_date="2016-04-26">
  <entry entry_type="Rfam" accession="RF00360" id="snoZ107_R87">
    <description>
<![CDATA[
Small nucleolar RNA Z107/R87
]]>
    </description>
    <comment>
<![CDATA[
Z107 and R87 are members of the C/D class of snoRNA which contain the C (UGAUGA) and␣
↪D (CUGA) box motifs. Most of the members of the box C/D family function in␣
↪directing site-specific 2'-O-methylation of substrate RNA
]]>
    </comment>
    <curation_details>
      <author>Moxon SJ</author>
      <seed_source>Moxon SJ</seed_source>
      <num_seqs>
        <seed>9</seed>
        <full>144</full>
      </num_seqs>
      <num_species>37</num_species>
      <type>Gene; snRNA; snoRNA; CD-box;</type>
      <structure_source>Predicted; RNAfold; Moxon SJ, Daub J, Gardner PP</structure_
↪source>
    </curation_details>
```

(continues on next page)

(continued from previous page)

```
    <cm_details num_states="">
      <build_command>cmbuild -F CM SEED</build_command>
      <calibrate_command>cmcalibrate --mpi CM</calibrate_command>
      <search_command>cmsearch --cpu 4 --verbose --nohmmonly -T 19 -Z 549862.597050␣
↪CM SEQDB</search_command>
      <cutoffs>
        <gathering>50.0</gathering>
        <trusted>50.2</trusted>
        <noise>49.8</noise>
      </cutoffs>
    </cm_details>
  </entry>
</rfam>
```

Using a script

Rfam API can also be used from a script written in any programming language, for example Python or Perl.

Python example script

```
import json
import requests

r = requests.get('http://rfam.org/family/RF00360?content-type=application/json')
print r.json()['rfam']['acc']
```

Perl example script

```
#!/usr/bin/perl

use strict;
use warnings;

use LWP::UserAgent;

my $ua = LWP::UserAgent->new;
$ua->env_proxy;

my $res = $ua->get(' http://rfam.org/family/snoZ107_R87?content-type=text%2Fxml' );

if ( $res->is_success ) {
  print $res->content;
}
else {
  print STDERR $res->status_line, "\n";
}
```

1.9.2 Endpoints

Family

Family description

Returns general information about an Rfam family, such as curation details, search parameters, etc.

Examples:

- http://rfam.org/family/RF00360?content-type=text/xml
- http://rfam.org/family/snoZ107_R87?content-type=application/json

Accession to ID

Returns the ID for the family with the given Rfam accession or ID.

Example:

http://rfam.org/family/snoZ107_R87/acc

Example output:

```
RF00360
```

ID to accession

Example output:

http://rfam.org/family/RF00360/id

Output:

```
snoZ107_R87
```

Secondary structure images

Returns the schematic secondary structure image for the family. The following types of secondary structure diagrams are supported:

- *cons* (sequence conservation)
- *fcbp* (basepair conservation)
- *cov* (covariation)
- *ent* (relative entropy)
- *maxcm* (maximum CM parse)
- *norm* (normal)
- *rscape* (R-scape[1] analysis of Rfam SEED alignment)
- *rscape-cyk* (secondary structure predicted by R-scape[1] based on Rfam SEED alignment)

Examples:

- http://rfam.org/family/snoZ107_R87/image/norm
- http://rfam.org/family/RF00360/image/cov

[1] http://eddylab.org/R-scape/

- http://rfam.org/family/RF00360/image/rscape
- http://rfam.org/family/RF00360/image/rscape-cyk

Covariance models

Returns the covariance model for the specified family.

Example: http://rfam.org/family/RF00360/cm

Sequence regions

Returns the list of all sequence regions for the specified families in tab-delimited format.

Note: Some families have too many regions to list. The server will return a status of `403 Forbidden` in these cases.

Examples:

- http://rfam.org/family/snoZ107_R87/regions (plain text)
- http://rfam.org/family/RF00360/regions?content-type=text%2Fxml

---

Phylogenetic trees

Tree data

Returns the raw data for the phylogenetic tree in NHX format based on seed alignment.

Example: http://rfam.org/family/RF00360/tree/

Tree image

Returns a PNG image showing the phylogenetic tree for the specified family based on seed alignment. The image can be labelled either using **species names** or **sequence accessions**.

Examples:

- http://rfam.org/family/RF00360/tree/label/species/image
- http://rfam.org/family/RF00360/tree/label/acc/image

Tree image map

Returns the HTML image map that is used in conjunction with the tree image to highlight tree nodes in the Rfam website.

Example:

- http://rfam.org/family/RF00360/tree/label/acc/map
- http://rfam.org/family/RF00360/tree/label/species/map

Note: The HTML snippet contains an `<img>` tag that automatically loads the tree image.

Structure mapping

Returns the mapping between an Rfam family, EMBL sequence regions and PDB residues. The plain text file has a tab-delimited format.

Examples:

- http://rfam.org/family/RF00002/structures (HTML)
- http://rfam.org/family/RF00002/structures?content-type=application/json
- http://rfam.org/family/RF00002/structures?content-type=text/xml

Alignments

The following methods can be used to return family alignments in various formats.

Hint: You can request a compressed version of the alignment by adding `gzip=1` to the URL.

Stockholm-format alignment

Returns the Stockholm-format seed alignment for the specified family.

Examples:

- http://rfam.org/family/RF00360/alignment
- http://rfam.org/family/RF00360/alignment?gzip=1

Formatted alignment

Returns the seed alignment for the specified family in one of the following formats:

- *stockholm* (standard Stockholm format - default)
- *pfam* (Stockholm with sequences on a single line conservation)
- *fasta* (gapped FASTA format)
- *fastau* (ungapped FASTA format)

Examples:

- http://rfam.org/family/RF00360/alignment/stockholm
- http://rfam.org/family/RF00360/alignment/pfam
- http://rfam.org/family/RF00360/alignment/fasta
- http://rfam.org/family/snoZ107_R87/alignment/fastau

1.9.3 Sequence searches

In addition to a sequence search user interface, it is possible to run single-sequence Rfam searches programmatically.

Running a search is a two step process:

1. submit the search sequence
2. retrieve search results

The reason for separating the operation into two steps rather than performing a search in a single operation is that the time taken to perform a sequence search will vary according to the length of the sequence searched. Most web clients, browsers or scripts, will simply time-out if a response is not received within a short time period, usually less than a minute. By submitting a search, waiting and then retrieving results as a separate operation, we avoid the risk of a client reaching a time-out before the results are returned.

The following example uses simple command-line tools to submit the search and retrieve results, but the whole process is easily transferred to a single script or program.

Save your sequence to file

It is usually most convenient to save your sequence into a plain text file, something like this:

```
$ cat test.seq
AGTTACGGCCATACCTCAGAGAATATACCGTATCCCGTTCGATCTGCGAA
GTTAAGCTCTGAAGGGCGTCGTCAGTACTATAGTGGGTGACCATATGGGA
ATACGACGTGCTGTAGCTT
```

The sequence should contain only valid sequence characters. You can break the sequence across multiple lines to make it easier to handle.

Submit the search

When you send a request to the server, you can specify the format of the response. The server supports JSON (application/json) and XML (text/xml) output. In the examples below we'll use the JSON output format by adding an `Accept` header to the request, specifying the media type `application/json`. You could use the "content-type" parameter on the URL, rather than setting a header.

```
curl -H 'Expect:' -F seq='<test.seq' -H "Accept: application/json" http://rfam.org/
↪search/sequence
```

Example output:

```
{
  "resultURL": "http://rfam.org/search/sequence/d9b451d8-96e6-4234-9dbb-aa4806925353",
  "opened": "2016-10-31 13:19:06",
  "estimatedTime": "3",
  "jobId": "d9b451d8-96e6-4234-9dbb-aa4806925353"
}
```

Wait for the search to complete

Having submitted the search, you now need to check the `resultURL` given in the response, which will be the URL that you used for submitting the search, but with a job identifier appended.

Although you can check for results immediately, if you poll before your job has completed you won't receive a full response. Instead, the HTTP response will have its status set appropriately and the body of the response will contain only string giving the status. You should ideally check the HTTP status of the response, rather than relying on the body of the response. See below for a table showing the response status codes that the server may return.

When writing a script to submit searches and retrieve results, **please add a short delay** between the submission and the first attempt to retrieve results. Most search jobs are returned within four to five seconds of submission, depending greatly on the length of the sequence to be searched. The `estimatedTime` given in the response provides a very rough estimate of how long your job should take. You may want to wait for this period before polling for the first time.

Retrieve results

The response that was returned from the first query includes a URL from which you can now retrieve results:

```
curl -H "Expect:" -H "Accept: application/json" http://rfam.org/search/sequence/
↪01d3c704-591a-4a85-b7c1-366496c5a63
```


```
{
      "closed": "2016-10-31 13:20:29",
      "searchSequence":
↪"AGTTACGGCCATACCTCAGAGAATATACCGTATCCCGTTCGATCTGCGAAGTTAAGCTCTGAAGGGCGTCGTCAGTACTATAGTGGGTGACCATATGC
↪",
      "hits": {
               "5S_rRNA": [{
                        "score": "104.9",
                        "E": "2.7e-24",
                        "acc": "RF00001",
                        "end": "119",
                        "alignment": {
                                 "user_seq": "#SEQ            1␣
↪AGUUACGGCCAUACCUCAGAGAAUAUACCGUAUCCCGUUCGAUCUGCGAAGUUAAGCUCUGAAGGGCGUCGUCAGUACUAUAGUGGGUGACCAUAUGGC
↪119        ",
                                 "hit_seq": "#CM             1␣
↪gccuGcggcCAUAccagcgcgaAagcACcgGauCCCAUCcGaACuCcgAAguUAAGcgcgcUugggCcagggUAGUAcuagGaUGgGuGAcCuCcUGgc
↪119        ",
                                 "ss": "#SS               (((((((((,,,,<<-<<<<<---<<--<<<<
↪<<_______>>-->>>>--->>----->>>>>-->>>><<-<<----<-<<-----<<____>>----->>->-->>->>>
↪))))))))):            ",
                                 "match": "#MATCH            :: U:C:GCCAUACC ::G:GAA␣
↪::ACCG AUCCC+U+CGA CU CGAA::UAAGC:C:: +GGGC: :G  AGUACUA  +UGGGUGACC+ ␣
↪UGGGAA+AC:A:GUGC:G:A ::+            ",
                                 "pp": "#PP                                  ␣
↪*************************************************************************************************
↪           ",
                                 "nc": "#NC                                                          ␣
↪                                                                                                   ␣
↪              "
                        },
                        "strand": "+",
                        "id": "5S_rRNA",
                        "GC": "0.49",
```


(continues on next page)

(continued from previous page)


```
                    "start": "1"
                }]
        },
        "opened": "2016-10-31 13:19:06",
        "numHits": 1,
        "started": "2016-10-31 13:20:08",
        "jobId": "99676096-9F6C-11E6-9647-5251D1B96DDE"
}
```


Warning: Old search results are regularly cleared out but results will be visible for **one week** after completion of the original search.

Server responses

Server responses include a standard HTTP status code giving information about the current state of your job. These are the possible status codes:

HTTP method	HTTP status code	Status description	Response body	Notes
POST	202	Accepted	PEND / RUN	The job has been accepted by the search system and is either pending (waiting to be started) or running. After a short delay, your script should check for results again.
POST	502	Bad gateway	Error message	There was a problem scheduling or running the job. The job has failed and will not produce results. There is no need to check the status again.
POST	503	Service unavailable	Error message	Occasionally the search server may become overloaded. If the error message suggests that the search queue is full, try submitting your search later.
GET	200	OK	Search results	The job completed successfully and the results are included in the response body.
GET	410	Gone	DEL	Your job was deleted from the search system. This status will not be assigned by the search system, but by an administrator. There was probably a problem with the job and you should contact the help desk for assistance with it.
GET	503	Service unavailable	HOLD	Your job was accepted but is on hold. This status will not be assigned by the search system, but by an administrator. There is probably a problem with the job and you should contact the help desk for assistance with it.
GET, POST	500	Internal server error	Error message	There was some problem accepting or running your job, but it does not fall into any of the other categories. The body of the response will contain an error message from the server. Contact the help desk for assistance with the problem.

1.9.4 Links

1.10 Public MySQL Database

Rfam provides a public read-only MySQL database containing the latest version of Rfam data. The database will be updated with each release. To access old versions of the database download SQL dumps from the FTP archive.

Hint: For advanced examples of using the public database, please see our paper in Current Protocols in Bioinformatics.

1.10.1 Connection details

Parameter	Value
host	mysql-rfam-public.ebi.ac.uk
user	rfamro
password	none
port	4497
database	Rfam

You can connect to the database on command line:

```
mysql --user rfamro --host mysql-rfam-public.ebi.ac.uk --port 4497 --database Rfam
```

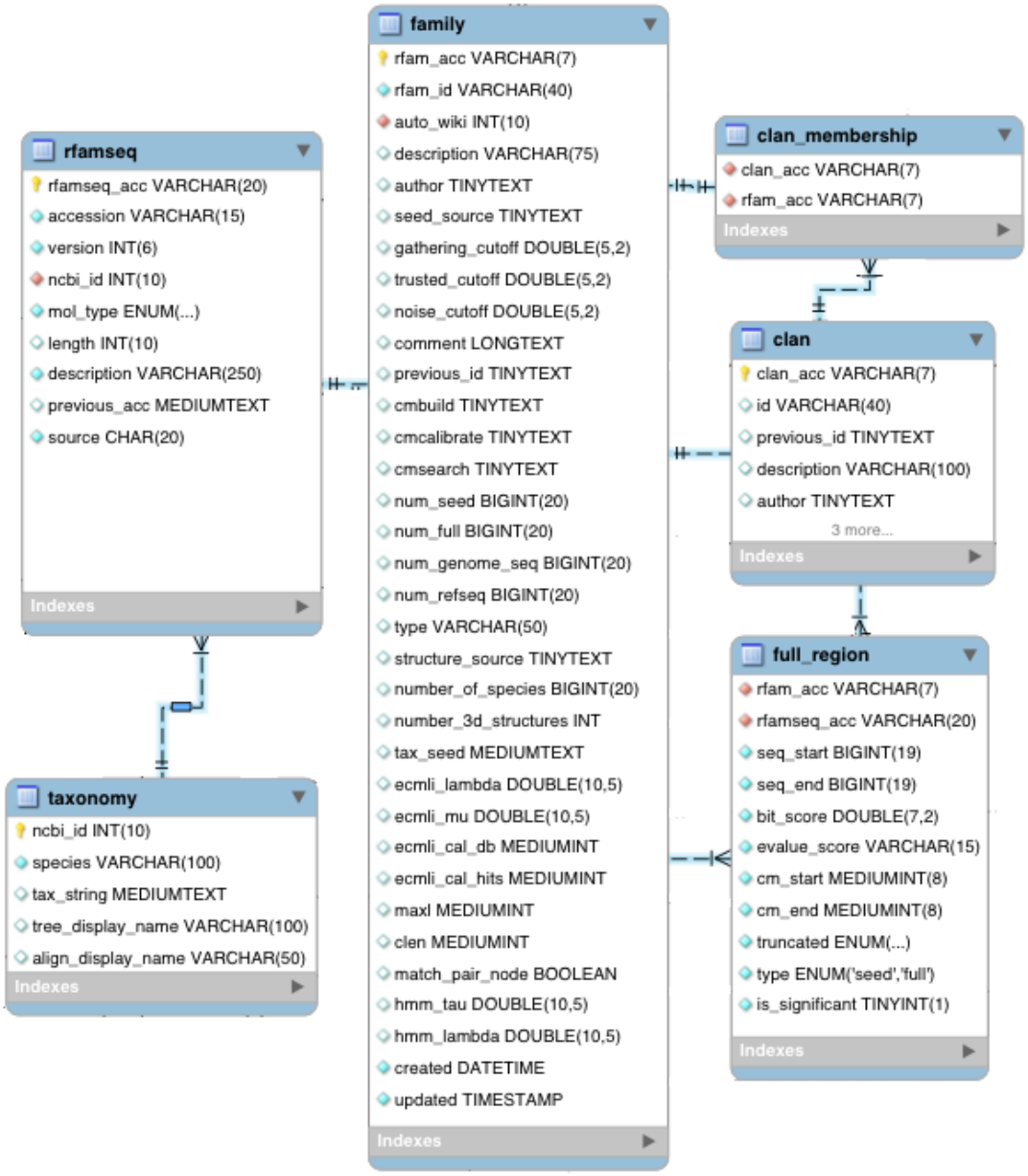
or use MySQL clients such as MySQL Workbench or Sequel Pro.

If your computer is behind a firewall, please ensure that outgoing TCP/IP connections to the corresponding ports are allowed.

1.10.2 Main tables

The most important tables are listed below and can be used as starting points for exploring the schema:

Table	Description
family	a list of all Rfam families and family specific information (family accession, family name, description etc.)
rfamseq	a list of all analysed sequences including INSDC accessions, taxonomy id etc.
full_region	a list of all sequences annotated with Rfam families including INSDC accessions, start and end coordinates, bit scores etc.
clan	description of all Rfam clans
clan_membership	a list of all Rfam families per clan
taxonomy	NCBI taxonomy identifiers



1.10.3 Example queries

Retrieve all rat sequence coordinates annotated with Rfam families

While it is possible to get a list of Rfam families found in a species using the taxonomy search, with an SQL query one can access sequence coordinates of each ncRNA:

```
SELECT fr.rfam_acc, fr.rfamseq_acc, fr.seq_start, fr.seq_end
FROM full_region fr, rfamseq rf, taxonomy tx
WHERE rf.ncbi_id = tx.ncbi_id
AND fr.rfamseq_acc = rf.rfamseq_acc
```

(continues on next page)

(continued from previous page)

```
AND tx.ncbi_id = 10116 -- NCBI taxonomy id of Rattus norvegicus
AND is_significant = 1 -- exclude low-scoring matches from the same clan
```

Example output:

```
RF01942        AABR05000009.1  211     327
RF00005        AABR05000052.1  4940    5008
```

Retrieve all snoRNA families found in Mammals

```
SELECT fr.rfam_acc, fr.rfamseq_acc, fr.seq_start, fr.seq_end, f.type
FROM full_region fr, rfamseq rf, taxonomy tx, family f
WHERE
rf.ncbi_id = tx.ncbi_id
AND f.rfam_acc = fr.rfam_acc
AND fr.rfamseq_acc = rf.rfamseq_acc
AND tx.tax_string LIKE '%Mammalia%'
AND f.type LIKE '%snoRNA%'
AND is_significant = 1 -- exclude low-scoring matches from the same clan
```

Example output:

```
RF00012        AAYZ01671298.1  83      298     Gene; snRNA; snoRNA; CD-box;
RF00012        AAYZ01122278.1  302     87      Gene; snRNA; snoRNA; CD-box;
```

1.11 Genome annotation

The Rfam library of covariance models can be used to search sequences (including whole genomes) for homologues to known non-coding RNAs, in conjunction with the Infernal software.

Before trying to annotate your own genome sequences on your local hardware or submitting lots of sequences to Rfam via the website, please check that the following resources do not provide the annotation for you:

- Ensembl
- Ensembl Genomes
- UCSC Genome Browser

Hint: For more details about genome annotation, please see our paper in Current Protocols in Bioinformatics.

1.11.1 Example of using Infernal and Rfam to annotate RNAs in an archaeal genome

The instructions below will walk you through how to annotate the *Methanobrevibacter ruminantium* genome (NC_013790.1) for non-coding RNAs using Rfam and Infernal. The files needed are included in the Infernal software package, which you will download in step 1.

1. Download, build and install Infernal from http://eddylab.org/infernal/

```
$ wget eddylab.org/infernal/infernal-1.1.2.tar.gz
$ tar xf infernal-1.1.2.tar.gz
$ cd infernal-1.1.2
$ make
```

If you do not have `wget` installed and in your path, download infernal-1.1.2.tar.gz here.

To compile and run a test suite to make sure all is well, you can optionally do:

```
$ make check
```

You don't have to install Infernal programs to run them. The newly compiled binaries are now in the *src* directory. You can run them from there. To install the programs and man pages somewhere on your system, do:

```
$ make install
```

By default, programs are installed in */usr/local/bin* and man pages in */usr/local/share/man/man1/*. You can change the */usr/localprefix* to any directory you want using the `./configure --prefix` option, as in `./configure --prefix /the/directory/you/want`.

Additional programs from the **Easel** library are available in *easel/miniapps/*. You can install these too if you'd like. Step 4 below involves the use of one of these Easel programs (esl-seqstat). If you do not install these programs, you can use the executable files in *easel/miniapps/*. To install them:

```
$ cd easel; make install
```

For more information on customizing the Infernal installation, see section 2 of the Infernal User's Guide.

2. Download the Rfam library of CMs from ftp://ftp.ebi.ac.uk/pub/databases/Rfam/CURRENT/Rfam.cm.gz and the Rfam clanin file from ftp://ftp.ebi.ac.uk/pub/databases/Rfam/CURRENT/Rfam.clanin .

```
$ wget ftp://ftp.ebi.ac.uk/pub/databases/Rfam/CURRENT/Rfam.cm.gz
$ gunzip Rfam.cm.gz
$ wget ftp://ftp.ebi.ac.uk/pub/databases/Rfam/CURRENT/Rfam.clanin
```

If you do not have `wget` installed and in your path, download the files ftp://ftp.ebi.ac.uk/pub/databases/Rfam/CURRENT/Rfam.cm.gz and ftp://ftp.ebi.ac.uk/pub/databases/Rfam/CURRENT/Rfam.clanin from a browser.

3. Use the Infernal program cmpress to index the Rfam.cm file

```
$ cmpress Rfam.cm
```

This step is required before cmscan can be run in step 5.

4. Determine the total database size for the genome you are annotating.

For the purposes of Infernal, the total database size is the number of nucleotides that will be searched, in units of megabases (Mb, millions of nucleotides). So, it is the **total number of nucleotides** in all sequences that make up the genome, **multiplied by two** (because both strands will be searched), and **divided by 1,000,000** (to convert to millions of nucleotides).

You will need to supply this number to Infernal to assure that the E-values reported by the cmscan program run in the next step are accurate.

You can use the esl-seqstat program from the Easel library that you built along with Infernal in step 1 to help with this. For this example, we will be annotating the genome of *Methanobrevibacter ruminantium*, an archaeon. The sequence file with this genome can be found in *infernal-1.1.2/tutorial/*, which you created in step 1. To determine the total size of this genome, do:

```
$ esl-seqstat infernal-1.1.2/mrum-genome.fa
```

Note: If you did not install the Easel miniapps in step 1, you can run `esl-seqstat` from infernal-1.1.2/easel/miniapps/esl-seqstat.

The output will include a line reporting the total number of nucleotides:

```
Total # of residues: 2937203
```

Because we want millions of nucleotides on both strands, we multiple this by 2, and divide by 1,000,000 to get 5.874406. This number will be used in step 5.

5. Use the cmscan program to annotate RNAs represented in Rfam in the *Methanobrevibacter ruminantium* genome.

```
$ cmscan -Z 5.874406 --cut_ga --rfam --nohmmonly --tblout mrum-genome.tblout --fmt 2 -
↪-clanin Rfam.clanin Rfam.cm tutorial/mrum-genome.fa > mrum-genome.cmscan
```

Note: The above cmscan command assumes you are in the infernal-1.1.2 directory from step 1. If not, you'll need to supply the paths to the tutorial/mrum-genome.fa and file within the infernal-1.1.2 directory.

Explanations of the command line options used in the above command are as follows:

`-Z 5.874406` the sequence database size in millions of nucleotides is 5.874406, it is the number computed in step 4. This option ensures that the reported E-values are accurate.

`--cut_ga` specifies that the special Rfam GA (gathering) thresholds be used to determine which hits are reported. See more in the section *Gathering cutoff*.

`--rfam` run in "fast" mode, the same mode used for Rfam annotation and determination of GA thresholds

`--nohmmonly` all models, even those with zero basepairs, are run in CM mode (not HMM mode). This ensures all GA cutoffs, which were determined in CM mode for each model, are valid.

`--tblout` a tabular output file will be created.

`--fmt 2` the tabular output file will be in format 2, which includes annotation of overlapping hits.

`--clanin` Clan information should be read from the file `Rfam.clanin`. This file lists which models belong to the same clan. Rfam clans are groups of models that are homologous and therefore it is expected that some hits to these models will overlap. For example, the LSU rRNA archaea and LSU rRNA bacteria models are both in the same clan.

6. Remove hits from the tabular output file that have overlapping hits with better scores. This step is explained below after a discussion of the cmscan output, in the section: *Removing lower-scoring overlaps from a tblout file*.

1.11.2 Understanding Infernal output

The above cmscan command will take at least several minutes and possibly up to about 30 minutes depending on the number of cores and speed of your computer. After it has finished, you will have two output files: `mrum-genome.cmscan` (standard output of cmscan) and `mrum-genome.tblout` (tabular output).

cmscan standard output

The first section of Infernal program's standard output is the header, telling you what program you ran, on what, and with what options:

```
# cmscan :: search sequence(s) against a CM database
# INFERNAL 1.1.2 (July 2016)
# Copyright (C) 2016 Howard Hughes Medical Institute.
# Freely distributed under a BSD open source license.
# - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - -
# query sequence file:                  /Users/nawrockie/src/infernal-1.1.2/tutorial/
↪mrum-genome.fa
# target CM database:                   Rfam.cm
# database size is set to:              5.9 Mb
# tabular output of hits:               mrum-genome.tblout
# tabular output format:                2
# model-specific thresholding:          GA cutoffs
# Rfam pipeline mode:                   on [strict filtering]
# clan information read from file:      Rfam12.2.claninfo
# HMM-only mode for 0 basepair models:  no
# number of worker threads:             8
# - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - -
```

The second section is a list of ranked top hits (sorted by E-value, most significant hit first). For `cmscan` output this section is broken down per-query sequence. In this example, there is only one sequence NC_013790.1. Here is the list of the top 25 hits (out of 78 total):

```
Query:       NC_013790.1  [L=2937203]
Description: Methanobrevibacter ruminantium M1 chromosome, complete genome
Hit scores:
 rank     E-value  score  bias  modelname                start     end   mdl trunc  ␣
↪gc  description
 ----   --------- ------ -----  -------------------- ------- -------   --- ----- ---
↪-  -----------
   (1) !         0 2763.5  45.1  LSU_rRNA_archaea        762872  765862 +  cm    no 0.
↪49  -
   (2) !         0 2755.0  46.1  LSU_rRNA_archaea       2041329 2038338 -  cm    no 0.
↪48  -
   (3) !         0 1872.9  45.1  LSU_rRNA_bacteria       762874  765862 +  cm    no 0.
↪49  -
   (4) !         0 1865.5  46.2  LSU_rRNA_bacteria      2041327 2038338 -  cm    no 0.
↪48  -
   (5) !         0 1581.3  41.5  LSU_rRNA_eukarya        763018  765851 +  cm    no 0.
↪49  -
   (6) !         0 1572.1  42.3  LSU_rRNA_eukarya       2041183 2038349 -  cm    no 0.
↪49  -
   (7) !         0 1552.0   4.1  SSU_rRNA_archaea       2043361 2041888 -  cm    no 0.
↪53  -
   (8) !         0 1546.5   4.1  SSU_rRNA_archaea        760878  762351 +  cm    no 0.
↪54  -
   (9) !         0 1161.9   3.7  SSU_rRNA_bacteria      2043366 2041886 -  cm    no 0.
↪53  -
  (10) !         0 1156.4   3.7  SSU_rRNA_bacteria       760873  762353 +  cm    no 0.
↪53  -
  (11) !  9.9e-293  970.4   4.6  SSU_rRNA_eukarya        2043361 2041891 -  cm    no 0.
↪53  -
  (12) !  9.9e-291  963.8   4.5  SSU_rRNA_eukarya         760878  762348 +  cm    no 0.
↪54  -
```

(continues on next page)

(continued from previous page)

```
 (13) !  7.7e-281  919.9   4.6  SSU_rRNA_microsporidia 2043361 2041891 -  cm    no 0.
↪53  -
 (14) !  5.4e-280  917.2   4.5  SSU_rRNA_microsporidia  760878  762348 +  cm    no 0.
↪54  -
 (15) !   1.1e-53  184.9   0.0  RNaseP_arch           2614544 2614262 -  cm    no 0.
↪43  -
 (16) !   6.9e-49  197.6   0.1  Archaea_SRP           1064321 1064634 +  cm    no 0.
↪44  -
 (17) !   6.8e-28  115.2   0.0  FMN                    193975  193837 -  cm    no 0.
↪42  -
 (18) !   4.9e-16   72.1   0.0  tRNA                   735136  735208 +  cm    no 0.
↪59  -
 (19) !     1e-15   71.0   0.0  tRNA                  2350593 2350520 -  cm    no 0.
↪66  -
 (20) !   1.1e-15   70.9   0.0  tRNA                  2680310 2680384 +  cm    no 0.
↪52  -
 (21) !   2.2e-15   69.7   0.0  tRNA                  2351254 2351181 -  cm    no 0.
↪62  -
 (22) !   2.5e-15   69.5   0.0  tRNA                   361676  361604 -  cm    no 0.
↪51  -
 (23) !   3.2e-15   69.2   0.0  tRNA                  2585265 2585193 -  cm    no 0.
↪60  -
 (24) !   3.9e-15   68.8   0.0  tRNA                  2585187 2585114 -  cm    no 0.
↪59  -
 (25) !   4.3e-15   68.7   0.0  tRNA                  2680159 2680233 +  cm    no 0.
↪67  -
```

The most important columns here are those labelled "E-value", "score", "modelname", "start" and "end", which are described below. For information on the other columns see the tutorial section (pages 18-19) of the Infernal User's Guide).

E-value The E-value is the statistical significance of the hit: the number of hits we'd expect to score this highly in a database of this size (measured by the total number of nucleotides) if the database contained only nonhomologous random sequences. The lower the E-value, the more significant the hit.

score The E-value is based on the bit score, which is in the "score" column. This is the log-odds score for the hit. Some people like to see a bit score instead of an E-value, because the bit score doesn't depend on the size of the sequence database, only on the covariance model and the target sequence. All reported hits here are above the model-specific Rfam GA bit score for that model because we used the `--cut_ga` option to cmscan.

modelname The name of the Rfam family/model this hit is to. The accession is not listed in this output, but is listed in the tabular output file, explained below.

start The start (first) position of the hit in the query sequence.

stop The stop (final) position of the hit in the query sequence. Immediately after this column is a single character denoting the strand of the hit: + for positive (Watson) strand and − for negative (Crick) strand. Also, for positive strand hits, the start position will always be less than or equal to the stop position, and for negative strand hits, the start position will always be greater than or equal to the stop position.

You may have noticed that some of these hits overlap with each other. For example, the LSU_rRNA_archaea and LSU_rRNA_bacteria hits from 762872-765862 and 762874-765862 almost completely overlap. This is because both models recognized this archael LSU rRNA sequence in this genome. Note that the LSU_rRNA_archaea score (2763.5 bits) is better than the LSU_rRNA_bacteria score (1872.9) indicating that the LSU_rRNA_archaea model is a better match (even though both hits have an E-value of 0).

When dealing with overlapping hits, the general recommendation is to keep the hit amongst all overlapping hits that has the best (lowest) E-value. If the E-values are equal, keep the hit with the highest bit score. In the tabular output file (discussed below), overlapping hits are annotated, making it easy to remove lower scoring overlaps, as explained in the section: *Removing lower-scoring overlaps from a tblout file*.

After the list of hits you will find the hit alignments for each hit. Each alignment is preceded by a summary of each hit. For hit #33, a tRNA hit (RF00005):

```
>> tRNA
 rank     E-value  score  bias mdl mdl from   mdl to       seq from      seq to       acc trunc   gc
 ----   --------- ------ ----- --- -------- --------     ---------- ----------       --- ----- ----
 (33) !   4.8e-14   65.0   0.0  cm        1       71 []     2130335     2130262 -  ..   1.00    no 0.55
```

This information is mostly redundant with the list of all hits at the top of the file, but is repeated here because it is useful to see adjacent to each hit alignment. After the summary, the hit alignment is displayed.

Understanding hit alignment annotation

```
>> tRNA
 rank     E-value  score  bias mdl mdl from   mdl to       seq from      seq to       acc trunc   gc
 ----   --------- ------ ----- --- -------- --------     ---------- ----------       --- ----- ----
 (33) !   4.8e-14   65.0   0.0  cm        1       71 []     2130335     2130262 -  ..   1.00    no 0.55

                            v          v                                                    NC      <-- negative scoring noncanonical basepairs
                     (((((((,,<<<<___..___._>>>>,<<<<<_______>>>>>,,,,<<<<<_______>>>>>))))))): CS      <-- model consensus secondary structure
           tRNA      1 GgagauaUAGCucAgU..GGU.AgaGCgucgGaCUuaaAAuCcgaaggcgcgGGUUCgAaUCCcgcuaucucCa 71      <-- model consensus sequence & coordinates
                       G:AG:U:U+G:: AGU  GGU A ::C U GG:CUU  AA:CC A ++C:CGGGUUCGAAUCCCG::A:CU:CA         <-- score contribution
  NC_013790.1 2130335 GCAGCUGUGGGUAUAGUcuGGUuAUUACUUGGGCCUUCCAAGCCUACAACCCGGGUUCGAAUCCCGGCAGCUGCA 2130262 <-- query sequence & coordinates
                       ********************************************************************************* PP      <-- expected alignment accuracy
```

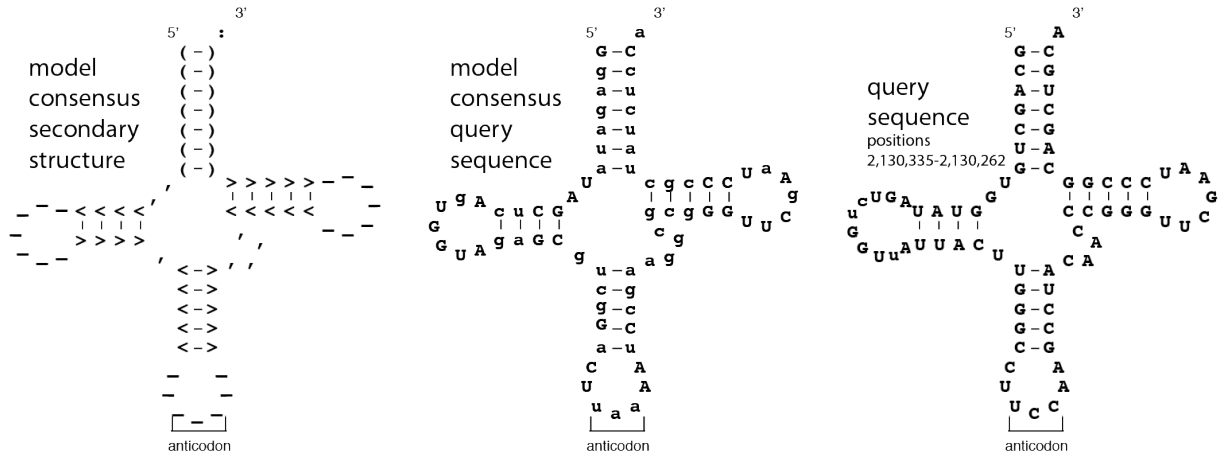


Fig. 6: Top: cmscan standard output of alignment of hit #33. Bottom: Three secondary structure diagrams showing the relationship between the alignment and the secondary structure of the Rfam tRNA model.

The alignment contains six lines. Start by looking at the second line which ends with CS. The line shows the predicted secondary structure of the query sequence in *WUSS format*.

> For more information see section 9 of the Infernal User's Guide.

The secondary structure on the left above shows how the CS line folds into the tRNA cloverleaf secondary structure.

The line above the CS line ends with NC and marks negative scoring non-canonical basepairs in the alignment with a v character. All other positions of the alignment will be blank. More specifically, the following ten types of basepairs

which are assigned a negative score by the model at their alignment positions will be marked with a v: A:A, A:C, A:G, C:A, C:C, C:U, G:A, G:G, U:U, and U:C. The NC annotation makes it easy to quickly identify suspicious basepairs in a hit. For this example, there is a single basepair that is negative scoring and non-canonical, it is the U:U pair between model positions 13 and 21.

The third line shows that consensus of the tRNA model. The highest scoring residue sequence is shown. Upper case residues are highly conserved. Lower case residues are weakly conserved or unconserved. Dots (.) in this line indicate insertions in the target sequence with respect to the model.

The fourth line shows where the alignment score is coming from. For a consensus basepair, if the observed pair is the highest-scoring possible pair according to the consensus, both residues are shown in upper case; if a pair has a score of 0, both residues are annotated by : characters (indicating an acceptable compensatory basepair); else, there is a space, indicating that a negative contribution of this pair to the alignment score. Note that the NC line will only mark a subset of these negative scoring pairs with a v, as discussed above. For a single-stranded consensus residue, if the observed residue is the highest scoring possibility, the residue is shown in upper case; if the observed residue has a score of 0, a + character is shown; else there is a space, indicating a negative contribution to the alignment score.

The fifth line, beginning with NC 013790.1 is the target sequence. Dashes (–) in this line indicate deletions in the target sequence with respect to the model.

The bottom line ends with PP. This line represents the posterior probability (essentially the expected accuracy) of each aligned residue. A 0 means 0-5%, 1 means 5-15%, and so on; 9 means 85-95%, and a * means 95-100% posterior probability. You can use these posterior probabilities to decide which parts of the alignment are well-determined or not. You'll often observe, for example, that expected alignment accuracy degrades around locations of insertion and deletion, which you'd intuitively expect.

Alignments for some searches may be formatted slightly differently than this example. Longer alignments to longer models will be broken up into blocks of six lines each - this alignment was short enough to be entirely contained within a single block.

cmscan tabular output

The cmscan tabular output file `mrum-genome.tblout` contains much of the information in the standard output, as well as some additional information in a tabular format that is easy to manipulate using common unix programs like `grep` and `awk`.

The top of the file has headers for each column. The first 25 hits are shown below:

```
#idx target name            accession query name           accession clan name mdl␣
↪mdl from   mdl to seq from   seq to strand trunc pass   gc  bias  score   E-value␣
↪inc olp anyidx afrct1 afrct2 winidx wfrct1 wfrct2 description of target
#--- ---------------------- --------- -------------------- --------- --------- --- ---
↪----- -------- -------- -------- ------ ----- ---- ---- ----- ------ --------- --- -
↪-- ------ ------ ------ ------ ------ ------ ----------------------
1    LSU_rRNA_archaea       RF02540   NC_013790.1          -         CL00112    cm    ␣
↪    1     2990   762872   765862      +    no    1 0.49  45.1 2763.5         0  !  ␣ ␣
↪^       -      -      -      -      -      - -
2    LSU_rRNA_archaea       RF02540   NC_013790.1          -         CL00112    cm    ␣
↪    1     2990  2041329  2038338      -    no    1 0.48  46.1 2755.0         0  !  ␣ ␣
↪^       -      -      -      -      -      - -
3    LSU_rRNA_bacteria      RF02541   NC_013790.1          -         CL00112    cm    ␣
↪    1     2925   762874   765862      +    no    1 0.49  45.1 1872.9         0  !  ␣ ␣
↪=       1  1.000  0.999      "      "      " -
4    LSU_rRNA_bacteria      RF02541   NC_013790.1          -         CL00112    cm    ␣
↪    1     2925  2041327  2038338      -    no    1 0.48  46.2 1865.5         0  !  ␣ ␣
↪=       2  1.000  0.999      "      "      " -
5    LSU_rRNA_eukarya       RF02543   NC_013790.1          -         CL00112    cm    ␣
↪    1     3401   763018   765851      +    no    1 0.49  41.5 1581.3         0  !  ␣
↪=       1  1.000  0.948      "      "      " -
```

(continues on next page)

(continued from previous page)

```
8  6    LSU_rRNA_eukarya        RF02543   NC_013790.1          -          CL00112    cm
   ↪     1     3401  2041183  2038349      -    no    1 0.49  42.3 1572.1          0  !
   ↪=       2  1.000  0.948      "      "      " -
9  7    SSU_rRNA_archaea        RF01959   NC_013790.1          -          CL00111    cm
   ↪     1     1477  2043361  2041888      -    no    1 0.53   4.1 1552.0          0  !
   ↪^       -      -      -        -      -      - -
10 8    SSU_rRNA_archaea        RF01959   NC_013790.1          -          CL00111    cm
   ↪     1     1477   760878   762351      +    no    1 0.54   4.1 1546.5          0  !
   ↪^       -      -      -        -      -      - -
11 9    SSU_rRNA_bacteria       RF00177   NC_013790.1          -          CL00111    cm
   ↪     1     1533  2043366  2041886      -    no    1 0.53   3.7 1161.9          0  !
   ↪=       7  0.995  1.000      "      "      " -
12 10   SSU_rRNA_bacteria       RF00177   NC_013790.1          -          CL00111    cm
   ↪     1     1533   760873   762353      +    no    1 0.53   3.7 1156.4          0  !
   ↪=       8  0.995  1.000      "      "      " -
13 11   SSU_rRNA_eukarya        RF01960   NC_013790.1          -          CL00111    cm
   ↪     1     1851  2043361  2041891      -    no    1 0.53   4.6  970.4  9.9e-293  !
   ↪=       7  1.000  0.998      "      "      " -
14 12   SSU_rRNA_eukarya        RF01960   NC_013790.1          -          CL00111    cm
   ↪     1     1851   760878   762348      +    no    1 0.54   4.5  963.8  9.9e-291  !
   ↪=       8  1.000  0.998      "      "      " -
15 13   SSU_rRNA_microsporidia  RF02542   NC_013790.1          -          CL00111    cm
   ↪     1     1312  2043361  2041891      -    no    1 0.53   4.6  919.9  7.7e-281  !
   ↪=       7  1.000  0.998      "      "      " -
16 14   SSU_rRNA_microsporidia  RF02542   NC_013790.1          -          CL00111    cm
   ↪     1     1312   760878   762348      +    no    1 0.54   4.5  917.2  5.4e-280  !
   ↪=       8  1.000  0.998      "      "      " -
17 15   RNaseP_arch             RF00373   NC_013790.1          -          CL00002    cm
   ↪     1      303  2614544  2614262      -    no    1 0.43   0.0  184.9   1.1e-53  !
   ↪*       -      -      -        -      -      - -
18 16   Archaea_SRP             RF01857   NC_013790.1          -          CL00003    cm
   ↪     1      318  1064321  1064634      +    no    1 0.44   0.1  197.6   6.9e-49  !
   ↪*       -      -      -        -      -      - -
19 17   FMN                     RF00050   NC_013790.1          -          -          cm
   ↪     1      140   193975   193837      -    no    1 0.42   0.0  115.2   6.8e-28  !
   ↪*       -      -      -        -      -      - -
20 18   tRNA                    RF00005   NC_013790.1          -          CL00001    cm
   ↪     1       71   735136   735208      +    no    1 0.59   0.0   72.1   4.9e-16  !
   ↪*       -      -      -        -      -      - -
21 19   tRNA                    RF00005   NC_013790.1          -          CL00001    cm
   ↪     1       71  2350593  2350520      -    no    1 0.66   0.0   71.0     1e-15  !
   ↪*       -      -      -        -      -      - -
22 20   tRNA                    RF00005   NC_013790.1          -          CL00001    cm
   ↪     1       71  2680310  2680384      +    no    1 0.52   0.0   70.9   1.1e-15  !
   ↪*       -      -      -        -      -      - -
23 21   tRNA                    RF00005   NC_013790.1          -          CL00001    cm
   ↪     1       71  2351254  2351181      -    no    1 0.62   0.0   69.7   2.2e-15  !
   ↪*       -      -      -        -      -      - -
24 22   tRNA                    RF00005   NC_013790.1          -          CL00001    cm
   ↪     1       71   361676   361604      -    no    1 0.51   0.0   69.5   2.5e-15  !
   ↪*       -      -      -        -      -      - -
25 23   tRNA                    RF00005   NC_013790.1          -          CL00001    cm
   ↪     1       71  2585265  2585193      -    no    1 0.60   0.0   69.2   3.2e-15  !
   ↪*       -      -      -        -      -      - -
26 24   tRNA                    RF00005   NC_013790.1          -          CL00001    cm
   ↪     1       71  2585187  2585114      -    no    1 0.59   0.0   68.8   3.9e-15  !
   ↪*       -      -      -        -      -      - -
```

(continues on next page)

(continued from previous page)

```
27 25   tRNA                    RF00005   NC_013790.1          -          CL00001     cm        ␣
   ↪     1       71  2680159  2680233       +     no     1 0.67   0.0    68.7   4.3e-15  !   ␣
   ↪*       -       -       -       -       -       - -
```

Each line has a whopping 27 fields. The most important ones are "seq from", "seq to", "strand", "E-value", "score", and "target name" and "accession" (Rfam model name and accession) and "query name" and "accession" (sequence name and accession), all of which (except the two accessions) were also included in the standard output file discussed above. The meanings of these columns should be clear from their names, but for a complete explanation of these and all other fields see Section 6 (target hits table format 2) of the Infernal User's Guide.

One column that requires explanation here is the **"olp"** (overlap) column, which indicates which hits overlap with one or more other hits. There are three possible characters in this column:

- ***** This hit's coordinates in the query sequence do not overlap with the query sequence coordinates of any other hits, on the same strand.
- **^** Indicates that this hit does overlap with at least one other hit on the same strand, but none of those hits are "better" hits. Here, hit A is "better" than hit B, if hit A's E-value is lower than hit B's E-value or if hit A and hit B have equal E-values but hit A has a higher bit score than hit B.
- **=** Indicates that this hit does overlap with at least one other hit on the same strand that is a "better" hit, given the definition of "better" above.

1.11.3 Removing lower-scoring overlaps from a tblout file

Using the values in the "olp" column of the tabular output file, you can easily remove all hits that have a higher scoring overlapping hit. This is recommended if you are annotating a genome or other sequence dataset. To do this for the example genome annotation file `mrum-genome.tblout`, and to save the remaining hits to a new file. `mrum-genome.deoverlapped.tblout`, use the following `grep` command:

```
grep -v " = " mrum-genome.tblout > mrum-genome.deoverlapped.tblout
```

1.11.4 Expected running times

CM searches are computationally expensive and searching large multi-Gb genomes with the roughly 2500 models in Rfam takes hundreds of CPU hours. However, you can parallelize by splitting up the input genome sequence file into multiple files (if the genome has multiple chromosomes) and running cmscan separately on each individual file. Also, you can run cmscan with multiple threads, as explained more below.

The following timings are from Table 2 of (Nawrocki et al., 2015). All searches were run as single execution threads on 3.0 GHz Intel Xeon processors.

Genome	Size (Mb)	CPU time (hours)	Mb/hour
Homo sapiens	3099.7	650	4.8
Sus scrofa (pig)	2808.5	460	6.1
Caenorhabditis elegans	100.3	20	5.2
Escherichia coli	4.6	0.46	10.2
Methanocaldococcus jannaschii	1.7	0.31	5.6

cmscan will run in **multithreaded mode** by default, if multiple processors are available. Running with 8 threads with 8 cores should reduce the running times listed in the table above by about 4-fold (reflecting about 50% efficiency versus single threaded).

1.11.5 Specificity

The Rfam/Infernal approach aims to be sufficiently generic to cope with **all types of RNAs**. A sequence can be searched using every model in exactly the same way.

In contrast, several tools are available that search for specific types of RNA, such as

- tRNAscan-SE for tRNAs
- RNAMMER for rRNA
- snoscan for snoRNAs
- SRPscan for SRP RNA

The generic Rfam approach has obvious advantages. However, the specialised programs often incorporate heuristics and family-specific information which may allow them to out-perform the general method. A comparison of Infernal versus some of these generic methods is presented in section 2.2 of a 2014 paper (by one of the authors of Infernal), available here.

1.11.6 Pseudogenes

ncRNA derived pseudogenes pose the biggest problem for eukaryotic genome annotation using Rfam/Infernal. Many genomes contain **repeat elements** that are derived from a non-coding RNA gene, sometimes in huge copy number. For example, Alu repeats in human are evolutionarily related to SRP RNA, and the active B2 SINE in mouse is recently derived from a tRNA.

In addition, specific RNA genes appear to have undergone massive **pseudogene expansions** in certain genomes. For example, searching the human genome using the Rfam U6 family yields over 1000 hits, all with very high score. These are not "false positives" in the sequence analysis sense, because they are closely related by sequence to the real U6 genes, but they completely overwhelm the small number (only 10s) of expected real U6 genes.

At present we don't have computational methods to distinguish the real genes from the pseudogenes (of course the standard protein coding gene tricks - in frame stop codons and the like - are useless). The sensible and precedented method for ncRNA annotation in large vertebrate genomes is to annotate the easy-to-identify RNAs, such as tRNAs and rRNAs, and then trust only hits with very high sequence identity (>95% over >95% of the sequence length) to an experimentally verified real gene. tRNAscan-SE has a very nice method for detecting tRNA pseudogenes.

Danger: We recommend that you use Rfam/Infernal for vertebrate genome annotation with **extreme caution**!

Nevertheless, Rfam/Infernal does tell us about important sequence similarities that are effectively undetectable by other means. However, in complex eukaryotic genomes, it is important to treat hits as sequence similarity information (much as you might treat BLAST hits), rather than as evidence of bona fide ncRNA genes.

1.12 Rfam cloud pipeline

The Rfam cloud environment provides access to the command-line interface for curating Rfam families. It uses the same software and the sequence database as the ones used by the Rfam team. The pipeline allows one to create a new RNA family or update an existing Rfam entry.

1.12.1 Background

The main Rfam family building pipeline is located on the EMBL-EBI computational cluster, so that only the EBI account holders can access it. In order to enable more users to contribute to Rfam, a new version of the pipeline was developed using a cloud infrastructure so that the EBI accounts are not needed. All families built using the cloud pipeline are **reviewed** by the *Rfam team* before the families are added to Rfam.

1.12.2 Requirements

1. A computer with internet access (Mac, Linux, or PC)
2. A command line environment supporting `ssh` (for example, bash)
3. An Rfam cloud account

1.12.3 Requesting an Rfam cloud account

Please *Contact us* to request access to the Rfam family building pipeline. If you intend to use the pipeline for teaching purposes, please let us know in advance to ensure that the pipeline can support the workload.

1.12.4 Connecting to Rfam cloud

Use the username and password provided by the Rfam team to `ssh` to Rfam cloud:

```
ssh <username>@cloud.rfam.org
```

To get access to an interactive session and start using the pipeline run the following command:

```
rfcloud --start
```

You should see a command line prompt:

Fig. 7: Rfam cloud command line prompt

To verify that the system works, try calling the `rfsearch` and `rfmake` scripts (you should see help messages explaining how to use the scripts):

```
rfsearch.pl -h
rfmake.pl -h
```

Tip:

Join Rfam Cloud on Slack to get help with the pipeline from the Rfam team

1.12.5 10 steps for building an Rfam family

Interested in editing an Rfam family? Skip to *Updating an existing Rfam family*.

1. Create a new folder

Create a new folder, for example *rfam_test* and navigate to it:

```
mkdir rfam_test
cd rfam_test
```

2. Prepare a SEED file

Each family has a *Seed alignment* file called `SEED` that contains a multiple sequence alignment of the confirmed instances of a family. To get started, you will need a *Stockholm format* file with at least 1 RNA sequence and a consensus secondary structure, for example see the tRNA seed alignment.

If you have a FASTA file called `file.fasta` with a **single RNA sequence**, convert it to Stockholm format and predict a consensus secondary structure using RNAfold (the `-r` option):

```
predict_ss.pl -infile <file.fasta> -outfile SEED -r
```

Alternatively, create a `SEED` file using the vi or nano text editors and paste the file contents from your local computer. See *Copying files to and from Rfam cloud* for instructions about moving files to and from Rfam cloud.

It is recommended that the sequences are named in the `accession:start-end` format where `accession` is an ENA, GenBank, or RNAcentral identifier, and `start-end` are the coordinates of the RNA in the accession (for example, `AB003409.1/96-167`). See the tRNA seed alignment for more examples. The sequence name cannot contain the parenthesis characters (`(` and `)`).

Once you have a Stockholm file called `SEED` in your working directory, proceed to the next step.

3. Find similar sequences using rfsearch

Build and calibrate a *Covariance model (CM)* based on your seed alignment and search for similar sequences in the *rfamseq* database:

```
rfsearch.pl -nodesc -relax -t 25 -cnompi
```

Option	Meaning
`-nodesc`	creates a required file called `DESC` that contains the description of the family. You only need to use the `-nodesc` flag the first time you run rfsearch, after that you will get an error if you use `-nodesc` because a `DESC` file already exists.
`-relax`	allow sequences not found in the *rfamseq* database to be included in the seed alignment (recommended)
`-cnompi`	do not use the MPI mode (this option should always be used)
`-t 30`	*Gathering cutoff* in bits. Usually 30 bits is a good starting point as most families are expected to have a threshold higher than 30.

This step can take a long time (up to 10 minutes or longer) depending on the size of the alignment and the availability of computational resources.

4. Choose a gathering threshold

The output files (choosing-gathering-threshold:Species file and choosing-gathering-threshold:Outlist file) can be used to determine the gathering threshold for this family (the bit score of the last true positive hit).

Note: For detailed instructions on how to select the threshold, see choosing-gathering-threshold:Choosing gathering threshold.

5. Add sequences to SEED (optional)

The *Seed alignment* needs to represent the taxonomic diversity and the structural features observed in different instances of the family. A seed alignment needs to have **at least 2 sequences** but a larger seed alignment is preferred.

Find an accession in the `outlist` file that you would like to add to the `SEED` (for example, `AB480043.1`):

```
grep AB480043.1 outlist >> addme
rfseed.pl addme
```

To remove sequences from `SEED` (if added in error, for example), create a file with a list of accessions you want to remove using `grep` as described above and call it *removeme*. Make sure the accession is exactly the same as in the `SEED` file, for example `NW_002196667.1/1438869-1438941`. Then run the following command:

```
rfseed.pl -d -n <removeme>
```

Consider **manually editing the alignment** on your local computer using RALEE or belvu and re-uploading it as explained in **Step 1**.

6. Repeat rfsearch with a new threshold

Steps 3 to 6 should be repeated until the seed alignment can no longer be improved:

```
rfsearch.pl -t <new_cutoff> -cnompi -relax -ignoresm
```

The `-ignoresm` option overrides the threshold set at the previous iteration and saved in the `DESC` file.

This process is known as **iteration** (see *Expanding the seed (iteration)* for more information).

7. Create required family files with rfmake

Once the cutoff has been chosen, all the required family files can be generated like this:

```
rfmake.pl -t <gathering_cutoff> -a
```

The `-a` option creates an `align` file with an alignment of all the sequences above the gathering threshold. For more information about setting the `-t` parameter, see choosing-gathering-threshold:Choosing gathering threshold.

After running rfmake you should:

- review the choosing-gathering-threshold:Align file to check that the threshold is set correctly.
- review the choosing-gathering-threshold:Taxinfo file to check that the taxonomic distribution of the family is correct.

Any unwanted sequences can be excluded by rerunning rfmake with a higher threshold `-t`.

8. Add metadata to the DESC file

Each family is described using in a `DESC` file (see the tRNA DESC file as an example). The following fields are required:

ID a unique ID, such as *tRNA* or *skipping-rope*. No spaces are allowed.

DE

a short description of the family.
Example: `DE GlmZ RNA activator of glmS mRNA`
Maximum **75 characters**.

AU Author name with an ORCID id. Multiple `AU` lines can be used. Example: `AU Eddy SR; 0000-0001-6676-4706`

SE Seed alignment source. Example: `SE Published; PMID:21994249;`

SS Secondary structure source. Examples:

- `SS Published; PMID:28977401;`
- `SS Predicted; mfold;`

TP One of Rfam RNA types. Example: *TP Gene; sRNA;*

DR A reference to a Gene Ontology or Sequence Ontology term. Multiple `DR` lines can be used. Example:

- `DR SO; 0000253; tRNA;`
- `DR GO; 0030533; triplet codon-amino acid adaptor activity;`

You may find the QuickGO website useful for finding GO terms. A link to a website can also be included, for example: `DR URL; http://telomerase.asu.edu/;`

CC A free text comment describing what is known about the RNA (function, taxonomic distribution, experimental validation etc). Maximum **80 characters per line**, but multiple `CC` lines can be used.

WK A Wikipedia link (you should create a new Wikipedia article or link to an existing one). Example: `WK Transfer_RNA`

To add literature references, use the following command that automatically imports information from PubMed:

```
add_ref.pl <pubmed_id>
```

The `GA`, `TC`, `NC`, `BM`, `CV`, `SM` lines are added automatically, please do not change them manually. The `RN`, `RM`, `RT`, `RA`, and `RL` lines are added by the `add_ref.pl` script. The `AC` field is assigned once the family is stored in the official Rfam database.

9. Perform quality control checks

The `rqc-all` script performs multiple quality controls on the family. It checks the file formats, the accessions, and the `DESC` file:

```
cd .. && rqc-all.pl rfam_test
```

10. Send SEED and DESC files for review

Download your `SEED` and `DESC` files to your local machine and send the files to the Rfam team for review by email or Slack.

See *Copying files to and from Rfam cloud* for instructions about moving files to and from Rfam cloud.

> **Danger:** We encourage you to **always keep a local copy of the important data**!

1.12.6 Updating an existing Rfam family

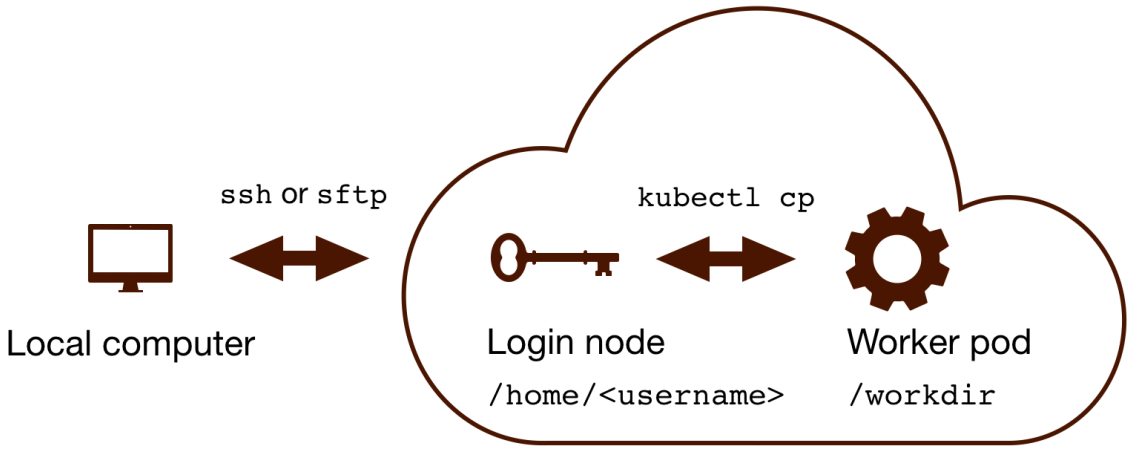
The only difference between creating a new family and updating an existing one is that the `SEED` and `DESC` files are retrieved from Rfam:

```
rfco.pl <RF0XXXX>
cd <RF0XXXX>
```

After that, follow the family building instructions: *3. Find similar sequences using rfsearch*.

1.12.7 Copying files to and from Rfam cloud

The Rfam cloud consists of a **login node** that handles the account login and **worker pods** which control the Rfam family building pipeline. When you run `ssh <username>@cloud.rfam.org` you are connected directly to your worker pod.



The login node and the worker pods currently have **different filesystems** which means that if you are on the worker pod you cannot see the files on the login node and vice versa. You can move files to and from login node using `scp` or `sftp` but then you need to use `kubectl cp` to make the files available on the worker pods.

Work on unifying the two filesystems is underway which should make moving files to and from Rfam more user-friendly.

Copying files to Rfam cloud

On your **local machine**:

```
scp SEED <username>@cloud.rfam.org:/home/<username>
```

This copies a file `SEED` to your login node. You can also use an SFTP client for this task (for example, CyberDuck on Mac and Windows).

On **worker pod**:

```
ssh <username>@cloud.rfam.org
kubectl get pod --selector=user=<username>,tier=frontend
```

Record the `pod_id` that looks like *rfam-login-pod-<username>-6b9f46fc76-67fhn*, then exit to the login node:

```
exit
```

On **login node**:

```
kubectl cp SEED <pod_id>:/workdir
```

Then get back to the worker pod:

```
kubectl exec -it <pod_id> bash
```

The file should appear in your `workdir` folder. You can specify other paths in the `kubectl cp` command to move the files to any subfolder.

Copying files from Rfam cloud

On **worker pod**:

```
ssh <username>@cloud.rfam.org
kubectl get pod --selector=user=<username>,tier=frontend
```

Record the `pod_id` that looks like *rfam-login-pod-<username>-6b9f46fc76-67fhn*, then exit to the login node:

```
exit
```

On **login node**:

```
kubectl cp <pod_id>:/workdir/SEED .
```

On your **local machine**:

```
scp <username>@cloud.rfam.org:/home/<username>/SEED .
```

1.12.8 Tips and tricks

- Filter out redundant sequences. For example, to remove redundancy from a file called *align* using 95% identify as a cutoff run:

```
esl-weight -f --idf 0.95 align
```

- Iteratively re-align seed sequences to the CM:

```
cmbuild --refine SEED.new CM.new SEED
```

1.12.9 Questions or comments?

Contact us by email, raise an issue on GitHub, or get in touch on Slack.

1.13 Extract ncRNA sequences

All Rfam ncRNA sequences become available on the ftp with every new release. The following is a tutorial on how to extract sequences using the public instance of the MySQL database and `esl-sfetch` tool.

Requirements:

1. MySQL Community Server, freely available here
2. `esl-sfetch` from Infernal's easel *miniapps*

---

1. Download and install the Infernal software. You can find additional information in the Infernal User's Guide.

See also:

Genome annotation section

2. Add Infernal tools to your **$PATH** using the following command:

```
> export PATH="/path/to/infernal-1.1.x/bin:$PATH"
```

3. Create a new directory and download all fasta files from the FTP using `wget` :

```
> mkdir rfam_sequences && cd rfam_sequences
> wget ftp://ftp.ebi.ac.uk/pub/databases/Rfam/CURRENT/fasta_files/* .
```

4. Decompress all sequence files and merge them in a single fasta file:

```
> gunzip *.gz
> cat *.fa > Rfam.fa
```

5. Index the unified sequence file using `esl-sfetch` :

```
> esl-sfetch --index Rfam.fa
```

Note: If the above command is successfull, you should see a `.ssi` file generated in your current directory

---

6. Create a `.sql` file with a SQL command that fetches the regions of interest.

Example query to retrieve all human ncRNAs:

```
select concat(fr.rfamseq_acc,'/',fr.seq_start,'-',fr.seq_end)
from full_region fr, genseq gs
where gs.rfamseq_acc=fr.rfamseq_acc
and fr.is_significant=1
```

(continues on next page)

(continued from previous page)

```
and fr.type='full'
and gs.upid='UP000005640' -- human upid
and gs.version=14.0;
```

Example query to retrieve all human snoRNAs:

```
select concat(fr.rfamseq_acc,'/',fr.seq_start,'-',fr.seq_end)
from full_region fr, genseq gs, family f
where gs.rfamseq_acc=fr.rfamseq_acc
and f.rfam_acc=fr.rfam_acc
and fr.is_significant=1
and fr.type='full'
and gs.upid='UP000005640' -- human upid
and f.type like '%snoRNA%'
and gs.version=14.0;
```

Example query to retrieve all Mammalian 5S ribosomal RNAs (RF00001):

```
select concat(fr.rfamseq_acc,'/',seq_start,'-',seq_end)
from full_region fr, rfamseq rs, taxonomy tx
where fr.rfamseq_acc=rs.rfamseq_acc
and tx.ncbi_id=rs.ncbi_id
and fr.rfam_acc='RF00001'
and tx.tax_string like '%Mammalia%'
and is_significant=1;
```

Note: In order for `esl-sfetch` to work with the Rfam fasta file, the regions need to be in the format: **rfamseq_acc/seq_start-seq_end**.

7. Fetch a list of accessions to extract from the database and save them in a `.txt` file using the MySQL database :

```
> mysql -urfamro -hmysql-rfam-public.ebi.ac.uk -P4497 --skip-column-names --database␣
↪Rfam < query.sql > accessions.txt
```

8. Extract the ncRNA sequences in the `.txt` file generated in **step 7** from the unified Rfam fasta file from **step 4** using `esl-fetch`:

```
> esl-sfetch -f Rfam.fa /path/to/accessions.txt > Rfam_ncRNAs.fa
```

1.14 How to link to Rfam?

Note: All links to Rfam should use the `rfam.org` domain. Please update any links referring to the `rfam.xfam.org` domain.

Here are some examples of linking to Rfam:

- Using Rfam accession (**recommended**):
  - http://rfam.org/family/RF00360
  - http://rfam.org/family?acc=RF00360

- Using Rfam ID:
  - http://rfam.org/family/snoZ107_R87
  - http://rfam.org/family?id=snoZ107_R87

Warning: Rfam accession numbers are more stable between releases than IDs. We **strongly** recommend that you link by Rfam accession (e.g. RF00360).

- Using "entry":

  You can also refer to a family by `entry`, although this is a convenience that should be used only if you're not sure if what you have is an accession or an ID.

  - http://rfam.org/family?entry=RF00360 or
  - http://rfam.org/family?entry=snoZ107_R87

1.15 Citing Rfam

Rfam makes use of a large amount of publicly available data, especially published multiple sequence alignments and secondary structures, and repackages these data in a single searchable and sustainable resource. We have made every effort to credit individual sources on family pages. If you find any of the data presented here useful, please also be sure to credit the primary source also.

1.15.1 Rfam references

Rfam 14: expanded coverage of metagenomic, viral and microRNA families

I. Kalvari, E.P. Nawrocki, N. Ontiveros-Palacios, J. Argasinska, K. Lamkiewicz, M. Marz, S. Griffiths-Jones, C. Toffano-Nioche, D. Gautheret, Z. Weinberg, E. Rivas, S.R. Eddy, R.D. Finn, A. Bateman, and A.I. Petrov

Nucleic Acids Research (2020) doi: 10.1093/nar/gkaa1047

Rfam 13.0: shifting to a genome-centric resource for non-coding RNA families

I. Kalvari, J. Argasinska, N. Quinones-Olvera, E.P. Nawrocki, E. Rivas, S.R. Eddy, A. Bateman, R.D. Finn, and A.I. Petrov

Nucleic Acids Research (2017) doi: 10.1093/nar/gkx1038

Non-coding RNA analysis using the Rfam database

I. Kalvari, E.P. Nawrocki, J. Argasinska, N. Quinones-Olvera, R.D. Finn, A. Bateman and A.I. Petrov

Current Protocols in Bioinformatics (2018) e51. doi: 10.1002/cpbi.51

Rfam 12.0: updates to the RNA families database

E.P. Nawrocki, S.W. Burge, A. Bateman, J. Daub, R.Y. Eberhardt, S.R. Eddy, E.W. Floden, P.P. Gardner, T.A. Jones, J.T. and R.D. Finn

Nucleic Acids Research (2014) doi: 10.1093/nar/gku1063

Rfam 11.0: 10 years of RNA families

S.W. Burge, J. Daub, R. Eberhardt, J. Tate, L. Barquist, E.P. Nawrocki, S.R. Eddy, P.P. Gardner and A. Bateman.

Nucleic Acids Research (2012) doi: 10.1093/nar/gks1005

Rfam: Wikipedia, clans and the "decimal" release

P.P. Gardner, J. Daub, J. Tate, B.L. Moore, I.H. Osuch, S. Griffiths-Jones, R.D. Finn, E.P. Nawrocki, D.L. Kolbe, S.R. Eddy and A. Bateman.
Nucleic Acids Research (2011) doi: 10.1093/nar/gkq1129

Rfam: updates to the RNA families database

P.P. Gardner, J. Daub, J.G. Tate, E.P. Nawrocki, D.L. Kolbe, S. Lindgreen, A.C. Wilkinson, R.D. Finn, S. Griffiths-Jones, S.R. Eddy and A. Bateman
Nucleic Acids Research (2008) Database Issue 37:D136-D140

The RNA WikiProject: community annotation of RNA families

J. Daub, P.P. Gardner, J. Tate, D. Ramsköld, M. Manske, W.G. Scott, Z. Weinberg, S. Griffiths-Jones and A. Bateman
RNA (2008) 12:2462-2464

Rfam: annotating non-coding RNAs in complete genomes

S. Griffiths-Jones, S. Moxon, M. Marshall, A. Khanna, S.R. Eddy, A. Bateman
Nucleic Acids Research (2005) Database Issue 33:D121-D124

Rfam: an RNA family database

S. Griffiths-Jones, A. Bateman, M. Marshall, A. Khanna and S.R. Eddy
Nucleic Acids Research (2003) 31(1):p439-441

1.15.2 Covariance models and stochastic context-free grammars

Annotating functional RNAs in genomes using Infernal

E.P. Nawrocki
Methods in Molecular Biology (2014) 1097:163-97

Infernal 1.1: 100-fold Faster RNA Homology Searches

E. P. Nawrocki, S. R. Eddy
Bioinformatics (2013) 29:2933-2935

Computational Identification of Functional RNA Homologs in Metagenomic Data

E. P. Nawrocki, S. R. Eddy
RNA Biology (2013) 10:1170-1179, 2013

Infernal 1.0: Inference of RNA Alignments

E.P. Nawrocki, D.L. Kolbe, S.R. Eddy
Bioinformatics (2009) Mar 23. [Epub ahead of print]

Local RNA Structure Alignment With Incomplete Sequence

D.L. Kolbe, S.R. Eddy
Bioinformatics (2009) Mar 20. [Epub ahead of print]

Query-dependent banding (QDB) for faster RNA similarity searches

E.P. Nawrocki, S.R. Eddy
PLoS Computational Biology (2007) 3(3):e56

A memory-efficient dynamic programming algorithm for optimal alignment of a sequence to an RNA secondary structure

S.R. Eddy

BMC Bioinformatics (2002) 2(3):18

Biological Sequence Analysis: Probabilistic Models of Proteins and Nucleic Acids

R. Durbin, S.R. Eddy, A. Krogh, G. Mitchison
Cambridge University Press (1999) ISBN 0-5216-2971-3

tRNAscan-SE: a program for improved detection of transfer RNA genes in genomic sequence

T.M. Lowe, S.R. Eddy
Nucleic Acids Research (1997) 25(5):955-964

RNA sequence analysis using covariance models

S.R. Eddy, R. Durbin
Nucleic Acids Research (1994) 22(11):2079-88

1.16 Privacy

This section outlines the ways in which the Rfam website handles information about users. This should not be read as a legal document, but as a description of how we handle information that could be considered sensitive. It should be read in conjunction with the privacy policy documents of the individual Rfam consortium member sites. If you have any concerns about the way that information is used in the website, please contact us at the address given at the bottom of the page and we will be more than happy to discuss your concerns.

Although we make every possible effort to keep this site and the data that it manipulates safe and secure, we make **no claim** to be able to protect sensitive or privileged information. If you are at all concerned about sensitive information being released, please do not use the site and consider installing the Rfam database and/or this website locally.

1.16.1 Google Analytics

We use Google Analytics (GA) to track the usage of this website. GA uses a single-pixel "web bug" image, which is served from every page, a javascript script that collects information about each request, and cookies that maintain information about your usage of the site between visits. You can read more about how GA works on the Google Analytics website, which includes a detailed description of how traffic is tracked and analysed.

We use the information generated by GA purely for audit and accounting purposes, and to help us assess the usefulness and popularity of different features of the site. It does not provide the ability to track individual users' usage of the site. However, GA does provides a high-level overview of the traffic that passes through the site, including such information as the approximate geographical location of users, how often and for how long they visited the site, etc.

We understand that this level of tracking may be worrying to some of our users. If you have any concerns about our use of Google Analytics, please feel free to contact us.

1.16.2 Browsing

All web servers maintain fairly detailed logs of their activity. This includes keeping a record of every request that they serve, usually along with the IP address of the client that made the request. This is true of the web servers that host the Rfam websites.

Although our servers do collect information about your IP address during the normal process of serving the Rfam website, we do not use this information explicitly. The Rfam group uses server logs **only** to help with development and debugging of the site.

1.16.3 Searches

The sequence search feature of the site allows you to upload a DNA or or RNA sequence to be searched against our library of CMs. The sequence that you upload is stored in a database and is retrieved by a set of scripts that actually perform the search. Although we do not have any information that could be used to link that sequence to you personally, you should be aware that the sequence itself **is accessible** to system administrators and other users who maintain the Rfam site.

The batch search function allows you to submit larger searches, the results of which are emailed to you. Obviously, this requires you to provide identifiable information, namely an email address. However, beyond the routine backups of our databases, we do not store any information about email addresses and sequences in the longer term and we make no attempt to keep track of the searches that a particular user may be performing.

Information from other types of search, such as a keyword search, is held only in the web server logs but, as described above, no attempt is made to interpret these logs except as part of development or debugging of the site.

1.16.4 Cookies

We use the following cookie to maintain some information about you between your visits to the site. The information that is stored cannot be used to identify you personally and cannot be used to track your usage of the site.

Cookie name	Purpose	Criteria
hide_posts	Keep track of whether blog posts have been hidden in home page	Optional

In addition to this Rfam-specific cookie, GA uses a series of cookies. You can read more about these in the GA documentation , or in EMBL-EBI's cookie policy.

If you are at all concerned about the use of cookies in the Rfam site, you are free to block all cookies from this site and you should not experience any problems. You may see some unintended behaviour, such as being notified of all new features every time you visit the index page, but the core functionality of the site should be unaffected.

1.16.5 Third-party javascript libraries

This site makes heavy use of javascript and relies on javascript libraries that are developed by various groups and companies. In order to improve the performance of the Rfam website, we no longer serve these files ourselves, but rely on files that are hosted on third-party web-servers. In particular, we use various files that are provided by the AJAX libraries APIs, hosted by google code, and components of the Yahoo! User Interface Library (YUI), hosted by Yahoo!

As these services are provided by commercial sites, it's likely that their usage will be carefully monitored by the companies that provide them. Although the Rfam site does not pass any information about you to these third-party sites, the sites themselves may use cookies to track your usage of the files that they serve. If you are concerned about the privacy implications of this monitoring, you may want to block cookies from the third-party hosting sites.

1.17 Website updates

1.17.1 Release 3.1

27th April 2016

- Fixed R-chie diagrams
- Improved loading time for sequence tabs on family pages

- Read full announcement on the blog.

1.17.2 Release 3.0

17th September 2014

- Removed references to full alignments throughout the site
- Added sequence motifs, in tab under the family pages and as a separate set of pages specific to each motif
- Internal changes to the sequence search system to use the EBI search infrastructure

1.17.3 Release 2.2

14th August 2012

- Added sunburst species trees
- Added biomart
- Improved single sequence search tool
- Added RefSeq regions to family page

1.17.4 Release 2.1

1st June 2011

- Added a "Browse by wikipedia title" page.
- Updated VARNA applet.
- Added Gene Ontology and Sequence Ontology links to family pages.

1.17.5 Release 2.0

15th April 2010

- Re-worked the alignment tab in family pages. Added new formats and new tools for viewing alignments, such as colorstock.
- We now provide secondary structure views using the VARNA applet.
- Rfam now includes clans. You can find clans via the browse pages and linked on pages for families that are members of clans.
- You can now select nodes in the species tree and download the list of sequence accessions or an alignment of the regions for the selected nodes.
- We now show sequence features using images taken from the European nucleotide archive (ENA) sequence feature viewer.

1.17.6 Release 1.5.2

9th October 2009

This was a maintenance release, needed to keep in sync with changes to the underlying codebase. The only significant change is the introduction of a new history mechanism for pages with a tab layout. The browser back button should now correctly take you back to the last tab that you viewed, rather than to the previous page altogether. Bookmarking tabs should now be possible too.

1.17.7 Release 1.5.1

9th June 2009

- Added Google Analytics code
- Added help pages
- Fixed a problem with viewing three-dimensional structures

1.17.8 Release 1.5

15th January 2009

- This release coincides with the release of Rfam 9.1. There are several changes and improvements throughout the site.

1.17.9 Release 1.4

7th January 2008

- Improved sequence validation for sequence searches.
- New help section on privacy.
- Sequence search defaults have been changed. Check the search form help text.
- Pfam domains drawn with molecular surfaces in AstexViewer.
- Fixed a bug in output of batch sequence searches.

1.17.10 Release 1.3

Bug fixes and other improvements.

1.17.11 Release 1.2

15th October 2007

- Reinstated taxonomy searches.
- Performance and stability improvements in IE.
- Metaseq data are now available.
- Find sequences using NCBI "GI" number.

1.17.12 Release 1.1

18th September 2007

Bug fixes and other improvements.

1.18 License

Rfam is freely available under the Creative Commons Zero ("CC0") licence.

1.19 Citing Rfam

If you use Rfam in your work, please consider citing the *Rfam references*.

1.20 Get in touch

If you have any questions or feedback, feel free to submit a GitHub issue or email us at **rfam-help@ebi.ac.uk**.

CHAPTER 2

Funding

Rfam is supported by the BBSRC and Wellcome and is developed at the EMBL-EBI.