Rfam Internal Docs

Release 1.0

Rfam Team

Apr 18, 2023

GENERAL:

This repository is intended for use by Rfam team members only. It contains internal documentation on processes, how-to's, notes, and general information regarding Rfam.

Note: This project is under active development.

CHAPTER

ONE

CONTENTS

1.1 Websites

There are two sets of machines that serve the live websites. PG virtual machines are in Hemel Hempstead (HH) and the OY VMs are located in Hinxton (HX). The load balancers ensure that most requests are handled by the PG machines with the OY machines serving as backup. The Rfam PG and OY VMs are run from separate directories and the update mechanism for each data-centre is different.

The test website, `preview.rfam.org`, is served by a HL (Harlow) web VM.

The preview website runs directly from the **rfam-live** database. The OY VMs use the **fb1-mysql-rfam-rel.ebi.ac.uk** database while the PG VMs use the **pg-mysql-rfam-rel.ebi.ac.uk** database.

1.1.1 Servers

- ves-pg-b7.ebi.ac.uk
- ves-pg-b9.ebi.ac.uk
- ves-oy-b7.ebi.ac.uk
- ves-oy-b9.ebi.ac.uk
- ves-hx-b7.ebi.ac.uk

1.1.2 Local development

The source code of the Rfam website is maintained on GitHub: rfam-website. Follow the Readme instructions to run the website locally using Docker.

1.1.3 Updating the website

The websites can be updated using **Jenkins** - simply choose the correct site (HX, OY, PG) from the dropdown website for the build `update_rfam_website`.

1.2 Databases

1.2.1 RfamLive

The main database used for ongoing curation. All new families are immediately visible in Rfam Live.

Host: mysql-rfam-live.ebi.ac.uk

1.2.2 RfamRel

A Gold Master copy of RfamLive generated at release time. Previous releases are available as separate schemas.

Host: mysql-rfam-rel.ebi.ac.uk

1.2.3 Rfam Web FB1

A fallback database used for the fallback web VMs located in Hinxton.

Host: fb1-mysql-rfam-rel.ebi.ac.uk

1.2.4 Rfam Web PG

Main database serving the main web VMs located in London.

Host: pg-mysql-rfam-rel.ebi.ac.uk

1.2.5 Rfam Public

A public database used by Rfam users (the users have read only access, see credentials on Rfam Docs).

Host: mysql-rfam-public.ebi.ac.uk

1.3 Rfam sequence search - Infernal cmscan

> ***NOTE:*** Since May 2021, the Rfam.cm file from the CURRENT Rfam FTP folder is fetched automatically every night by both production and wwwdev cmscan services. If Rfam is silently updated 1 day before the public announcement, the cmscan service should already be updated and no email is necessary.

Note that there will be a delay between the public Rfam release and the cmscan service update. If this delay is not acceptable, follow the manual update procedure using email.

The batch Rfam sequence search is powered by the Infernal cmscan program which is run by the EBI web production team. The Infernal service is available in several ways:

Web form: http://www.ebi.ac.uk/Tools/rna/infernal_cmscan/

REST API: http://www.ebi.ac.uk/Tools/webservices/services/rna/infernal_cmscan_rest

Every release Rfam updates the covariance models so we need to update the Rfam.cm file used by the Infernal service. When a new version of Infernal becomes available, we may also need to update the cmscan command, cmscan binary, or EBI help for cmscan.

1. Email www-prod@ebi.ac.uk using the template below:

To: www-prod@ebi.ac.uk Subject: Infernal cmscan service update

Hello,

Could you please update the Infernal cmscan service to use a new set of covariance models from release 14.6 that are available here:

http://ftp.ebi.ac.uk/pub/databases/Rfam/.14.6/Rfam.cm.gz

Please let me know if you need more information. Many thanks in advance!

2. Verify that the update worked.

To test that the search works as expected, navigate to

http://www.ebi.ac.uk/Tools/rna/infernal_cmscan/ (production environment)

or

http://wwwdev.ebi.ac.uk/Tools/rna/infernal_cmscan/ (testing environment)

and try searching with a sequence from a new family that is only present in the new release. Make sure that the sequence is found by the correct family.

1.4 Text search

Data for the text search is stored here `/nfs/production/agb/rfam/search_dumps/current_release`, and is indexed nightly.

Description of indexed fields: http://www.ebi.ac.uk/ebisearch/metadata.ebi?db=rfam

The steps to update all data for the text search are run during each release, except for that which runs as part of the PDB pipeline.

1.5 `rfsearch.pl` notes

Families which require special `rfsearch.pl` flags:

accession	recommended command	reason
RF00017	`rfsearch.pl -ignoresm -cut_ga`	too many hits otherwise, and it takes more than 1 week
RF02543	`rfsearch.pl -cgtailn 200`	allows calibration to finish successfully, otherwise you get this error 'Error: –gtailn =250 cannot be used, there's only 215.000 hits per Mb in the histogram! Lower or use –tailp.'

Possible `rfsearch.pl` error:

```
ERROR unable to fetch description for JZJH011 at /homes/nawrocki/git/Rfam/rfam-family-
↪pipeline/Rfam/Lib/Bio/Rfam/FamilyIO.pm line 2070, <RTBL> line 48.
```

If you see an error like this: rerun with `-scpu 0`. This error is due to a bug in infernal 1.1.4 (actually in easel) that was fixed in commit 8d300ef, so should be fixed in next release of infernal. https://github.com/EddyRivasLab/easel/pull/66/commits/8d300ef3899f69365be02635416333d0f30ccbe9

1.6 rfci.pl notes

For large families, it is recommended you login to a node with at least 64G of RAM (`bsub -M 64000M -q research -Is $SHELL`). The overlap check requires a lot of RAM. There may be other steps that require a lot of RAM as well.

Table of running times for commits (`rfci.pl`) (compiled 10/22:

acc	family id	seed #seq	full #seq	rfci.pl flags*	runtime
RF00177	SSU_rRNA_bacteria	99	37125	`-i coding -i overlap`	17 min
RF01960	SSU_rRNA_eukarya	90	39618	`-i coding -i overlap`	9 min
RF02540	LSU_rRNA_archaea	91	48683	`-i coding -i overlap`	23 min
RF02542	SSU_rRNA_microsporidia	46	35934	`-i coding -i overlap`	11 min
RF02543	LSU_rRNA_eukarya	88	54013	`-i coding -i overlap`	7 min
RF03064	RAGATH-18	1420	1771		9 min

- `-i spell -i missing -preseed` also used for *all* above `rfci.pl` commands

Miscellaneous notes:

Jiffy script that loads info to RfamLive: `/homes/nawrocki/git/Rfam/rfam-family-pipeline/Rfam/Scripts/jiffies/update_seed_dependent_tables_for_family.pl`

Note this is only a local file not in the git repo.

Code that updates the RfamLive tables is here:

Rfam/Schemata/{RfamLive,RfamDB}/ResultSet/Taxonomy.pm Rfam/Schemata/{RfamLive,RfamDB}/ResultSet/SeedRegion.pm Rfam/Schemata/{RfamLive,RfamDB}/ResultSet/Rfamseq.pm Rfam/Schemata/{RfamLive,RfamDB}/ResultSet/RnacentralMatch.pm

1.7 Release Process

The release process is carried out through the running of these steps

The process is mostly automated, however some steps still require manual updates. Until the automation process is running smoothly, you may find it easier to run the workflows in the given order, to ease potential troubleshooting.

1. Generate annotated files
2. Update PDB mapping
3. Run view process
4. Run clan competition
5. Prepare RfamLive
6. Generate FTP files
   a. manual step to generate database_files folder
7. Generate rfam2go
8. Stage RfamLive
9. Update text search
10. Copy from production folder to release ftp

1.8 Release Legacy Steps

These steps may be useful for reference.

1.8.1 Generate annotated SEED files

Export SEED files from SVN using generate_ftp_files.py:

- to export all files (recommended 16GB RAM to checkout large families):

```
python generate_ftp_files.py --acc all --seed --dest-dir /path/to/seed/files/dest/dir
```

- to export accessions listed in a file

```
python generate_ftp_files.py -f /path/to/rfam_accession_list.txt --seed --dest-dir /path/
↪to/seed/files/dest/dir
```

- Alternatively, use writeAnnotatedSeed.pl:

```
perl writeAnnotatedSeed.pl RFXXXXX
```

1.8.2 Generate annotated CM files

Export plain CM files using generate_ftp_files.py:

- to export all files (recommended 16GB RAM to checkout large families)

```
python generate_ftp_files.py --acc all --cm --dest-dir /path/to/seed/files/dest/dir
```

- to export accessions listed in a file python

```
generate_ftp_files.py -f /path/to/rfam_accession_list.txt --cm --dest-dir /path/to/CM/
↪files/dest/dir
```

- alternatively use writeAnnotatedCM.pl:

```
perl writeAnnotatedCM.pl RFXXXXX
```

- Rewrite CM file and descriptions from SEED using seed-desc-to-cm.pl:

Required files:

- $CM_no_desc - a CM file to add DESC to (could be 1 CM or all CMs in a single file)
- $SEED_with_DESC - a seed file with DESC lines (could be 1 seed or all seeds in a single file)

```
filter out DESC lines to avoid duplicates as some CMs already have DESC lines
grep -v DESC $CM_no_desc > Rfam.nodesc.cm
perl seed-desc-to-cm.pl $SEED_with_DESC Rfam.nodesc.cm > Rfam.cm

check that Rfam.cm contains the correct number of families
cmstat Rfam.cm | grep -v '#' | wc -l

check the number of DESC lines - should be 2 * number of families
```

(continues on next page)

(continued from previous page)

```
grep DESC Rfam.cm | wc -l

generate the final archive
gzip -c Rfam.cm > Rfam.cm.gz

split annotated Rfam.cm into individual .cm files and create Rfam.tar.gz:

delete any existing files
rm -f RF0*.cm

get a list of Rfam IDs found in Rfam.cm
grep ACC Rfam.cm | sed -e 's/ACC\s+//g' | sort | uniq > list.txt

create an index file to speed up cm retrieval
cmfetch --index Rfam.cm

create individual cm files
while read p; do echo $p; cmfetch Rfam.cm $p > "$p.cm" ; done <list.txt

create an archive
tar -czvf Rfam.tar.gz RF0*.cm

remove temporary files
rm RF0*.cm
```

The CM file is used for PDB mapping. The SEED and CM files are stored in FTP archive and are also available from the Rfam website.

1.8.3 Load annotated SEED and CM files into rfam_live

This enables the SEED and CM download directly from the Rfam website. Requires new Rfam.cm and Rfam.seed files.

Use load_cm_seed_in_db.py:

```
python load_cm_seed_in_db.py /path/to/Rfam.seed /path/to/Rfam.cm
```

Alternatively, use populateAnnotatedFiles.pl to process families one by one:

```
perl populateAnnotatedFiles.pl RFXXXXX /path/to/RFXXXXX.cm /path/to/RFXXXXX.seed
```

1.8.4 Clan competition

Clan competition is a quality assurance measure ran as a pre-processing release step aiming to reduce redundant hits of families belonging to the same clan.

1. Create a destination directory for clan competition required files

```
mkdir ~/releaseX/clan_competition     mkdir ~/releaseX/clan_competition/sorted
```

2. Generate clan files using clan_file_generator.py:

- recommended 16GB RAM

```
python clan_file_generator.py --dest-dir ~/releaseX/clan_competition --clan-acc CL00001 -
↪-cc-type FULL
```

–cc-type: Clan competition type FULL/PDB

–clan-acc: Clan accession to compete

–all: Compete all Rfam clans

-f: Only compete clans in file

3. Sort clan files in dest-dir based on rfamseq_acc (col2) using linux sort command and store then in the sorted directory:

```
sort -k2 -t $'\t' clan_file.txt > ~/releaseX/clan_competition/sorted/clan_file_sorted.txt
```

or for multiple files cd ~/releaseX/clan_competition and run:

```
for file in ./CL*; do sort -k2 -t $'\t' ${file:2:7}.txt > sorted/${file:2:7}_s.txt; done
```

4. Run clan competition using clan_competition.py:

```
python clan_competition.py --input ~/releaseX/clan_competition/sorted --full
```

–input: Path to ~/releaseX/clan_competition/sorted

–pdb: Type of hits to compete PDB (pdb_full_region table)

–full: Type of hits to compete FULL (full_region table)

1.8.5 Prepare rfam_live for a new release

1. Populate rfam_live tables using populate_rfamlive_for_release.py:

python populate_rfamlive_for_release.py –all

2. Make keywords using make_rfam_keywords_table.pl:

perl make_rfam_keywords_table.pl

3. Update taxonomy_websearch table using updateTaxonomyWebsearch.pl (:warning: requires cluster access):

perl updateTaxonomyWebsearch.pl

1.8.6 Running view processes

:warning: Requires cluster access and updating the PDB fasta file. Note that using PDB view processes plugin will be discontinued in favour of regular updates of the entire pdb_full_region table.

1. Create a list of tab separated family accessions and their corresponding uuids using the following query:

```
select rfam_acc, uuid from _post_process where status='PEND';
```

2. Update the PDB sequence file and the path in the PDB plugin

3. Launch view processes on the EBI cluster:

```
python job_dequeuer.py --view-list /path/to/rfam_uuid_pairs.tsv --dest-dir /path/to/
↪destination/directory
```

–view-list: A list with tab separated rfam_acc, uuid pairs to run view processes on –dest-dir: The path to the destination directory to generate shell scripts and log to

1.8.7 Update PDB mapping

This step requires a finalised Rfam.cm file with the latest families, including descriptions (see FTP section for instructions).

```
1. Get PDB sequences in FASTA format

wget ftp://ftp.wwpdb.org/pub/pdb/derived_data/pdb_seqres.txt.gz

gunzip pdb_seqres.txt.gz

2. Split into individual sequences

perl -pe "s/\>/\/\/\n\>/g" < pdb_seqres.txt > pdb_seqres_sep.txt

3. Remove protein sequences using collateSeq.pl:

perl collateSeq.pl pdb_seqres_sep.txt

mv pdb_seqres_sep.txt.noprot pdb_seqres_sep_noprot.fa

4. Remove extra text from FASTA descriptor line

awk '{print $1}' pdb_seqres_sep_noprot.fa > pdb_trimmed_noprot.fa

5. Replace any characters that are not recognised by Infernal

sed -e '/^[^>]/s/[^ATGCURYMKSWHBVDatgcurymkswhbvd]/N/g' pdb_trimmed_noprot.fa > pdb_
↪trimmed_noillegals.fa

6. Split into 100 files

mkdir files

seqkit split -O files -p 100 pdb_trimmed_noillegals.fa

7. Run cmscan in parallel

cd files
cmpress Rfam.cm
bsub -o part_001.lsf.out -e part_001.lsf.err -M 16000 "cmscan -o part_001.output --
↪tblout part_001.tbl --cut_ga Rfam.cm pdb_trimmed_noillegals.part_001.fa"
...
bsub -o part_100.lsf.out -e part_100.lsf.err -M 16000 "cmscan -o part_100.output --
↪tblout part_100.tbl --cut_ga Rfam.cm pdb_trimmed_noillegals.part_001.fa"

To check that all commands completed, check that every log file contains Successfully␣
↪completed:

cat part_*lsf.out | grep Success | wc -l
```

(continues on next page)

(continued from previous page)

```
8. Combine results

cat *.tbl | sort | grep -v '#' > PDB_RFAM_X_Y.tbl

9. Convert Infernal output to pdb_full_region table using infernal_2_pdb_full_region.py:

python infernal_2_pdb_full_region.py --tblout /path/to/pdb_full_region.tbl --dest-dir /
↪path/to/dest/dir

The script will generate a file like pdb_full_region_YYYY-MM-DD.txt.

10. Create a new temporary table in rfam_live

create table pdb_full_region_temp like pdb_full_region;
```

11. Manually import the data in the .txt dump into rfam_live using Sequel Ace or similar

:warning: mysqlimport or LOAD DATA INFILE do not work with rfam_live because of secure-file-priv MySQL setting.

11. Examine the newly imported data and compare with pdb_full_region
12. Rename pdb_full_region to pdb_full_region_old and rename pdb_full_region_temp to pdb_full_region
13. Clan compete the hits as described under Clan competition section using the –PDB option
14. List new families with 3D structures
    - number of families with 3D before select count(distinct rfam_acc) from pdb_full_region_old where is_significant = 1;
    - number of families with 3D after select count(distinct rfam_acc) from pdb_full_region where is_significant = 1;
    - new families with 3D select distinct rfam_acc from pdb_full_region where is_significant = 1 and rfam_acc not in (select distinct rfam_acc from pdb_full_region_temp where is_significant = 1);

1.8.8 Stage RfamLive for a new release

Create MySQL dumps using mysqldump:

1. Create a new MySQL dump to replicate the database on REL and PUBLIC servers:

```
export MYSQL_PWD=rfam_live_password

bsub -o mysqldump.out -e mysqldump.err "mysqldump -u <user> -h <hostname> -P <port> --
↪single-transaction --add-locks --lock-tables --add-drop-table --dump-date --comments --
↪allow-keywords --max-allowed-packet=1G rfam_live > rfam_live_relX.sql"
```

To create a MySQL dump of a single table:

```
mysqldump -u username  -h hostname -P port -p --single-transaction --add-locks --lock-
↪tables --add-drop-table --dump-date --comments --allow-keywords --max-allowed-
↪packet=1G database_name table_name > table_name.sql
```

Restore a MySQL database instance on a remote server

```
2. Move to the directory where a new MySQL dump is located

    cd /path/to/rfam_live_relX.sql

3. Connect to the MySQL server (e.g. PUBLIC)

    mysql -u username  -h hostname -P port -p

4. Create a new MySQL Schema

    Create schema rfam_X_Y;

5. Select schema to restore MySQL dump

    Use rfam_X_Y;

6. Restore the database

    source rfam_live_relX.sql
```

1.8.9 Generate FTP files

1. Generate annotated tree files

Generate new tree files for the release using generate_ftp_files.py:

- to export all files (recommended 16GB RAM to checkout large families) python generate_ftp_files.py –acc all –tree –dest-dir /path/to/tree/files/dest/dir
- to export accessions listed in a file python generate_ftp_files.py -f /path/to/rfam_accession_list.txt –tree –dest-dir /path/to/tree/files/dest/dir

alternatively use writeAnnotatedTree.pl:

perl writeAnnotatedTree.pl RFXXXXX

2. Generate rfam2go files

Create a new rfam2go export by running rfam2go.pl:

rfam2go.pl > /path/to/dest/dir/rfam2go

Add a header line to rfam2go with release and date.

!version date: <YYYY/MM/DD>

!description: A mapping of GO terms to Rfam release <XX.X>.

!external resource: https://rfam.org/

!citation: Kalvari et al. (2020) Nucl. Acids Res. 49: D192-D200

!contact: rfam-help@ebi.ac.uk

Create md5 checksum of rfam2go file:

cd rfam2go && md5sum * > md5.txt

Run GO validation and review warnings using validate_rfam2go.pl:

perl validate_rfam2go.pl goselect selectgo rfam2go

1.8.10 Generate Rfam.full_region file

Export ftp_Rfam_full_region.sql:

export MYSQL_PWD=rfam_live_password

mysql -u -h -P –database rfam_live < sql/ftp_rfam_full_reqion.sql > /path/to/Rfam.full_region

gzip Rfam.full_region

1.8.11 Generate Rfam.pdb file

Export ftp_rfam_pdb.sql:

```
export MYSQL_PWD=rfam_live_password

mysql -u <user> -h <host> -P <port> --database rfam_live < sql/ftp_rfam_pdb.sql > /path/
↪to/Rfam.pdb

gzip Rfam.pdb
```

1.8.12 Generate Rfam.clanin file

Generate a new Rfam.clanin file using clanin_file_generator.py:

python clanin_file_generator.py /path/to/destination/directory

1.8.13 Generate database_files folder

The option –tab of mysqldump is used to generate dumps in both .sql and .txt formats, where:

table.sql includes the MySQL query executed to create a table

table.txt includes the table contents in tabular format

The main rfam_live database is running with the secure-file-priv setting so it is not possible to export using the –tab option directly. A workaround is to copy dump files on a laptop, and use a local MySQL database for export.

Create a new database dump using mysqldump:

```
    mysqldump -u <user> -h <host> -P <port> -p --single-transaction --add-locks --lock-
↪tables --add-drop-table --dump-date --comments --allow-keywords --max-allowed-
↪packet=1G --tab=/tmp/database_files rfam_local
```

Run the database_file_selector.py python script to create the subset of database_files available on the FTP:

```
    cd /releaseX/database_files
    gzip *.txt
    python database_file_selector.py --source-dir /tmp --dest-dir /releaseX/database_
↪files
```

Zip database_files directory and copy to remote server:

```
tar -czvf database_files.tar.gz /releaseX/database_files     scp database_files.tar.gz␣
↪username@remote.host:/some/location
```

Restore database_files on FTP

```
tar -xzvf /some/location/database_files.tar.gz .
```

1.8.14 Generate fasta_files folder

Export new fasta files for all families in Rfam using fasta_file_generator.py:

```
bsub -M 10000 -o fasta_generator.lsf.out -e fasta_generator.lsf.err "python fasta_file_
↪generator.py --seq-db /path/to/rfamseq.fa --rfam-seed /path/to/releaseX/Rfam.seed --
↪all --outdir /path/to/ftp/fasta_files"
```

1.8.15 Update Rfam text search

Description of indexed fields: https://www.ebi.ac.uk/ebisearch/metadata.ebi?db=rfam

1.8.16 Generate new data dumps

Requirements:

The directory for the text search index must have the following structure:

release_note.txt

families

clans

genomes

motifs

full_region

Move to the main rfam-production repo and setup the django environment:

```
source django_settings.sh
```

Create an output directory for a new release index and all required subdirectories:

```
cd /path/to/relX_text_search

mkdir clans

mkdir families

mkdir full_region

mkdir genomes

mkdir motifs
```

Create new XML dumps:

```
cd /path/to/relX_text_search

python rfam_xml_dumper.py --type F --out families

python rfam_xml_dumper.py --type C --out clans

python rfam_xml_dumper.py --type M --out motifs

python rfam_xml_dumper.py --type G --out genomes

python rfam_xml_dumper.py --type R --out full_region
```

For more information on launching XML dumps as LSF jobs see lsf_rfam_xml_dumper.sh.

Validate xml dumps using xml_validator.py:

```
python xml_validator.py --input /path/to/relX_text_search/families --log

python xml_validator.py --input /path/to/relX_text_search/clans --log

python xml_validator.py --input /path/to/relX_text_search/motifs --log

python xml_validator.py --input /path/to/relX_text_search/genomes --log

python xml_validator.py --input /path/to/relX_text_search/full_region --log
```

Check that error.log files in each folder is empty.

Get the total number of entries:

```
grep -r 'entry id' . | wc -l
```

Create release_note.txt file:

```
release=14.5     release_date=15-Mar-2021     entries=2949678
```

1.8.17 Index data on dev and prod

Change directory to text_search on the cluster:

cd_main && cd search_dumps

Index data on dev:

unlink rfam_dev

ln -s /path/to/xml/data/dumps rfam_dev

Index data on prod:

unlink current_release

ln -s /path/to/xml/data/dumps current_release

The files in rfam_dev and current_release folders are automatically indexed every night.

1.8.18 Create new json dumps for RNAcentral

Create a new region export from Rfam using Rfam2RNAcentral.pl:

Extract SEED regions

```
perl Rfam2RNAcentral.pl SEED > /path/to/relX/rnacentral/dir/rfamX_rnac_regions.txt
```

Extract FULL region

```
perl Rfam2RNAcentral.pl FULL >> /path/to/relX/rnacentral/dir/rfamX_rnac_regions.txt
```

Split regions into smaller chunks using basic linux split comman

mkdir chunks && cd chunks && split -n 3000 /path/to/relX/rnacentral/dir/rfamX_rnac_regions.txt rnac_ –additional-suffix='.txt'

-n: defines the number of chunks to generate (2000 limit on LSF)

Notes:

- This command will generate 3000 files named like rnac_zbss.txt
- Use -l 500 option for more efficient chink size

Create a copy of the fasta files directory

mkdir fasta_files && cd fasta_files && wget http://ftp.ebi.ac.uk/pub/databases/Rfam/CURRENT/fasta_files/* .
NOTE: Rfam2RNAcentral regions need to match the exact same release the fasta_files were created for. Use the Public MySQL database if rfam-live have been updated

Unzip all fasta files and index using esl-sfetch:

gunzip * .gz for file in ./RF*.fa; do esl-sfetch –index $file; done

Copy the Rfam.seed file from the FTP to the fasta_files directory and index using esl-sfetch`:

wget http://ftp.ebi.ac.uk/pub/databases/Rfam/CURRENT/Rfam.seed.gz && gunzip Rfam.seed.gz esl-sfetch –index Rfam.seed

Launch a new json dump using rnac2json.py:

- fasta file directory: Create a new fasta files directory
- json files directory: Create a new json files output directory

python rnac2json.py –input /path/to/chunks –rfam-fasta /path/to/fasta_files –outdir /path/to/json_files

1.9 Release Checklist

This is a template checklist. It can be copied to the GitHub issue for each release.

1.9.1 Data updates

- Export annotated CM and SEED files
- Update PDB mapping
- Run view processes
- Run clan competition
- Run make_rfam_keywords_table.pl
- Update family_ncbi
- Update the family table (number_of_species and num_full fields)
- Update the version table
- Update RNAcentral descriptions

1.9.2 Update FTP archive

- Create .release FTP directory
- database_files
- fasta_files
- genome_browser_hub
- rfam2go
- COPYING
- USERMAN
- README including Section 5
- Rfam.clanin
- Rfam.cm.gz
- Rfam.full_region.gz
- Rfam.pdb.gz
- Rfam.seed.gz
- Rfam.seed_tree.gz
- Rfam.tar.gz
- .tsv release stats file

1.9.3 Pre Announce

- Update EBI text search on wwwdev
- Update the website
- Add release graphic
- Update pages with new data (e.g. microRNA or viral project pages)
- Merge new code into rfam-website master

- Update the REL MySQL database
- Stop Apache in OY
- Update the OY web production MySQL database
- Update rfamweb_local.conf (fields ebi_search and connect_info) in OY
- Deploy new website code in OY
- Start Apache in OY
- Verify OY VMs directly
- Repeat for PG
- Move.release FTP directory to release
- Update CURRENT FTP symlink
- Update Public MySQL database using Sequel Ace
- Load new data into a new database called rfam_xx_x
- Rename the Rfam database to rfam_xx_y
- Rename the rfam_xx_x database to Rfam
- Delete old databases to save space on PUB
- Update EBI cmscan search - check that it automatically picks up the new covariance models from the production FTP location
- Update RNAcentral sequence search
- Update rfam_local.py config to latest db

1.9.4 Announce

- Publish blog post with the Rfam tag
- Post on Twitter

1.9.5 Post Announce

- Close GitHub release project
- Review GitHub issues
- Backup old text search files - tar.gz folder in search_dumps
- Update EBI text search in production
- Make sure production website uses production search index
- Update Rfam models in R2DT
- Update Rfam taxonomy and create a new GitHub release
- Create a release checklist for the next release
- update config files
  - rfam-production/config/rfam_local.py

1.10 Codon Migration

This collection has notes and details on the steps taken to migrate from noah (old EBI cluster) to codon. All of the data needed to be copied from several different paths. We had to make updates to hardcoded paths in the code, and ensure all the configs were updated. Should a migration need to be repeated, the following checklist may act as a start:

- autobundle the Perl code and re-install the modules
- manual (cpan) install of Perl modules
- copy all data, files, folders

1.11 Codon Setup

The paths mostly used by Rfam on the codon cluster are:

- software packages

`/hps/software/users/agb/rfam`

- production data

`/nfs/production/agb/rfam`

- release FTP data

`/nfs/ftp/public/databases/Rfam/`

- website code, html reports generated by rfreport
- must be on a datamover node to access
- currently this info is mounted from (old) noah to the HX web VM
- we need to migrate VM from HX to HL to mount from codon to the preview website

`/nfs/public/rw/xfam/rfam/`

Virtual users:

- rfamprod
- xfm_adm

1.12 Software Installation

1.12.1 Infernal

- use the `--prefix` option with configure to ensure software is installed to the correct location, for example:

```
cd /hps/software/users/agb/rfam/
bsub -Is $SHELL
cd packages/
curl -OL http://eddylab.org/infernal/infernal-1.1.2.tar.gz
tar -xvzf infernal-1.1.2.tar.gz
cd infernal-1.1.2
./configure --help
cd infernal-1.1.2
```

(continues on next page)

(continued from previous page)

```
./configure --prefix=/hps/software/users/agb/rfam
make
make install
```

1.12.2 R-scape

```
mv rscape rscapev1.5.16
wget http://eddylab.org/software/rscape/rscape.tar.gz
tar -xvzf rscape.tar.gz
mv rscape_* rscape
rm rscape.tar.gz
cd rscape
./configure --help
./configure --prefix=/hps/software/users/agb/rfam
make
make install
```

- R-scape then needs to be added to the PATH variable
- As of July 2022, we are using R-scape version 2.0.0.g

1.12.3 R

```
R-2.6.0/bin/R CMD BATCH --no-save  plot_outlist.R
```

1.12.4 bedToBigBed

```
 cd /hps/software/users/agb/rfam/
 cd bin/
 wget http://hgdownload.cse.ucsc.edu/admin/exe/linux.x86_64/bedToBigBed
 chmod +x ./bedToBigBed
```

1.12.5 aspell

```
wget https://ftp.gnu.org/gnu/aspell/aspell-0.60.8.tar.gz
cd ..
tar -xvzf packages/aspell-0.60.8.tar.gz
ls bin/
cd aspell-0.60.8/
./configure --prefix=/hps/software/users/agb/rfam
make
make install
cd ..
aspell -v
```

1.12.6 seqkit

```
cd packages/
wget https://github.com/shenwei356/seqkit/releases/download/v2.2.0/seqkit_linux_amd64.
↪tar.gz
cd ..
tar -xvzf packages/seqkit_linux_amd64.tar.gz
cp seqkit bin/
seqkit version
```

1.12.7 misc.

```
wget http://hgdownload.cse.ucsc.edu/admin/exe/linux.x86_64/hubCheck
wget http://hgdownload.cse.ucsc.edu/admin/exe/linux.x86_64/faSplit
wget http://hgdownload.cse.ucsc.edu/admin/exe/linux.x86_64/fetchChromSizes
chmod +x hubCheck
chmod +x faSplit
chmod +x fetchChromSizes
```

1.12.8 netxflow

- module available on codon, but installed a later version for our own use
- nextflow jobs will not run on codon from an interactive job, you must submit the jobs from a login node and monitor their progress with the output or the .nextflow.log
- getting this error on codon, as are other users, no solution from IT as of yet

```
[rfamprod@hl-codon-30-03 rfam-production]$ nextflow run pdb_mapping/pdb_mapping.nf

N E X T F L O W ~ version 20.07.1

Launching `pdb_mapping/pdb_mapping.nf` [backstabbing_stallman] - revision: e7ea86f012

Signal already used by VM: HUP
```

1.13 Copying Data

1.13.1 paths

- grep /nfs/ on all code in Rfam to get a list of distinct paths in the code
- Rfam.conf
- rfam-family-pipeline/Rfam/Conf/rfam_local.conf
- rfamprod .bashrc
- all aliases
- rh74 all

- soft all
- rfamprod folder

1.13.2 datamover queue

- need to use the datamover queue to transfer between noah and codon, e.g.

```
bsub -n 4 -q datamover "/hps/software/copytools/msrsync/msrsync
hps/nobackup/production/xfam/rfam/RELEASES /nfs/production/agb/rfam/"
```

1.13.3 folders to copy

- folders to copy in to `/nfs/production/agb/rfam`
  - /hps/ebi/nobackup/production/xfam/rfam/RELEASES
  - /nfs/ebi/production/xfam/users/rfamprod
  - /hps/ebi/nobackup/production/xfam/rfam/
  - /hps/ebi/nobackup2/production/xfam/rfam/
  - /nfs/ebi/production/xfam/rfam/rfamseq
- Copying data from all aliases
  - alias cd_prod='cd /nfs/production/xfam/users/rfamprod'
  - alias cd_rh74='cd /nfs/production/xfam/rfam/rfam_rh74'
  - /nfs/production/agb/rfam/software = rh74 stuff
  - alias cd_rfamp='cd /nfs/production/xfam/users/rfamprod/rfamp/'
  - alias cd_rfam='cd /nfs/production/xfam/rfam/rfam_rh74/production_software/rfam_production/rfam-family-pipeline/Rfam'
  - alias cd_soft='cd /nfs/production/xfam/rfam/rfam_rh74/software'
  - alias cd_lib='cd /nfs/production/xfam/rfam/rfam_rh74/production_software/rfam_production/rfam-family-pipeline/Rfam/Lib/Bio/Rfam'
  - alias cd_hps='cd /hps/nobackup/production/xfam/rfam'
  - alias cd_gpfs='cd /gpfs/nobackup/xfam/rfam' - no such file or directory
  - alias cd_code='cd /nfs/production/xfam/users/rfamprod/code'
  - alias cd_rel='cd /hps/nobackup/production/xfam/rfam/RELEASES'
  - alias cd_main='cd /nfs/production/xfam/rfam'
  - alias cd_gens='cd /hps/nobackup/production/xfam/rfam/genome_datasets'
  - alias cd_rfamseq='cd /nfs/production/xfam/rfam/rfamseq'
  - alias cd_megs='cd /hps/nobackup/production/xfam/rfam/METAGENOMICS'
  - alias cd_hps2='cd /hps/nobackup2/production/rfam'
  - alias cd_mir='cd /hps/nobackup/production/xfam/rfam/RELEASES/14.3'
  - alias cd_ftp='cd /ebi/ftp/pub/databases/Rfam'

  - alias search_dumps='cd /nfs/production/xfam/rfam/search_dumps'
  - /nfs/ftp/public/databases/Rfam/
  - /nfs/ftp/pub/databases/Rfam/.preview/

1.14 Perl

1.14.1 Installing Modules

- create an autobundle of the current perl modules

```
perl -MCPAN -eautobundle
```

- when finished you will see a message like

  ```
  Wrote bundle file /homes/user/.cpan/Bundle/Snapshot_2022_03_03_00.pm
  ```

- copy this line, you will need the path later on
- now run

```
perl -MCPAN -e 'install Bundle::Snapshot_2022_03_03_00'
```

- this will read and install what is listed in the snapshot file
- may need to change the lib directory in .cpan/CPAN/MyConfig.pm so that 'my' perl modules are installed where wanted (in this case `/hps/software/users/agb/rfam/perl/lib`)
- needed to manually install some modules along the way with CPAN install, e.g.

```
cpan -f install File::ShareDir::Install
cpan -f install Inline::C
cpan -f install Data::Printer
cpan -f install Config::General
cpan -f install DBIx::Class::Schema
cpan -f install DateTime
cpan -f install DateTime::Format::MySQL
cpan -f install MooseX::NonMoose
cpan -f install Bio::Annotation::Reference
cpan -f install File::Touch
cpan -f install IPC::Run
cpan -f install Term::ReadPassword
cpan -f install JSON
cpan -f install JSON::XS

cpan -f install DBIx::Class
cpan -f install Catalyst::Devel
cpan -f install Catalyst::Utils
cpan -f install DBIx::Class
cpan -f install Catalyst::Model
cpan -f install DBIx::Class::ResultSet

cpanm -l /hps/software/users/agb/pfam/perl/lib/perl5 --force My::Package
```

1.14.2 Bio-Easel

- installing Bio-Easel following these instructions
  - used 0.15
- need to install with perl 5.26 (not 5.34 as had be done - needed to reinstall)
- had issues with paths installing in `/hps/software/users/agb/rfam`, currently installed in `/homes/rfamprod`

1.15 Code Changes

1.15.1 PDB Pipeline

- Change all hardcoded paths to params - in config file
- Update paths in config file
- Update PYTHONPATH variable in config
- Need to copy over search dumps
- Check slack config
- Will need to update database configs on codon

1.15.2 To submit jobs

- had to change the queue names in several scripts in the family pipeline repo https://github.com/Rfam/rfam-family-pipeline/
- run rfmake with `-local`
- run rfsearch with `-scpu 0`