FRAMA User's Guide

Martin Bens

v20151019

Contents

Licence **2**

Introduction **2**

Input **2**

Requirements **2**

Bioinformatic Software 2

Perl Modules 3

R Packages 3

Installation **4**

Manual Installation of external software 4

Automatic Installation of external software 4

Run **5**

Cleanup **6**

Configuration **6**

mandatory variables 6

optional 7

Software parameter 8

Output files **10**

important files 10

intermediate output 11

Licence

This software was developed at the Leibniz Institute on Aging - Fritz Lipmann Institute (FLI; http://www.leibniz-fli.de/) under a mixed licensing model. This means that researchers at academic and non-profit organizations can use it for free, while for-profit organizations are required to purchase a license. By downloading the package you agree with conditions of the FLI Software License Agreement for Academic Non-commercial Research (FLI-LICENSE).

Introduction

FRAMA is a **transcriptome assembly and mRNA annotation pipeline**, which utilizies external and newly developed software components. Starting with RNA-seq data and a reference transcriptome, FRAMA performs 4 steps:

```
1) de novo transcript assembly (Trinity),
2) gene symbol assignment (best bidirectional blastn hit) and
3) fusion detection and scaffolding
4) contig annotation (CDS, mRNA boundaries).
```

Further details: [Unpublished]

Input

All you need is a reference transcriptome in GenBank format and RNA-seq data in FastQ format. You can also provide orthologs to your reference transcripts from other species. The additional homologs are used for CDS inference.

Requirements

FRAMA runs on Linux and is written in Perl (5.10.0), R (3.0.3) and GNU Make (3.81). FRAMA does not require any compilation, but relies on common bioinformatic applications to be installed. The installation of all external software packages might seem like a daunting task, but your package manager might bring you halfway through (see Installation).

Bioinformatic Software

Software	Link	Description
Trinity	http://trinityrnaseq.sourceforge.net/	mandatory

Software	Link	Description
samtools	http://samtools.sourceforge.net	mandatory
bamtools	https://github.com/pezmaster31/bamtools	mandatory
bowtie1	http://bowtie-bio.sourceforge.net/bowtie1	mandatory
bowtie2	http://bowtie-bio.sourceforge.net/bowtie2	mandatory
EMBOSS	http://emboss.sourceforge.net	mandatory
MAFFT	http://mafft.cbrc.jp/alignment/software	mandatory
GENSCAN	http://genes.mit.edu/license.html	mandatory
RepeatMasker	http://www.repeatmasker.org/	optional
CD-HIT-EST	http://weizhong-lab.ucsd.edu/cd-hit/	optional
TGICL	http://compbio.dfci.harvard.edu/tgi/software/	optional
WU-BLAST	http://blast.advbiocomp.com	mandatory

In case you do not use WU-BLAST:

Software	Link	Description
NCBI-BLAST	http://www.ncbi.nlm.nih.gov/books/NBK279671/	mandatory
GenblastA	http://genome.sfu.ca/genblast/download.html	mandatory

Perl Modules

Available via CPAN.

Module	Version
BioPerl	1.006924
Parallel::ForkManager	0.7.5
Set::IntSpan	1.19
FileHandle::Unget	0.1628

R Packages

Package	Version
plyr	1.8.3
ggplot2	1.0.1
reshape	0.8.5
gridExtra	2.0.0
annotate	1.44
GO.db	3.0
KEGG.db	3.0

Installation

In addition to FRAMA, you have to install all third-party tools described as 'mandatory' in the table above. Depending on your Linux platform, your package manager might bring you half the way through (see Manual Installation / Automatic Installation).

Installing FRAMA is quick and easy. Download and unpack this repository and make sure to set the permission to execute FRAMA. You can add FRAMA to your $PATH or create a symlink to FRAMA in one of the directories in $PATH.

Here is a suggest workflow, which adds *FRAMA* to your `$PATH`:

```
unzip FRAMA.zip
cd FRAMA/
chmod u+x FRAMA
PATH=$(pwd):$PATH
export PATH
# run example
FRAMA example/testing.cfg
```

Manual Installation of external software

For instance, on Ubuntu (15.04, Vivid Vervet) :

```
sudo apt-get install perl default-jre r-base-core \
    ncbi-blast+ mafft emboss bowtie bowtie2 cd-hit \
    bamtools samtools parallel libc6-i386 build-essential \
    bioperl libparallel-forkmanager-perl libset-intspan-perl \
    libfilehandle-unget-perl r-cran-ggplot2 r-cran-plyr \
    r-cran-reshape
```

Left to install manually:

- `Trinity, GENSCAN, Genblasta, RepeatMasker, TGICL`
- R-packages: `gridExtra, annotate, GO, KEGG.db`

Automatic Installation of external software

On 64bit platforms, `SETUP` attempts to download and install (as non-root) missing software packages in very naive way. This might fail due to different/missing library/compiler versions on your system. Required additional prerequesites for `SETUP` include but are not limited to:

```
cmake
zlib >= 1 (zlib1g-dev)
ncurses >= 5 (libncurses5-dev)
jre >= 1.7.0
g++
libc6 (libc6-i386) # genscan, tgicl
```

Start automatic installation:

```
cd FRAMA
bash SETUP
```

GENSCAN must be downloaded manually, due to licence restrictions.

Run

Make sure all mandatory parameters are specified in the configuration file (see Configuration section). Then, call FRAMA with the appropriate configuration file.

```
FRAMA configuration_file
```

That's all. In case of aborts, consult logfiles and remove incomplete results. Rerunning the above command will complete remaining tasks.

Same as above, but shows all called processes.

```
FRAMA configuration_file verbose
```

Start from scratch (removes all created files beforehand).

```
FRAMA configuration_file scratch
```

FRAMA uses GNU make as a backbone. Parameters other than `verbose` and `scratch` (and `full-cleanup`, `cleanup`) are forwarded to make. For example, the following lists all tasks without executing them.

```
FRAMA configuration_file -n
```

Cleanup

FRAMA creates a lot of intermediate files. See "output files" for further information about each file. We provide to two cleaning methods:

```
FRAMA configuration_file full-cleanup
```

This will keep all important files: `sequences-mRNA.fasta`, `sequences-CDS.fasta`, `transcript_catalogue.gbk`, `summary`, `tables/`

```
FRAMA configuration_file cleanup
```

additionally keeps: `transcripts/`, `trinity/`

Configuration

Take a look at and try to run the provided example file in `PATH_TO_FRAMA/example/testing.conf` before running FRAMA on your own data set.

This also serves as a template for your custom configuration.

mandatory variables

The following depends mostly on your `$PATH` variable. Specify path to **directories**(!) of executables for each program that is not in your `$PATH`. Otherwise, remove line or leave empty.

```
PATH_BAMTOOLS     :=
PATH_BOWTIE       :=
PATH_CD_HIT_EST   := /home/user/src/cd-hit/
PATH_EMBOSS       := /home/user/src/EMBOSS/bin/
PATH_GENSCAN      :=
PATH_GENSCAN_MAT  := (point to actual file)
PATH_MAFFT        := /home/user/src/mafft/
PATH_PERL         :=
PATH_REPEATMASKER :=
PATH_RSCRIPT      := /home/user/src/R/bin/
PATH_SAMTOOLS     :=
PATH_TGICL        :=
PATH_TRINITY      :=
PATH_BLAST        :=
```

Indicate whether WU- or NCBI-BLAST should be used [0 WU, 1 NCBI].

```
NCBI_BLAST := 1
```

Store intermediate and final files in specified location. Make sure that enough space is available to store intermediate output of trinity, blast results, read alignments, …).

```
OUTPUT_DIR := /data/output
```

Input reads in fastq format. In case of paired end data, indicate elements of pair by "R1" and "R2" in filename (Example: `sampleA_R1.fq`, `sampleA_R2.fq`). All files must be in the same format (one of fastq, fasta, gzipped).

```
READ_DIR := /data/reads/
```

Reference transcriptome in GenBank format as provided by NCBI:

`http://ftp.ncbi.nlm.nih.gov/genomes/[YOUR_REF_SPECIES]/RNA/rna.gbk.gz`

```
REF_TRANSCRIPTOME := /data/human.gb
```

Specify taxonomy id of species to assemble. FRAMA connects to NCBI (once) to fetch necessary species information.

```
SPEC_TAXID := 458603
```

We use genome wide annotation packages from Bioconductor to assign functional annotation to the resulting transcript catalogue. Provide (and install) the annotation package corresponding to your reference species.

```
OPT_ANNOTATION := org.Hs.eg.db
```

optional

If you already have extracted mRNA and CDS sequences in FASTA format, provide them to FRAMA. Additionally, you can add a repeat (soft) masked FASTA of your reference sequence in order to skip RepeatMasking step.

```
REF_TRANSCRIPTOME_FASTA             := /data/human_mRNA.fa
REF_TRANSCRIPTOME_FASTA_MASKED      := /data/human_mRNA.fa.masked
REF_TRANSCRIPTOME_FASTA_CDS         := /data/human_cds.fa
REF_TRANSCRIPTOME_FASTA_CDS_MASKED := /data/human_cds.fa.masked
```

CDS inference is based on the coding sequence of the orthologous reference transcript. You can extend the number of orthologs used to infere the appropriate CDS by providing a table with mappings between orthologous transcript from different species. The first column must contain accession of the reference transcript. Add one column for each species you want to use and use 'NA' to indicate unknown orthologs. Additionally, specify taxonomy ID of each species in the first line (starting with #, tab separated). Keep in mind, that we perform a multiple sequence alignments with all coding sequences. Therefore, the number of species used will have an influence on runtime. Additionally, you must provide a fasta file containing all coding sequences mentioned in table (`ORTHOLOG_FASTA`).

```
ORTHOLOG_TABLE := /data/ortholog_table.csv
ORTHOLOG_FASTA := /data/ortholog_cds.fa
```

Example content `ORTHOLOG_TABLE` (also, take a look at `exampe/ortholog_table.csv`)

```
#9606   10090   10116   9615
NM_130786       NM_001081067    NM_022258       NA
NM_001198819    NM_001081074    NM_133400       XM_534776
NM_001198818    NM_001081074    NM_133400       XM_534776
```

We keep a note in GenBank output about the sequence name and species used to annotated the CDS. If multiple equally valid coding sequencing are found, the first species in `SPECIES_ORDER` will be used. Please specify the order of columns (0-based) in `ORTHOLOG_TABLE` to indicate your preferred order of species. Example:

```
SPECIES_ORDER := 0,2,1
```

Specify the primary processing steps you want to apply to the raw trinity assembly (space separated list) in preferred order. Possible steps are: `cd-hit` and `tgicl`. Leave empty to skip primary processing.

```
ASSEMBLY_PREPROCESS := cd-hit tgicl
```

Soft masks repeats in assembly and reference. Set to 0 if you want to skip repeat masking.

```
REPEAT := 1
```

Software parameter

!Consult manual for external software!

Number of cpus. This will be used for any software which runs in parallel.

```
OPT_CPUS := 2
```

If SGE is available (qsub), it will be used for blast jobs. Specify number of jobs.

```
OPT_MAX_SGE := 20
```

Trinity

Single end (s) or paired end (pe) reads?

```
OPT_READTYPE := s
```

Consult trinity manual.

Added automatically: `--no_cleanup`

```
OPT_TRINITY   := --JM 10G --seqType fa
OPT_BUTTERFLY :=
```

RepeatMasker

Repeat masking reference/assembly.

Added automatically: `-xsmall -par OPT_CPUS`

```
OPT_REPEAT_REF_TRANSCRIPTOME := -species human -engine ncbi
OPT_REPEAT_ASSEMBLY          := -species human -engine ncbi
```

CD-HIT-EST

Added automatically: `-T OPT_CPUS`

```
OPT_CD_HIT_EST :=
```

TGICL

Added automatically: `-c OPT_CPUS`

```
OPT_TGICL :=
```

misassembled contigs

Used to detect fusion transcript. Specify maximum overlap (`-max-overlap`) between CDS regions (specifically: blast hits by coding sequences of reference transcriptome), minimum length of alignment (`-min-frac-size`), identity (`min-identity`) and coverage (`min-coverage`) thresholds.

```
OPT_FUSION := -max-overlap 5.0 -min-frac-size 200 -min-identity 70.0
  -min-coverage 90.0
```

BLAST

BLAST and GENBLASTA Paramater, respectively.

Added automatically: `–wordmask=seg lcmask –topcomboN 3 –cpus 1`

```
OPT_BLAST :=
```

SBH requiremnts

Specify minimum required identity and coverage to consider hit as SBH.

```
OPT_SBH := –identity=70.0 –coverage=30.0
```

Scaffolding

Specify minimum required identity and contig coverage of blast hit to consider contig as possible scaffolding fragment.

```
OPT_FRAGMENTS   := –identity 70.0 –query–coverage 90.0
```

Specify minimum overlap between fragments in alignment to apply filtering rules (example: keeps sequence with higher similarity to reference if fragments differ over 98% in overlap, if overlap exceed 66% of contig length)

```
OPT_SCAFFOLDING := –fragment–overlap 66.0 –fragment–identity 98.0
```

CDS prediction

Add '-predictions' if you don't want to use predicted coding sequences (XM Accessions) for CDS inference. Don't use if your reference contains "XM" Accessions [TODO].

```
OPT_PREDICTCDS := –predictions
```

Output files

important files

File	Description
transcriptome.gbk *	GenBank file describing **all annotated sequences**.

File	Description
transcriptome_CDS.fa	Fasta with **coding sequences**.
transcriptome_mRNA.fa	Fasta with **transcript sequences** (w/o introns; clipped ends).
transcriptome_CDS.csv	Coordinates of CDS for mRNA sequences.
assembly_pripro.fa	Trinity assembly after primary processing.
annotation.pdf	General overview of transcript catalogue
annotation.csv	Table containing summary for each annotated transcript.

*mRNA feature instead of 'gene' feature to limit mRNA boundaries in case of misassembled contigs

functional annotations (based on reference)

Table containing GO Terms associated with each annotated transcript. Also, overview of covered GO Terms and genes in total (genes_per_ontology) and in more detail (genes_per_path).

```
tables/gene_ontology.csv
tables/gene_ontology_genes_per_ontology.csv
tables/gene_ontology_genes_per_path.csv
```

Same as above, but for KEGG Pathways.

```
tables/kegg.csv
tables/kegg_covered.csv
tables/kegg_genes_per_path.csv
```

intermediate output

trinity/

Trinity output (including intermediates).

transcripts/

Running FRAMA creates a lot intermediate output which might come in handy in downstream analysis. Each transcript assignment is stored in a separate directory in

```
transcripts/
```

with the naming pattern according to assigned ortholog.

```
transcripts/SYMBOL_ACCESSION/
```

This directory includes the following files:

Result in GenBank format.

```
_final.gbk
```

Raw GENSCAN output.

```
CDS_genscan.txt
```

Assignment of transcript accession to GENSCAN prediction based on blast hits.

```
CDS_genscan_annotated.txt
```

Multiple sequence alignment with orth. species requested in `ORTHOLOG_TABLE`

```
CDS_alignment.aln
```

BLAST databases for reference and assembly.

```
db/
```

BLAST results including average for each HSP-group (`avg_*`) and best hit per query (`best_*`).

```
blast/raw_*
blast/avg_*
blast/best_*
```