MIRfix

Documentation

Contents

Installation

1.1 Conda environment

The easiest way to install MIRfix or its dependencies is using bioconda. For this to work please first install conda, then configure conda to use the bioconda channel.

In short, first install conda, then run
"'conda config –add channels defaults"'
"'conda config –add channels bioconda"'
"'conda config –add channels conda-forge"'
Once conda is installed and configured, you can simply install MIRfix:
"'conda create -n mirfix mirfix "' will install MIRfix in the conda environment mirfix.

You can than use "'conda activate mirfix"' and run "'bash runMIRfix.sh"' from there.

Or you install all dependencies for MIRfix into a conda environment as follows:

In the *envs* directory you can find a file *MIRfix.env*, a simple
"'conda env create -f MIRfix.env"' will install all dependencies in the conda environment mirfix.

You can than use "'conda activate mirfix"' and run "'bash runMIRfix.sh"' from a local clone of the git repository or the unpacked source.

1.2 Pre-requests for manual installation

1.2.1 Python (version and modules)

This pipeline is coded with python version: **Python 2.7.15**. Please make sure to use the same version to avoid any error. Usually errors are raised when using different python versions. Also, special modules are needed to run this pipeline. Please make sure that the following packages, modules, libraries, etc are installed.
∗*The following group should be automatically installed with Python 2.7.15*:

- module *__future__* to use module *division*
- *os* module
- *operator* module
- *math* module

- *sys* module
- *re* module
- *Tkinter* package, and please make sure *tkFileDialog & Tkconstants* are included in the package

∗*In addition, the following needs to be installed:*

- *Bio* package (biopython)
  and please make sure that these are included:
  - Package SeqIO
  - Module *seq*
  - (sub)Package *AlignIO*
  - The submodule *Bio.Align.Applications*(to use ClustalwCommandline class) in the (sub)Package *Align*
- *matplotlib* library to use :
  - *pyplot* and
  - *matplotlib.backends.backend_pdf*

∗*Python needs some dependencies, one of the most known is:*

- *numpy* package

1.2.2 Alignment tools (clustalw and dialign2)

Two different alignment tools are called in the script,
clustalw (CLUSTAL 2.0.12 multiple sequence alignment)
and
dialign2 (*Anchored DIALIGN* you can find it at http://dialign.gobics.de/).
As this tool requires setting an environment variable pointing to the substitution matrices downloaded with dialign installation, the user is asked to upload this directory in the GUI of the pipeline. We recommend copying this folder to your work directory for simplicity.

1.2.3 Vienna RNA package (RNAfold and RNAalifold)

RNAfold and *RNAalifold* are secondary structure prediction tools, which calculate the minimum free energy (MFE).
It is found at https://www.tbi.univie.ac.at/RNA/#download
HINT: in the .py script add your own path to the RNA package in the line:
e.g: sys.path.append('*/local-path/vrna249/lib64/python2.7/site-packages/*')

Inputs

The pipeline requires 7 inputs:

- 4 Files
- 2 Directories
- 1 Number (number of the flanking nucleotides)

2.1 The Files and its format

The four files are:

1. The precursors file (fasta file, .fa) OR list of precursor files (text file, .txt)
2. The mature sequences file (fasta file, .fa)
3. The mapping file (text file, .txt)
4. List of the sources fast files (text file, .txt)

IMPORTANT: In all the following, the order of the attributes is **necessary** and has to follow the order given in **roman numbers**

2.1.1 The precursors file and the list of files

Precursors file(s) is a fasta file that contains the microRNA precursors. The sequences are considered as records. Each record must be as follows:

I II III+IV V VI
>*Precursor-name PrecursorID Species-Name Family-name “stem-loop”*
...**the sequence**

I, II, III+IV are the main attributes that have to be provided for each record(precursor).

Example:
>cel-let-7 MI0000001 Caenorhabditis elegans let-7 stem-loop
UACACUGUGGAUCCGGUGAGGUAGUAGGUUGUAUAGUUUGGAAUAUUACCACCGGUGAACUAU

I: cel-let-7 *(Precursor-name)*
II: MI0000001 *(PrecursorID)*–>Unique ID

III+IV: Caenorhabditis elegans *(Species-Name)*
V: let-7 *(microRNA-Family-name)*
VI: *"stem-loop"*

(see *examples/inputs/MIPF0000159.fa* in the supplementary materials)

The List of files(precursors files/families):
This is needed in case of processing more than one precursors files (families) in the same time.
Each line, has to contain the name of one precursor file, with or without *".fa"* extension.
For example, if the 3 files (families) MIPF0000002.fa, MIPF0000600.fa and test.fa are 3 different precursors files. Create a list of them in a text file, e.g:

"list.txt": OR	"list.txt":
MIPF0000002	MIPF0000002.fa
MIPF0000600	MIPF0000600.fa
test	test.fa

(see *examples/inputs/list.txt* in the supplementary materials)

2.1.2 The mature sequences file

It is a fasta file containing **all** mature sequences of the **all** given precursors, followed by an empty line for processing.
(Note: Make sure always to use *enter* (\n) to add an empty line at the end of the mature sequences file (i.e. The cursor should be at a new empty line).) The records must be as follows:

```
    I               II        III+IV              V
>Mature-name MatureID Species-Name microRNA-Family-name+orientation(5p or 3p)
```
+orientation(5p or 3p)
...**sequence**

I and II, are the main attributes that have to be provided for each record

Example:
>cel-let-7-3p MIMAT0015091 Caenorhabditis elegans let-7-3p
CUAUGCAAUUUUCUACCUUACC

I: cel-let-7-3p *(Mature-name)*
II: MIMAT0015091 *(MatureID)*–>Unique ID
III+IV: Caenorhabditis elegans *(Species-Name)*
V: let-7-3p *(microRNA-Family-name+orientation(5p or 3p))*

(see *examples/inputs/mature.fa* in the supplementary materials)

2.1.3 The mapping file

This text file maps between the data in the precursors and mature sequences files and is also ended with an empty line for processing.

Each line of this file is a new record. (Note: Make sure always to use *enter* (\n) to add an empty line at the end of the mature sequences file (i.e. The cursor should be at a new empty line).) The records must be as follows:

I	II	III	IV	V	VI	VII
FamilyID	Family-name	PrecursorID	Precursor-Name	MatureID	Mature-coordinates	Mature-name

Please Note The family ID of a group of precusors, has to be always the name of the fasta file of this group of precusors (family). i.e. *"FamilyID.fa"* is the name of the fasta file of the precusors belong to the same family or group.

Mature-coordinates, are the start and end of each mature sequence in its given precursor, separated by 2 dots "..", e.g
12..34: start position is 12 and end position is 34. However, if the coordinates are not available or not easy to get, just replace the numbers by 'X' and 'Y' or any other alphabets (e.g X..Y).
The representation at the beginning of this section, shows an entry for a precursor with only one mature sequence. In other cases, the precursor could have 2 or more annotated mature sequences as in `miRBase`. In the cases of more than one annotated mature sequence, simply the **MatureID, Mature-coordinates, and Mature-name** must be added, respectively, following to V, VI and VII. In the following are two examples of, one mature and 2 matures cases:

one annotated mature miRNAs:

I	II	III	IV	V	VI	VII
MIPF0000600	mir-634	MI0015115	ppy-mir-634	MIMAT0016073	61..82	ppy-miR-634

from I to VII, respectively:
FamilyID *(as mentioned above, it is also the name of the fasta file containing all the related precursors. i.e. in this example, the precursor of ID:MI0015115, exists in the fasta file MIPF0000600.fa)*, Family-name, PrecursorID, Precursor-Name, MatureID, Mature-coordinates and Mature-name.

Two annotated mature miRNAs:

I	II	III	IV	V	VI	VII	VIII
MIPF0000002	let-7	MI0000001	cel-let-7	MIMAT0000001	MIMAT0015091	17..38	60..81

IX	X
cel-let-7-5p	cel-let-7-3p

The difference here, is that VI is the *MatureID**, VIII is the *Mature-coordinates** and X is the *Mature-name**

(see *examples/inputs/mapping.txt* in the supplementary materials)

2.1.4 List of the sources fast files

List of fast files that can be genomes or transcripts or any fasta sequences, which are the sources of the uploaded precursors. Each line of this file, is the *directory + the name of the fast file (precursor(s) source).*
please note that the line (*directory + the name of the fasta file*) must contain the name of the species as it is given in the record of the precursor (but it is not case sensitive i.e. regardless small and capital letters).

Example:

The fasta file:

`/genomes/Metazoan-Animals/Caenorhabditis_elegans/ENSEMBL/Caenorhabditis_elegans_WS235.fa`
OR
`/genomes/Metazoan-Animals/Caenorhabditis_ELEGANS/ENSEMBL/test.fa`

is the source fasta file(genome or any other sequence file)

for the precursor:
>cel-let-7 MI0000001 Caenorhabditis elegans let-7 stem-loop
UACACUGUGGAUCCGGUGAGGUAGUAGGUUGUAUAGUUUGGAAUAUUACCACCGGUGAACU

As in this example, it doesn't matter where the name of the species appear, but the name should appear at least once in the source line − (*directory+fasta file name*) and as it is written in the related precursor's record (but not case sensitive). As you see, *"elegans"* has different cases but written correctly, so in both example lines it is correct.

(see *examples/inputs/genomes_list.txt* in the supplementary materials)

2.2 The Directories

There are 3 directories as inputs for this pipeline:

1. The user's output directory (location)
2. The directory of the input fasta files
3. The matrices directory

2.2.1 The output directory (location)

The user is requested to specify the desired output folder, where all the results will be automatically stored. In this folder, for each family or fasta file, a folder with the name of the precursor's file (family) will be created. This file will contain all the related results, and the folder's name is *"familyname(filename).out"*. This is explained in the *"output"* section.

2.2.2 The directory of the input fasta files

The directory (folder) of the target file(s) listed (The list explained in *2.1.1*, *"The List of files"*) to be processed in the pipeline. i.e. in case the list contains more than one file, make sure that the all listed files are in the same directory (folder).

2.2.3 The matrices directory

Anchored dialign is one of the two alignment tools used in this pipeline, and this tool needs some files and a BLOSUM matrix. A folder containing *USER_GUIDE*, BLOSUM matrix and other files, is downloaded with the installation of the Dialign2 tool. This folder must be opened as a directory in the pipeline, to set the environment variable to this directory, as dialign requests.

2.3 The Numbers

2.3.1 Number of flanking nucleotides

In case using GUI a related dropdown list, contains numbers from *zero* to *50*, representing the number of the nucleotides the user prefers to keep upstream of the 5' mature sequence and downstream of the 3' mature sequence in the precursors. When nothing is chosen from the dropdown list, the default number of the flanking nucleotides is *10*.
In case of using the command line version, this number will be added to the given parameters as described in section *3.2*

2.3.2 Number of threads

In case using GUI a related dropdown list, contains numbers from *one* to *50*, representing the number of threads (parallel processes) the user prefer to run. When nothing is chosen from the dropdown list, the default number is *one*. In case of using the command line version, this number will be added to the given parameters as described in section *3.2*

Running the pipeline

This pipeline can be run in two ways: Using the simple *GUI* or using the *command line*.

3.1 Using the GUI

After installing all the pre-requests and the python script file *MIRfix.py*. Run *MIRfix.py* from the command line, from the directory where the script file is installed/saved. For example:
[ali@homedirectory]$ python <path-to-script>/MIRfix.py

After running this command, a graphic user interface will be launched Figure 3.1, that guides you through uploading the files explained in the previous section (*Inputs*).

Uploading the files using GUI

Figure 3.1 shows the GUI of the pipeline, and the numbers from 1 to 9 are:

(1) The output location (directory) (*section 2.2.1*)
(2) The directory of the file(s) in the list. (*section 2.2.2*)
(3) The list of files to process which can contain only one file.(*section 2.1.1*)
(4) The list of the genomes related to the uploaded precursors (*section 2.1.4*)
(5) The mapping file (*section 2.1.3*)
(6) The mature sequences fasta file (*section 2.1.2*)
(7) The matrices file (*section 2.2.3*)
(8) The number of the desired flanking nucleotides (Default 10) (*section 2.3.1*)
(9) The desired number of threads (processors) to run the code (minimum 1)(*section 2.3.2*)
(10)The submission button after uploading all related files and opening the directories

In case any input is missing, the text of the corresponding button will turn *red*, and when files or directories are entered and uploaded the directories and the file names will appear in *green* on the corresponding buttons (fields).

3.2 Using the command line

Using the commandline the program can be run faster, as it allows multiprocessing the files. The code takes 9 ordered parameters as follows:
(1) The desired number of threads (processors) to run the code (minimum 1)(*section 2.3.2*)
(2) The output location (directory) (*section 2.2.1*)
(3) The directory of the file(s) in the list. (*section 2.2.2*)
(4) The list of files to process which can contain only one file.(*section 2.1.1*)
(5) The list of the genomes related to the uploaded precursors (*section 2.1.4*)
(6) The mapping file (*section 2.1.3*)
(7) The mature sequences fasta file (*section 2.1.2*)
(8) The number of the desired flanking nucleotides (Default 10) (*section 2.3.1*)
(9) The matrices file (*section 2.2.3*)

In case the installation was done with conda environment, no need for the parameter (9) and this will be automatically called.

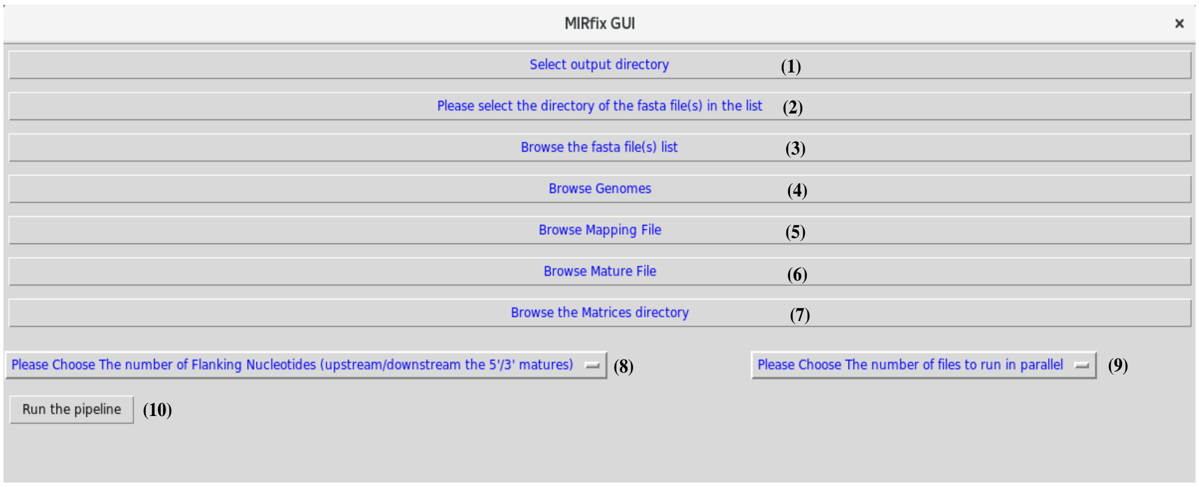
MIRfix GUI
Select output directory (1)
Please select the directory of the fasta file(s) in the list (2)
Browse the fasta file(s) list (3)
Browse Genomes (4)
Browse Mapping File (5)
Browse Mature File (6)
Browse the Matrices directory (7)
Please Choose The number of Flanking Nucleotides (upstream/downstream the 5'/3' matures) (8)
Please Choose The number of files to run in parallel (9)
Run the pipeline (10)

Figure 3.1: GUI window

Outputs

For each given file, a result folder will be created in the choosen output directory. The name of the folder will be *filename.out*. For example, for the file *"myprecursors.fa"*, the folder *"myprecursors.out"* will be created in the output directory defined by the user and containing all the result files that are explained in this section.
The output will be:

1. Editing the original uploaded files
2. Newly created files

4.1 Edited uploaded files

The **mature** sequences and the **mapping** file will be edited only if at least one precursor has no annotated mature sequence.

4.1.1 Editing the uploaded mature file (original file)

When a mature sequence is predicted, it will be added to the original uploaded mature file following the same format of the record with few changes.
The general record's form:

```
   I             II            III
```
>**mature-name matureID species-name**
...nucleotide sequence of the predicted mature.

I: **Mature-name**: Is the name of the precursor, plus *"-mat"* (*precursorname-mat*).
II: **MatureID**: As it is very important that the IDs should be unique, this ID consists of two parts, it is a combination of the matureID and the precursorID separated by " / ". The first part is the matureID of the most similar mature sequence, to the predicted one (this part is explained in the paper). The second part is the precursorID of the processed precursor (precursor with predicted mature).
III: **Species-name**: Obviously, the species name is the name of the species of the processed precursor.

Example: The following precursor without annotated mature sequence, i.e. this precursor has no reference line in the mapping file.

Ip IIp IIIp IVp
>ppy-mir-296 MI0014874 Pongo pygmaeus miR-296 stem-loop
AGGACCCUUCCGGAGGGCCCCCCCUCAAUCCUGUUGUGCCUAAUUCAGAGGGUUGGGUGGAGG

And its predicted mature based on the mature sequence with ID:
Im IIm
>mmu-miR-296-5p MIMAT0000374 Mus musculus miR-296-5p

Then, the predicted mature ID will be:
I II III
>ppy-mir-296-mat MIMAT0000374/MI0014874 Pongo pygmaeus
I: is Ip + *"-mat"*
II: is IIm + IIp
III is IIIp

For the full example, see the supplementary materials (*examples/inputs/mature.txt*); the last line is added to the uploaded mature file, after processing the precursors.

4.1.2 Editing uploaded mapping file (original file)

Accordingly, a new line will be added to the mapping file, it contains the new corresponding IDs, names, positions, etc. The following line is **added** to the **end** of the mapping file and it corresponds to the example in *4.1.1*:

MIPF0000159 mir-296 MI0014874 ppy-mir-296 MIMAT0000374/MI0014874 13..33 ppy-mir-296-mat

For the full example, see the supplementary materials (*examples/inputs/mapping.txt*); the last line is added to the uploaded mapping file, after processing the precursors.

4.2 Newly created files

As this pipeline works at two levels and in order to meet the possible users' interest in tracking the results, the files of each level are saved into the output folder. In this section the files are separated into to subsections, the first shows the names and the explanation of the outputs of the precursors' level before the final processing (alignment level). The second explains the last outputs and the alignment level outputs. In addition to the latter subsections, the statistical and summary files are also explained in this section.

4.2.1 Precursors' Level outputs

4.2.1.1 Predicted mir* sequences

A fasta file containing all the predicted mir* sequences, for the precursors with one annotated mature sequence and the precursors without any annotated mature sequence.

Please note that the IDs of these sequences, contain the ID of the related precursor at the end, because in some cases the same annotated miRNA refers to more than one precursor.
The name of this file: *precrusorsfilename-mirstar.fa*

(see *examples/main_outputFiles/MIPF0000159-mirstar.fa* in the supplementary materials)

4.2.1.2 mir* mapping file

A txt file matches each predicted mature ID to its precursor ID with the coordinates (the start and end positions of the predicted mature sequence in its precursor). This file represents the results at the end of the precursors' level. In case of any changes at the alignment level, the numbers are updated in the file *precrusorsfilename-FinalCoor.txt* (section 4.2.2.1)
The name of this file: *precrusorsfilename-mirstar-map.txt*

(see *examples/main_outputFiles/MIPF0000159-mirstar-map.txt* in the supplementary materials)

4.2.1.3 Precursors without annotated mature miRNAs

Fasta file includes all precursors without annotated mature sequence(s).
The name of this file: *nomat-precursorsfilename.fa*

(see *examples/main_outputFiles/nomat-MIPF0000159.fa* in the supplementary materials)

4.2.1.4 Precursors result file - fasta file

Fasta file including all processed precursors.
The name of this file: *precursorsfilename-Final.fasta*

(see *examples/main_outputFiles/MIPF0000159-Final.fasta* in the supplementary materials)

Please note that In case of any further changes in these precursors, produced by the final correction process (alignment level), additional file will be created *see 4.2.2.2*.

4.2.1.5 Precursors result file - Stockholm file

Alignment file for the precursors in Stockholm format.
The name of this file: *precursorsfilename.stk*
In case this file is created without sequences, it is just because the final result contains one or no sequences (i.e. no sense to create alignment file).
(see *examples/main_outputFiles/MIPF0000159.stk* in the supplementary materials)

4.2.1.6 Alignment output sequences

The sequences alignment used to create the previous Stockholm format files.
The name of this file is: *precrusorsfilename-Final.fa*

(see *examples/main_outputFiles/MIPF0000159-Final.fa* in the supplementary materials)

4.2.2 Alignment Level outputs (Final stage)

4.2.2.1 Final coordinates txt file

The final mapping of the mir and mir* sequences to their precursors, and the corresponding coordinates. Each line contains in the following order:
(PrecursorID) (mirID) (mir*ID) (mir, start..end positions) (mir, -start..end positions)
The name of this file: *precrusorsfilename-FinalCoor.txt*

(see *examples/main_outputFiles/MIPF0000159-FinalCoor.txt* in the supplementary materials)

4.2.2.2 Corrected precursors result file - fasta file

At the end, one more process applied and this file will be created only if any precursor from the precursors' result file is corrected (changed).
The name of this file: *precursorsfilename-corrected.fasta*

(see *examples/main_outputFiles/MIPF0000159-corrected.fasta* in the supplementary materials)

4.2.2.3 Corrected precursors result file - Stockholm file

Alignment file for the corrected precursors result file (if it exists), in Stockholm format.
The name of this file: *precursorsfilenamecorrected.stk*

(see *examples/main_outputFiles/MIPF0000159corrected.stk* in the supplementary materials)

4.2.2.4 Corrected predicted mir* sequences

Same as file in *section 4.2.1.1*, with changing the mir* sequences related to the corrected precursors (corrected misaligned).
The name of this file: *precursorsfilename-mirstar-corrected.fa*

(see *examples/main_outputFiles/MIPF0000159-mirstar-corrected.fa* in the supplementary materials)

Sub-output files

Two (one in case no changes at the alignment level) files are produced with *".anc"* extension and these are the anchoring points used for dialign2. The file *precrusorsfilename-Final.anc* is produced at the precursor' level and the file *precrusorsfilename-.anc* is produced at the last stage (alignment level).

4.2.3 Statistics outputs

4.2.3.1 PDF format figure

The name of this file is *precursorsfilenamestatistics.pdf*
(see *examplesmain_outputFiles/MIPF0000159statistics.pdf* in the supplementary materials)

The bars are divided into 5 groups and please note that the white bars refer to *zero*, Figure 4.1 :

a) Total number of precursors (*processed/removed*)

b) Original sequences distributions (*without annotated mature/with 1 annotated mature/with more than one annotated matures*)

c) Changed sequences (*totally changed/corrected misaligned(covers changed and non-changed precursors)*)

d) Precursors without annotated mature(s) (*predicted/non-predicted miRNAs*

e) Entropy of the alignment (*new entropy/original entropy.* NB: The entropies are equal in 2 cases: the original and the final result are same, or the final result includes no or only one sequence)

4.2.3.2 Summary txt file

This file includes more statistical details and numbers, as well as the IDs of the precursors included in the statistics.
The name of this file: *precursorsfilename-summ.txt*

(see *examplesmain_outputFiles/MIPF0000159-summ.txt* in the supplementary materials)

4.2.3.3 JSON format file

Besides the txt summary file, a json file format contains the most useful statistical numbers to be easily parsed and read by machines.
The name of this file is: *precursorsfilename.json*

(see *examplesmain_outputFiles/MIPF0000159.json* in the supplementary materials)

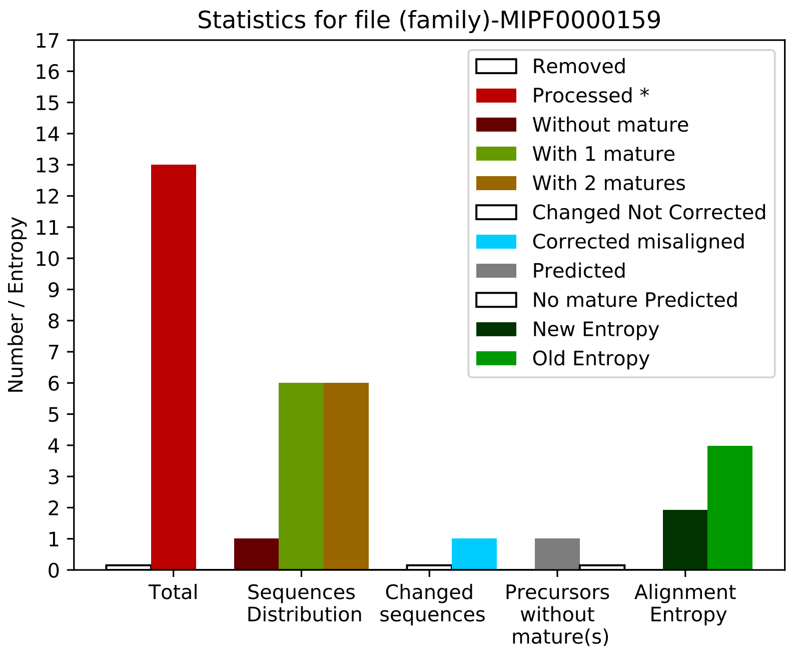

Figure 4.1: Example of the statistics output figure. The white columns are zeros.