FasParser Manual

[For version 2.8.0 | 2019-4-20]

Sun Yan-Bo

Kunming Institute of Zoology, Chinese Academy of Science

Contents

Introduction

Background

With the development of sequencing technology in recent times, a great number of molecular sequences (DNA and RNA) have been generated. Molecular analyses based on these sequences have become one of the most important measures for assessing their potential biological significance. The increase in the amount of available sequence data has made its manipulation tricky, especially for those without programming experience. Hence, it has now become necessary to develop one or more user-friendly software to perform such analyses in a batch mode.

Common manipulations may include extracting/removing subsequences or subalignments, estimating the sequence similarity and identifying the non-homologous sequence(s), converting between file formats, and/or alignment trimming to exclude poorly aligned regions from a multiple sequence alignment. While, with the increase in the amount of available sequence data, it made sequence manipulation tricky, especially for those without programming experience.

To achieve an objective of simplifying many common tasks in sequence manipulation for biologists, I provide a new program package named 'FasParser' for manipulating sequence files. It is designed with a user-friendly GUI and batch processing modes, which allows users to handle multiple sequence files in a simple way. Presently, the main functions of FasParser2 involve: (1) batch performing alignment construction with Muscle (Edgar, 2004), Mafft (Katoh, et al., 2002), or Prank (Loytynoja and Goldman, 2005), (2) alignment trimming on poorly-aligned regions, (3) alignment format conversion between FASTA and either PHYLIP, PAML, or NEXUS, (4) sorting & classifying sequences according to accession numbers or

sequence length, which can also rename sequences, (5) concatenating & merging sequences for a particular set of samples from multiple sequence files, (6) translation and ORF prediction, (7) extracting and filtering sequences according to ID, sequence length, or sequence similarity, (8) detecting positive selection with CodeML (Yang, 2007), (9) designing PCR primers with Primer3 (Rozen and Skaletsky, 2000), and (10) extracting consensus-aligned regions between different aligner results (for a same gene). Moreover, there has been designed a specialized Editor for FasParser (v2.0+) to easily viewing and editing sequences.

Citations

If you use this package in your work, please cite these references:

1. *Yan-Bo Sun.* ***FasParser: a package for manipulating sequence data****, Zoological Research 38(2): 110-112, 2017*
2. *Yan-Bo Sun.* ***FasParser2: A Graphical Platform for Batch Manipulating Biological Sequence, Bioinformatics,*** *in press.*

Below is a non-exhaustive list of publications related to the programs integrated:

1. *Edgar RC. 2004.* ***MUSCLE: a multiple sequence alignment method with reduced time and space complexity****. BMC Bioinformatics, 5: 113.*
2. *Katoh K, Misawa K, Kuma K, Miyata T. 2002.* ***MAFFT: a novel method for rapid multiple sequence alignment based on fast Fourier transform****. Nucleic Acids Research, 30: 3059-3066.*
3. *Loytynoja A & Goldman N. 2005.* ***An algorithm for progressive multiple alignment of sequences with insertions****. Proceedings of the National Academy of Sciences of the United States of America, 102: 10557-10562.*
4. *Nei M & Gojobori T. 1986.* ***Simple methods for estimating the numbers of synonymous and nonsynonymous nucleotide substitutions****. Molecular Biology and Evolution, 3: 418-426.*
5. *Untergasser A, et al. (2012)* ***Primer3--new capabilities and interfaces****. Nucleic Acids Res 40(15):e115.*

6. *Yang Z (2007)* ***PAML 4: phylogenetic analysis by maximum likelihood****. Molecular Biology and Evolution 24(8):1586-1591.*

Installation

The 'FasParser' has been developed into a standalone Windows System Application (compiled and tested on Windows 7/10). It can run on most Windows systems with no installation of other programs.

Download the newest setup program (i.e. 'FasParserX.X_setup.exe') from https://github.com/Sun-Yanbo/FasParser/releases to your disk, and then double click it to install the whole package. Normally, it is well using the default installation parameters (by clicking the Next button to end). After successful installation, you would get a screen of the Home page of this package (Figure 1).

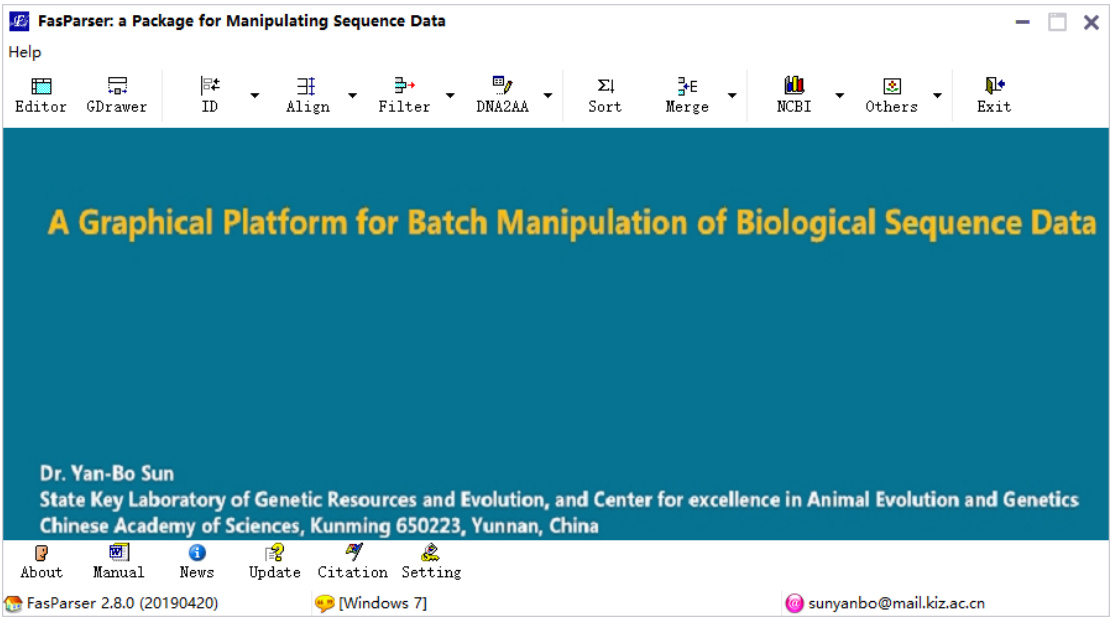


Figure 1. The Home page of the FasParser package.

Package Usage

[Open multiple files]

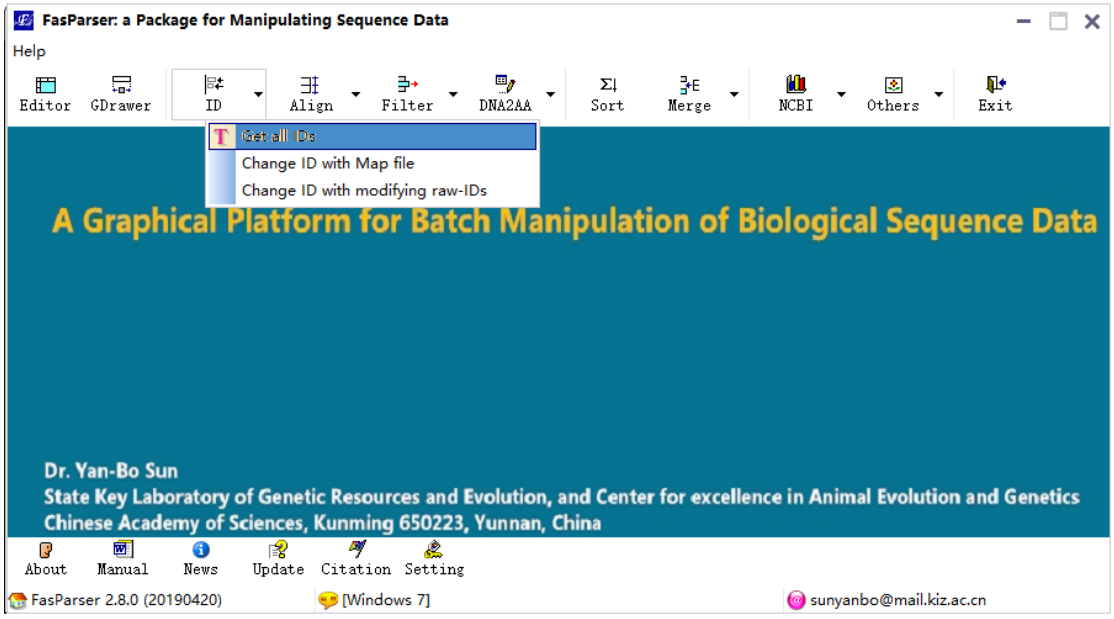
There are 3 different ways to select multiple files into the checking list box:

➔Using the button "Open";

➔Drag the files into the checking list box;

➔Drag the folder into the checking list box;

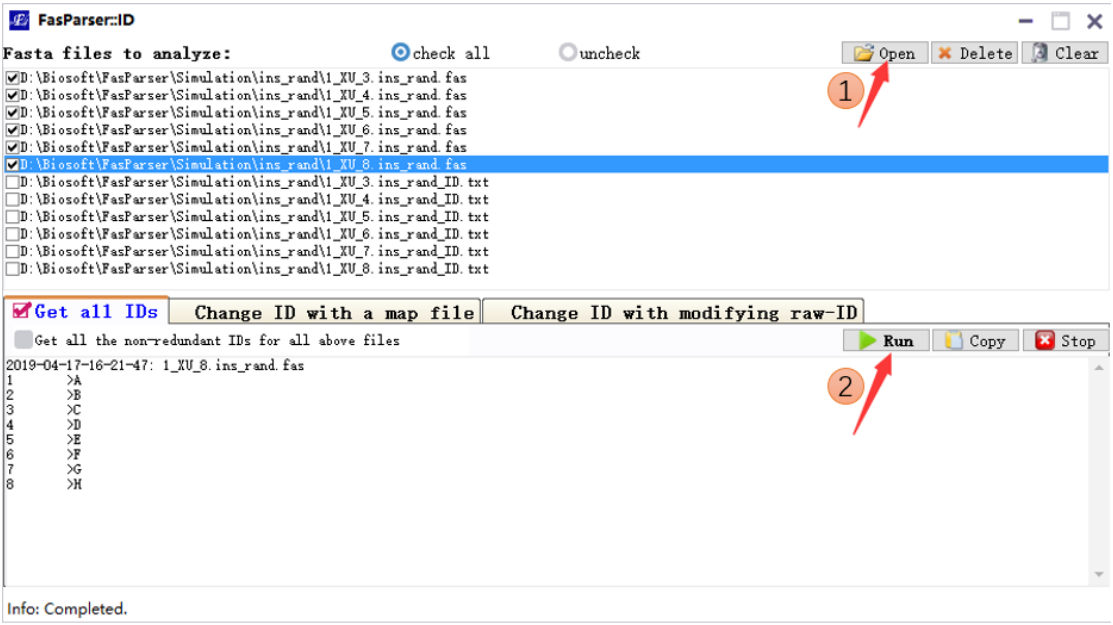
[ID]: Extract or Rename FASTA IDs



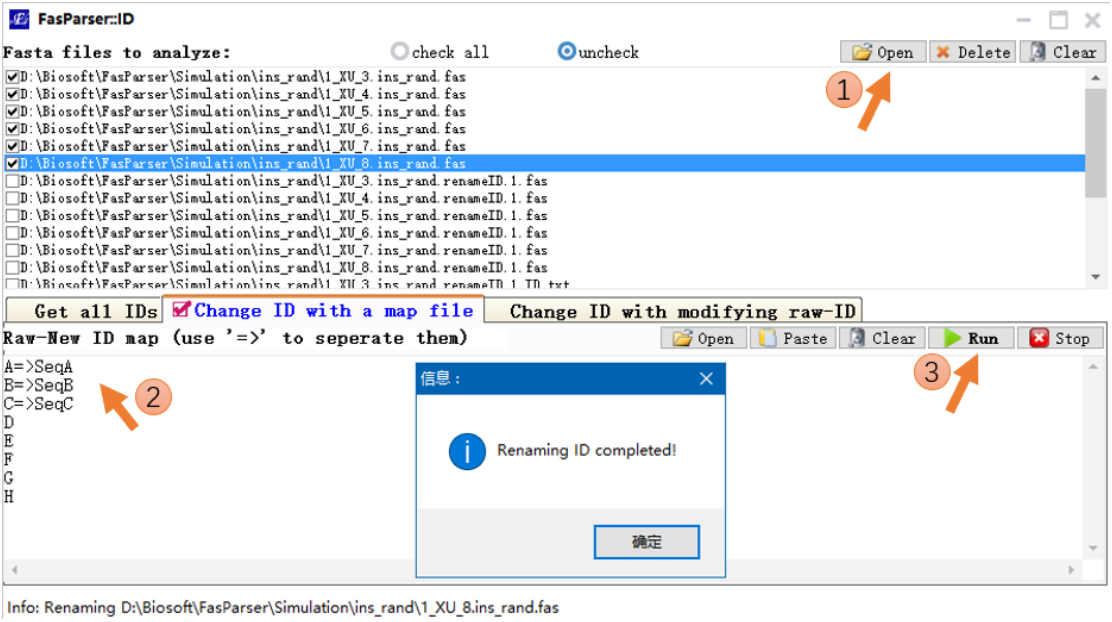
This function is designed to

1) extract IDs of multiple Fasta files;
2) change raw IDs to another according to the user provided raw-new ID map;
3) change raw IDs based on modifying the raw IDs.

➔Extracting IDs:

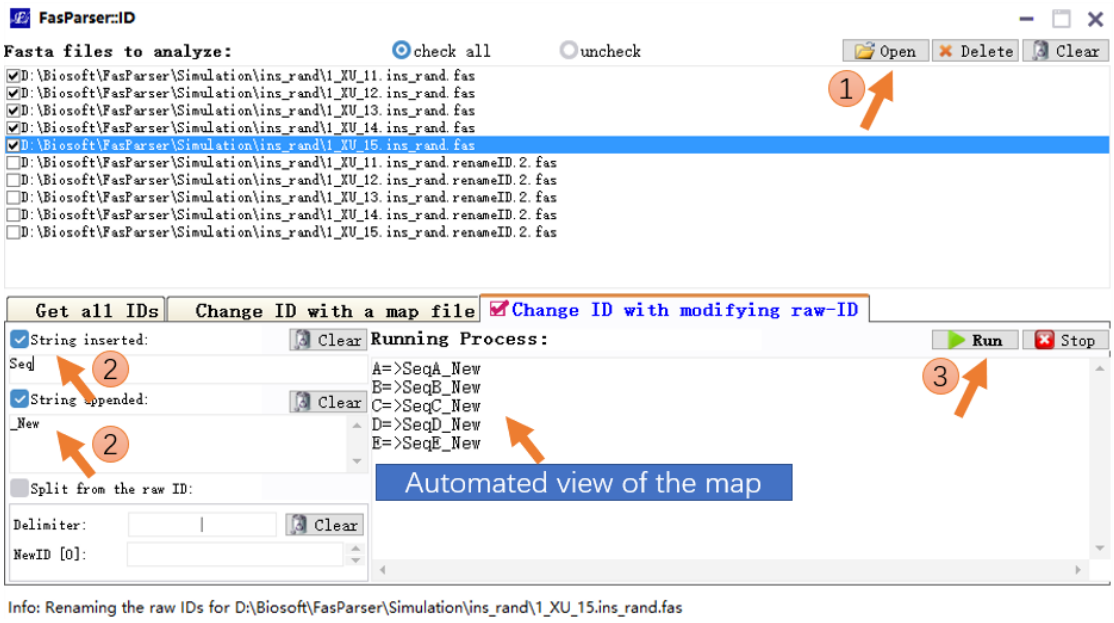


➔Changing raw IDs based on defined ID map file:



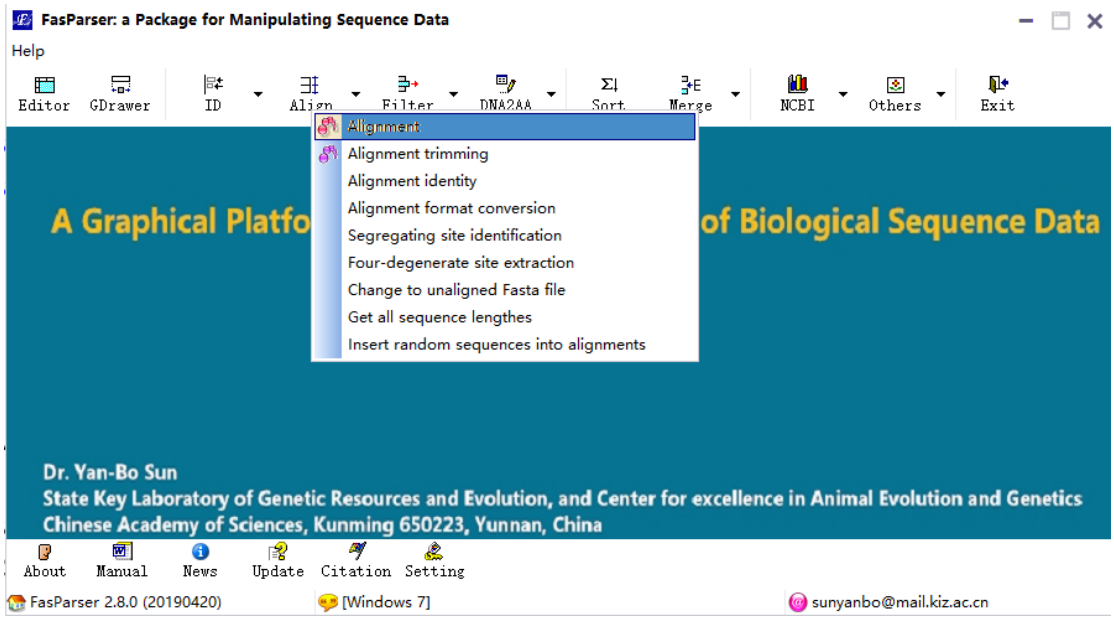
*Please note the symbol "=>" should be present between the raw and new IDs.

➔Changing IDs based on modifying the raw IDs:



*The program can split the raw IDs based on defined delimiter, and add some other strings before the ID, and/or append some strings at the end of ID.

[Align]



This functional module contains 9 different functions to analyze FASTA files, including:

1) Alignment construction,

2) Alignment trimming,
3) Alignment format conversion,
4) Alignment identity calculation,
5) Segregating site identification,
6) 4-degenerate site extraction,
7) Calculate sequence lengths, and
8) Insert random segments into alignment.

➔Alignment construction

This program is designed to **construct alignments for multiple FASTA files**. There are 4 aligners (MUSCLE, KALIGN2, MAFFT, and PRANK, see below reference) available in FasParser. The first two aligners are always faster than Prank, but Prank could generate more accurate results. In addition, the first two aligners can auto recognize the input type (nuclear DNA or protein sequence), while, the Prank should be told the sequence type; user should select the corresponding aligner by yourself (including Prank-DNA, Prank-Codon, and Prank-AA, Figure 2).

The running process information can be viewed in the bottom text box. User should use **Open** button to select the Fasta files to analyze. The output files will be saved in the folder which contains the input files, with '*. <aligner>.fas'. **Please note all the default parameters of the aligners are used by FasParser, and you can add some other parameters in the parameter textbox (see below).**

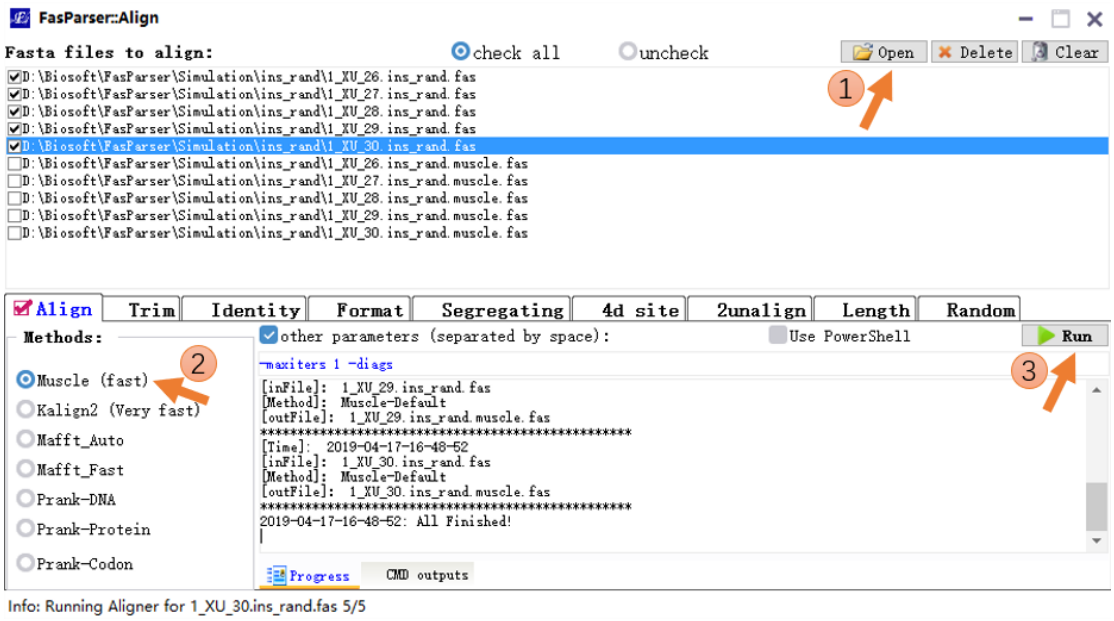


Figure 2. Overview of alignment construction

*The aligned files can be displayed in the checking list box and can be viewed by double clicks.

➔Alignment trimming

Alignment quality is of great importance for downstream analyses, like phylogenetic inference and positive selection detection. The low-quality alignment always produces false positive results, like wrong phylogenetic positions of species. So, it is much better to strip the poorly-aligned regions in the raw alignments before conducting downstream evolutionary analyses. The 'Trim' function provides a simple and efficient trimming method to filter the low-quality regions in each alignment. [It is more suitable for preparing high-quality alignments for positive selection detection]

'Trim' will apply a sliding window method to identify which regions hold low-similarity sequences based on a dynamic cutoff that will be estimated from the total alignment (1000 times). There are two steps to trim an alignment. The 1st step

is for the gap-near regions, and the window will be sliding from the gap "-" to both left and right ends. The 2nd step is for the block regions. After the 1st step trimming, the 2nd will trim again from the left end to right.

User should tell which type of seqs you provided: codon or DNA, and then you can click the "Run" button to start the trimming analyses for multiple files (Figure 3). The output files will be saved in the folder which contains the input files, with '*. trimC.fas' or '*. trimD.fas' for codon or DNA alignment, respectively.

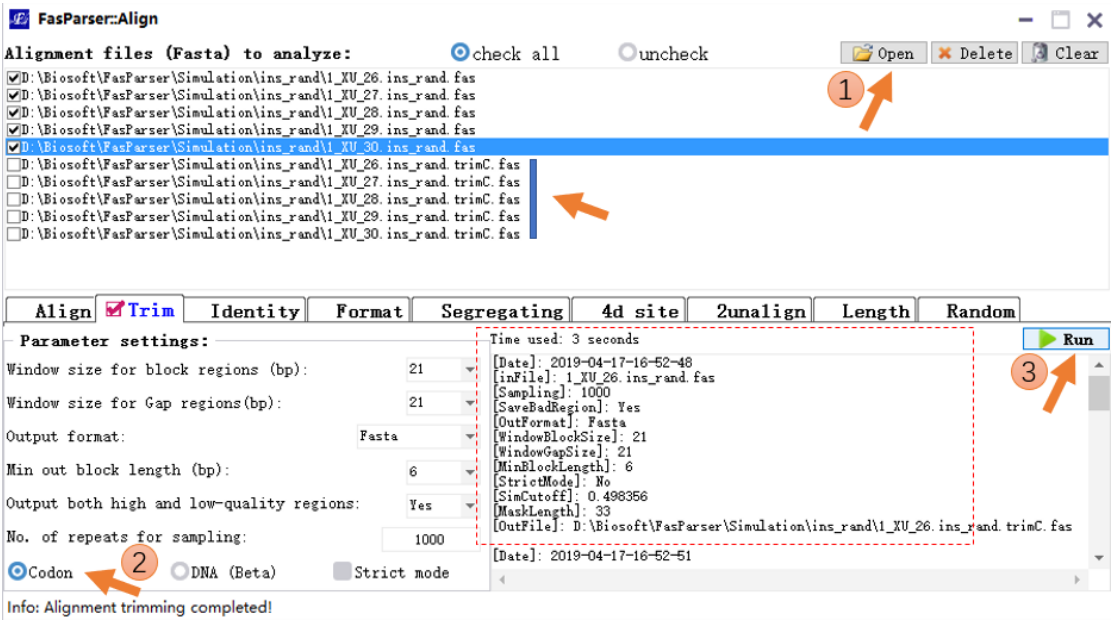


Figure 3. Overview of alignment trimming.

➔Alignment identity calculation

It is important to know the conservation level for a gene or a segment among different species (always refer to the species you studied). If the mean identity of an alignment is too low, it might be due to rapid evolution of this gene or non-homologous prediction for one or more species in this alignment. For each alignment, 'Sim' (Similarity) provides three identity calculations: the minimum

pair-wise identity within the alignment (min.); the maximum pair-wise identity (max.); and the mean identity (mean).

Similarity, user should select files to analyze through clicking the 'Open' button, and then click the "Run" button to start the estimates (Figure 4).

Please note that during the similarity estimates, the gaps can be taken into account according to "Score = Score -1" if one base match a gap "-". You can avoid it by unchecking the selection box.

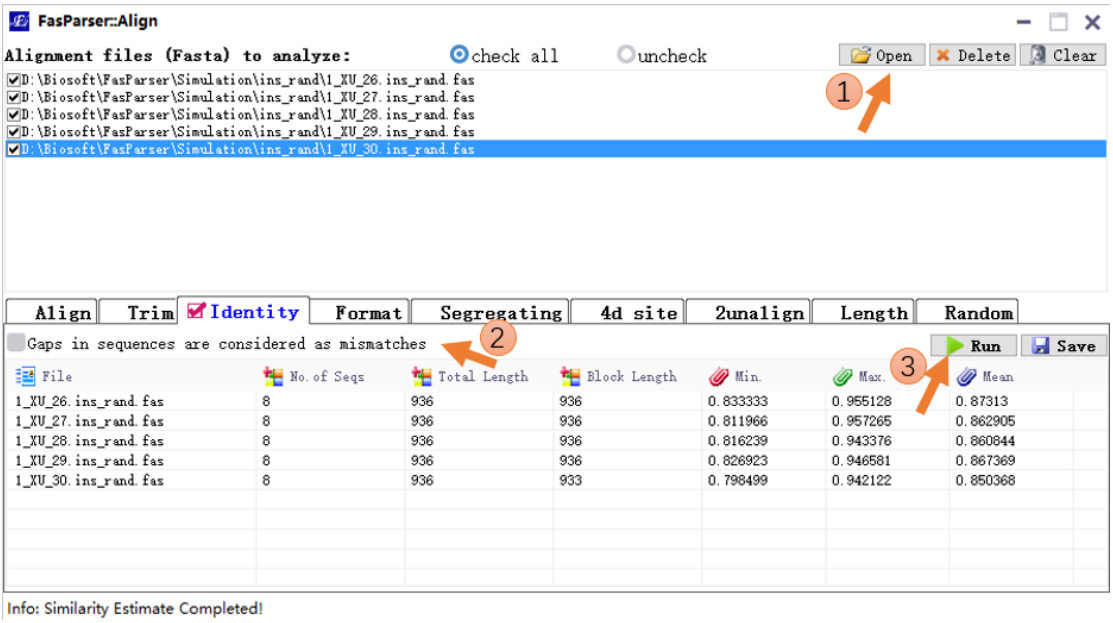


Figure 4. Overview of alignment similarity estimates.

➔Alignment format conversion

This function is used to convert the FASTA files to other formatted ones. Presently, there are 4 different formats available: FASTA, PHYLIP (Phylipi and Phylips), PAML, and NEXUS (Figure 5). The output files will be saved in the folder which contains the input files, with suffix of '*. <format>'. The result files are also displayed in the window and can be viewed by double clicks.

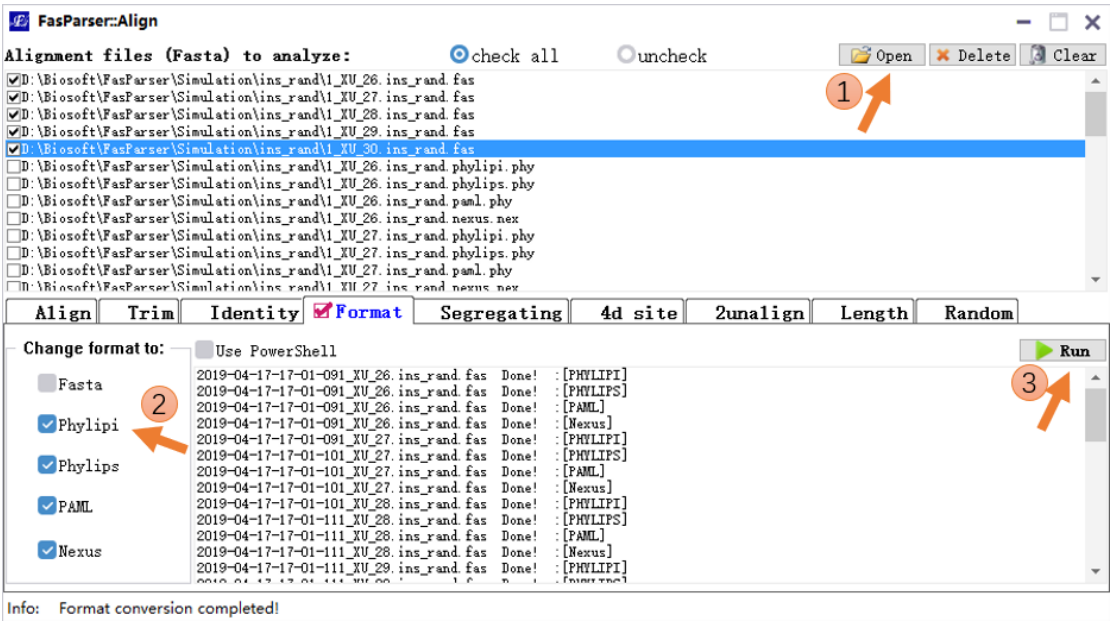
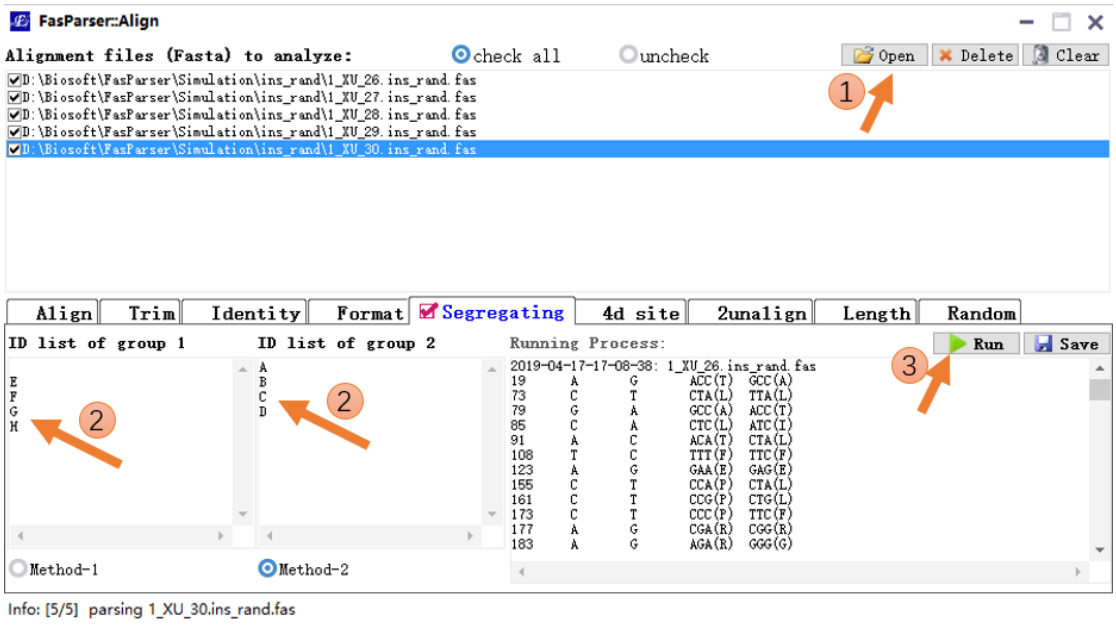


Figure 5. Overview of alignment format conversion.

➔Segregating site extraction

FasParser provides two methods to extract segregating sites whose bases are same in a group of sequences, but different from the other group of sequences. Method-1 allows the bases of one site are same among one group of sequences, but the bases of other sequences can be different. For example, one position in the 1st group of sequences are all "A", while the other sequences can be either G, C, or T. This method needs you to select one group of IDs.

Method-2 is more rigorous, which require the base of the same position are same in each group of sequences. For example, the 1st group is "A", and the others are all "G". This method needs you to set two different groups of IDs, and there is no overlap between the two groups of IDs.



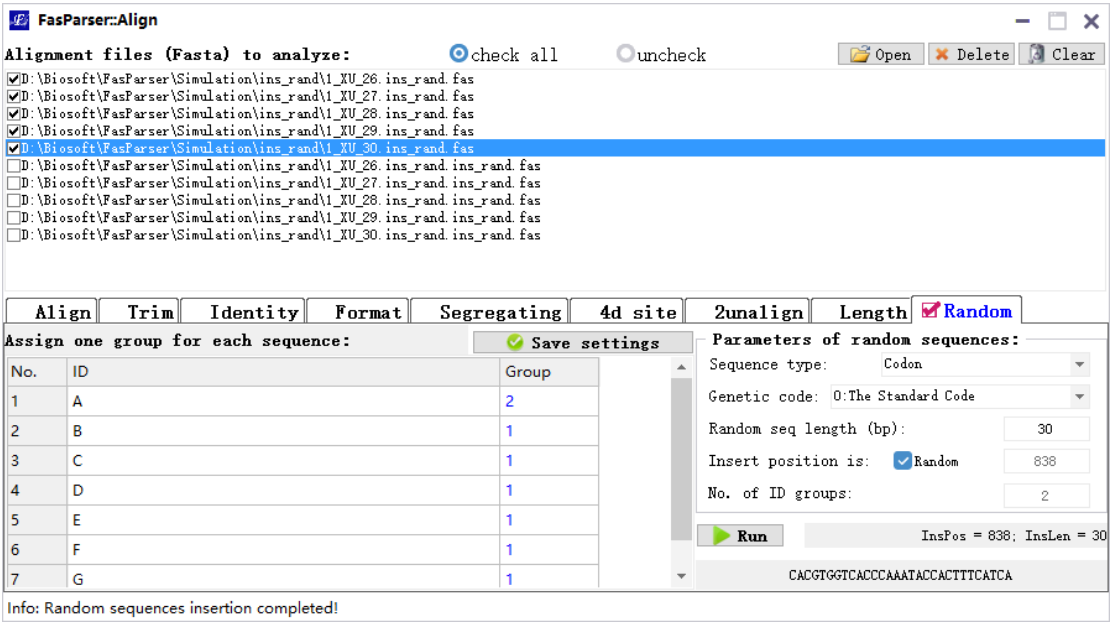
You can also view the changes in the Editor (see below):

➔Insert random segments into alignment

This function is designed for inserting a random segment into 1 or more sequences, and inserting another segment into other sequences at the same position of the alignments. This manipulation can be used to evaluate the alignment efficiency of different aligners in dealing with the non-homologous sequences, as well as the trimming efficiencies of different alignment trimming methods in removing the

non-homologous sequences.

To do this, you should set 2 or more groups of IDs, for each of them one random segment will be inserted into their sequences. The length of inserted random sequence, and the insertion position of alignment can be set by the user.



[Merge]

➔Concatenate sequences with same ID

This program is used to 'concatenate' sequences of the same IDs from multiple FASTA files. It is much useful in phylogenetic analyses, especially when users generated multiple loci sequences for a set of samples and want to generate a "*super*" sequence by concatenating all the loci sequences together for each sample.

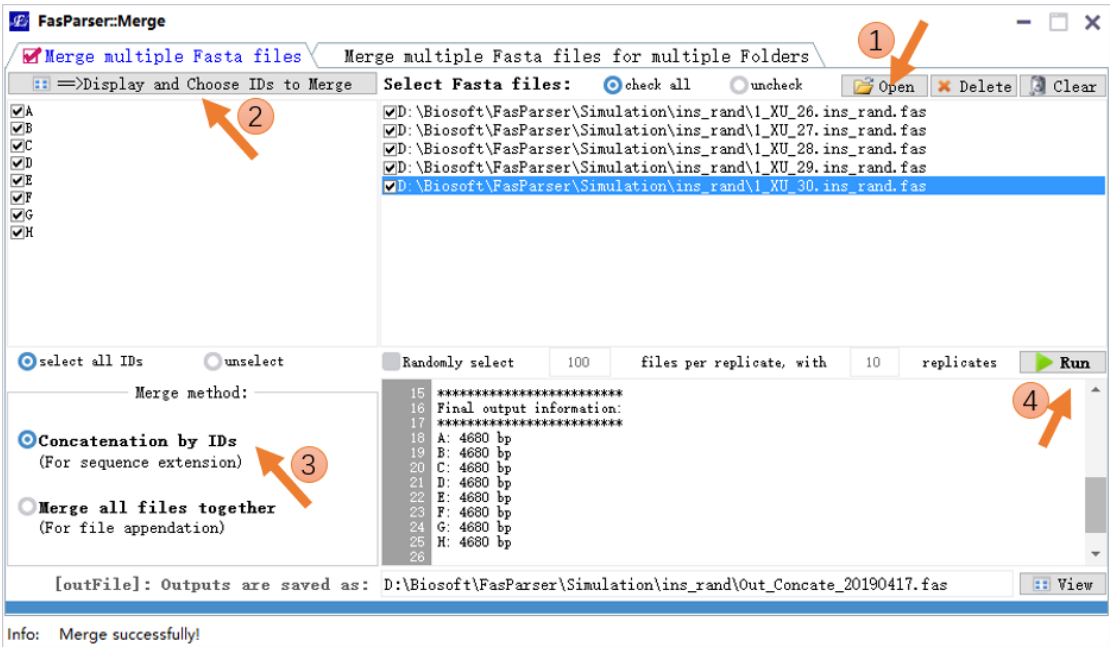
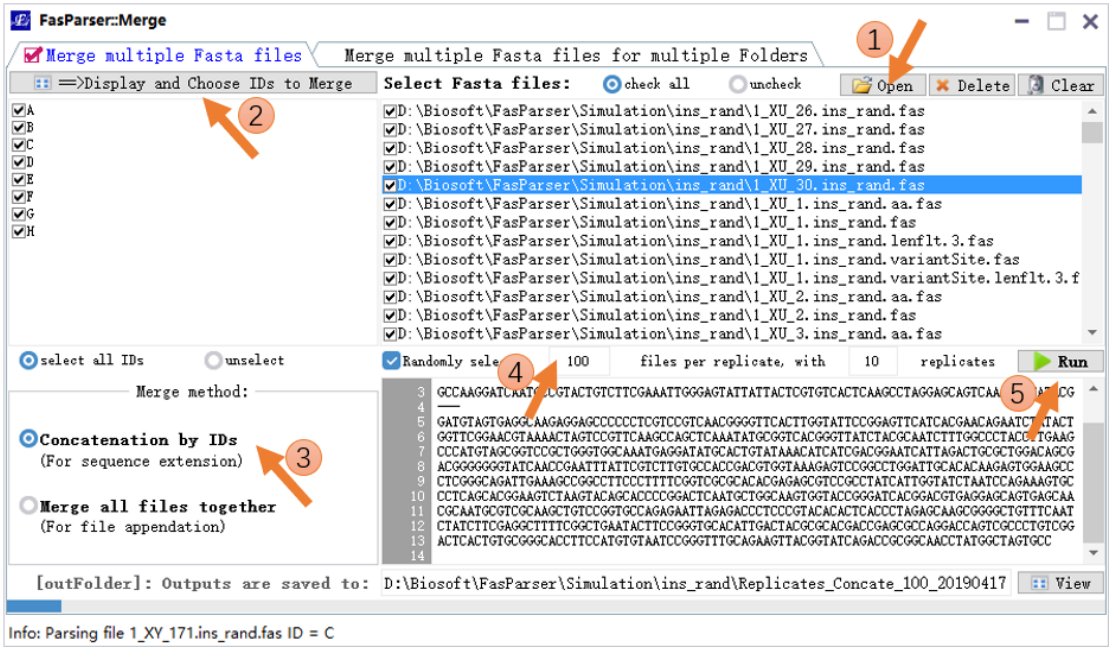


Figure 6.1. Overview of sequence concatenation manipulation.

Please note that for concatenate, if the raw FASTA files are not aligned before, the final concatenated FASTA file should be aligned first before conducting some other analyses. (We recommend user to construct alignment for each file before, and each file must have a same ID list).

To start this analysis, user should define an output file name and its location (can be automated created by FasParser). User can also set which IDs should be concatenated by editing the ID list in the single-selection list box (left-up, see below Figure). Clicking the View button can use the user's default execute program to open the result file (e.g. Text).

➔Randomly select FASTA file to concatenate



➔Combine FASTA files together (append)

In addition to the concatenate, user may just want to 'combine' all the selected files together. It is easy to achieve this by using the Merge function in FasParser (Figure 6.2):

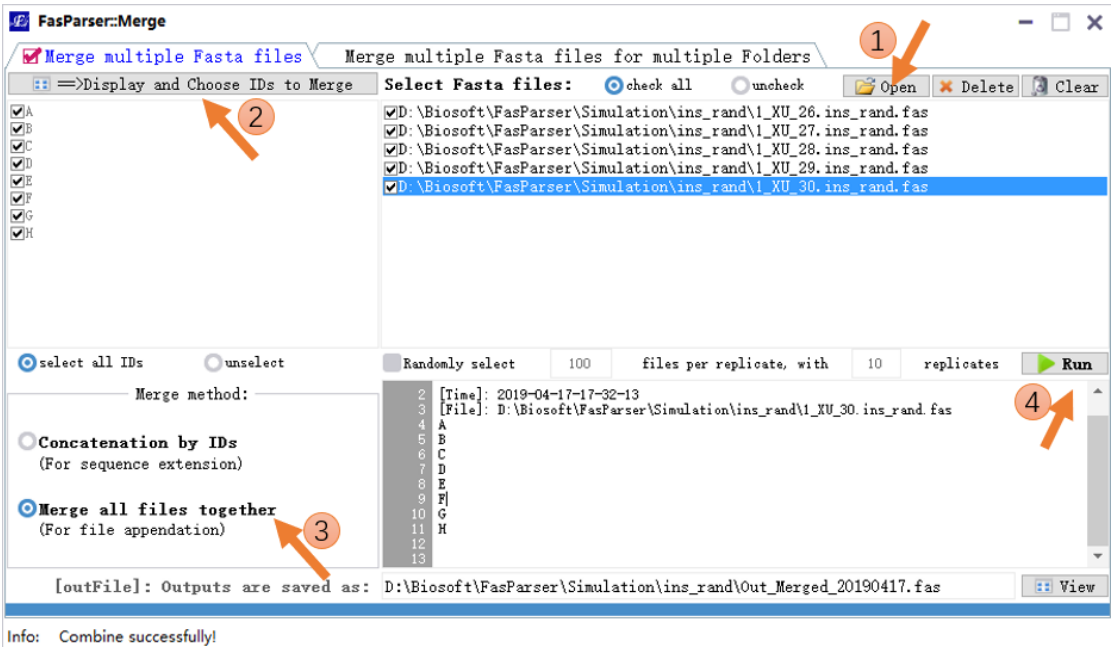


Figure 6.2. Overview of sequence combine manipulation

[Sort]

➔Sort Sequences

This program will allow users to sort their FASTA files according to the **ID names**, **sequence lengths**, or **a provided list of IDs**. Part of this function (with the ID list provided by users) is much like the **Extraction** analysis in [Filter] module.

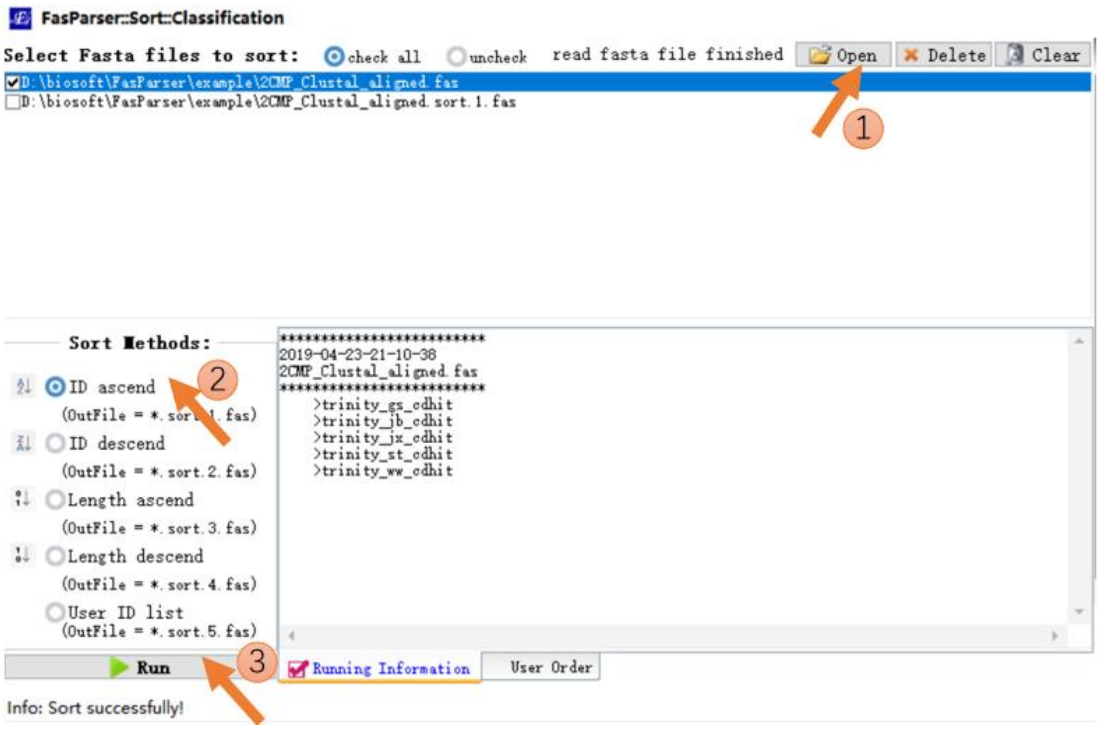
Please note that **the ID can be automatedly recognized from the raw FASTA files, according to** **the first continuous string before symbols space, ".", and "|"**. For example, if the raw ID in a FASTA file is:

'>Uma_R000001.2 locus=scaffold79:384179:406202:-'

You can use the ID 'Uma_R000001.2' to search its sequence. Sometimes, the ID 'Uma_R000001' is also ok, if there is only one targeted ID. If the provided IDs cannot be recognized, there will be no sequence reported. Another example. You can use 'gi|947195581' to represent the sequence ID :

'>gi|947195581|ref|XP_006139827.2| PREDICTED…'

The output files will be saved in the folder which contains the input files, with '*.sort.1.fas' , '*.sort.2.fas', '*.sort.2.fas', '*.sort.3.fas', '*.sort.4.fas', and '*.sort.5.fas' for for ID ascend, ID descend, length-ascend, length-descend, and user-provided ID list, respectively.



➔Rename sequences (similar with ID module)

With the provided ID list (right-below textbox), the program can also rename the raw sequence IDs. To achieve that, the ID list should have two columns, the first column refers to the raw ID, and the second column refers to the new ID, the separator between the two columns must be "=>", as below:

```
Uma_R000001.2=>R000001.2
Uma_R000003.2
Uma_R000002.2=>R000002
…
```

Under the above query IDs, program will use `R000001.2` to replace `Uma_R000001.2`, and use `R000002` to replace `Uma_R000002.2`. If there is only one column, the raw IDs will not be changed (see below Figure). The output files will be saved in the folder which contains the input files, with '*. sort.5.fas'.

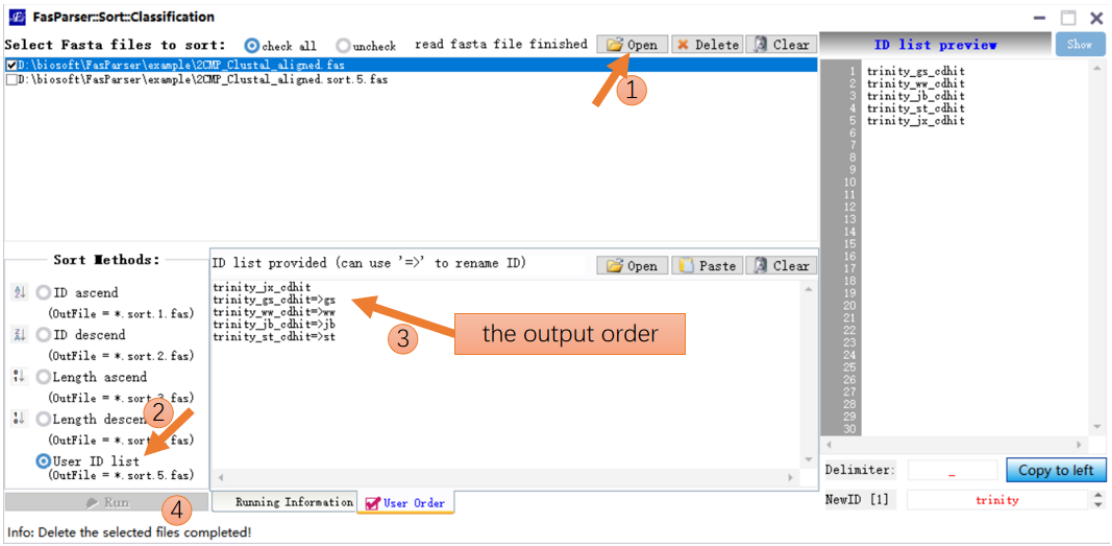


Figure 7. Overview of Sort & Rename sequences.

[Filter]

This program is designed to perform some extraction and filtration analyses within FASTA files. They include several much common sequence manipulations. With this module, users can extract or remove a set of sequences from the raw FASTA file based on query IDs (or keywords), based on the positions per codon, based on the sequence length, or based on similarity with a provided reference sequence.

➔Extraction & Filtration based on IDs

Based on this function, user can extract or remove some sequences from one or more FASTA files. Just select the files you want to analyze, and then input the IDs (you want to query) in the left-below textbox. Select the appropriated operation: *extract* or *remove*, whether or not with reverse complement operation, and then click the "*Run*" button (Figure. 8.1). The output files will be saved in the folder which contains the input files, with '*.extract.fas' or '*.removed.fas'.

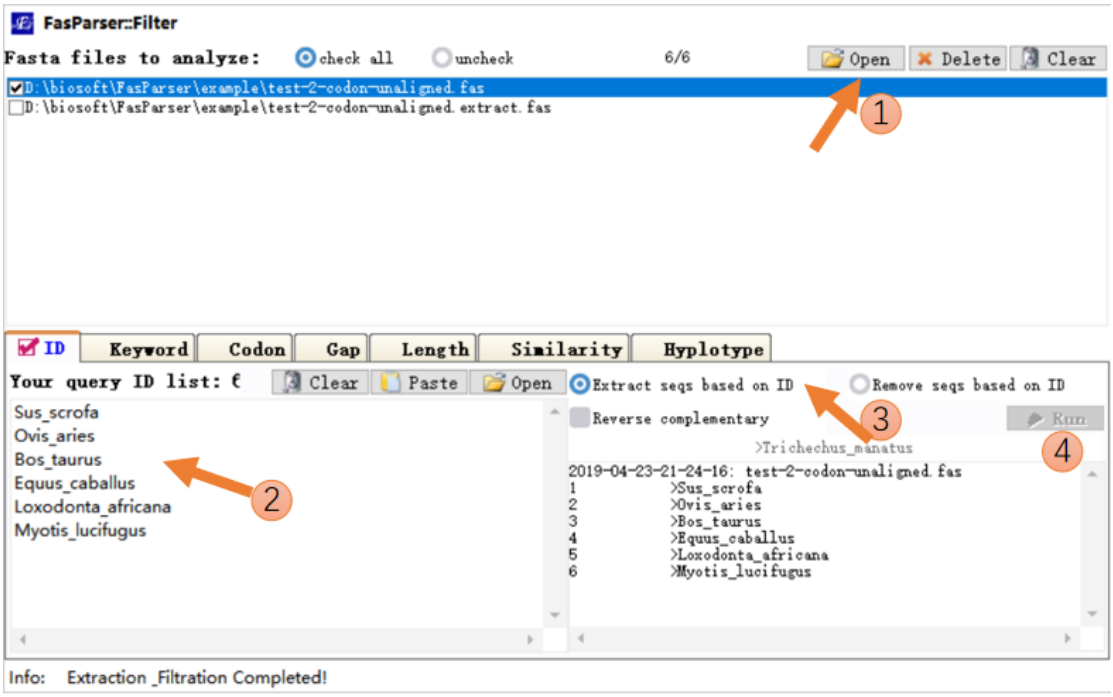


Figure 8.1. Overview of extraction & filtration based on query IDs.

➔Extraction based on keywords

The "Keyword" function in this module also provides a Classification tool, which means that you can use one or more keywords to extract sequences from raw FASTA files and save them into separate FASTA files. The keywords can be the genus name, species name, or some other words, which should be part of the raw IDs (Figure 7.2).

The user should input the keywords in the left-below textbox, and then click the "*Run*" button to start this analysis. The output files will be saved in the folder which contains the raw input files, with suffix of '*.class.[keyword].fas'.

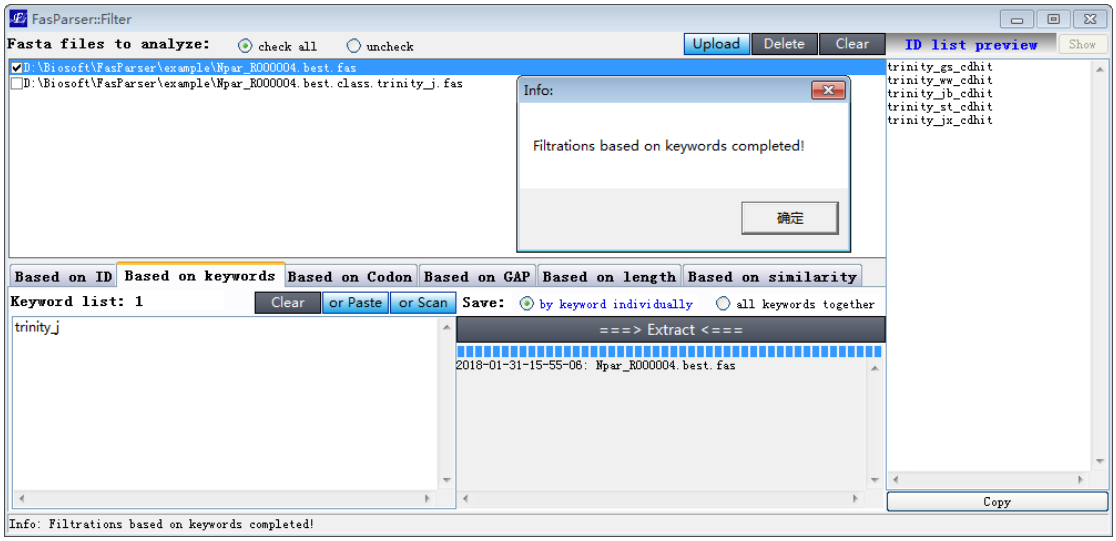


Figure 7.2. Overview of extraction & filtration based on ID keywords.

➔Extract 3-site columns for Codon sequences

Based on this function, user can extract each-site columns of codon sequences for multiple FASTA files. Just select the files you want to analyze, and then click the "*Run*" button (Figure. 8.2). The output files will be saved in the folder which contains the input files, with '*.codon1.fas', '*.codon2.fas', and '*.codon3.fas'.

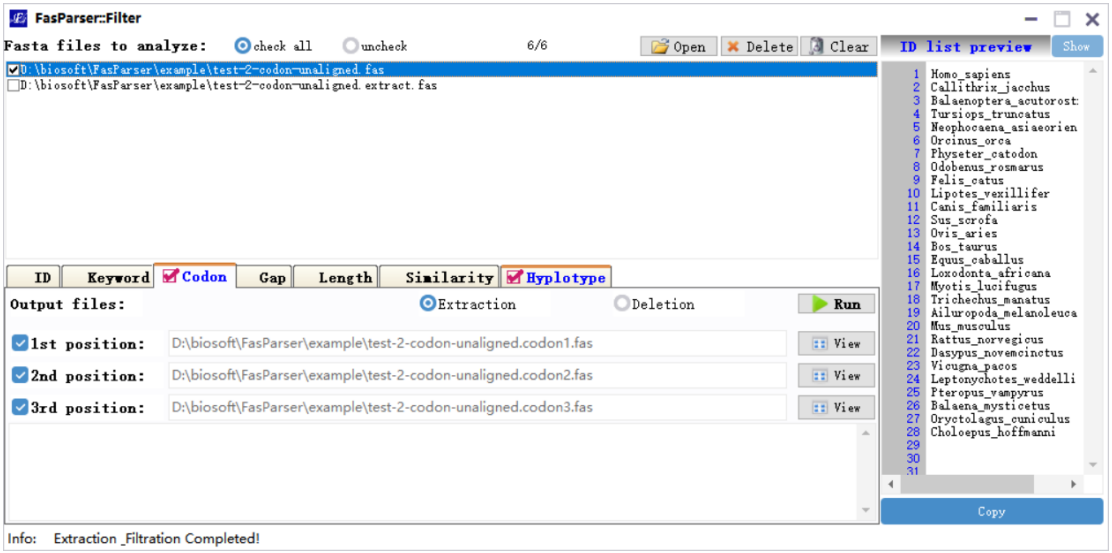


Figure 8.2. Overview of extraction of site columns for codon sequences.

➔Filtration of columns based on Gap frequency

This is a common function to cut the raw alignment by deleting columns with many gaps ('-'). This function can also fill up the missing data (with 'N') at the beginning and ending of an alignment, which is useful for some phylogenetic analyses. To perform this analysis, user should provide a gap-frequency cutoff value, and then select appropriated operation, like analyzing which region, and how to deal with the columns with a lower value (than the cutoff). Click the '*Run*' button to start this analysis (Figure 8.3).

The output files will be save in the folder which contains the raw input files, with a name format of "*.cut.end.fas" or "*.cut.all.fas".

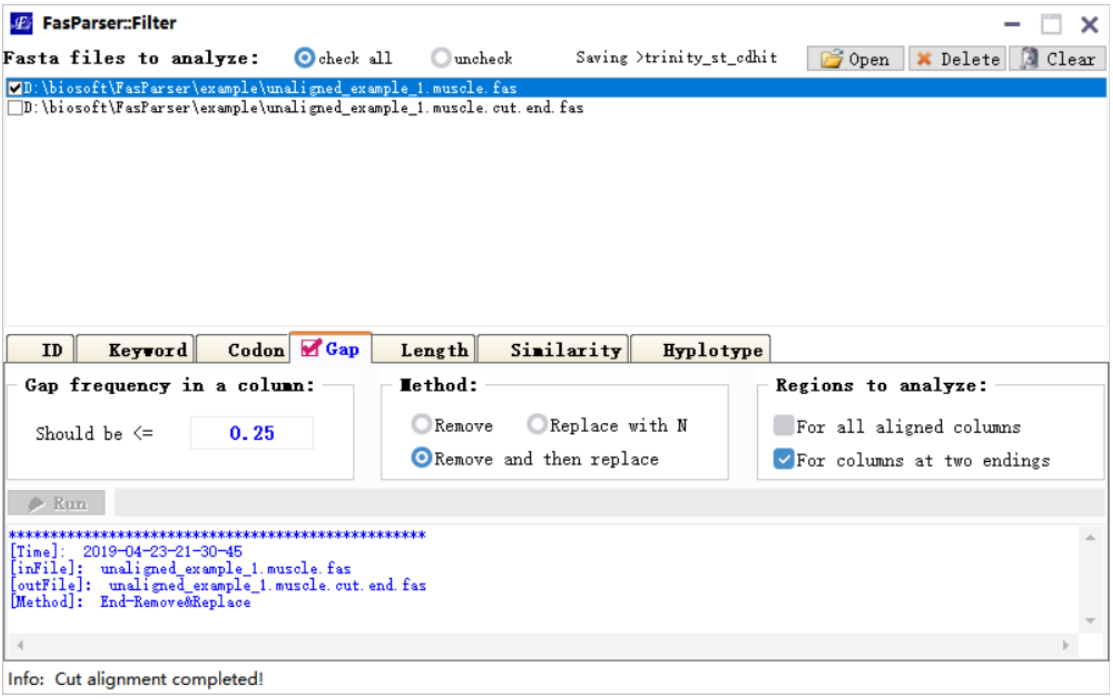


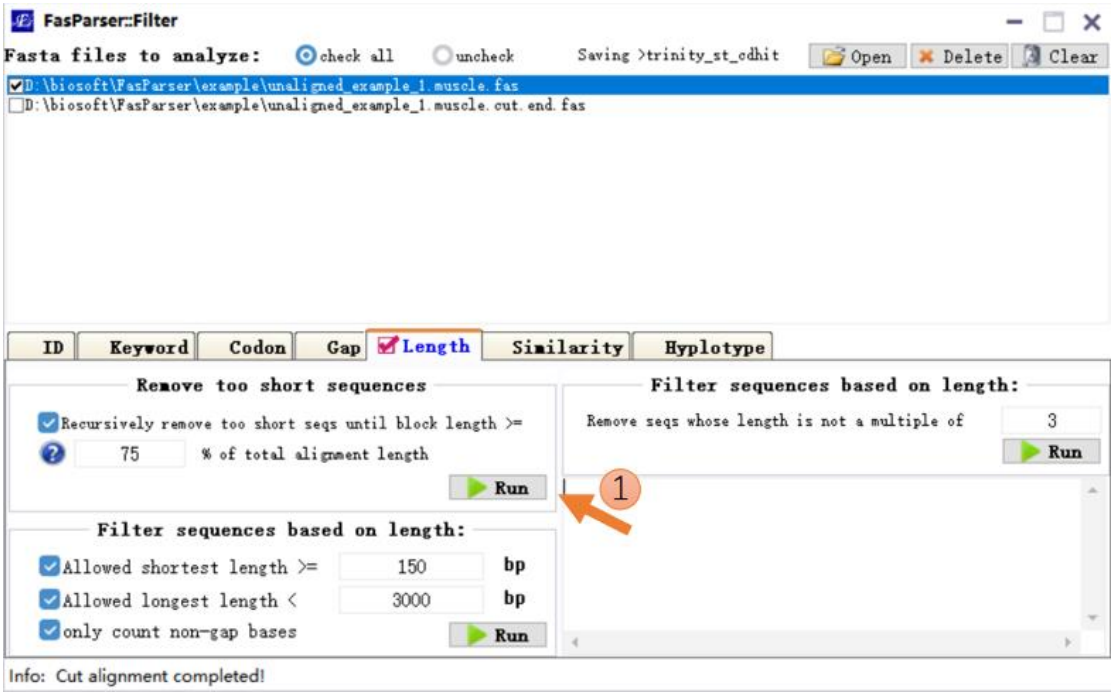
Figure 8.3. Overview of the column-filtration based on Gap frequency.

➔Filtration of sequences based on sequence length

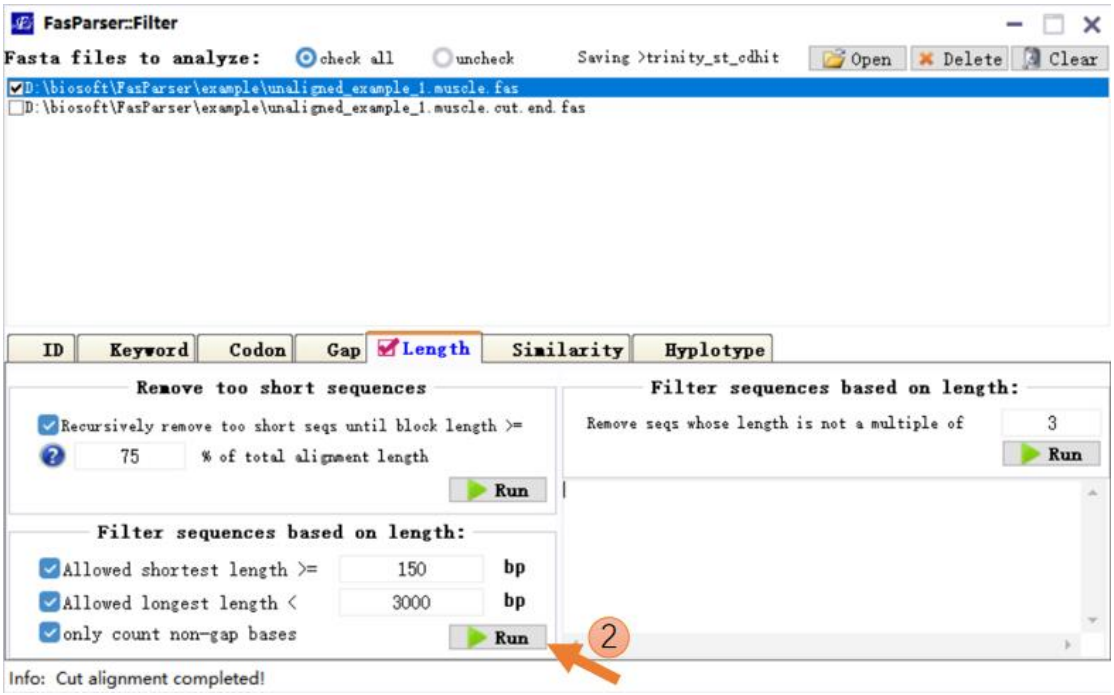
Maybe there are some too short sequences in a file, which would significantly influence the final available sequence length. So, it is usually useful to identify and

remove the too short sequences. To do this, FasParser provides some different methods.

The 1st method is designed for alignment files (slower). The program will auto-estimate the block (no-gap regions) length under removal of each single sequence, and then identify the bad sequences whose removal can increase significantly the final block lengths.



The 2nd method is designed for non-aligned files (faster). This method will identify directly the too short sequences according to their sequence lengths. The formula used is "mean - 3*std" if there are more than 100 sequences. If there are less than 100 sequences, Grubbs method will be used.



User can also use the 3-rd method to predict pseudo-genes according whether the sequence length is multiple of 3.

➔Filtration based on similarity with a reference sequence

This function is designed for manipulating multiple sequences, which are always collected from different sources without few sequence checking tasks. There may be some non-homologous sequences in the raw inputted sequences, or the sequences come from different parts of a same gene. Before downstream analyses, it is much better to do some filtration task on these sequences to improve their availability in other analyses.

To do this work, user should provide a reference sequence that is from your previous or other peoples' work to identify the non-homologous sequences and cut the input sequences to a same length based on the alignment result with the reference sequence.

After setting appropriated parameters, just click the "Run" button to run this analysis. There would generate 2 or 4 output files for each input file, which contain the final alignment (cut to a length based on the reference sequence), as well as the

homologous identification results (Figure. 8.4).

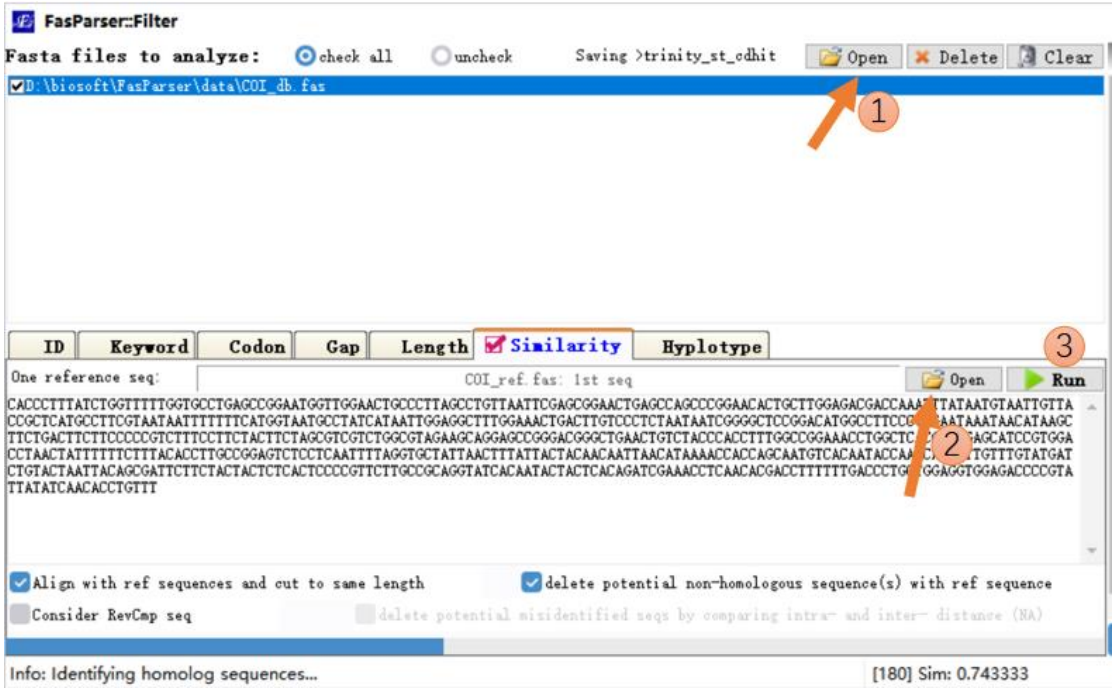


Figure. 8.4. Overview of the sequence filtration based on similarity with a reference sequence.

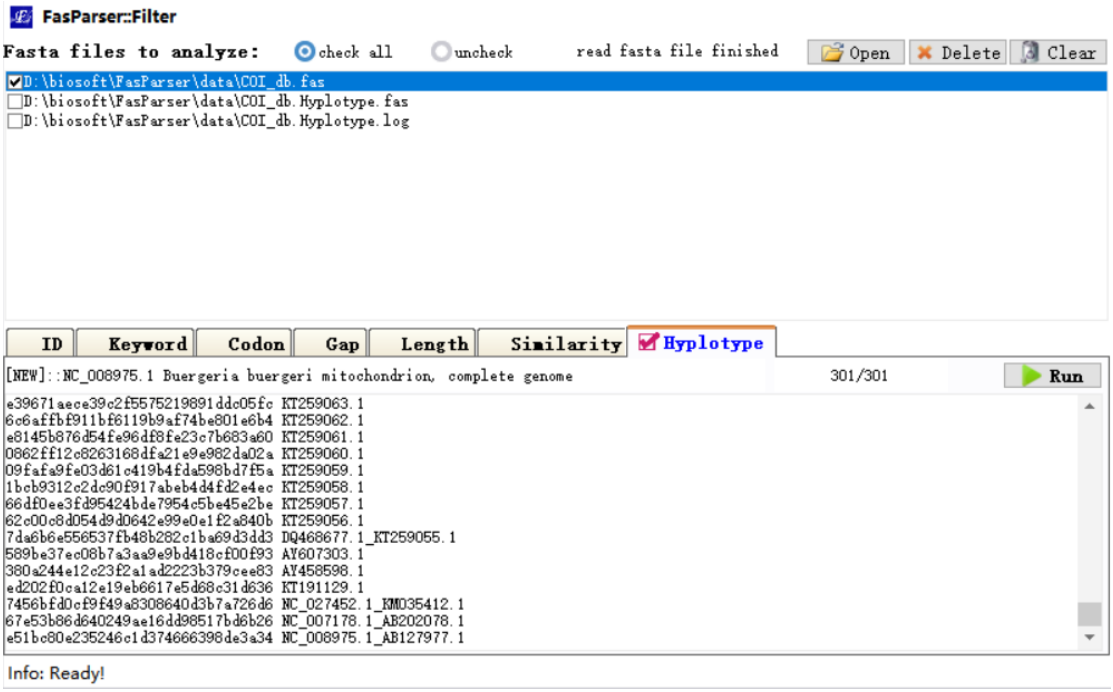
The running log file will tell you which sequence would be non-homologous sequence with the reference one (see below Figure).

```
[Identification of Non-homologs]:
FJ230952.1 Buergeria sp. BTBP1 cytochrome oxidase subunit I (COI) gene, partial cd0.493976   [Bad!]
HQ141556.1 Polypedates leucomystax isolate PolyLeuco17474_TH cytochrome b (cytb) g0.452607   [Bad!]
HQ141555.1 Polypedates leucomystax isolate PolyLeuco1147_TH tRNA-Glu gene, partial0.457031   [Bad!]
HQ141552.1 Polypedates leucomystax isolate PolyLeuco17475_TH tRNA-Glu gene, partia0.466934   [Bad!]
HQ141548.1 Polypedates leucomystax isolate PolyLeuco17473_TH tRNA-Glu gene, partia0.45776    [Bad!]
HQ141547.1 Polypedates leucomystax isolate PolyLeuco17496_TH tRNA-Glu and cytochro0.439122   [Bad!]
AY607303.1 Polypedates leucomystax voucher MVZ 222058 NADH dehydrogenase subunit 10.510802   [Bad!]
DQ468684.1 Rhacophorus moltrechti voucher A538 cytochrome oxidase subunit I (COI) 0.770517   [Good]
DQ468683.1 Chirixalus palpebralis voucher A342 cytochrome oxidase subunit I (COI) 0.767477   [Good]
DQ468682.1 Kurixalus idiootocus voucher A127 cytochrome oxidase subunit I (COI) ge0.767477   [Good]
DQ468681.1 Kurixalus eiffingeri voucher A120 cytochrome oxidase subunit I-like (CO0.753799   [Good]
DQ468680.1 Kurixalus eiffingeri voucher 11320 (CE19) cytochrome oxidase subunit I 0.75228    [Good]
DQ468679.1 Kurixalus wangi voucher 11328 (CE06) cytochrome oxidase subunit I (COI)0.762918   [Good]
DQ468678.1 Kurixalus eiffingeri voucher 11333 (CE03) cytochrome oxidase subunit I 0.744681   [Good]
DQ468677.1 Kurixalus berylliniris voucher 11311 (CE01X) cytochrome oxidase subunit0.758359   [Good]
KP996848.1 Liuixalus hainanus haplotype W2 cytochrome oxidase subunit I (COI) gene0.759358   [Good]
KP996847.1 Liuixalus hainanus haplotype W1 cytochrome oxidase subunit I (COI) gene0.759358   [Good]
KP996846.1 Polypedates impresus haplotype LC0805099 cytochrome oxidase subunit I (0.761141   [Good]
KP996845.1 Polypedates braueri haplotype LC0805096 cytochrome oxidase subunit I (C0.743316   [Good]
KP996844.1 Polypedates braueri haplotype LC0805095 cytochrome oxidase subunit I (C0.743316   [Good]
```

➔Predicting hyplotypes

There may be more than one sequences are the same within one FASTA file, to determine which sequences are same (the same hyplotype), the "**Hyplotype**" function in the "**Filter**"

module is assigned to address this issue.



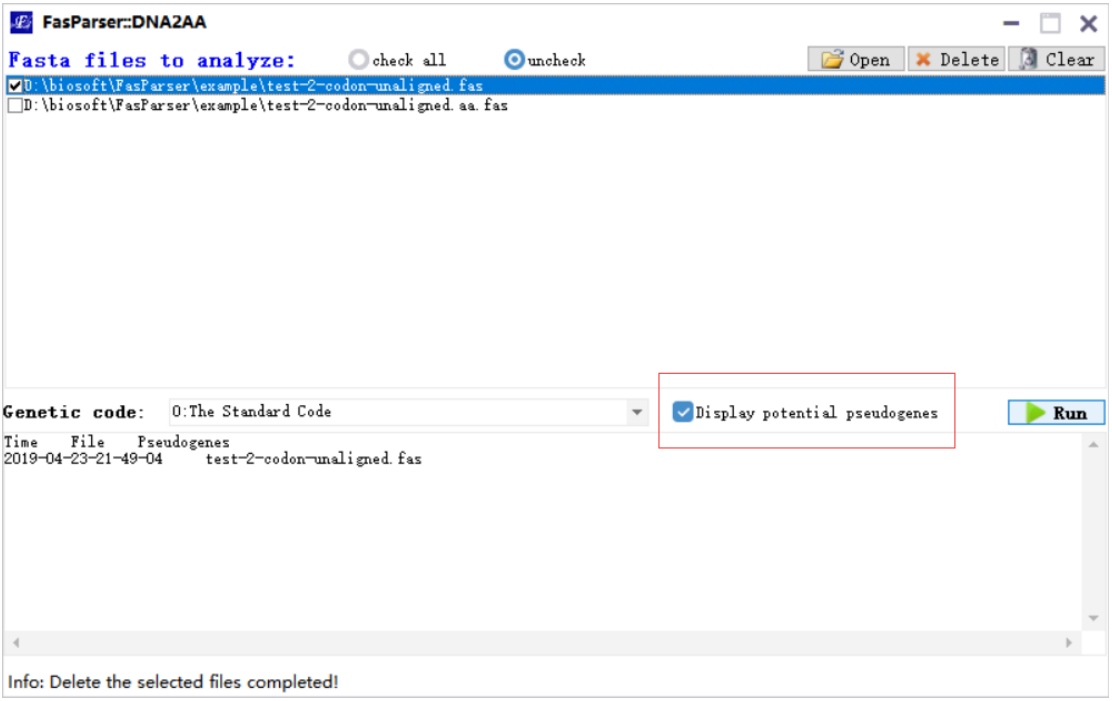
And the output FASTA file will be save like below, as the IDs combined which have the same sequences :

```
>KP996837.1↓
AGCTCTTAGCCTTTTAATTCGGGCTGAATTGGCCCAACCTGGCACCCTATTAGGGGATGACCAGATTTATA
>KP996836.1↓
AGCTCTTAGCCTCTTAATTCGGGCTGAATTGGCCCAACCTGGCACCCTATTAGGGGATGACCAAATTTATA
>KP996834.1_KP996833.1_KP996832.1_KP996831.1↓
AGCTCTTAGCCTCTTAATTCGGGCTGAATTGGCCCAACCTGGCACCCTATTAGGAGATGACCAAATTTATA
>KP996830.1↓
AGCTCTTAGCCTCTTAATTCGGGCTGAATTGGCCCAACCTGGCACCCTATTAGGGGATGACCAGATTTATA
>KP996848.1_KP996847.1_KP996829.1↓
AGCTCTTAGCCTTTTAATTCGGGCTGAATTGGCCCAACCTGGCACCCTATTAGGGGATGACCAGATTTATA
>KP996828.1↓
TGCCCTTAGCCTGCTTATCCGCGCTGAACTATCCCAACCCGGCACCCTGCTCGGTGATGATCAAATCTATA
>KP996826.1↓
ATCTCTTAGCTTACTTATCCGAGCTGAACTTGCCCAACCCGGTACTCTTCTCGGCGATGACCAAATTTATA
```

[DNA2AA]

This module is used to translate the DNA sequences to protein sequences, or back-translate the protein sequences to Codon sequences, or predict the open-reading frames of multiple sequences.

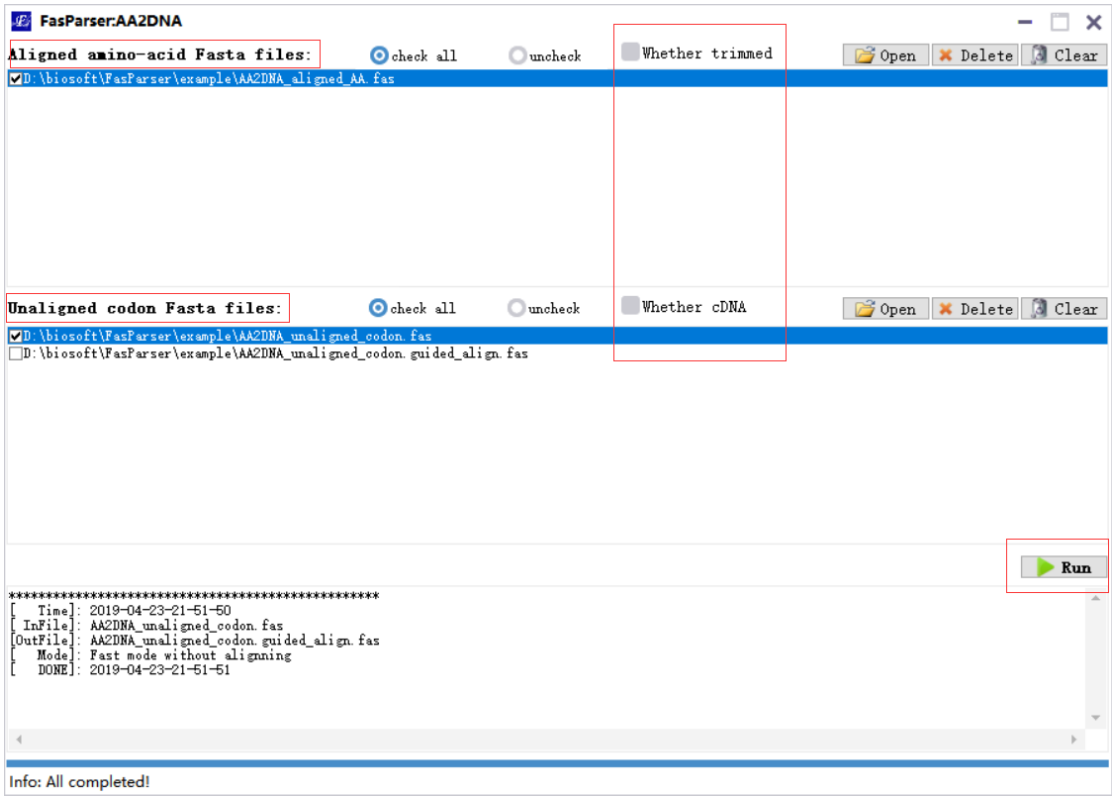
➔DNA2AA: translate the codons to proteins



User can use this function to predict the potential pseudogenes which have stop codons within the sequences.

➔AA2DNA: back-translate to codons from aligned proteins

This function is used to generate codon alignments guided by the amino-acid alignments. The raw DNA sequence files (unaligned) and the amino-acid alignment files (aligned) are needed for this function. It is always used in transcriptome analyses. For example, you have 1000 genes with 5 species. Each gene has been translated to amino-acid sequences and aligned by Muscle or others. You can use this function to generate codon alignments based on the aligned amino-acid sequences.



➔Predict ORF (Open-Reading Frame)

This program is designed to identify the ORF for cDNA sequence(s). The cDNA sequences can be organized into a FASTA file or as single sequence. There are 2 choices to obtain the ORFs: a) get the longest one if there are present more than one potential ORF in the input cDNA sequence, and b) get all the potential ORF sequences (Figure 9).

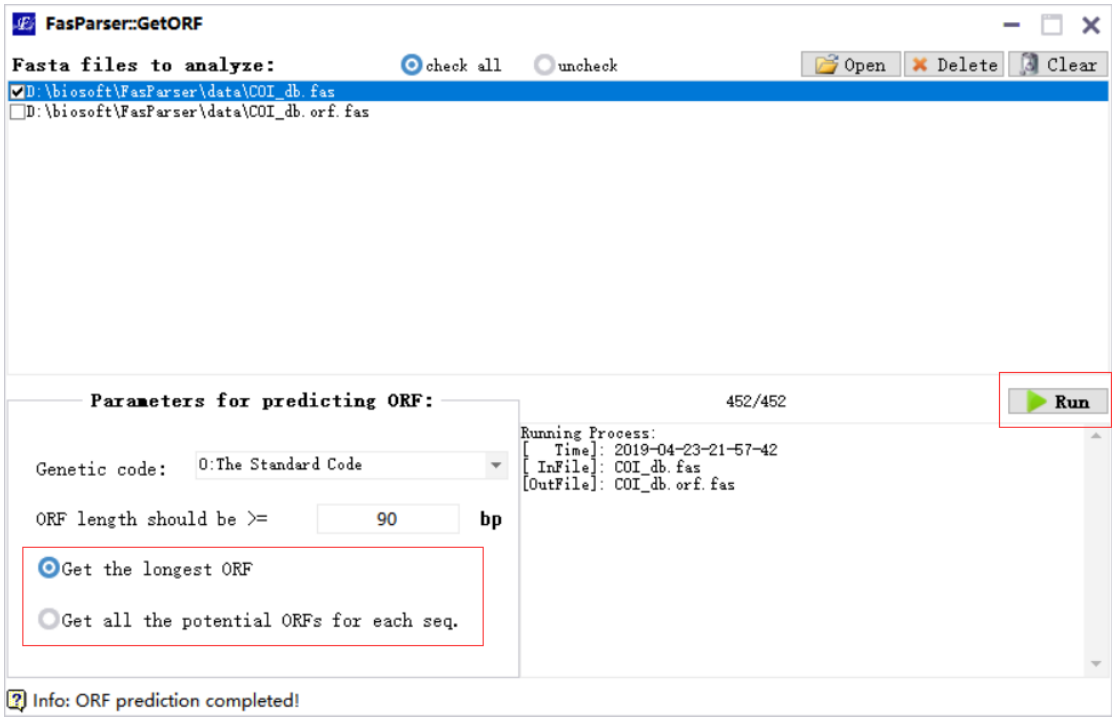


Figure 9. overview of identification of ORFs for sequences.

[Others]

This module contains some other tools to perform sequence analysis, like compare two DNA sequences and identify their mutations (or to calculate their dN/dS), compare two alignments of the same gene, and thus to obtain some consensus aligned regions, primer designing, and positive selection detection with the PAML package (CodeML program).

➔Compare two DNA sequences

The program "CMP2Seq" is designed to easily count and view differences between two DNA sequences at both DNA and codon levels. Under the codon level, the program can also estimate the total number of synonymous (S) and non-synonymous (N) sites for the first sequence and then calculate the number of

synonymous and non-synonymous substitutions between the two sequences. To do that, users just need to put two sequences into the two above textboxes and click the Run button. The program could then provide a view the differences between the two sequences in colors and the summary of identified mutations or substitutions (Figure 10).

Figure 10. Overview of comparison between two sequences.

➔Compare two alignments of the same gene

This program "CMP2Align" is designed to compare different alignments of a same gene that could be generated by different aligners. It is well known that there is almost no current method correctly aligns the entire sequence, and different aligners always correctly align different regions. One simple method is to identify the overlapped regions between different aligner-generated alignments, which might be useful for some other analyses, like phylogenetic reference and/or

positive selection detection. This function needs users to **input two alignments through the above scan buttons** and **click the "*Run*"** button to view their overlaps (Figure 11). *This is the only function currently cannot run at the batch mode.

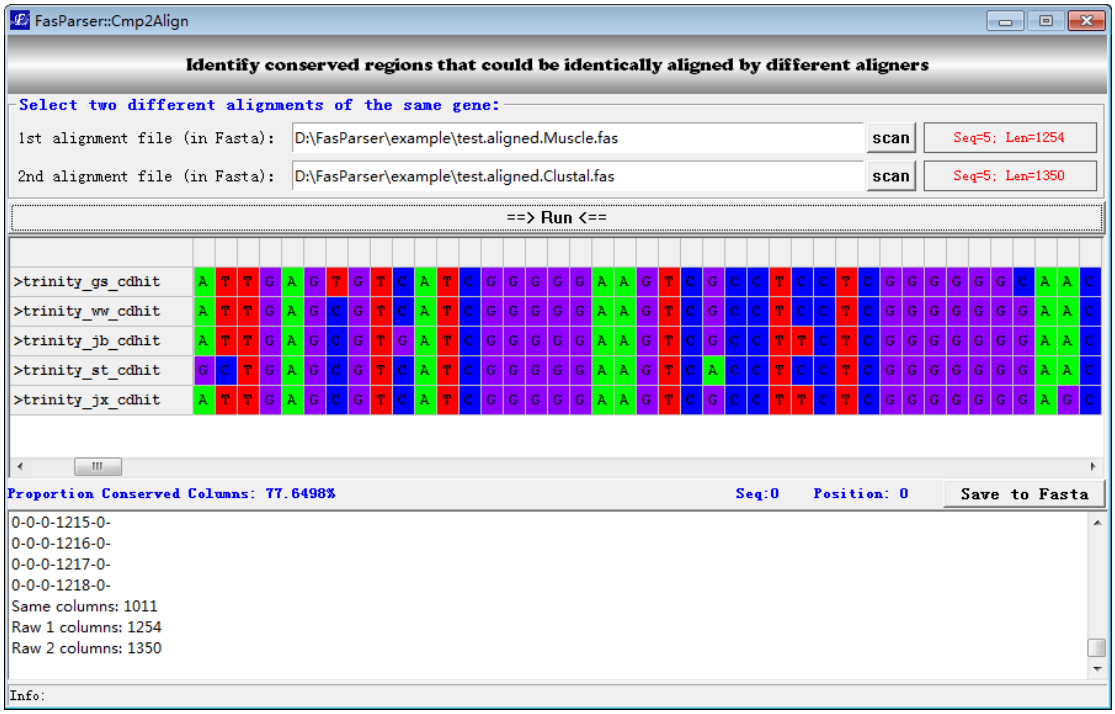


Figure 11. Overview of the estimates of consensus alignment.

➔PCR primer designing

This is an interface to design PCR primers with an excellent Primer designing tool (Primer3). To do this, jst input the DNA template (either in Fasta or pure DNA sequence) and modify the parameters. Click the "Run" button will obtain the primer results if all run properly (Figure 12).

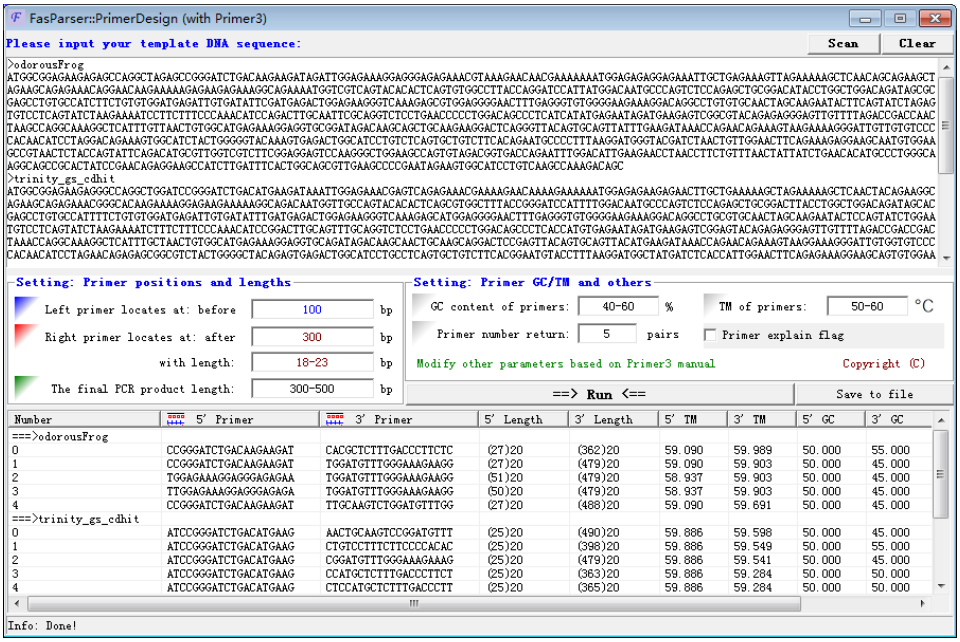


Figure 12. Overview of the primer designing function.

➔Positive selection detection

FasParser provided a batch mode to run codeml in PAML package (Yang et al. 2007). It is much better for user to read the manual of PAML to understand the parameter settings and add proper symbols to the phylogenetic tree (like below, Figure 13).

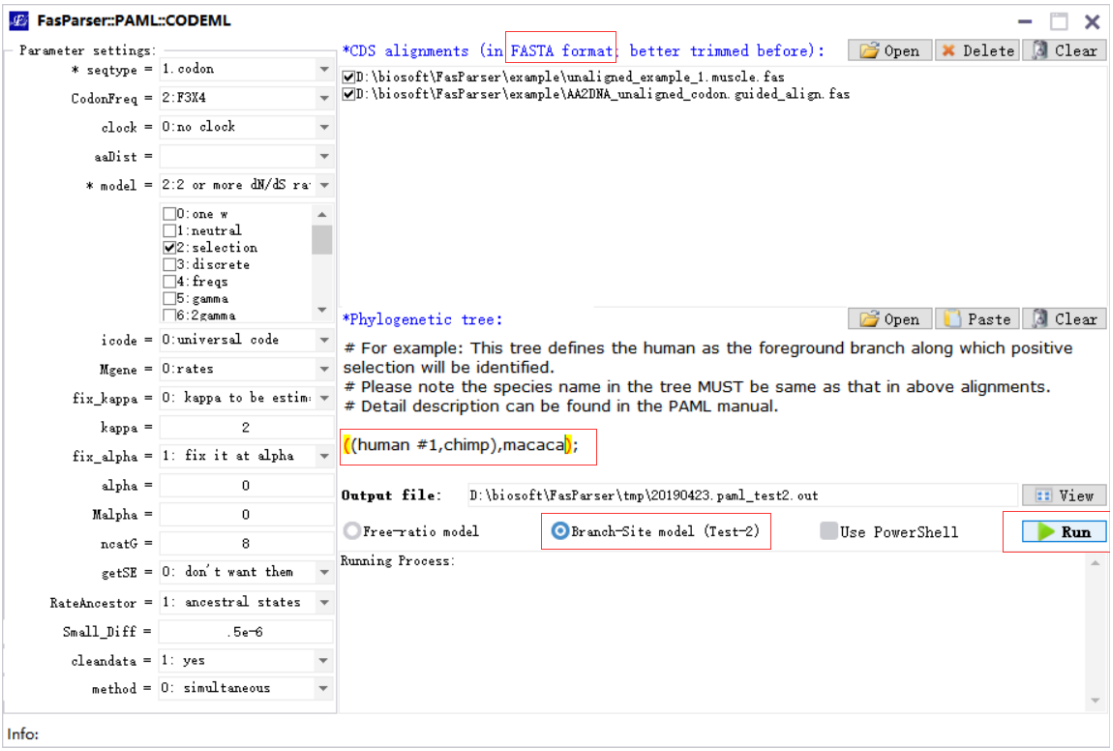


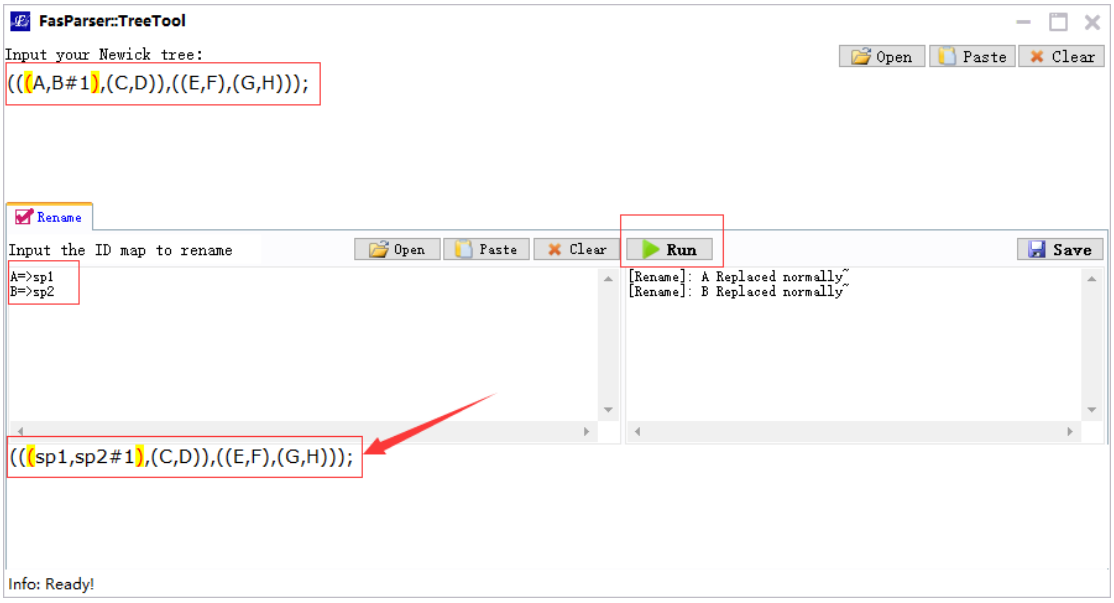
Figure 13. Positive selection detection.

Below is an example of running output:

GeneID	Ln Alternative	Ln Null	Deta	Sig.
ENSACAP00000000005.3.best.fas	-4271.375567	-4271.375567	0	
ENSACAP00000000006.3.best.fas	-6538.150389	-6538.150389	0	
ENSACAP00000000008.3.best.fas	-1092.144468	-1092.144468	0	
ENSACAP00000000009.3.best.fas	-2101.001357	-2094.010118	13.9829	P < 0.05
ENSACAP00000000010.3.best.fas	-1165.844923	-1165.844923	0	

➔Rename species names for Newick Trees

After phylogentic tree construction, user always needs to rename multiple tips of the tree. This function is designed to address this issue, to save more manual operations. To start this manipulation, you just need provide a rename map file, which contains the raw and new names, and they are separated by symbol "=>" (see below Figure).

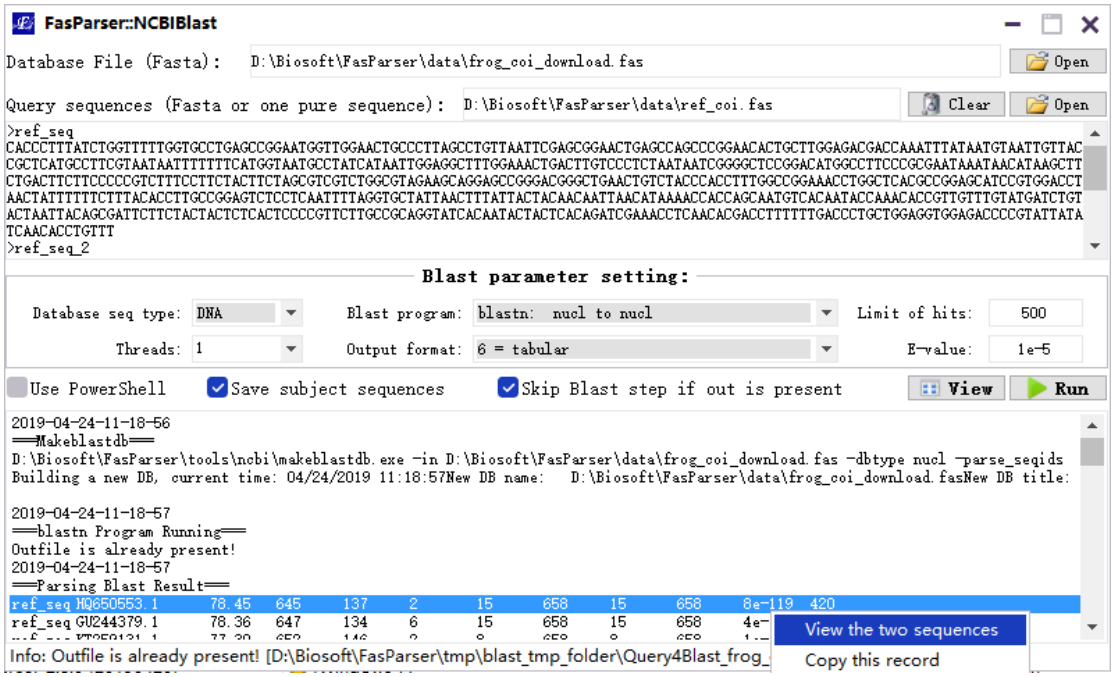


[NCBI]

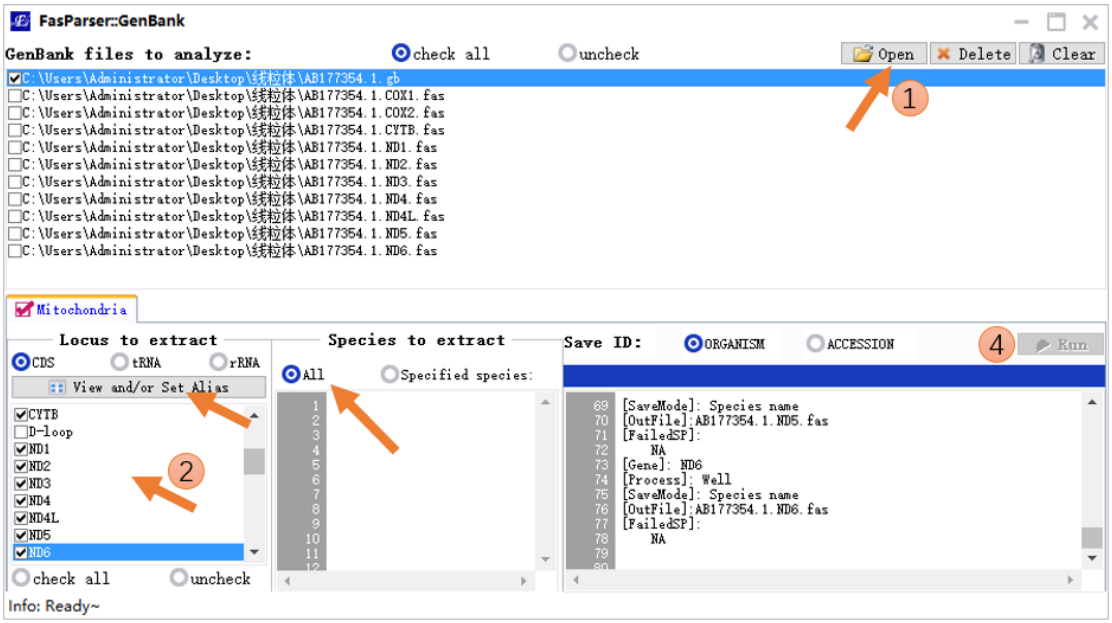
➔NCBI local blast parser

This function provides a simple interface to perform NCBI local blast, as well as a blast output parser function. User just need to provide the database name (in FASTA format) and the query file name (in FASTA format), and then select the appropriated Blast program to run.

For the parser, it is now only suitable for Tabular Text Output (with output = 6). The parser will help you to automatedly view the query sequence and its hit sequence (from the database) by right click “View the two sequences”.



➔Mitochondiral Genome GenBank Parser:



User can set the genes and/or species to extract, and can also set the output ID to be species name or accession number. The program will display the running process, where are present the information of species whose gene sequences are not found.

[GeneDrawer]

FasParser (since version 2) provides a gene structure drawing fiction. It is very easy to use. It can be run in two different mode, based on the data type of inputs. It now can read either gene annotation or gene sequences (Figure 14). Presently, the drawer function has litter photo-editing interfaces, which will be modified in the future versions.

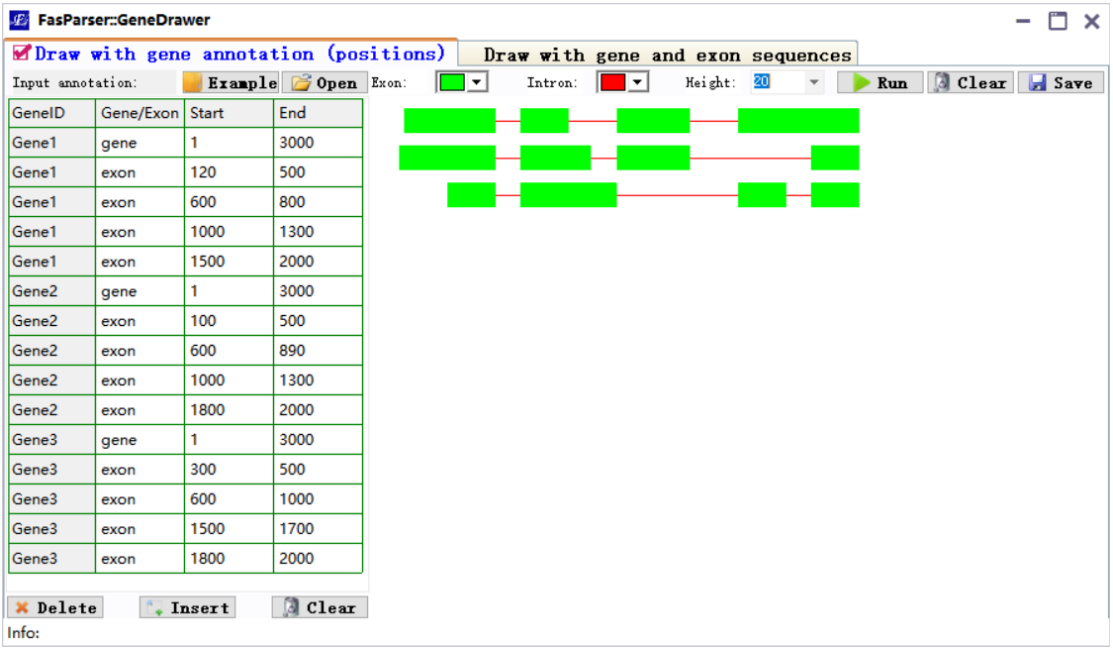
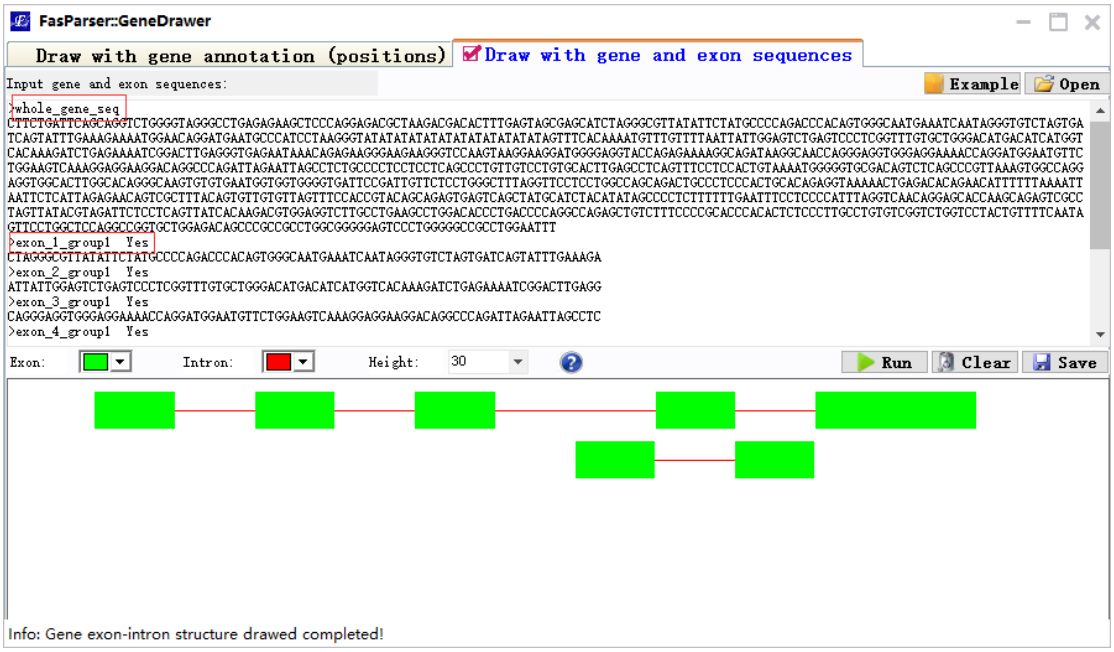


Figure 14. Overview of gene structure drawing function.

Sometimes, users want to draw the gene structure based on whole gene sequence (with presence of both exons and introns) and the exon sequences. During this analysis, sequence matching will be performed when the exon sequence and the gene sequence are from the same species (there is no variation). Otherwise, sequence alignment will be performed first, and then determine the positions of each exon, when the exons are from different species.

User should tell the program which sequence is the whole gene sequence, and which exon are from different species (Yes or No; see below Figure).



Contact

If you have any question, or program bug, please feel free to contact the me (Yan-Bo Sun). My email address is sunyanbo@mail.kiz.ac.cn

FasParser also has a bug report interface ([Bug]), through which you can also contact me and report the errors of this program. I believe FasParser will grow better under your suggestions.