FasParser Manual

[For version 2.1.1]

Sun Yan-Bo (edit at 2017-12-25)

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 achieve the objective of simplifying many common tasks in sequence manipulation for biologists, we 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 use 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*** *unpublished.*

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 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).

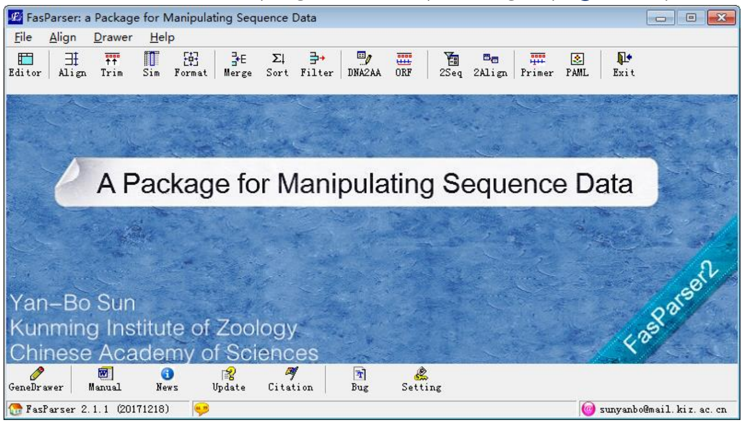
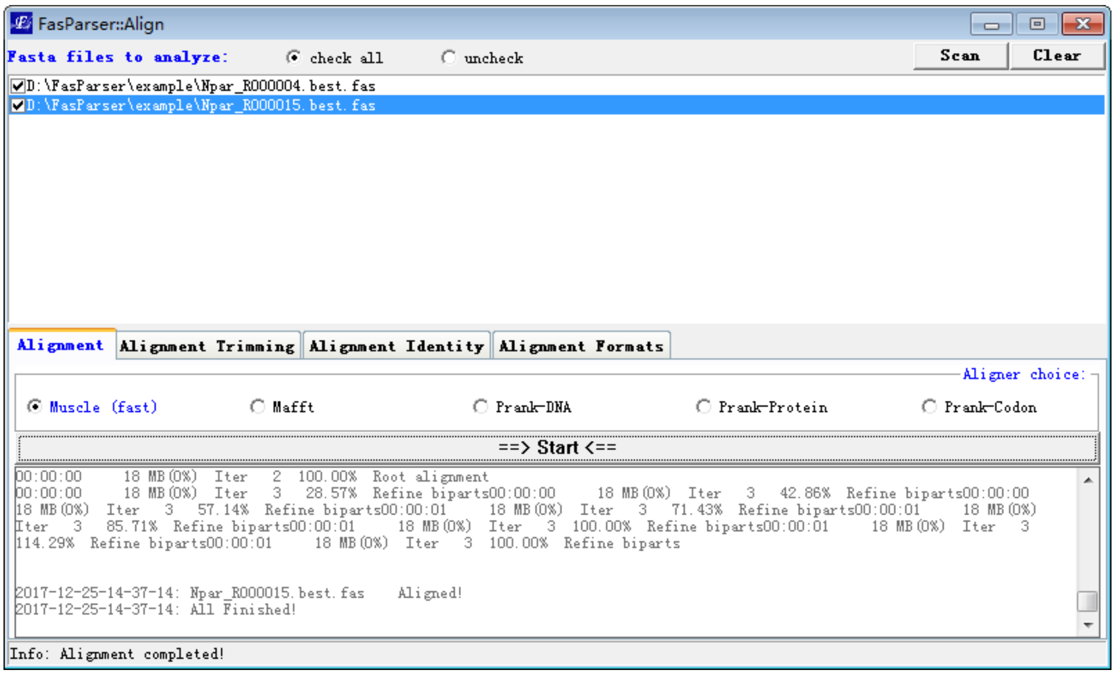


Figure 1. The Home page of the FasParser package.

Package Usage

[Align]

This program is designed to **construct alignments for multiple FASTA files**. There are 3 aligners (MUSCLE, MAFFT, and PRANK, see below reference) have been integrated into 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 '*Scan'* 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'



```
D:\FasParser\example\Npar_R000004.best.fas
D:\FasParser\example\Npar_R000015.best.fas
```

```
00:00:00      18 MB(0%)  Iter   2  100.00%  Root alignment
00:00:00      18 MB(0%)  Iter   3   28.57%  Refine biparts00:00:00      18 MB(0%)  Iter   3   42.86%  Refine biparts00:00:00
18 MB(0%)  Iter   3   57.14%  Refine biparts00:00:01      18 MB(0%)  Iter   3   71.43%  Refine biparts00:00:01      18 MB(0%)
Iter   3   85.71%  Refine biparts00:00:01      18 MB(0%)  Iter   3  100.00%  Refine biparts00:00:01      18 MB(0%)  Iter   3
114.29%  Refine biparts00:00:01      18 MB(0%)  Iter   3  100.00%  Refine biparts


2017-12-25-14-37-14: Npar_R000015.best.fas    Aligned!
2017-12-25-14-37-14: All Finished!
```

```
Info: Alignment completed!
```

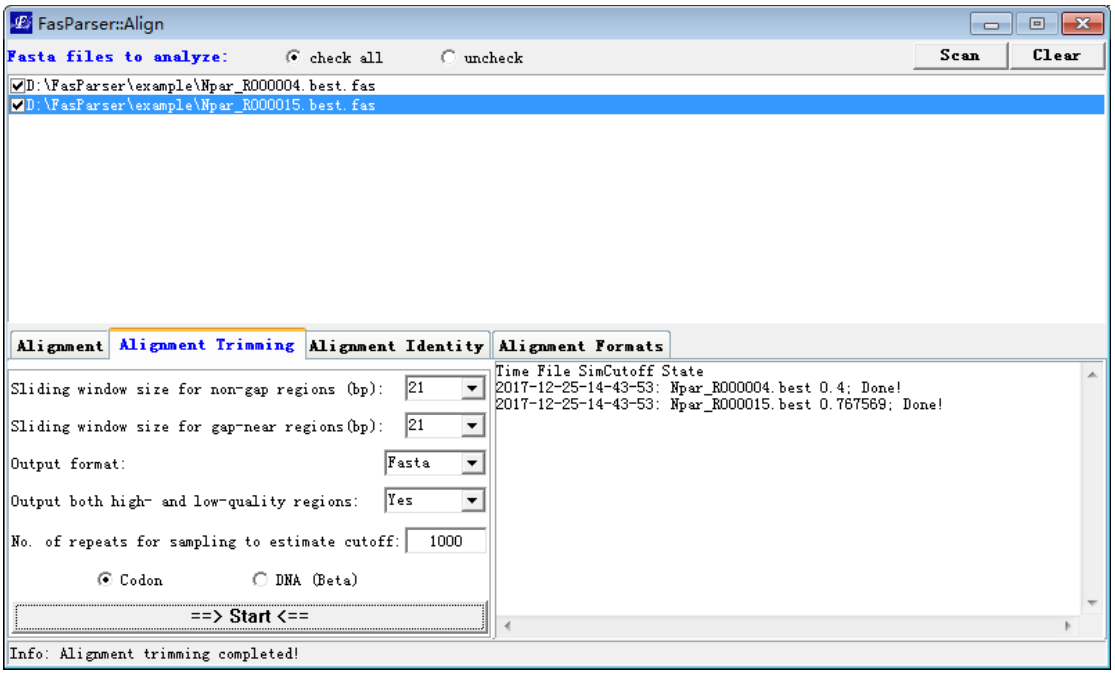
Figure 2. Overview of alignment construction

[Trim]

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 with PAML]

'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 "Start" 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, respectively.



```
☑D:\FasParser\example\Npar_R000004.best.fas
☑D:\FasParser\example\Npar_R000015.best.fas
```

```
Sliding window size for non-gap regions (bp): 21
Sliding window size for gap-near regions(bp): 21
Output format: Fasta
Output both high- and low-quality regions: Yes
No. of repeats for sampling to estimate cutoff: 1000
[✓ Codon]   DNA (Beta)
```

```
Time File SimCutoff State
2017-12-25-14-43-53: Npar_R000004.best 0.4; Done!
2017-12-25-14-43-53: Npar_R000015.best 0.767569; Done!
```

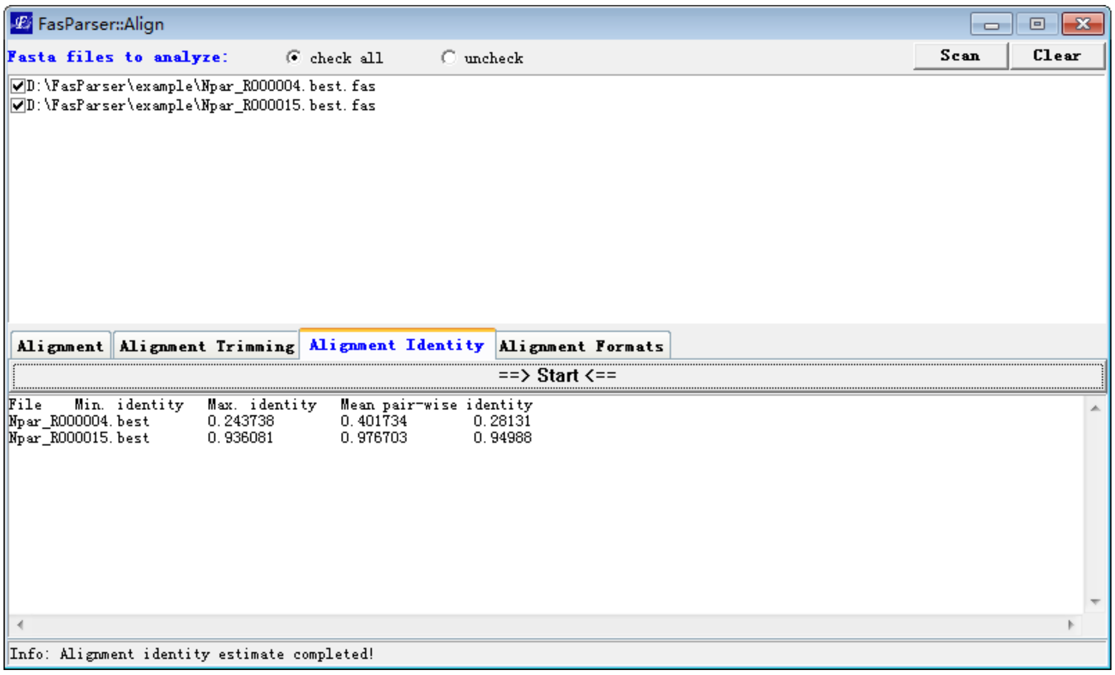
```
Info: Alignment trimming completed!
```

Figure 3. Overview of alignment trimming.

[Sim]

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 '*Scan*' button, and then click the "*Start*" button to start the estimates (Figure 4).



D:\FasParser\example\Npar_R000004.best.fas
D:\FasParser\example\Npar_R000015.best.fas

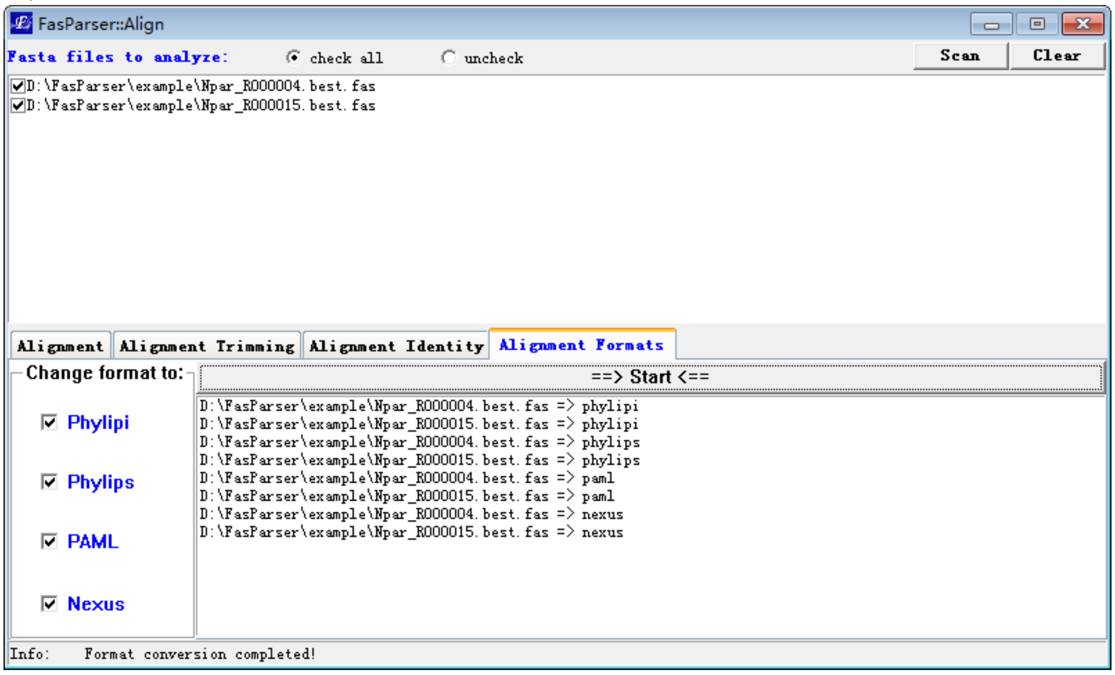
```
File     Min. identity    Max. identity    Mean pair-wise identity
Npar_R000004.best        0.243738         0.401734         0.28131
Npar_R000015.best        0.936081         0.976703         0.94988
```

Info: Alignment identity estimate completed!

Figure 4. Overview of alignment similarity estimates.

[Format]

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 '*. <format>.fas'.



D:\FasParser\example\Npar_R000004.best.fas
D:\FasParser\example\Npar_R000015.best.fas

☑ Phylipi
☑ Phylips
☑ PAML
☑ Nexus

```
D:\FasParser\example\Npar_R000004.best.fas => phylipi
D:\FasParser\example\Npar_R000015.best.fas => phylipi
D:\FasParser\example\Npar_R000004.best.fas => phylips
D:\FasParser\example\Npar_R000015.best.fas => phylips
D:\FasParser\example\Npar_R000004.best.fas => paml
D:\FasParser\example\Npar_R000015.best.fas => paml
D:\FasParser\example\Npar_R000004.best.fas => nexus
D:\FasParser\example\Npar_R000015.best.fas => nexus
```

Info: Format conversion completed!

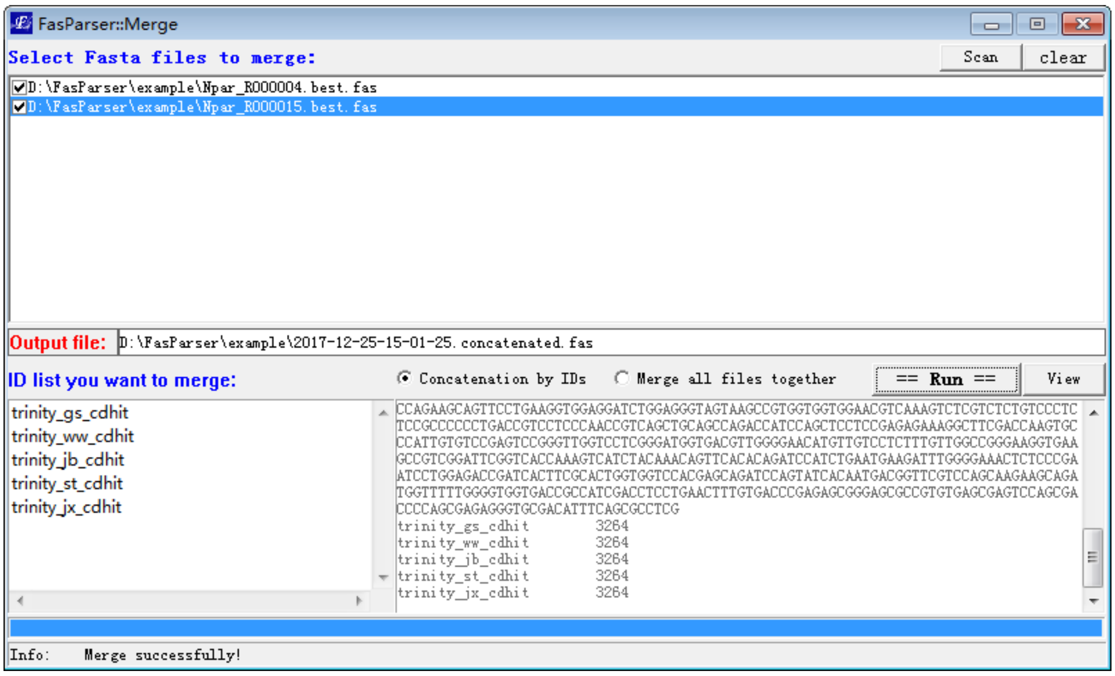
Figure 5. Overview of alignment format conversion.

[Merge]

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

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 define which IDs will be concatenated by editing the ID list in the textbox (left-below, Figure 6).



```
[✓] D:\FasParser\example\Npar_R000004.best.fas
[✓] D:\FasParser\example\Npar_R000015.best.fas
```

Output file: D:\FasParser\example\2017-12-25-15-01-25.concatenated.fas

(● Concatenation by IDs) (Merge all files together)

```
trinity_gs_cdhit
trinity_ww_cdhit
trinity_jb_cdhit
trinity_st_cdhit
trinity_jx_cdhit
```

```
CCAGAAGCAGTTCCTGAAGGTGGAGGATCTGGAGGGTAGTAAGCCGTGGTGGTGGAACGTCAAAGTCTCGTCTCTGTCCCTC
TCCGCCCCCCTGACCGTCCTCCCAACCGTCAGCTGCAGCCAGACCATCCAGCTCCTCCGAGAGAAAGGCTTCGACCAAGTGC
CCATTGTGTCCGAGTCCGGGTTGGTCCTCGGGATGGTGACGTTGGGGAACATGTTGTCCTCTTTGTTGGCCGGGAAGGTGAA
GCCGTCGGATTCGGTCACCAAAGTCATCTACAAACAGTTCACACAGATCCATCTGAATGAAGATTTGGGGAAACTCTCCCGA
ATCCTGGAGACCGATCACTTCGCACTGGTGGTCCACGAGCAGATCCAGTATCACAATGACGGTTCGTCCAGCAAGAAGCAGA
TGGTTTTTGGGGTGGTGACCGCCATCGACCTCCTGAACTTTGTGACCCGAGAGCGGGAGCGCCGTGTGAGCGAGTCCAGCGA
CCCCAGCGAGAGGGTGCGACATTTCAGCGCCTCG
trinity_gs_cdhit        3264
trinity_ww_cdhit        3264
trinity_jb_cdhit        3264
trinity_st_cdhit        3264
trinity_jx_cdhit        3264
```

```
Info:    Merge successfully!
```

Figure 6. Overview of sequence concatenation manipulation.

[Sort]

Sort Sequences

This program will allow users to sort their FASTA files according to either 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 is automatedly recognized from the raw ID, which is** 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…'
```

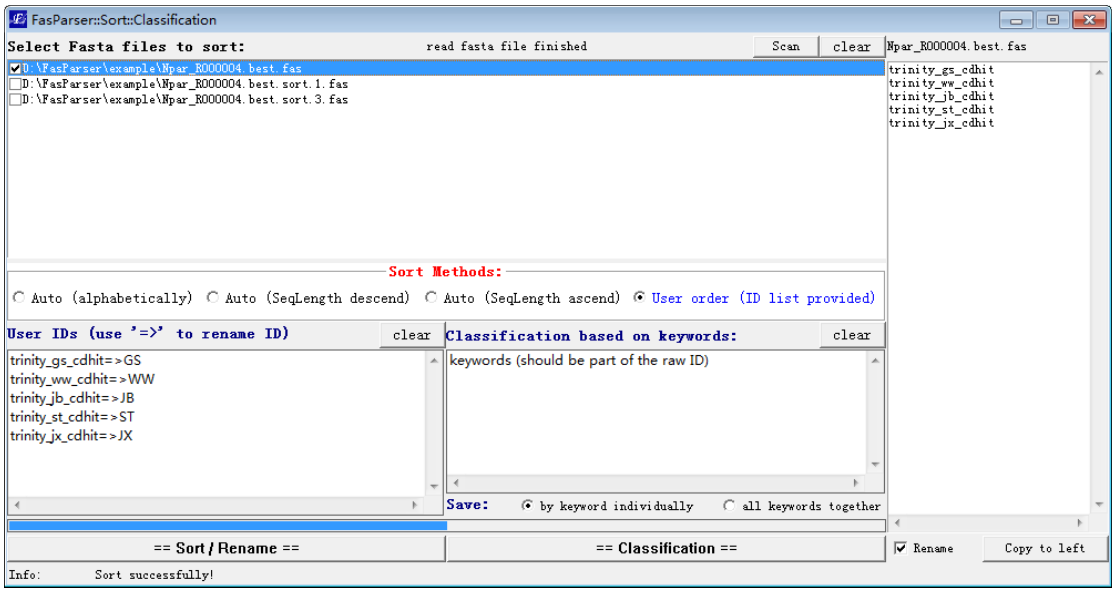
The output files will be saved in the folder which contains the input files, with '*. sort.1.fas' for alphabetically, '*. sort.2.1.fas' and '*. sort.2.2.fas' for length-descend and length-ascend, respectively.

Rename sequences

With the provided ID list (**left-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 be retained (Figure 7.1). The output files will be saved in the folder which contains the input files, with '*. sort.3.fas'.



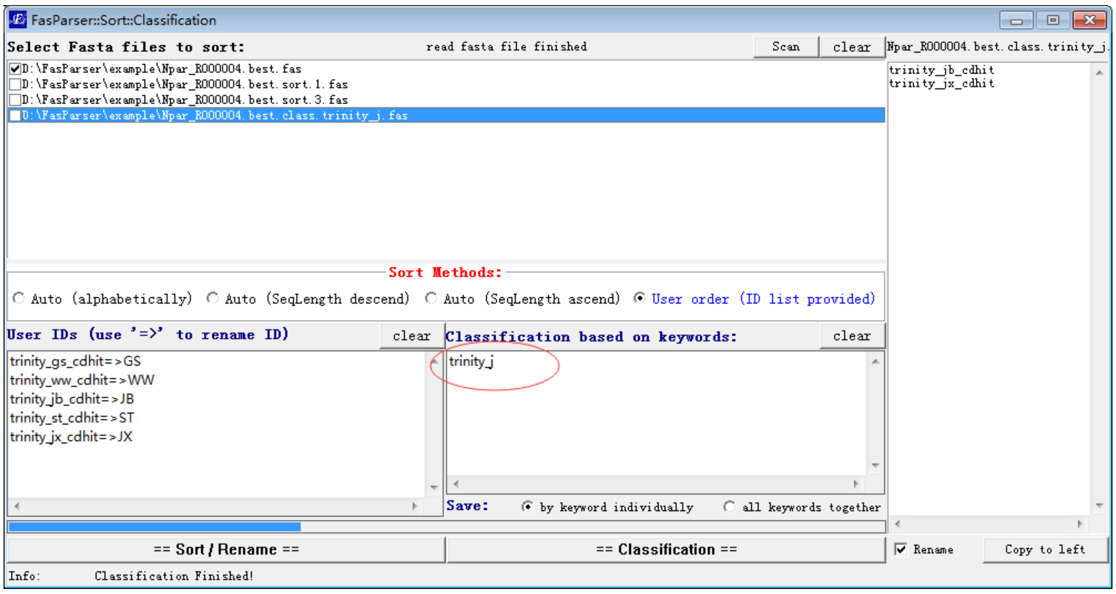
```
FasParser::Sort::Classification
Select Fasta files to sort:        read fasta file finished        Scan  clear   Npar_R000004.best.fas
☑D:\FasParser\example\Npar_R000004.best.fas                                      trinity_gs_cdhit
☐D:\FasParser\example\Npar_R000004.best.sort.1.fas                               trinity_ww_cdhit
☐D:\FasParser\example\Npar_R000004.best.sort.3.fas                               trinity_jb_cdhit
                                                                                 trinity_st_cdhit
                                                                                 trinity_jx_cdhit
Sort Methods:
○ Auto (alphabetically)  ○ Auto (SeqLength descend)  ○ Auto (SeqLength ascend)  ● User order (ID list provided)
User IDs (use '=>' to rename ID)   clear   Classification based on keywords:   clear
trinity_gs_cdhit=>GS                       keywords (should be part of the raw ID)
trinity_ww_cdhit=>WW
trinity_jb_cdhit=>JB
trinity_st_cdhit=>ST
trinity_jx_cdhit=>JX
                                           Save:  ● by keyword individually  ○ all keywords together
== Sort / Rename ==                        == Classification ==               ☑ Rename   Copy to left
Info:     Sort successfully!
```

Figure 7.1. Overview of Sort & Rename sequences.

Classify sequences

In addition, this section also provides a Classification function, which means that you can use one or more keywords to extract sequences from a raw FASTA file 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 right-below textbox, and then click the "*Classification*" button to run this analysis. The output files will be saved in the folder which contains the input files, with '*.class.[keyword].fas'.



```
FasParser::Sort::Classification
Select Fasta files to sort:        read fasta file finished        Scan  clear   Npar_R000004.best.class.trinity_j.
☑D:\FasParser\example\Npar_R000004.best.fas                                      trinity_jb_cdhit
☐D:\FasParser\example\Npar_R000004.best.sort.1.fas                               trinity_jx_cdhit
☐D:\FasParser\example\Npar_R000004.best.sort.3.fas
☐D:\FasParser\example\Npar_R000004.best.class.trinity_j.fas
Sort Methods:
○ Auto (alphabetically)  ○ Auto (SeqLength descend)  ○ Auto (SeqLength ascend)  ● User order (ID list provided)
User IDs (use '=>' to rename ID)   clear   Classification based on keywords:   clear
trinity_gs_cdhit=>GS                       trinity_j
trinity_ww_cdhit=>WW
trinity_jb_cdhit=>JB
trinity_st_cdhit=>ST
trinity_jx_cdhit=>JX
                                           Save:  ● by keyword individually  ○ all keywords together
== Sort / Rename ==                        == Classification ==               ☑ Rename   Copy to left
Info:     Classification Finished!
```

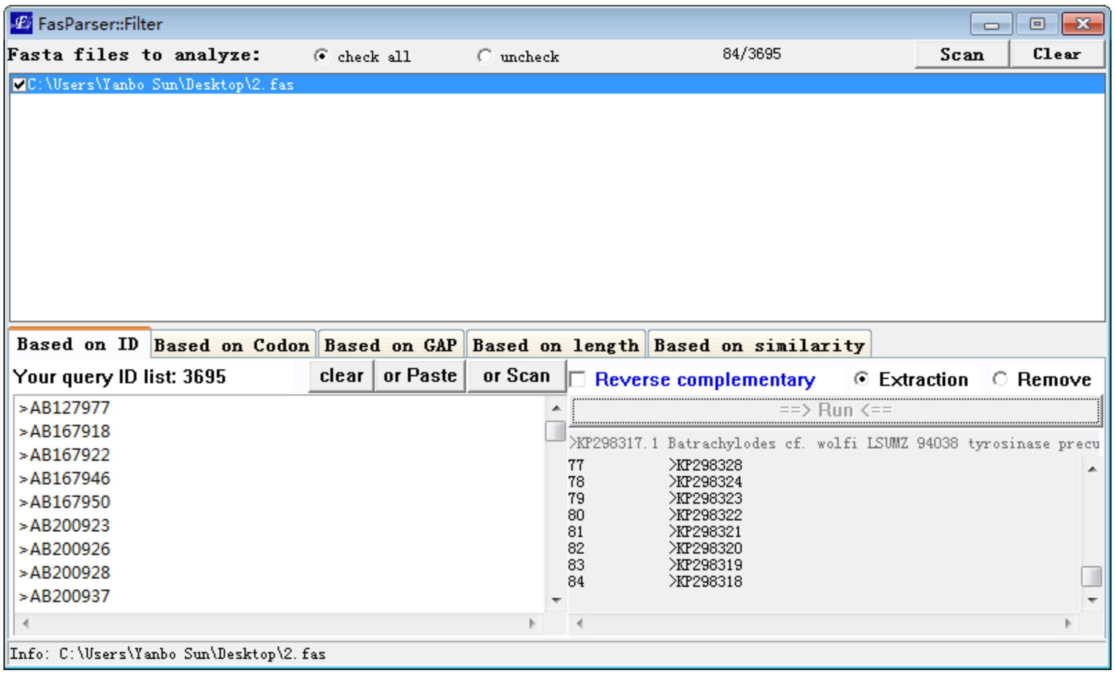
Figure 7.2. Overview of Classification function.

[Filter]

This program is designed to perform some extraction and filtration analyses within a particular FASTA file. It is a much common sequence manipulation. With this program, users can extract or remove a set of sequences from the raw FASTA file based on query IDs, the positions per codon, and can also filter the sequences based on the sequence length, or 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'.



FasParser::Filter

Fasta files to analyze: [✓ check all] uncheck 84/3695

C:\Users\Yanbo Sun\Desktop\2.fas

Your query ID list: 3695

```
>AB127977
>AB167918
>AB167922
>AB167946
>AB167950
>AB200923
>AB200926
>AB200928
>AB200937
```

Reverse complementary [✓ Extraction] Remove

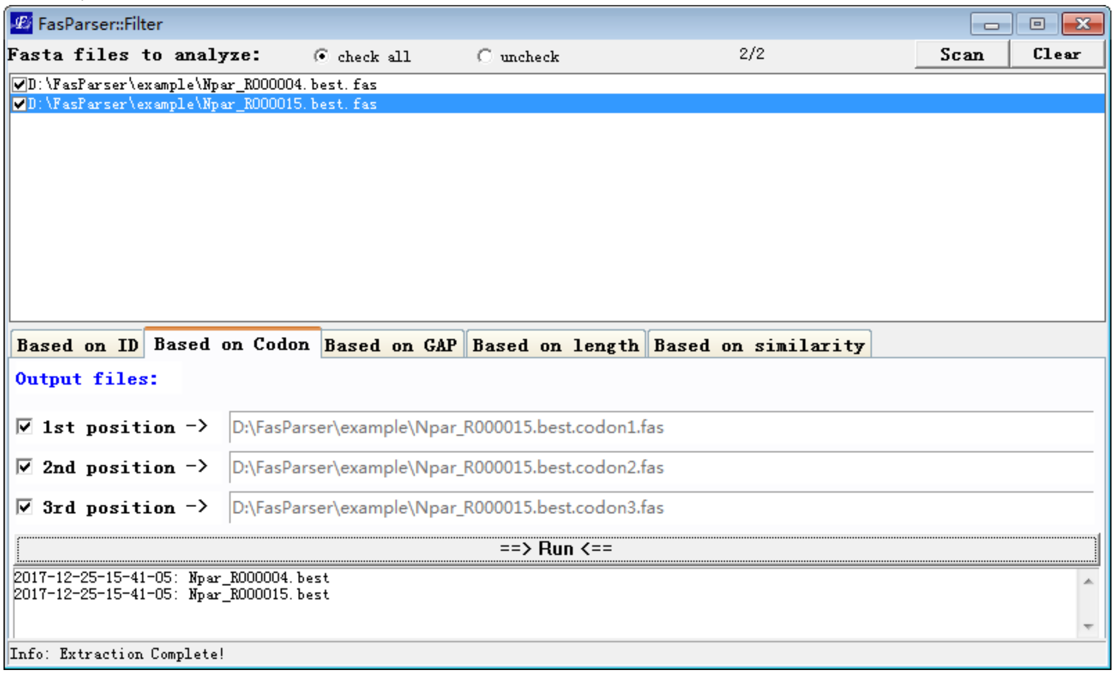
```
>KP298317.1 Batrachylodes cf. wolfi LSUMZ 94038 tyrosinase precu
77          >KP298328
78          >KP298324
79          >KP298323
80          >KP298322
81          >KP298321
82          >KP298320
83          >KP298319
84          >KP298318
```

Info: C:\Users\Yanbo Sun\Desktop\2.fas

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

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'.



FasParser::Filter

Fasta files to analyze: [✓ check all] uncheck 2/2 Scan Clear

☑ D:\FasParser\example\Npar_R000004.best.fas
☑ D:\FasParser\example\Npar_R000015.best.fas

Output files:

☑ 1st position -> D:\FasParser\example\Npar_R000015.best.codon1.fas
☑ 2nd position -> D:\FasParser\example\Npar_R000015.best.codon2.fas
☑ 3rd position -> D:\FasParser\example\Npar_R000015.best.codon3.fas

==> Run <==

```
2017-12-25-15-41-05: Npar_R000004.best
2017-12-25-15-41-05: Npar_R000015.best
```

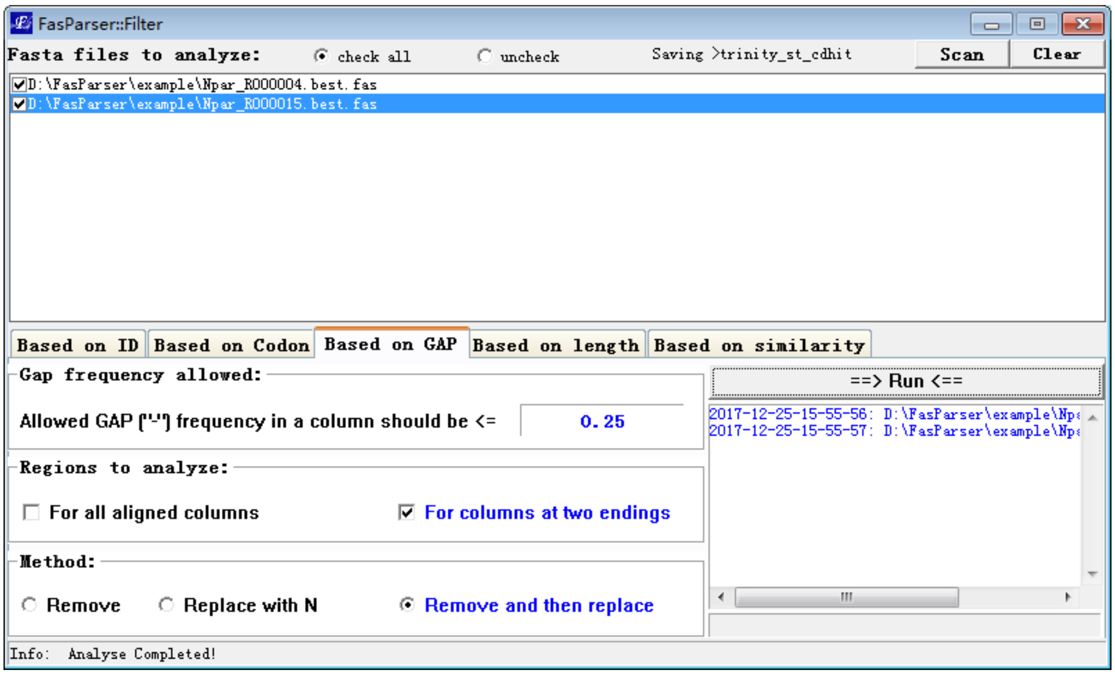
Info: Extraction Complete!

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".


FasParser::Filter

Fasta files to analyze: [✓ check all] uncheck   Saving >trinity_st_cdhit   Scan   Clear

```
☑ D:\FasParser\example\Npar_R000004.best.fas
☑ D:\FasParser\example\Npar_R000015.best.fas
```

Based on ID | Based on Codon | Based on GAP | Based on length | Based on similarity

Gap frequency allowed:

Allowed GAP ('-') frequency in a column should be <= 0.25

Regions to analyze:

☐ For all aligned columns   ☑ For columns at two endings

Method:

Remove   Replace with N   [✓ Remove and then replace]

==> Run <==

```
2017-12-25-15-55-56: D:\FasParser\example\Npa
2017-12-25-15-55-57: D:\FasParser\example\Npa
```

Info: Analyse Completed!

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.

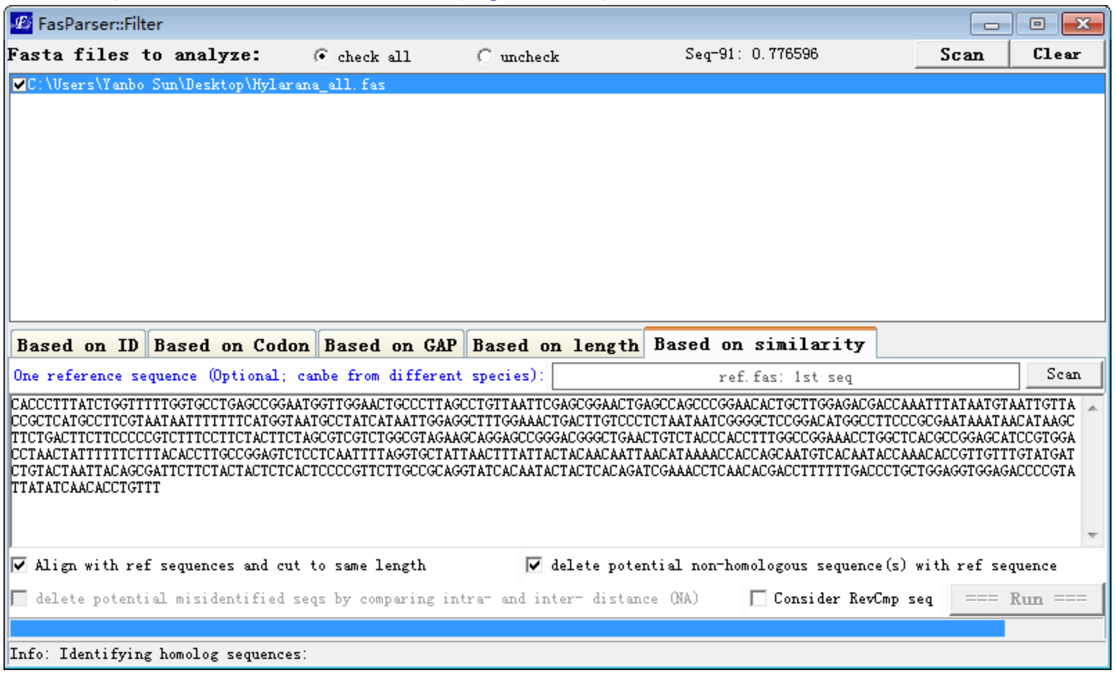
The 2nd method is designed for non-aligned files (faster). This method will identify directly the too short sequences according to their sequence lengths. The formula used is "mean - 3*std" if there are more than 100 sequences. If there are less than 100 sequences, Grubbs method will be used.

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).



Seq-91: 0.776596

☑ C:\Users\Yanbo Sun\Desktop\Hylarana_all.fas

One reference sequence (Optional; canbe from different species): ref.fas: 1st seq

```
CACCCTTTATCTGGTTTTTGGTGCCTGAGCCGGAATGGTTGGAACTGCCCTTAGCCTGTTAATTCGAGCGGAACTGAGCCAGCCCGGAACACTGCTTGGAGACGACCAAATTTATAATGTAATTGTTA
CCGCTCATGCCTTCGTAATAATTTTTTTCATGGTAATGCCTATCATAATTGGAGGCTTTGGAAACTGACTTGTCCCTCTAATAATCGGGGCTCCGGACATGGCCTTCCCGCGAATAAATAACATAAGC
TTCTGACTTCTTCCCCCGTCTTTCCTTCTACTTCTAGCGTCGTCTGGCGTAGAAGCAGGAGCCGGGACGGGCTGAACTGTCTACCCACCTTTGGCCGGAAACCTGGCTCACGCCGGAGCATCCGTGGA
CCTAACTATTTTTTCTTTACACCTTGCCGGAGTCTCCTCAATTTTAGGTGCTATTAACTTTATTACTACAACAATTAACATAAAACCACCAGCAATGTCACAATACCAAACACCGTTGTTTGTATGAT
CTGTACTAATTACAGCGATTCTTCTACTACTCTCACTCCCCGTTCTTGCCGCAGGTATCACAATACTACTCACAGATCGAAACCTCAACACGACCTTTTTTGACCCTGCTGGAGGTGGAGACCCCGTA
TTATATCAACACCTGTTT
```

☑ Align with ref sequences and cut to same length

☑ delete potential non-homologous sequence(s) with ref sequence

☐ delete potential misidentified seqs by comparing intra- and inter- distance (NA)

☐ Consider RevCmp seq

=== Run ===

Info: Identifying homolog sequences:

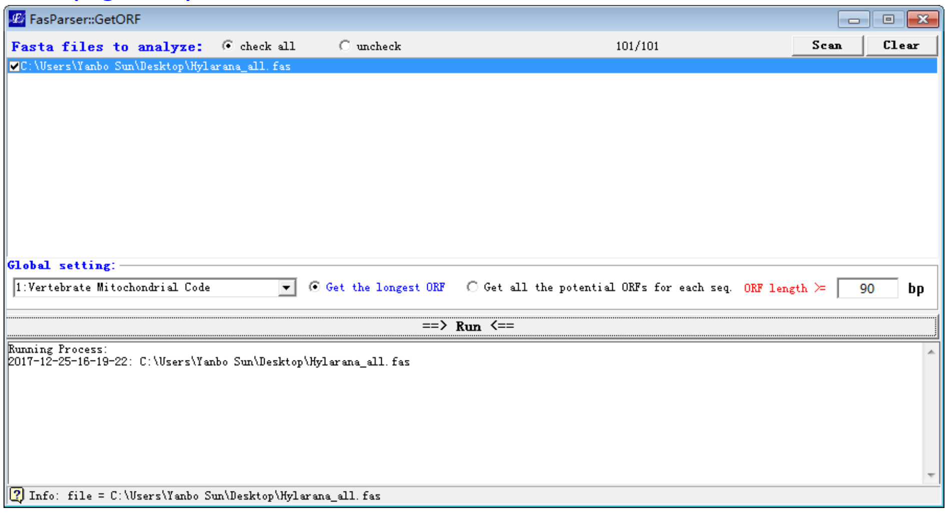
Figure. 8.4. Overview of the sequence filtration based on similarity with a reference sequence.

[DNA2AA]

This function is used to translate the DNA sequences to protein sequences. There are two modes available. If users have many FASTA files, you can use the batch mode to conduct such analyses, with genetic code selected. The output files will be saved into the folder which contains the raw input files, with a filename as "*.aa.fas". You can also input a pure DNA sequence or a FASTA file under the manual mode, the program can automate recognize them.

[ORF]

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).

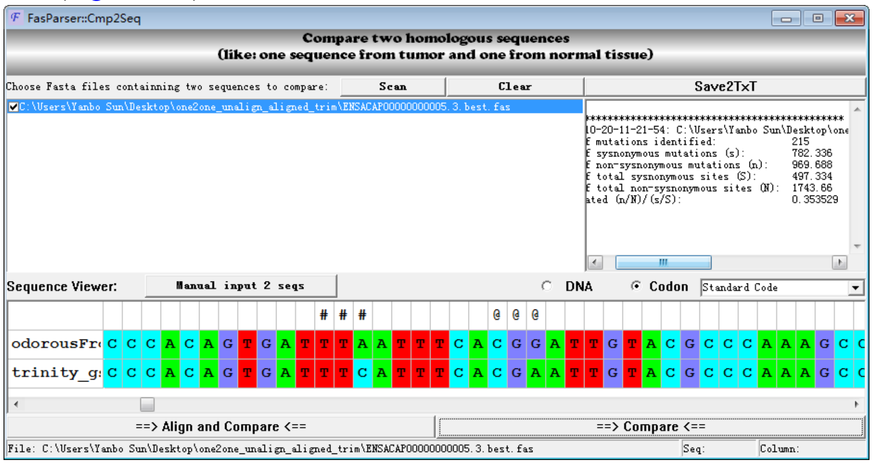


```
FasParser::GetORF
Fasta files to analyze:  (•) check all   ( ) uncheck        101/101     Scan   Clear
☑C:\Users\Yanbo Sun\Desktop\Hylarana_all.fas

Global setting:
1:Vertebrate Mitochondrial Code  (•) Get the longest ORF  ( ) Get all the potential ORFs for each seq.  ORF length >= 90 bp
==> Run <==
Running Process:
2017-12-25-16-19-22: C:\Users\Yanbo Sun\Desktop\Hylarana_all.fas

Info: file = C:\Users\Yanbo Sun\Desktop\Hylarana_all.fas
```

Figure 9. overview of identification of ORFs for sequences.

[2Seq]

The program "2Seq" 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).

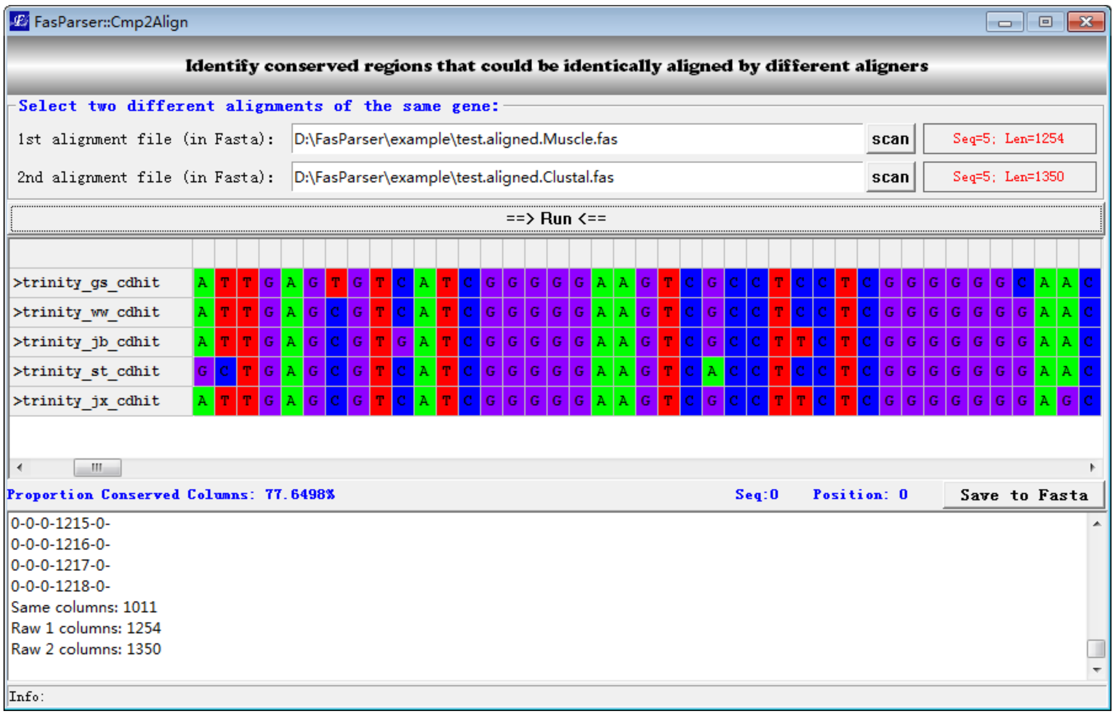


```
FasParser::Cmp2Seq
Compare two homologous sequences
(like: one sequence from tumor and one from normal tissue)
Choose Fasta files containning two sequences to compare:   Scan   Clear   Save2TxT
☑C:\Users\Yanbo Sun\Desktop\one2one_unalign_aligned_trim\ENSACAP00000000005.3.best.fas

**************************************************
10-20-11-21-54: C:\Users\Yanbo Sun\Desktop\one
f mutations identified:                 215
f sysnonymous mutations (s):            782.336
f non-sysnonymous mutations (n):        969.688
f total sysnonymous sites (S):          497.334
f total non-sysnonymous sites (N):      1743.66
ated (n/N)/(s/S):                       0.353529

Sequence Viewer:  Manual input 2 seqs     ( ) DNA  (•) Codon  Standard Code
                                   # # #                   @ @ @
odorousFr  C C C A C A G T G A T T T A A T T T C A C G G A T T G T A C G C C C A A A G C C
trinity_g  C C C A C A G T G A T T T C A T T T C A C G A A T T G T A C G C C C A A A G C C
==> Align and Compare <==                    ==> Compare <==
File: C:\Users\Yanbo Sun\Desktop\one2one_unalign_aligned_trim\ENSACAP00000000005.3.best.fas   Seq:   Column:
```

Figure 10. Overview of comparison between two sequences.

[2Align]

This program "2Align" 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.



Identify conserved regions that could be identically aligned by different aligners

Select two different alignments of the same gene:

1st alignment file (in Fasta): D:\FasParser\example\test.aligned.Muscle.fas Seq=5; Len=1254

2nd alignment file (in Fasta): D:\FasParser\example\test.aligned.Clustal.fas Seq=5; Len=1350

```
>trinity_gs_cdhit  A T T G A G T G T C A T C G G G G G A A G T C G C C T C C T C G G G G G G C A A C
>trinity_ww_cdhit  A T T G A G C G T C A T C G G G G G A A G T C G C C T C C T C G G G G G G G A A C
>trinity_jb_cdhit  A T T G A G C G T G A T C G G G G G A A G T C G C C T T C T C G G G G G G G A A C
>trinity_st_cdhit  G C T G A G C G T C A T C G G G G G A A G T C A C C T C C T C G G G G G G G A A C
>trinity_jx_cdhit  A T T G A G C G T C A T C G G G G G A A G T C G C C T T C T C G G G G G G G A G C
```

Proportion Conserved Columns: 77.6498%   Seq:0   Position: 0

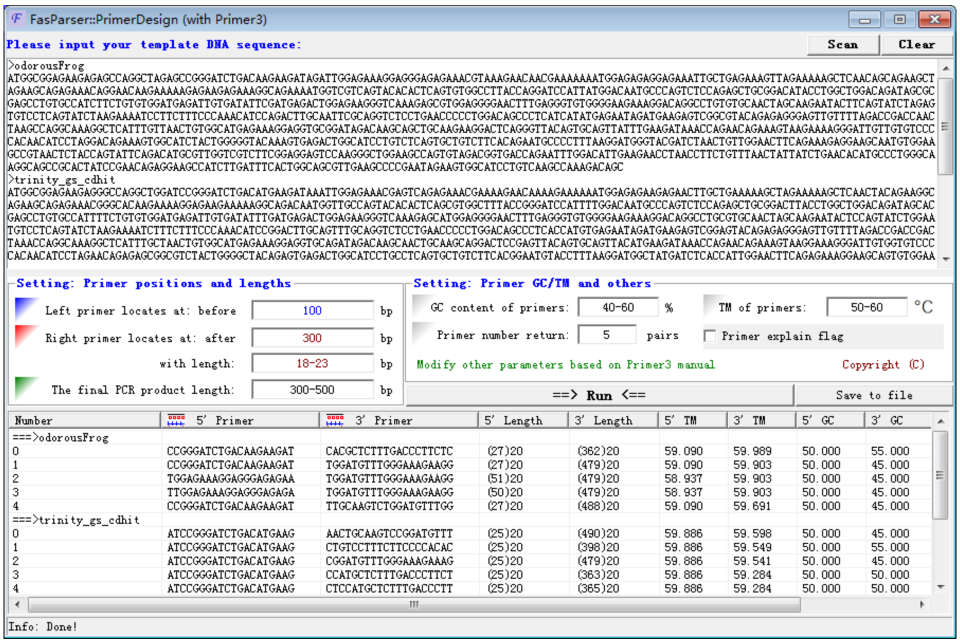
```
0-0-0-1215-0-
0-0-0-1216-0-
0-0-0-1217-0-
0-0-0-1218-0-
Same columns: 1011
Raw 1 columns: 1254
Raw 2 columns: 1350
```

Info:

Figure 11. Overview of the estimates of consensus alignment.

[Primer]

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).


FasParser::PrimerDesign (with Primer3)

Please input your template DNA sequence:

```
>odorousFrog
ATGGCGGAGAAGAGAGCCAGGCTAGAGCCGGGATCTGACAAGAAGATAGATTGGAGAAAGGAGGGAGAGAAACGTAAAGAACAACGAAAAAAATGGAGAGAGGAGAAATTGCTGAGAAAGTTAGAAAAAGCTCAACAGCAGAAGCT
AGAAGCAGAGAAACAGGAACAAGAAAAAGAGAAGAGAAAGGCAGAAAATGGTCGTCAGTACACACTCAGTGTGGCCTTACCAGGATCCATTATGGACAATGCCCAGTCTCCAGAGCTGCGGACATACCTGGCTGGACAGATAGCGC
GAGCCTGTGCCATCTTCTGTGTGGATGAGATTGTGATATTCGATGAGACTGGAGAAGGGTCAAAGAGCGTGGAGGGGAACTTTGAGGGTGTGGGGAAGAAAGGACAGGCCTGTGTGCAACTAGCAAGAATACTTCAGTATCTAGAG
TGTCCTCAGTATCTAAGAAAATCCTTCTTTCCCAAACATCCAGACTTGCAATTCGCAGGTCTCCTGAACCCCCTGGACAGCCCTCATCATATGAGAATAGATGAAGAGTCGGCGTACAGAGAGGGAGTTGTTTTAGACCGACCAAC
TAAGCCAGGCAAAGGCTCATTTGTTAACTGTGGCATGAGAAAGGAGGTGCGGATAGACAAGCAGCTGCAAGAAGGACTCAGGGTTACAGTGCAGTTATTTGAAGATAAACCAGAACAGAAAGTAAGAAAAGGGATTGTTGTGTCCC
CACAACATCCTAGGACAGAAAGTGGCATCTACTGGGGGTACAAAGTGAGACTGGCATCCTGTCTCAGTGCTGTCTTCACAGAATGCCCCTTTAAGGATGGGTACGATCTAACTGTTGGAACTTCAGAAAGAGGAAGCAATGTGGAA
GCCGTAACTCTACCAGTATTCAGACATGCGTTGGTCGTCTTCGGAGGAGTCCAAGGGCTGGAAGCCAGTGTAGACGGTGACCAGAATTTGGACATTGAAGAACCTAACCTTCTGTTTAACTATTATCTGAACACATGCCCTGGGCA
AGGCAGCCGCACTATCCGAACAGAGGAAGCCATCTTGATTTCACTGGCAGCGTTGAAGCCCCGAATAGAAGTGGCATCCTGTCAAGCCAAAGACAGC
>trinity_gs_cdhit
ATGGCGGAGAAGAGGGCCAGGCTGGATCCGGGATCTGACATGAAGATAAATTGGAGAAACGAGTCAGAGAAACGAAAAGAACAAAAGAAAAAATGGAGAGAAGAGAACTTGCTGAAAAAGCTAGAAAAAGCTCAACTACAGAAGGC
AGAAGCAGAGAAACGGGCACAAGAAAAGGAGAAGAAAAAGGCAGACAATGGTTGCCAGTACACACTCAGCGTGGCTTTACCGGGATCCATTTTGGACAATGCCCAGTCTCCAGAGCTGCGGACTTACCTGGCTGGACAGATAGCAC
GAGCCTGTGCCATTTTCTGTGTGGATGAGATTGTGATATTTGATGAGACTGGAGAAGGGTCAAAGAGCATGGAGGGGAACTTTGAGGGTGTGGGGAAGAAAGGACAGGCCTGCGTGCAACTAGCAAGAATACTCCAGTATCTGGAA
TGTCCTCAGTATCTAAGAAAATCTTTCTTTCCCAAACATCCGGACTTGCAGTTTGCAGGTCTCCTGAACCCCCTGGACAGCCCTCACCATGTGAGAATAGATGAAGAGTCGGAGTACAGAGAGGGAGTTGTTTTAGACCGACCGAC
TAAACCAGGCAAAGGCTCATTTGCTAACTGTGGCATGAGAAAGGAGGTGCAGATAGACAAGCAACTGCAAGCAGGACTCCGAGTTACAGTGCAGTTACATGAAGATAAACCAGAACAGAAAGTAAGGAAAGGGATTGTGGTGTCCC
CACAACATCCTAGAACAGAGAGCGGCGTCTACTGGGGCTACAGAGTGAGACTGGCATCCTGCCTCAGTGCTGTCTTCACGGAATGTACCTTTAAGGATGGCTATGATCTCACCATTGGAACTTCAGAGAAAGGAAGCAGTGTGGAA
```

Setting: Primer positions and lengths

Left primer locates at: before 100 bp
Right primer locates at: after 300 bp
with length: 18-23 bp
The final PCR product length: 300-500 bp

Setting: Primer GC/TM and others

GC content of primers: 40-60 %
TM of primers: 50-60 °C
Primer number return: 5 pairs
☐ Primer explain flag

Modify other parameters based on Primer3 manual

Copyright (C)

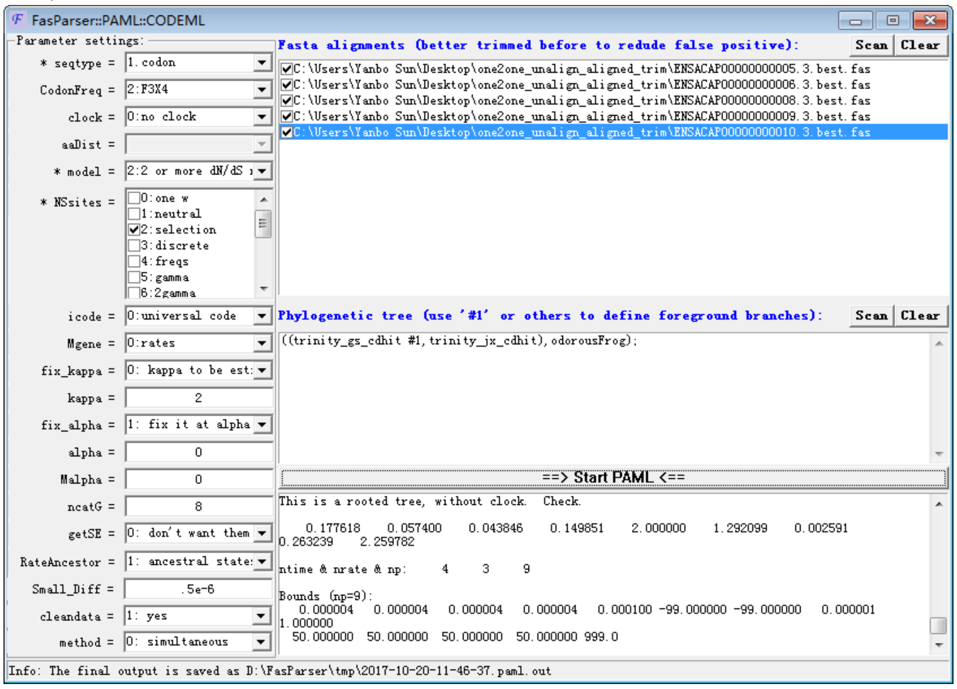
Number	5' Primer	3' Primer	5' Length	3' Length	5' TM	3' TM	5' GC	3' GC
===>odorousFrog								
0	CCGGGATCTGACAAGAAGAT	CACGCTCTTTGACCCTTCTC	(27)20	(362)20	59.090	59.989	50.000	55.000
1	CCGGGATCTGACAAGAAGAT	TGGATGTTTGGGAAAGAAGG	(27)20	(479)20	59.090	59.903	50.000	45.000
2	TGGAGAAAGGAGGGAGAGAA	TGGATGTTTGGGAAAGAAGG	(51)20	(479)20	58.937	59.903	50.000	45.000
3	TTGGAGAAAGGAGGGAGAGA	TGGATGTTTGGGAAAGAAGG	(50)20	(479)20	58.937	59.903	50.000	45.000
4	CCGGGATCTGACAAGAAGAT	TTGCAAGTCTGGATGTTTGG	(27)20	(488)20	59.090	59.691	50.000	45.000
===>trinity_gs_cdhit								
0	ATCCGGGATCTGACATGAAG	AACTGCAAGTCCGGATGTTT	(25)20	(490)20	59.886	59.598	50.000	45.000
1	ATCCGGGATCTGACATGAAG	CTGTCCTTTCTTCCCCACAC	(25)20	(398)20	59.886	59.549	50.000	55.000
2	ATCCGGGATCTGACATGAAG	CGGATGTTTGGGAAAGAAAG	(25)20	(479)20	59.886	59.541	50.000	45.000
3	ATCCGGGATCTGACATGAAG	CCATGCTCTTTGACCCTTCT	(25)20	(363)20	59.886	59.284	50.000	50.000
4	ATCCGGGATCTGACATGAAG	CTCCATGCTCTTTGACCCTT	(25)20	(365)20	59.886	59.284	50.000	50.000

Info: Done!

Figure 12. Overview of the primer designing function.

[PAML]

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).


FasParser::PAML::CODEML

Parameter settings:

```
* seqtype = 1. codon
CodonFreq = 2:F3X4
    clock = 0:no clock
   aaDist =
  * model = 2:2 or more dN/dS r
* NSsites = ☐0:one w
            ☐1:neutral
            ☑2:selection
            ☐3:discrete
            ☐4:freqs
            ☐5:gamma
            ☐6:2gamma
    icode = 0:universal code
    Mgene = 0:rates
fix_kappa = 0: kappa to be est:
    kappa = 2
fix_alpha = 1: fix it at alpha
    alpha = 0
   Malpha = 0
    ncatG = 8
    getSE = 0: don't want them
RateAncestor = 1: ancestral state:
Small_Diff = .5e-6
cleandata = 1: yes
   method = 0: simultaneous
```

Fasta alignments (better trimmed before to redude false positive):

```
☑C:\Users\Yanbo Sun\Desktop\one2one_unalign_aligned_trim\ENSACAP00000000005.3.best.fas
☑C:\Users\Yanbo Sun\Desktop\one2one_unalign_aligned_trim\ENSACAP00000000006.3.best.fas
☑C:\Users\Yanbo Sun\Desktop\one2one_unalign_aligned_trim\ENSACAP00000000008.3.best.fas
☑C:\Users\Yanbo Sun\Desktop\one2one_unalign_aligned_trim\ENSACAP00000000009.3.best.fas
☑C:\Users\Yanbo Sun\Desktop\one2one_unalign_aligned_trim\ENSACAP00000000010.3.best.fas
```

Phylogenetic tree (use '#1' or others to define foreground branches):

```
((trinity_gs_cdhit #1,trinity_jx_cdhit),odorousFrog);
```

```
This is a rooted tree, without clock.   Check.

    0.177618    0.057400    0.043846    0.149851    2.000000    1.292099    0.002591
0.263239    2.259782

ntime & nrate & np:     4     3     9

Bounds (np=9):
   0.000004    0.000004    0.000004    0.000004    0.000100 -99.000000 -99.000000    0.000001
1.000000
  50.000000   50.000000   50.000000   50.000000 999.0
```

Info: The final output is saved as D:\FasParser\tmp\2017-10-20-11-46-37.paml.out

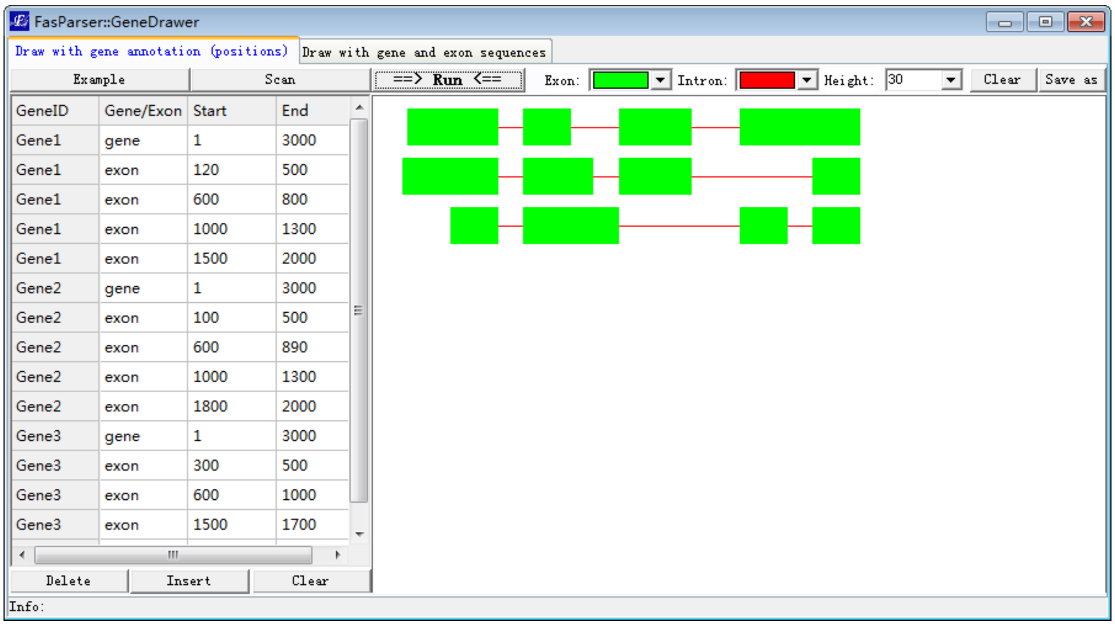
Figure 13. Positive selection detection.

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
```

[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.



Exon: 🟩 Intron: 🟥 Height: 30

GeneID	Gene/Exon	Start	End
Gene1	gene	1	3000
Gene1	exon	120	500
Gene1	exon	600	800
Gene1	exon	1000	1300
Gene1	exon	1500	2000
Gene2	gene	1	3000
Gene2	exon	100	500
Gene2	exon	600	890
Gene2	exon	1000	1300
Gene2	exon	1800	2000
Gene3	gene	1	3000
Gene3	exon	300	500
Gene3	exon	600	1000
Gene3	exon	1500	1700

Figure 14. Overview of gene structure drawing function.

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.