ConCat v-10 Manual

Alignment concatenation and analysis utility

Ambuj Kumar, University of Florida
Email: ambuj@ufl.edu
Kimball-Braun Lab group

Contents

List of Figures

List of Tables

1. Introduction

ConCat is a biopython based alignment concatenation utility designed to obtain alignment super matrix from list of alignment files. It is divided into three modules (ConCat-Align, ConCat-Core, ConCat-Analyze). ConCat-Align module provides dynamics sequence alignment options with multiple input/output formats. ConCat-Core module is for concatenation, annotation handling and storage. ConCat-Analyze module conducts post analysis of super matrix obtained from ConCat-Core module. Workflow of ConCat is shown below:

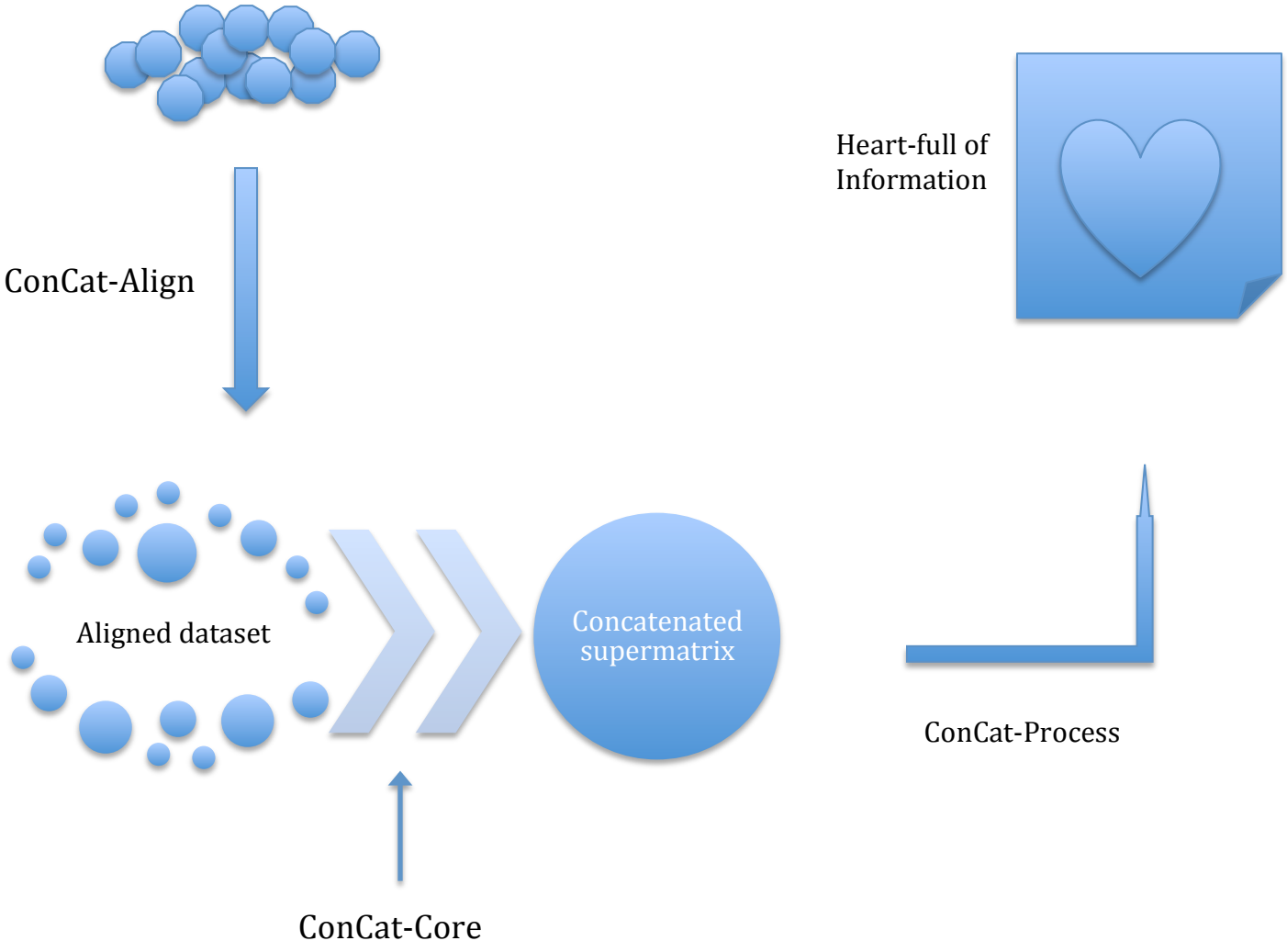

Fig 1. ConCat Workflow

Different script files separate each module so the user can choose to run individual modules at a time.

ConCat produces rich character annotations obtained by user-selected functions applied along with the concatenation process. The program can handle input file in fasta, phylip-interleaved, phylip-relaxed and nexus format. ConCat has several alignment processing functions that helps user to create and store publication ready annotations and files. It extracts genome IDs (if available) from the taxon IDs and stores it in an excel sheet in a publication ready format. Furthermore, it checks for taxon name spelling errors within the alignment file, identifies the missing taxa in each alignment file and stores these information in nexus output as taxset data. Moreover, it allows user to select or reject specific taxa while conducting concatenation, edit taxa name by adding or removing user supplied nomenclature group and creates useful bins by classifying different input alignment file datasets according to user defined criterion.

1.1. Input/Output

Creating, handling and storing annotation is one of the most important features of ConCat, which is why the nexus is a preferred input format in ConCat. Although, ConCat allows multiple input/output formats including fasta (.fas), nexus (.nex), phylip-interleaved (.phy) and phylip-relaxed (.phy). ConCat has "–CA" and "-convert" arguments to handle non-nexus input files. These arguments allows user to first convert non-nexus input files to nexus format and then run the analysis or select to convert and run the analysis simultaneously. ConCat also allows user to select fasta, nexus, phylip-interleaved and phylip-relaxed output formats.

1.2. ConCat block

ConCat allows users to define the alignment file type, define files to run RNA structure mapping and supply user defined RNA structure through ConCat block option. Structure of ConCat block is shown below:

```
begin ConCat;
        Ali_type = DNA;
        RNA_type = False;
        RNA_Struc = Species_Name, (((..(….))….).),6;
end;
```

ConCat block can be defined in nexus alignment input file after the first line that contains #NEXUS. This block has three variables and each carries special set of information.

Ali_type variable takes alignment type information from user. User can allow ConCat to create RaxML partition file by defining the alignment type in ConCat block. ConCat takes the user defined alignment type from each file and extract their corresponding nucleotide positions from the concatenated super matrix to create a final RaxML partition file. This variable also allows ConCat to distinguish between DNA and amino acid alignment data while calculating alignment entropy and RCV values.

RNA_type variable allows user to label the alignment files for which ConCat is supposed to create RNA structure data and map it over the concatenated alignment matrix. ConCat uses RNAfold package to obtain the RNA structure data from the consensus sequence obtained from the input alignment file. Program will skip this step if RNAfold program is not installed on the users system.

RNA_Struc variable allows user to supply predefined RNA structure to map over the concatenated super matrix data. It contains three sub-variables Species_name, structure and structure starting position. So if user has following alignment file:

Homo_sapiens ACTAGATACAGATACGATCAGATCA
Gorilla_gorilla ACTAGATAGAGAAACGATCAGATCA
Macaca_mulata ACTAGATACAGAAACGAACCGCTCA

And Homo_sapiens RNA structure "(((..(....))....).)" starting at position 5th then the ideal way of RNA_Struc variable representation is

RNA_Struc = Homo_sapiens, (((..(....))....).), 5;

2. Usage/Options

2.1. Start ConCat via command line

Steps to run ConCat via command line:

a) Open terminal and move to the ConCat directory.
b) Store your input files in the "Input" directory
c) If input files are in nexus format the run

python ConCat.py

or else run

python ConCat.py -CA -ftype phylip-relaxed -otype phylip-relaxed

for phylip-relaxed input and output files. Select –ftype and –otype argument inputs accordingly.

2.2. ConCat-Core options

2.2.1. –ftype, –otype and –CA

–ftype and –otype arguments are used for passing input and output file type respectively. Bothe arguments allow user to select from 5 options (fasta, nexus, phylip, phylip-interleaved, phylip-relaxed). If a user has input files in fasta format and requires output super matrix file in phylip-relaxed format, then the typical command line operation will be

python ConCat-Core.py -CA –ftype fasta –otype phylip-relaxed

–CA argument first converts all the –ftype file format files to nexus file format, which is then imported by ConCat-Core module. Output is produced in nexus as well as in –otype file format.

ConCat uses nexus as default input format when the –ftype argument is not supplied.

2.2.2. –convert

–convert argument allows user to convert between different file formats.

python ConCat-Core.py –convert –ftype fasta –otype nexus

2.2.3. –spell

Spelling errors are common in taxon naming. ConCat-Core –spell argument checks for possible spelling mistakes by comparing the taxon name in the input alignment files.

2.2.4. –block

By passing –block argument user tells ConCat-Core module to check ConCat block in all the alignment files. ConCat-Core creates RaxML partition file if Ali_type variable is intiated in the ConCat block in the input alignment files.

-block argument is important when user wants to create partition file, supply RNA structure or create and map RNA structure on the final super matrix data. Therefore, it is important to initiate –block argument when defining ConCat block in the input alignment file or using –RNA argument.

Example:

```
python ConCat-Core.py -CA –ftype fasta –otype phylip-relaxed -block –RNA
```

2.2.5. –RNA

-RNA argument can be used to create and map RNA structures from the input alignment file. ConCat uses RNAfold program to generate RNA structure for the input files that has ConCat block RNA_type variable set as True. Program checks for the RNAfold program on the users system and skips this step if the RNAfold program is not installed.

2.2.6. –pipe

-pipe argument is very useful when user has database IDs attached with the taxa name separated by pipes.

Example:
Cricetulus_griseus|NM_001246726.1
Nannospalax_galili|XM_008839441.1
Microtus_ochrogaster|XM_005368436.1
Rattus_norvegicus|NM_207592.1

-pipe argument allows ConCat-Core module to extract these IDs and store it as a Taxset in nexus output file. It also creates a publication ready database ID excel sheet as output.

2.2.7. –shannon

-shannon argument allows user to obtain an estimate of variability within each alignment input files. ConCat has separate modules to calculate Shannon entropy for DNA and protein alignments.
For protein alignment, ConCat first divides each amino acid into 12 unique groups on the basis of their chemical similarity.

```
'u': ['D', 'E']
'b': ['R', 'K']
'i': ['I', 'V']
'l': ['L', 'M']
'f': ['F', 'W', 'Y']
'n': ['N', 'Q']
's': ['S', 'T']
'c': ['C']
'h': ['H']
'a': ['A']
'g': ['G']
'p': ['P']
```

For DNA sequence A, C, G and T forms their own group. Gaps (-) are considered as a separate group in entropy calculation. Finally the entropy values for a particular alignment position is calculated by using following equation:

$$\xi = -\sum_{i=1}^{n} p_i \log p_i$$

Where p_i is the fraction of residue of amino acid group type and n is the number of amino acid group types. These values are averaged over the alignment length to obtain overall entropy.

2.2.8. –rcv

Relative Composition Variability (RCV) value is a good indication of alignment quality. Similar to Entropy calculation, ConCat has separate modules for amino acid and DNA alignment RCV calculations. Similar to entropy calculation function, RCV calculation for amino acid alignment is performed by grouping amino acids into 12 groups on the basis of their chemical composition. A, C, G and T are the only groups taken into account while performing RCV calculation for DNA alignment.

$$RCV = \sum_{i=1}^{n} (|A_i - A^*| + |T_i - T^*| + |C_i - C^*| + |G_i - G^*|) / n \times t$$

where A_i, T_i, C_i, and G_i are the numbers of each nucleotide for the i^{th} taxon. A^*, T^*, C^*, and G^* are averages across the n taxa, and t is the number of sites. Constant sites were excluded for all χ^2 and RCV calculations.

2.2.9. –OV

To calculate observed variability (OV), all sequences for a given position are compared in a pair-wise fashion. Mismatches are scored as 1 and matches as 0; the mean value amongst all the comparisons for a given position is used as the measure of character variability in the subsequent data sorting:

$$OV = \sum_{p=1}^{k} d_{ij} / p$$

Here k is the number of pair-wise comparisons made for a given position and d_{ij} is the score of character variability in each pair-wise comparison made (can be either 0 or 1). If n is the number of aligned sequences which do not have a gap at the given alignment position, then $k = (n^2 - n)/2$.

2.2.10. –GC

This argument calculates GC content of each input alignment file.

2.2.11. –RY

-RY argument asks user for filenames to conduct RY coding either on all the coding positions or for 3rd coding position. File names can be supplied though a text file having one file name in each line and RY coding position separated by commas.

ASPM.nex, 3
CENPJ.nex, all
WDR62.nex, all

ConCat-Core extracts these filenames and selects the RY coding position by matching it with the concatenated supermatrix nexus Charsets. The program performes RY coding for 3rd position if the position supplied by user is “3”, whereas it performes RY coding for all position if the position supplied by user is “all”.

2.2.12. –addT and –remT

-addT and –remT arguments are used for editing taxa name while performing concatenation. –addT takes taxon genus, class, order, phylum and kingdom names as

input from Taxanomy.csv file. Use –addT Phyum to add phylum name to an existing taxa name.

```
python ConCat-Core.py –addT Phylum
```

This will extract Phylum names of each taxon from Taxanomy.scv file and add it to the corresponding taxon names. Use –remT to remove the last segment of taxa name from the final concatenated super matrix object.

```
python ConCat-Core.py –remT
```

2.2.13. –inc and –exc

-inc and –exc arguments allows user to select specific set of taxa from input alignment files to perform concatenation. These two arguments take input from and authority file. –inc argument limits the concatenation process for set of taxa supplied by the user via authority file whereas –exc argument limits the concatenation process for set of taxa which are absent in the authority file.

2.2.14. –pbin

-pbin argument invokes a function that creates multiple 0-25, 25-75 and 75-100 percentile bins (defined as ConCat_Bin) by characterizing the input alignment data on the basis of their corresponding RCV, GC and Entropy values.

```
python ConCat-Core.py –rcv –GC –shannon –pbin
```

This will generate 0-25, 25-75 and 75-100 percentile bins each for the data characterized by their corresponding RCV, GC and entropy values.

2.2.15. –rbin, -ebin, -gcbin

These binning functions takes bin range from user as input and creates one bin of each argument type for the range supplied by user.

```
python ConCat-Core.py –rcv –GC –shannon –rbin 0.1-0.3 –ebin 0.2-0.4 –gcbin 35-50
```

2.2.16. –ugcbin

-ugcbin argument allows user to set values for bin partitioning. If user enters –ugcbin 20 then the program creates and populates 5 bins with range 0-20, 20-40, 40-60, 60-80 and 80-100, whereas if 15 is supplied then the program forms 7 bins with range 0-15, 15-30, 30-45, 45-60, 60-75, 75-90, 90-100. –pbin argument is required to run –ugcbin.

python ConCat-Core.py –GC –pbin –ugcbin 15