**Guidelines for Package 'FuzzyID2'**

**July 29, 2017**

**Title** A software package for large dataset species identification via DNA barcodes, using Hidden Markov Models and fuzzy set methods.

**Description** A Python pipeline for species identification using DNA barcodes, includinganimal COI barcodes, plant barcodes, eDNA, and metabarcoding.

**Version** 2.0

**License** GPL-2

**Authors** Zhi-yong SHI, Cai-qing YANG, Mengdi-HAO and Ai-bing ZHANG

**Maintainer** Zhi-yong SHI and Ai-bing ZHANG<zhiyong-shi@163.com; zhangab2008@mail.cnu.edu.cn>

**Citation** Researchers using FuzzyID2 in a published paper should cite the following article: Shi ZY, Yang CQ, Hao MD, and Zhang AB (2017). A software package for large dataset species identification via DNA barcodes, using Hidden Markov Models and fuzzy set methods. Molecular Ecology and Resources. xxx.

**Introduction**

FuzzyID2 is a software package for species identification via DNA barcoding. It is an extension of a fuzzy-set-theory-based approach (Zhang *et al.* 2012) that can handle large DNA barcoding datasets.

FuzzyID2 performs a two-step searching strategy that incorporates a Hidden Markov Model (HMM) algorithm (HMM; Yada 1996; Eddy 1998) and a fuzzy membership function calculation (Zhang *et al.* 2012). Hidden Markov Models are firstly used to narrow the searching scope from the whole reference library to a much smaller sub-dataset, and then K2P and 1NN algorithms (Kimura 1980; Kim *et al*. 2010; Collins *et al*. 2012) are used to find the nearest-neighbor as the best hit. If preferred, F81 or GTR distances (Felsenstein 1981; Tavare 1986) can be used instead of K2P. Once the potential species is found from the dataset, its fuzzy membership value is calculated to give a confidence level of species assignment (Zhang *et al*. 2012).

FuzzyID2 is programmed using C++ and Python, and runs under the Linux operating system. Reference datasets are stored in the database management system SQlite3 (http://www.sqlite.org). The HMMER 3.0 software suite (Finn *et al.* 2011; Eddy 2011) is utilized to improve processing speed, and Bio++ v2.1.0 suite libraries (Dutheil *et al.* 2006) are embedded for sequence alignments and K2P (or F81 or GTR) distance analysis. The running environment of this approach was a CentOS6.4 system with a C++ program compiled by G++4.4.7. The hardware environment included an Intel(R) Xeon(R) E5620@ 2.40GHz CPU with 16 cores and 32 threads, and a server with 32G memory. We have compiled the software on 32-bit and 64-bit linux distributions. FuzzyID2 is free software released under the CNU General Public License and is available at https://github.com/zhangab2008/FuzzyID2.git. The FuzzyID2 distribution includes a user manual and a sample dataset. FuzzyID2 runs under a command-line interface.

**Installation and Compilation:**

**1. Installation**

1. Upload the FuzzyID2 software package “FuzzyID2.tar.gz” to the linux server.
2. Open the linux terminal console in the FuzzyID2 software direction.
3. Type “tar zxvf FuzzyID2.tar.gz” to decompress the software package, and then type “cd FuzzyID2” to change direction.

**2. Compilation**

1) Compile command for 32bit:

g++ main.cpp mainFunction.cpp commonFunction.cpp -o FuzzyID2\_i686 -lsqlite3 -lbpp-core -lbpp-seq -lbpp-phyl -Wl,-rpath ./lib -L ./lib -I ./include

ln -s FuzzyID2\_i686 FuzzyID2

2) Compile command for 64bit:

g++ main.cpp mainFunction.cpp commonFunction.cpp -o FuzzyID2\_x86\_64 -lsqlite3 -lbpp-core -lbpp-seq -lbpp-phyl -Wl,-rpath ./lib64 -L ./lib64 -I ./include

ln -s FuzzyID2\_x86\_64 FuzzyID2

**Usage:**

**1. Option**

-c Specify an operational mode to run.

There are three operational modes to run in FuzzyID2.

-c Theta1 Calculate the maximum intraspecific genetic distance of each species which is contained in the reference database.

Example: ./FuzzyID2 -c Theta1 -m K2P -d Lepidoptera

-c Theta2 Calculate the minimum interspecific genetic distance between each species and its nearest-neighbor in the reference database.

Example: ./FuzzyID2 -c Theta2 -m K2P -d Lepidoptera

-c ID Calculate the pairwise genetic distance between the query sequence and reference barcodes. The nearest-neighbor barcode from reference dataset with the smallest genetic distance to the query sequence will become the final hit.

Example: ./FuzzyID2 -c ID -m K2P -d Lepidoptera -in query.fas -out outPut.csv

-d Specify a reference database.

-in Specify a query sequence file name.

The sequence file should be in *fasta* format.

-m Specify a nucleotide substitution model.

There are three nucleotide substitution models available in FuzzyID2: K2P (default), JC69, GTR and P(p-distance), and should be consistent during reference Database (DB) construction and subsequent species identification.

-out Specify an output file name for species identification.

The result file is provided in CSV format.

-rb set the marker in the rough search step

-mb set the marker in the precise search step

\* Parameters rb and mb can only be applied to plant barcodes.

**2. Construct the local reference database**

The reference barcode datasets should be provided in *fasta* format or BOLD’s *tsv* format, and the file name should contain the marker name (such as Bees\_COI.fas). It will be stored in a sqlite3 database by a python script (FuzzyID2\_makeDB.py), together with a config file (config.txt, contains parameters used for reference database construction), and the local reference database and the Hmm database will be built automatically.

The sequence name of a reference barcode in the *fasta* format file is required as “>sequenceID\_familyName\_genusName\_speciesName”.

1. Upload the reference barcode file to the linux server.
2. Open the linux terminal console in the FuzzyID2 directory.
3. Type “python3 FuzzyID2\_makeDB.py” command in the console and press Enter.
4. Input the config file name such as “config.txt” and press Enter to start.

**3. Parameters for reference database construction (config.txt)**

To make things easier, a configure file to control the running of the program is provided. The meanings of the parameters are listed below:

(1) refDBName=Bee #reference database name

(2) refMarker=COI #input dataset barcode marker, split with '|' when using multiple markers (such as metabarcodes or plant references, e.g. 'refMarker=rbcL|trnL|ITS2')

(3) refFileFormat=fasta #input reference dataset file format (tsv or fasta, tsv files can be downloaded from BOLD)

(4) refFileName=./datasets/Bee/Bee\_COI.fas #input reference dataset file path and name, split with '|' when using multiple markers, refFileName should contain the marker name.

(5) refRoughMarker= #barcode marker used in the rough scan step, to be set only under the plant barcode scenario.

(6) refMDMarker= #barcode marker used in the precise search step, to be set only under the plant barcode scenario.

(7) roughMaxNum=20 #maximum barcode number for each species to form the rough scan database, based on the reference dataset.

(8) roughMinNum=5 #minimum barcode number for each species, if the barcode number for a species is less than this number, then all barcodes from it will be used to form the rough scan database.

(9) roughPercent=0.3 #percent of barcodes of each species forming the rough scan database.

(10) deleteRepeatSeq=off #reserve only one sequence if barcode sequences are consistent (off or on), for use with a very large dataset.

(11) seqClean=on #replace '-','N' to '', to filter out a sequence whose length is not in the range (Max\_length\_DNA, Min\_length\_DNA) (off or on).

(12) Max\_length\_DNA=1000 #barcode length check, based on refMarker

(13) Min\_length\_DNA=30 #barcode length check, based on refMarker

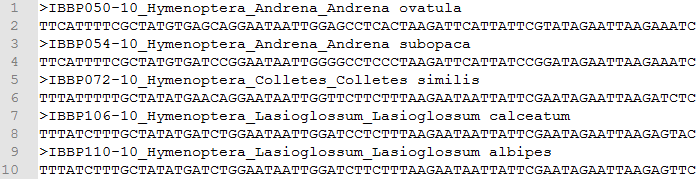
(14) estimateFuzzyPara=on #estimate parameters theta1 and theta2 for each species during database construction step (on) or after ID step (off, used for large datasets).

(15) substitutionModel=K2P #used for genetic distance estimation (K2P, JC69, GTR, P) (working when estimateFuzzyPara=on)

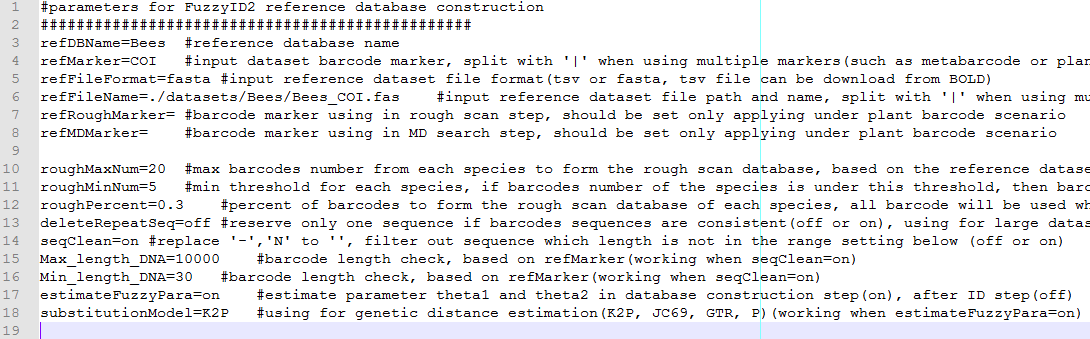
**Examples:**

**1. Traditional single DNA barcode identification:**

1. The reference dataset file “Bees\_COI.fas” is uploaded to the linux server and stored in the folder “datasets” in FuzzyID2 directory. The reference dataset file is prepared in fasta format with the requisite ID. “>sequenceID\_family\_genus\_species” as its comment line.



1. Prepare a config file (config.txt)



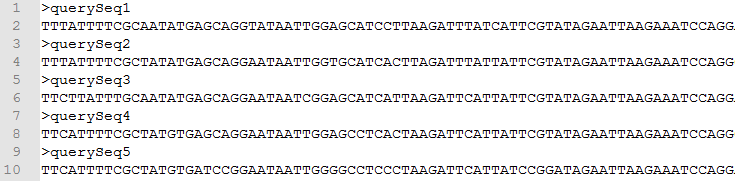
1. In the FuzzyID2 directory, a Python script “FuzzyID2\_makeDB.py” is used to build the local reference database.

Command: python3 FuzzyID2\_makeDB.py

Input config file name (such as config.txt)

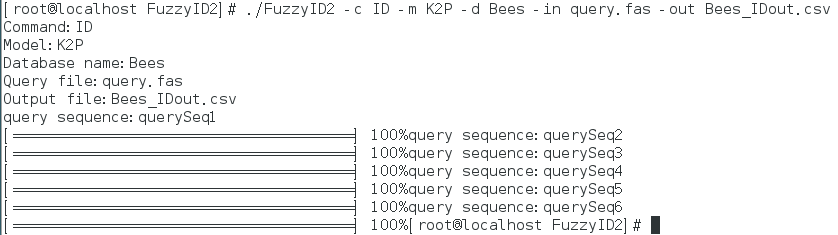


1. The file of query sequences, “query.fas”, is uploaded to the linux server and stored in the FuzzyID2 directory.

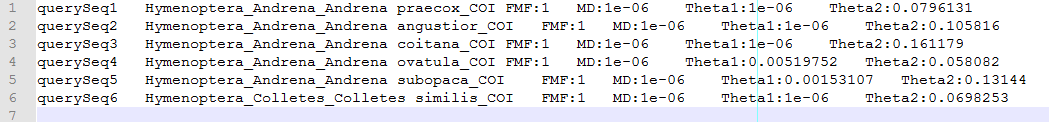


1. Barcode identification

Command: ./FuzzyID2 -c ID -m K2P -d Bees -in query.fas -out Bees\_IDout.csv



1. Output of running “Bees\_IDout.csv” is as follows:

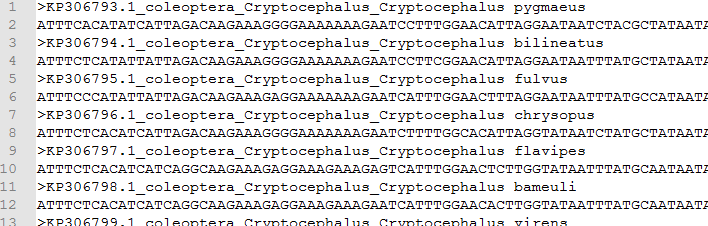


The column names are: query sequence ID, assigned taxonomy of the query sequence, fuzzy membership value, minimum genetic distance between query sequence and assigned species, maximum intraspecific genetic distance of the assigned species, and minimum interspecific genetic distance of the assigned species.

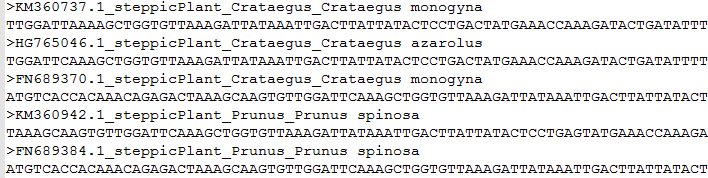
**2. Metabarcodes:**

1. The reference dataset files “ref\_COI.fas”, “ref\_rbcL.fas” and “ref\_trnL.fas” are uploaded to the linux server and retained under the folder “datasets” in the FuzzyID2 directory.

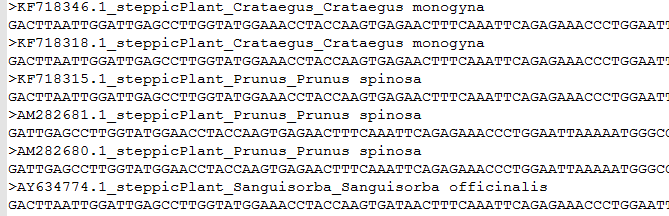
Ref\_COI.fas



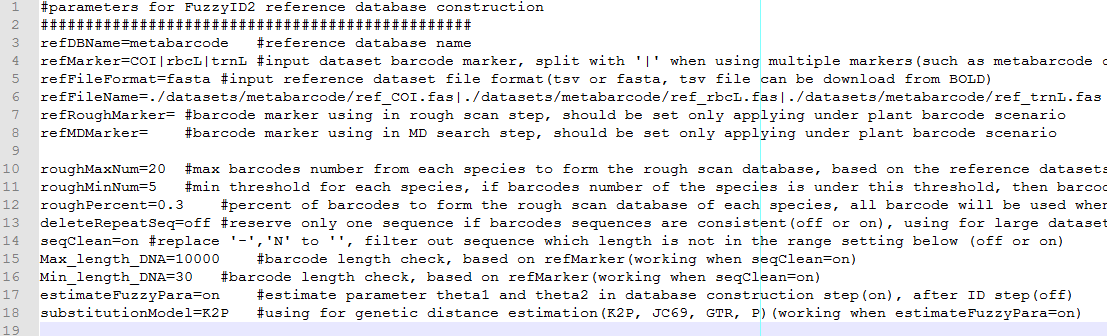
Ref\_rbcL.fas



Ref\_trnL.fas



1. Prepare a config file (config.txt)



1. In the FuzzyID2 directory, the Python script “FuzzyID2\_makeDB.py” is used to build the local reference database.

Command: python3 FuzzyID2\_makeDB.py

Input config file name (such as config.txt)

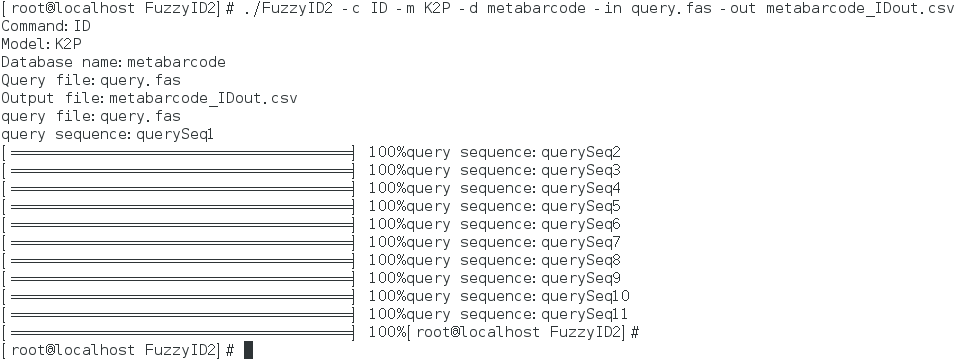


1. Upload the query sequence file “query.fas” to the linux server and save in the FuzzyID2 directory.

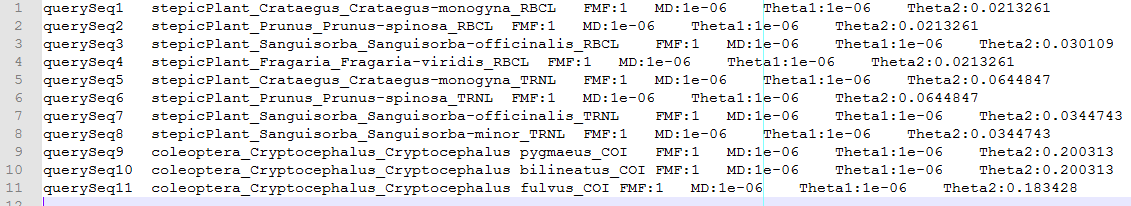


1. Barcode identification

Command: ./FuzzyID2 -c ID -m K2P -d metabarcode -in query.fas -out metabarcode\_IDout.csv



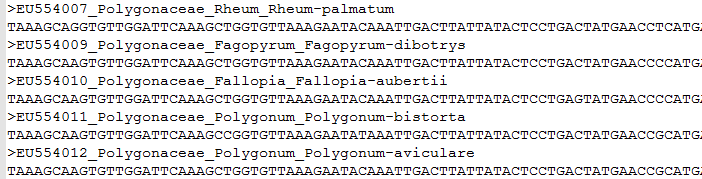
1. Output is presented in the file “metabarcode\_IDout.csv”, shown as following.



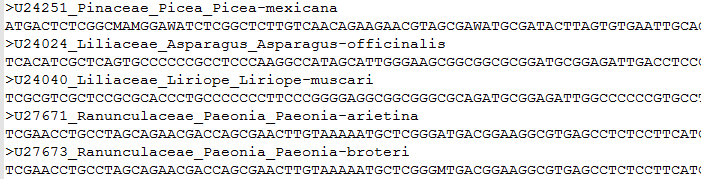
**3 Plant barcodes:**

1. Upload the reference dataset file “Plant\_rbcL.fas” and “Plant\_its.fas” to the linux server and store in the folder “datasets” in the FuzzyID2 directory.

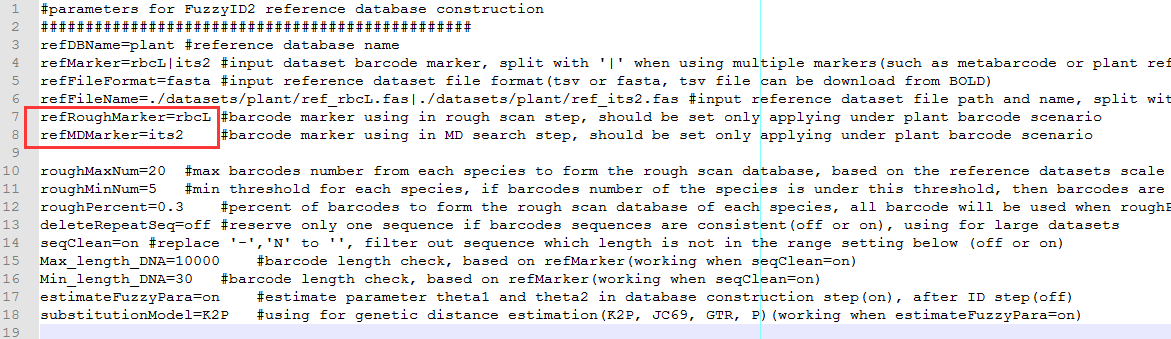
Plant\_rbcL.fas



Plant\_its.fas



1. Prepare the config file (config\_plant.txt)



1. In the FuzzyID2 directory, the Python script “FuzzyID2\_makeDB.py” is used to build the local reference database.

Command: python3 FuzzyID2\_makeDB.py

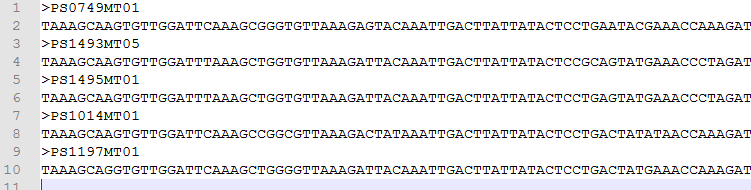
Input config file name “config\_plant.txt”



1. Upload the query sequence files “query\_rbcL.fas” and “query\_its.fas” to the linux server and save in the FuzzyID2 directory.

Note: The sequence ID should be a one-to-one correspondence with the two marker datasets. The marker *rbcL,* withitsslow evolutionary rate, is used to reduce the search scope from the whole reference dataset to the genus level, and then a marker with rapid evolutionary rate, such as *its,* is used to calculate pair-wise genetic distances to find the most similar reference sequence as the final hit at that genus.

Query\_rbcL.fas



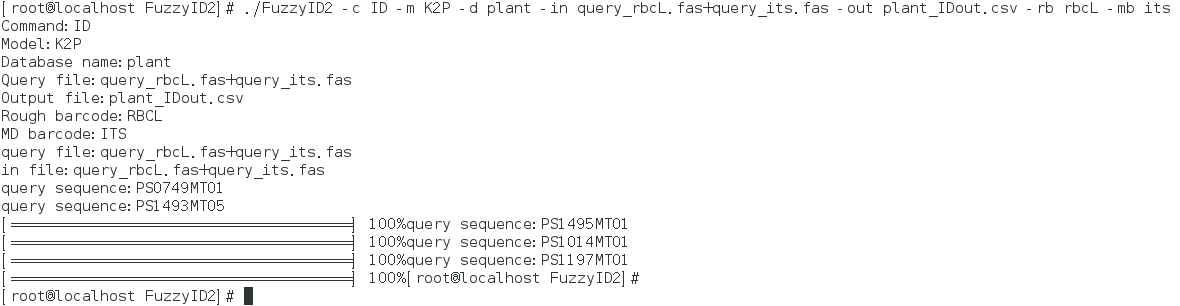
Query\_its.fas



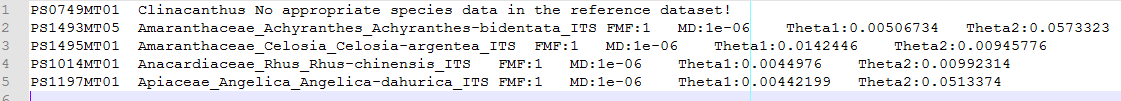
1. Identify the query sequence.

Command: ./FuzzyID2 -c ID -m K2P -d plant -in query\_rbcL.fas+query\_its.fas -out plant\_IDout.csv -rb rbcL -mb its

Note: the two query files are split with ‘+’, and parameters ‘rb’ and ‘mb’ should be specified.



1. Output run on “plant\_IDout.csv” is shown as follows:



**Reference**

Collins RA, Boykin LM, Cruickshank RH, Armstrong KF (2012) Barcoding’s next top model: an evaluation of nucleotide substitution models for specimen identification. *Methods in Ecology and Evolution*, **3**, 457-465.

Dutheil J, Gaillard S, Bazin E, Glemin S, Ranwez B, Galtier N, Belkhir K (2006) Bio++: a set of C++ libraries for sequence analysis, phylogenetics, molecular evolution and population genetics. *BMC Bioinformatics*, **7**, 188-193.

Eddy SR (1998) Profile hidden Markov models. *Bioinformatics Review*, **14**, 755-763.

Eddy SR (2011) Accelerated profile HMM searches. *PLoS Computational Biology*, **7**, e1002195.

Felsenstein J (1981) Evolutionary trees from DNA sequences: a maximum likelihood approach. *Journal of Molecular Evolution*, **17**, 368-376.

Finn RD, Clements J, Eddy SR (2011) HMMER web server: interactive sequence similarity searching. *Nucleic Acids Research*, **39**, W29-37.

Kim S, Eo HS, Koo H, Choi JK, Kim W (2010) DNA barcode-based molecular identification system for fish species. *Molecules and Cells*, **30**, 507-512.

Kimura M (1980) A simple method for estimating evolutionary rates of base substitutions through comparative studies of nucleotide sequences. *Journal of Molecular Evolution*, **16**, 111-120.

Tavare S (1986) Some probabilistic and statistical problems in the analysis of DNA sequences. *Lectures on Mathematics in the Life Sciences*, **17**, 57-86.

Yada T, Ishikawa M, Tanaka H, Asai K (1996) Extraction of hidden Markov model representations of signal patterns in DNA sequences. *Pacific Symposium on Biocomputing*, 686-696.

Zhang AB, Muster C, Liang HB, Zhu CD, Crozier R, Wan P, Feng J, Ward RD (2012) A fuzzy-set-theory-based approach to analyse species membership in DNA barcoding. *Molecular Ecology*, **21**, 1848-1863.

Note: The guideline for BOLD-downloaded sequences cleaning: (1) filter out sequences that are inconsistent within the dataset. For example, filter out “COI-3p” sequences from dataset of “COI-5p”. (2) filter out sequences which are without taxon information (for instance, no genus name). (3) remove gaps (but IUPAC codes were kept). (4) formart the comment like the following “>sequenceID\_family\_genus\_species” format.