MolD

Manual

08.07.2020

Authors Alexander Fedosov <fedosovalexander@gmail.com>

Citations:

Fedosov A.E., Achaz G., Puillandre N. 2019. Revisiting use of DNA characters in taxonomy with MolD - a tree independent algorithm to retrieve diagnostic nucleotide characters from monolocus datasets. *BioRxiv*. DOI: 10.1101/838151.

Disclaimer

Copyright (C) Alexander Fedosov, Guillaume Achaz, Nicolas Puillandre. This program is free software: you can redistribute it and/or modify it under the terms of the GNU General Public License as published by the Free Software Foundation, either version 3 of the License, or (at your option) any later version. This program is distributed in the hope that it will be useful, but WITHOUT ANY WARRANTY; without even the implied warranty of MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the GNU General Public License for more details.

Operating systems

Mac OSX, Windows, and Linux are supported

System requirements

Python 2.7 installed

Input

DATAFILE

The input file is in text (or any compatible) format, in which each line corresponds to one sequenced specimen and contains three space-separated records:

1. species name,
2. taxon name of the query level (in the example below correspond to genus level),
3. nucleotide sequence (with all the sequences aligned across the data set).

The names of the taxa to be diagnosed correspond to the **second** column.

EXAMPLE

Genera of the family Conidae (Gastropoda) – Puilllandre et al. 2014

```
Conasprella_alisi Conasprella TATAAGATTTTGGCTTTTACCTCCTGCCCTTCTTTTACTCCTTTCTTCAGCT
Conasprella_alisi Conasprella TATAAGATTTTGGCTTTTACCTCCTGCCCTTCTTTTACTCCTTTCTTCAGCT
Conasprella_alisi Conasprella TATAAGATTTTGGCTTTTACCTCCTGCTCTTCTTTTACTCCTTTCTTCAGCT
Conasprella_alisi Conasprella TATAAGATTTTGGCTTTTACCTCCTGCCCTTCTTTTACTCCTTTCTTCAGCT
Conasprella_alisi Conasprella TATAAGATTTTGGCTTTTACCTCCTGCCCTTCTTTTACTCCTTTCTTCAGCT
Conasprella_baileyi Conasprella TATAAGATTTTGACTTTTGCCTCCGGCCCTTCTTTTACTTCTTTCTTCAGCC
Conasprella_baileyi Conasprella TATAAGATTTTGACTTTTGCCCCCGGCCCTTCTTTTACTTCTTTCTTCAGCC
Conasprella_boholensis Conasprella TATAAGATTTTGACTTTTACCTCCTGCGCTTCTTTTACTTCTTTCTTCAGCT
Conasprella_boucheti Conasprella TATAAGATTTTGACTTTTACCTCCCGCACTTCTTTTACTTCTTTCTTCAGCT
Conasprella_comatosa Conasprella TATAAGATTTTGACTTTTACCTCCTGCGTTGCTTCTACTCTTATCTTCAGCT
Conasprella_coriolisi Conasprella TATAAGATTTTGACTTTTACCCCCTGCGTTGCTTCTACTCCTATCTTCAGCT
Conus_adamsonii Conus TATGAGTTTTTGGCTTCTTCCTCCTGCGCTTTTACTCCTTCTGTCTTCGGCT
Conus_variegatus Conus TATAAGTTTCTGGCTTCTTCCTCCTGCACTTTTACTTCTTTTATCATCAGCT
Conus_anemone Conus TATAAGTTTTTGGCTTCTTCCTCCTGCTTTGTTGCTTCTCTTATCGTCTGCT
Lilliconus_sagei Lilliconus TATAAGCTTCTGACTTTTACCTCCTGCTTTATTACTTTTATTGTCTTCTGCT
Lilliconus_sagei Lilliconus TATAAGCTTCTGACTTTTACCTCCTGCTTTATTACTTTTATTGTCTTCTGCT
Profundiconus_barazeri Profundiconus CATGAGTTTTTGATTATTACCTCCTGCTTTATTACTTTTGTTATCATCAGCT
Profundiconus_maribelae Profundiconus TATAAGCTTTTGATTATTACCTCCTGCTTTATTACTTTTATTATCATCAGCT
Profundiconus_loyaltiensis Profundiconus TATAAGTTTCTGGTTATTACCTCCTGCTTTATTGCTTTTATTATCCTCAGCT
Profundiconus_neocaledonicus Profundiconus CATAAGCTTTTGACTATTACCTCCTGCTTTATTACTTTTATTATCATCAGCT
Profundiconus_neocaledonicus Profundiconus CATAAGCTTTTGACTATTACCTCCTGCTTTATTACTTTTATTATCATCAGCT
Profundiconus_neocaledonicus Profundiconus CATAAGCTTTTGACTATTACCTCCTGCTTTATTACTTTTATTATCATCAGCT
Pygmaeconus_traillii Pygmaeconus TATAAGTTTTTGGCTTTTACCTCCTTCTCTTTTATTGCTTTTAGCATCTGCT
Californiconus_californicus Californiconus TATAAGCTTTTGACTTTTACCCCCTGCTTTGTTATTACTTCTATCATCAGCT
```

PARAMETERS FILE

Contains all the parameters that should be provided after ‘=‘ without a space.

1. INPUT / OUTPUT

-INPUT_FILE – input alignment file with complete path.
-OUTPUT_FILE – name of the output file with complete path

2. TAXON PARAMETERS (NO DEFAULTS - no parameters entered will lead to an error).

qTAXA (**Focus taxa**)
Arguments:

`[Taxon1,Taxon2,Taxon3]`	A comma separated list of taxa to be diagnosed in square brackets and without spaces.
`ALL`	if all taxa in the dataset are to be diagnosed.
`>N`	if all taxa with more than N sequences available (where N is a natural number) to be diagnosed.

`Taxon_rank` **(Taxon rank)**:
Arguments:

1	for species
2	for supraspecific taxa

`Code gaps as characters:`

`Yes`	dashes ('-') in the alignment are transformed into 'D', which is treated as an independent characters
`No`	dashes are treated as missing data ('N')

3. ADVANCED PARAMETERS FOR pDNC RECOVERY

For explanation see '*Review of MolD*' below or *Fedosov et al. 2019*. If you don't want to set them, don't enter anything after '=', and the defaults will be used.

`Cutoff`	-integer, denoting a (default **100**) denoting number of informative positions to be considered, or -integer prepended by ('>') denoting cut-off value (see below). If this option selected, all informative positions with cut-off value above specified will be considered.
`NumberN`	Number of ambiguously called nucleotides allowed, integer (default **5**).
`Number_of_iterations`	Number of recursions of MolD, integer (default **10000**).
`MaxLen1`	Maximum length of raw pDNCs, integer (default **12**).
`MaxLen2`	Maximum length of refined pDNCs, integer (default **7**).

4. PARAMETERS OF ARTIFICIAL DATASETS (only sDNSs).

`Pdiff`	Percent difference between original and modified sequence, integer (default **1** for species-level taxa, **3** for for supraspecific taxa).
`NmaxSeq`	Max number of sequences per taxon to modify. Integer (default 10)

`Scoring` Sets threshold of sDNC robustness scoring (default stringent). 100 artificial datasets are created to score the sDNC. If the sDNC remains diagnostic in requested (defined by value of threshold), or higher number of artificial datasets in **TWO** consequtive runs, the sDNC is output.

Arguments:

`lousy`	66
`moderate`	75
`stringent`	90
`very_stringent`	95

Run

Run in terminal:

```
python /PATH_TO/MolD_sDNC_20-07.py –i /PATH_TO/MolD_parameters
```

Review of the MolD

(For term definition and theoretical background see:

Fedosov A.E., Achaz G., Puillandre N. 2019. Revisiting use of DNA characters in taxonomy with MolD - a tree independent algorithm to retrieve diagnostic nucleotide characters from monolocus datasets. *BioRxiv*. DOI: 10.1101/838151)

The MolD algorithm is divided into five consecutive steps. At **first** step sequences are sorted by taxon (as defined by the column 2 of the input) and the positions conserved within each taxon are identified.

At the **second** step, each of the positions shared by all focus taxon sequences is assigned a ***cut-off*** value, which corresponds to the number of sequences in the alignment with different (from the focus taxon) nucleotide at this position. The positions that are conserved across the entire data set have a minimum cut-off value of 0 (i.e. non-informative). The positions that correspond to type 1 characters (see Fedosov et al. 2019, Fig. 1) and allow to immediately diagnose the focus taxon, have a maximum cut-off value (equal to the total number of non-focus-taxa sequences in the data set). Desired size of this subset (parameter **cutoff,** by default set to 100), or the threshold cut-off value (>N) can be set by user.

The third step contains main functionality of the MolD algorithm implemented in two recursive functions. The **step_reduction_complist** function initiates a raw DNC, and picks up positions from CPP one-by-one in random order. For each picked position the non-focus taxa sequences that differ at this position from the focus taxon sequences are identified and excluded from further comparisons, and the picked position is appended to the raw DNC. Thus, the set of non-focus taxa sequences is reduced step-by-step until its length equals zero, a condition at which the function terminates, and the raw DNC is output. The output raw DNC allows unambiguous differentiation of the focus taxon members in an analyzed data set, but it is usually redundant (i.e. it includes more positions than necessary). The function **step_reduction_complist** is run repeatedly, where the parameter ***Number_of_iterations*** (default 10,000) defines the number of iterations that generate a pool of raw DNCs. The non-identical raw DNCs, each comprising no more than a predefined number of nucleotide positions (parameter ***Maxlen1***, default 12), make up the input of the **RemoveRedundantPositions** function. The latter function removes redundant positions from the raw DNCs by picking and discarding positions in each DNC one-by-one, and each time checking whether the thus shortened combination remains diagnostic for the focus taxon or not. The non-identical refined DNCs are retained, if their length is equal to or less than a pre-defined threshold value (parameter ***Maxlen2***, default 7). Each of the refined DNCs therefore defines a minimal and sufficient condition for a nucleotide sequence (and a corresponding specimen) to belong to the focus taxon (pDNCs). The steps 3 is executed 5 times to overcome random effect, and to ensure thorough sampling of the informative positions.

Two pDNCs may overlap by one or several nucleotide positions, or share no positions; in the latter case the two pDNCs are termed independent pDNCs (see Fedosov et al. 2019). In the case that all identified pDNCs share one or more nucleotide positions (i.e. no independent combinations are identified), such position(s) present in all pDNCs are termed key positions. The key position(s) are crucial for diagnosing a taxon, because a substitution at this position even in one sequence attributed to a focus-taxon would immediately make the focus-taxon impossible to diagnose with the selected genetic marker. On the contrary, when *n* independent DNCs are recovered, *n* substitutions would be needed to make a focus taxon undiagnosable; the likelihood of the latter scenario is obviously much lower. At the **fourth** step the set of refined pDNCs is analyzed to output the 25 shortest pDNCs, a set of independent pDNCs, or (if present), key position(s). In the case that no pDNCs were recovered for a pre-defined set of DNA sequences, an exception is raised.

At the **fifth** step the set of pDNCs is converted into the sDNC that fulfills pre-defined requirements of robustness (for detailed explanation of the nature of both see fedosov et al. 2019). The robustness of a DNC is tested by generating 100 artificial DNA sequence datasets (derived from the original dataset, but different from it), and checking in how many of them the tested DNC will remain 1) shared by all focus taxon sequences, and 2) diagnostic for the focus taxon. When the DNC is tested 1 mismatch is allowed in the positions involved in the sDNC in a focus taxon sequences: in this case the sDNC will still be considered valid, if remaining positions constitute a valid DNC. To generate artificial datasets a pre-defined number of random nucleotide substitutions is introduced in nucleotide sequences:

- Only in the nucleotide positions that are not conserved across the dataset.
- Without phylogenetic pattern.
- The probability of change to other nucleotide is proportional to the nucleotides frequencies in this position across the dataset. For example, if in the position **P** only ‘**A**’ and ‘**G**’ are present in the dataset, ‘**A**’ in modified sequence will be changed only to ‘**G**’.

Different proportion of sequences is modified in such a manner, depending on the rank of a query taxon, and the number of sequences in the dataset:

- - user defines maximum number of sequences per taxon to be modified (**NMaxSeq**, default = 10)
- - for species, **half** of the sequences in the alignment is modified (***PrSeq* =0.5**), given that **NMaxSeq** is not exceeded, if the total number of records does not exceed **500**. A **quarter** of sequences in the alignment is modified (***PrSeq* =0.25**), given that **NMaxSeq** is not exceeded, if the total number of records exceeds **500.** The number of introduced substitutions is such to make a derived sequence 1% divergent from the original one (***Pdiff*=1**).
- - for genera and higher taxa, only 10% of the sequences are modified (***PrSeq*=0.1**), given that **NMaxSeq** is not exceeded. The number of introduced substitutions is such to make a derived sequence 3% divergent from the original one (***Pdiff*=3**).

Thus each sDNC is scored by assigning a number from 0 (when a DNC failed in all 100 artificial datasets) to 100 (when it worked for all 100). This algorithm is used repeatedly: one informative position is appended to the DNC and then it is scored; if the desired criteria of robustness (see below) are met, the generated sDNC is output; if not, another position is appended and thus the extended DNC is scored again, and so on. If the length of DNC exceeds the arbitrarily selected of 10 positions, and the criteria of robustness are still not fulfilled, either the best scoring sDNC is output with a warning (if the robustness threshold was exceeded), or a message is output to inform a user that no reliable diagnostic combination was recovered.