Measuring Annotated Homology (HAM) and Homology Annotation Conflict (HAC)

User's Guide

By: Nawar Malhis

Installing HAM and HAC:

- Download: `wget orca.msl.ubc.ca/nmshare/ham.tar.gz`
- Unpack: `tar -zxvf ham.tar.gz`
- Install blast+: `sudo apt install ncbi-blast+`

Input data:

Input data should be in an annotated fasta format such that each sequence is annotated with a single line of annotation; for each sequence, we need a fasta header line with a unique accession, a sequence line, and an annotation line. Annotations can include '1', '0', and '-'. Where '-'is used as a mask that is neither '1' nor '0'. The annotated fasta format file can start with lines marked with '#' as comments. Example of a sequence in an annotated fasta format with a single annotation line:

```
>Q86FP9
MKHFAILILAVVASAVVMAYPERDSAKEGNQEQERALHVKVQKRTDGDADYDEYEEDGTTPTPDPTAPTAKPRLRGNKP
00000000000000000000000000000000---------00000000000000000000000000001111100000
```

All input files (datasets) need to be in the same data directory. Our input files `TR2008.af`, `TS2008.af`, `TR2021.af`, `TS2021.af`, `TR2022.af`, and `TS2022.af` are in the data directory '`~/ham_test_data/`'. The data directory can be located anywhere.

Optional parameters for `ham.py` and `hac.py`:

- Identity cut off, default 80%: -ico 80
- Minimum aligned size, default 10: -msz 10
- Number of threads, default 8: -num_threads 8

HAM: Measuring Annotated Homology

This includes two tools, `ham.py` and `ham_mask_homologous.py`. The first, `ham.py`, identifies homologous regions between the two input files. The second, `ham_mask_homologous.py`, masks those regions identified by `ham.py`.

Example:

First, we run `ham.py` for each of our three training/testing datasets to identify shared homologous regions.

```
(py311) ~/Tools/HAM$ python3 ham.py -in1 TS2008.af -in2 TR2008.af -p ~/ham_test_data/
```

A results directory is created inside our data directory, and three files are added:

- `ham-details-TS2008-TR2008.tsv`: includes a list of the one-to-one residue homology between the two input files.

- TR2008-homology-to-TS2008.af: this is the same TR2008.af input file with three extra annotation lines added, H0, shows the '0' annotations of homologous residues in TS2008.af. 'H1' shows the '1' annotations of homologous residues, and 'H-'shows the '-'annotations of homologous residues. Example line:

  ```
  >PDB:1a3b_I
  ITYTDCTESGQDLCLCEGSDVCGKGNKCILGSNGEENQCVTGEGTPKPQSHNDGDFEEIPEEYLQ
  00000000000000000000000000000000000000000000000000000111111111110
  ................000000000000................................
  ..................................................111111...
  ........................--------.............................
  ```

- TS2008-homology-to-TR2008.af: just like with TR2008-homology-to-TS2008.af. This file identifies regions in TS2008.af that are homologous to those in TR2008.af.

```
(py311) ~/Tools/HAM$ python3 ham.py -in1 TS2021.af -in2 TR2021.af -p ~/ham_test_data/
(py311) ~/Tools/HAM$ python3 ham.py -in1 TS2022.af -in2 TR2022.af -p ~/ham_test_data/
```

Then, we run TS2008.af against TR2021.af and TR2022.af.

```
(py311) ~/Tools/HAM$ python3 ham.py -in1 TS2008.af -in2 TR2021.af -p ~/ham_test_data/
(py311) ~/Tools/HAM$ python3 ham.py -in1 TS2008.af -in2 TR2022.af -p ~/ham_test_data/
```

Finally, we run ham_mask_homologous.py to mask all homologous residues.

```
(py311) ~/Tools/HAM$ python3 ham_mask_homologous.py -in TS2008.af -p ~/ham_test_data/
```

ham_mask_homologous reads TS2008.af from the data directory, then search for all files in the results directory that start with "TS2008-homology-to-", process them one at a time by masking TS2008.af residues that are homologous. So, in this example, ham_mask_homologous will consider TS2008-homology-to-TR2008.af, TS2008-homology-to-TR2021.af, and TS2008-homology-to-TR2022.af.

The final masked dataset is saved in the data directory as ham-masked-TS2008.af

However, python3 ham_mask_homologous.py -in TS2021.af -p ~/ham_test_data/ will only consider TS2021-homology-to-TR2021.af

HAC: Homology Annotation Conflict

This includes two tools, hac.py and hac_resolve_conflict.py. The first, hac.py, identifies homologous regions with conflicting annotations between the sequences of the input file. The second, hac_resolve_conflict.py, enables us to resolve these conflicting annotations identified by hac.py by reannotating them with either '1', '0', or '-'(masking).

Example:

Our input files TR2008.af and TS2008.af are in the data directory '~/ham_test_data/'. First, we run hac.py for each input file:

```
(py311) ~/Tools/HAM$ python3 hac.py -in TR2008.af -p ~/ham_test_data/
```

This generates two files:

- hac-details-TR2008.tsv: includes a list of the one-to-one residue homology between the sequences of the input file.

- `TR2008-annotation-conflict.af`: this is the same `TR2008.af` input file with two extra annotation lines added, H0 shows the '0' annotations of the homologous residues in `TR2008.af`. and 'H1' shows the '1' annotations. Example sequence with annotations in the `TR2008-annotation-conflict.af` file:

```
>PDB:1a3b_I
ITYTDCTESGQDLCLCEGSDVCGKGNKCILGSNGEENQCVTGEGTPKPQSHNDGDFEEIPEEYLQ
0000000000000000000000000000000000000000000000000000111111111110
0000000000000000000000000000000000000000000000000000......00000
.....................................................111111111111111
```

To resolve the conflict, we can choose one of three priorities:

```
python3 hac_resolve_conflict.py -in TR2008.af -p ~/ham_test_data/ -pr '01'
```

This converts conflicting annotations of '0' and '1' into '1'. Thus, the updated annotation to the above sequence is:

```
0000000000000000000000000000000000000000000000000000111111111111111
```

And the final dataset with resoved annotations is saved in the data directory as `TR2008-resolved-01.af`

```
python3 hac_resolve_conflict.py -in TR2008.af -p ~/ham_test_data/ -pr '10'
```

This converts conflicting annotations of '0' and '1' into '0'. Thus, the updated annotation to the above sequence is:

```
0000000000000000000000000000000000000000000000000000011111100000
```

And the final dataset with resoved annotations is saved in the data directory as `TR2008-resolved-10.af`

```
python3 hac_resolve_conflict.py -in TR2008.af -p ~/ham_test_data/ -pr '-'
```

This converts conflicting annotations of '0' and '1' into '-'. Thus, the updated annotation to the above sequence is:

```
0000000000000000000000000000000000000000000000000000----111111-----
```

And the final dataset with resoved annotations is saved in the data directory as `TR2008-resolved-masked.af`