HH-suite for sensitive sequence searching based on HMM-HMM alignment

User Guide

Version 2.0.0, January 2012

We are very grateful for bug reports! Please contact us at soeding@genzentrum.lmu.de

Summary

The HH-suite is an open-source software package for sensitive sequence searching based on the pairwise alignment of hidden Markov models (HMMs). It contains HHsearch [1] and HHblits [2] among other programs and utilities. HHsearch takes as input a multiple sequence alignment (MSA) or profile HMM and searches a database of HMMs (e.g. PDB, Pfam, or InterPro) for homologous proteins. HHsearch is often used for protein structure prediction to detect homologous templates and to build highly accurate query-template pairwise alignments for homology modeling. In the CASP9 competition (2010), a fully automated version of HHpred based on HHsearch and HHblits was ranked best out of 81 servers in template-based structure prediction. HHblits can build high-quality MSAs starting from single sequences or from MSAs. It transforms these into a query HMM and, using an iterative search strategy, adds significantly similar sequences from the previous search to the updated query HMM for the next search iteration. Compared to PSI-BLAST, HHblits is faster, up to twice as sensitive and produces more accurate alignments. HHblits uses the same HMM-HMM alignment algorithms as HHsearch, but it employs a fast prefilter that reduces the number of database HMMs for which to perform the slow HMM-HMM comparison from tens of millions to a few thousands.

References:

[1] Söding J. (2005)
Protein homology detection by HMM-HMM comparison.
Bioinformatics **21**, 951-960.

[2] Remmert M., Biegert A., Hauser A., and Söding J. (2011)
HHblits: Lightning-fast iterative protein sequence searching by HMM-HMM alignment.
Nat. Methods, epub Dec 25, doi: 10.1038/NMETH.1818.

Contents

1 Introduction

The HH-suite is an open-source software package for highly sensitive sequence searching and sequence alignment. Its two most important programs are HHsearch and HHblits. Both are based on the pairwise comparison of *profile hidden Markov models* (HMMs).

Profile HMMs are a concise representation of *multiple sequence alignments* (MSAs). Like sequence profiles, they contain for each position in the master sequence the probabilities to observe each of the 20 amino acids in homologous proteins. The amino acid distributions for each column are extrapolated from the homologous sequences in the MSA by adding *pseudocounts* to the amino acid counts observed in the MSA. Unlike sequence profiles, profile HMMs also contain position-specific gap penalties. More precisely, they contain for each position in the master sequence the probability to observe an insertion or a deletion after that position (the log of which corresponds to gap-open penalties) and the probabilities to extend the insertion or deletion (the log of which corresponds to gap-extend penalties). A profile HMM is thus much better suited than a single sequence to find homologous sequences and calculate accurate alignments. By representing both the query sequence and the database sequences by profile HMMs, HHsearch and HHblits are more sensitive for detecting remotely homologous proteins than methods based on pairwise sequence comparison or profile-sequence comparison.

HHblits can build high-quality multiple sequence alignments (MSAs) starting from a single sequence or from an MSA. Compared to PSI-BLAST, HHblits is faster, finds up to two times more homologous proteins and produces more accurate alignments. It uses an iterative search strategy, adding sequences from significantly similar database HMMs from a previous search iteration to the query HMM for the next search. Because HHblits is based on the pairwise alignment of profile HMMs, it needs its own type of databases that contain multiple sequence alignments and the corresponding profile HMMs instead of single sequences. The HHblits databases uniprot20 and nr20 are generated regularly by clustering the UniProt database from EBI and the nonredundant (nr) database from the NCBI into groups of similar sequences alignable over at least 80 % of their length and down to $\sim 20\%$ pairwise sequence identity. These databases can be downloaded together with HHblits. HHblits uses the HMM-HMM alignment algorithms in HHsearch, but it employs a fast prefilter that reduces the number of database HMMs for which to perform the slow HMM-HMM comparison from tens of millions to a few thousands. At the same time, the prefilter is sensitive enough to reduce the sensitivity of HHblits only marginally in comparison to HHsearch.

By generating highly accurate and diverse MSAs, HHblits can improve almost all downstream sequence analysis methods, such as the prediction of secondary and tertiary structure [1, 2], of membrane helices, functionally conserved residues, binding pockets, protein interaction interfaces, or short linear motifs. The accuracy of all these methods depends critically on the accuracy and the diversity of the underlying MSAs, as too few or too similar sequences do not add significant information for the predictions. As an example, running the popular PSIPRED secondary structure prediction program [1] on MSAs generated by HHblits instead of PSI-BLAST improved the accuracy of PSIPRED significantly even without retraining PSIPRED on the HHblits alignments [3].

HHsearch takes as input an MSA (e.g. built by HHblits) or a profile HMM and searches a database of HMMs for homologous proteins. Many HHsearch databases can be downloaded (see next section). The pdb70 database, for instance, consists of profile HMMs for a set of representative sequences from the PDB database; the Pfam, InterPro and CDD domain databases are large collections of curated MSAs and profile HMMs for conserved, functionally annotated domains. HHsearch is often used to predict the domain architectures and the functions of domains in proteins by finding similarities to domains in the pdb70, Pfam, InterPro or other databases.

In addition to the command line package described here, two interactive web servers at `http://hhpred.tuebingen.mpg.de` [4, 5] and `http://hhblits.genzentrum.lmu.de` run HHsearch and

HHblits. They offer extended functionality, such as Jalview applets for checking query and template alignments, histogram views of alignments, and building 3D models with MODELLER.

In the CASP9 competition (Critical Assessment of Techniques for Protein Structure Prediction) in 2010, a fully automated version of HHpred based on HHsearch and HHblits was ranked best out of the 81 servers in template-based structure prediction, the category most relevant for biological applications, while having an average response time of minutes instead of days like most other servers [6] (http://predictioncenter.org/casp9/groups_analysis.cgi?type=server&tbm=on).

2 Installation of the HH-suite and its databases

The HH-suite source code, executable RPM and DPKG packages for most Linux 64 bit platforms, MAC OS X, and BSD Unix, utility scripts in Perl, and databases for HHblits and HHsearch can be downloaded at

```
ftp://toolkit.genzentrum.lmu.de/HH-suite/
```

2.1 Installation

Installation under Linux from source code

1. Downloading: Download the sources from ftp://toolkit.genzentrum.lmu.de/HH-suite/, for example

   ```
   $ mkdir ~/programs/hh/
   $ cd ~/programs/hh/
   $ wget ftp://toolkit.genzentrum.lmu.de/HH-suite/hhsuite_latest.tar.gz
   ```

2. Then unzip and untar the file

   ```
   $ tar -xzvf hhsuite_latest.tar
   ```

3. Compilation: Run the Makefile in `src/`:

   ```
   $ cd src/
   $ make
   ```

   This compiles all programs and creates the binaries in `src/`. Binaries are by default static. If you encounter missing library errors, also make sure you have the static versions installed, e.g. glibc-static.

4. Installation: Either install in current directory:

   ```
   $ make install
   ```

   Or set `INSTALL_DIR` to the base directory (absolute path), e.g. `/usr/local`, where you want to install:

   ```
   $ make install INSTALL_DIR=/usr/local
   ```

5. Set HHLIB: In your shell, set the environment variable HHLIB to `$INSTALL_DIR/lib/hh`, e.g (for bash, zsh, ksh):

   ```
   $ export HHLIB=/usr/local/lib/hh
   ```

   HHsearch and HHblits look for the column state library file `cs219.lib` and the context library file `context_data.lib` in `$HHLIB/data/`. The hh-suite perl scripts also read HHLIB to locate

   Put the location of your binaries into your path, e.g.

   ```
   $ export PATH=$PATH:<install_dir/bin>:$HHLIB/scripts
   ```

   To save you typing these commands every time you open a new shell, you can also add the following lines to the `.profile` or `.bashrc` file in your home directory:

   ```
   export HHLIB=/usr/local/lib/hh
   PATH=$PATH:<install_dir/bin>:$HHLIB/scripts
   alias hhblits='hhblits -d <path/uniprot20 or path/nr20>'
   ```

   The last line defines default parameters for hhblits.

6. Perl paths: Specify paths to BLAST, PSIPRED, PDB, DSSP etc. in `hh/scripts/HHPaths.pm` They are read by the perl scripts in hh-suite.

Installation under x86 64bit Linux with the red hat package manager RPM'

If you use a RPM based distribution like Scientific Linux (SL), Red Hat Enterprise Linux (RHEL) or CentOS we provide precompiled x86_64 packages for Version 6.x, which might also work on Version 5.x and other RPM based distros like SuSE.

1. Run the RPM package:

   ```
   > rpm -hvU ftp://toolkit.genzentrum.lmu.de/HH-suite/hh-suite-latest.rpm
   ```

Then follow instructions under point 5 and 6 of the first subsection.

Installation under x86 64bit Linux with the Debian package manager DPKG

To be written.

Installation under x86 64bit Max OS X

To be written.

Installation under x86 64bit BSD Unix

To be written.

2.2 HHblits databases

To build MSAs using iterative HHblits searches, you need either the uniprot20 or the nr20 HHblits database. Both yield MSAs of equivalent quality and diversity, so which one you should choose depends on what sequence annotation and name formats you prefer. The pdb70 and scop70 databases are not suited for iterative searches as they do not cover the entire sequence space. They are useful as an alternatives to the versions formatted for HHsearch (next subsection), because HHblits is almost as sensitive as HHsearch while being much faster. The following HHblits databases can be downloaded at `ftp://toolkit.genzentrum.lmu.de/HH-suite/hhblits_dbs/`: **!!!!!! Andy: please set softlinks to previous paths !!!!!**

```
 1 uniprot20    based on UniProt database from EBI, clustered to 20 % seq. identity
 2 nr20         based on nonredundant db from NCBI, clustered to 20% seq. identity
 3 pdb70        representatives from PDB (70% max. sequence identity), updated weekly
 4 scop70       representatives from SCOP (70% max. sequence identity)
 5 pfamA        Pfam A database from Sanger Inst., http://www.sanger.ac.uk/Software/Pfam/
```

The HHblits databases consist of eight files, which all start with the name of the database, followed by different extensions:

```
<dbname>.cs219                column state sequences for prefiltering
<dbname>.cs219.sizes          number of sequences and characters in <dbname>.cs219
<dbname>_hhm_db               packed, concatenated HMM models in HHM format
<dbname>_hhm_db.index         index file for packed HMM model file
<dbname>_hhm_db.index.sizes   number of lines in <dbname>_hhm_db.index
<dbname>_a3m_db               packed, concatenated file with MSAs in A3M format
<dbname>_a3m_db.index         index file for packed MSA file
<dbname>_a3m_db.index.sizes   number of lines in <dbname>_a3m_db.index
```

The last three files are not needed for a single search iteration when no output MSA is requested. To get started, download the uniprot20 or nr20 database files. For example:

```
> cd /home/soeding/hh  % change to hh-suite directory
> mkdir hhblits_dbs; cd hhblits_dbs
> wget ftp://toolkit.genzentrum.lmu.de/HH-suite/hhblits_dbs/uniprot20_<date>.tar.gz‘
> tar -xzvf uniprot20_<date>.tar.gz‘
```

2.3 HHsearch databases

HHsearch databases have a simple, human-readable format. They consist of a single file that is a concatenation of all HMM files in HHsearch/HHblits hhm format. The following HHsearch databases can be downloaded at `ftp://toolkit.genzentrum.lmu.de/HH-suite/hhsearch_dbs/`:

!!!!!! Andy: please set softlinks to previous paths !!!!!

```
 1* pdb70       representatives from PDB (70% max. sequence identity), updated weekly
 2* scop70      representatives from SCOP (70% max. sequence identity)
 3* PfamA       http://www.sanger.ac.uk/Software/Pfam/
 4* SMART       http://smart.embl-heidelberg.de/, downloaded from NCBI site
 5* PfamB       based on ProDom, downloaded from Pfam site
 6* COG         http://www.ncbi.nlm.nih.gov/COG/new/
 7* KOG         http://www.ncbi.nlm.nih.gov/COG/new/
```

```
 8* CD/NCBI     http://www.ncbi.nlm.nih.gov/Structure/cdd/cdd.shtml
 9  Panther     http://www.pantherdb.org/, from InterPro
10  TIGRFAMs    http://tigrblast.tigr.org/web-hmm/, from InterPro
11  PIRSF       http://pir.georgetown.edu/pirsf/, from InterPro
12  Superfamily http://supfam.mrc-lmb.cam.ac.uk/SUPERFAMILY/, from InterPro
13  CATH/Gene3D http://cathwww.biochem.ucl.ac.uk/latest/, from InterPro
```

The eight databases marked by asterisks can be downloaded with HMMs in HHsearch format (*.hhm.tar files) and the multiple sequence alignments (MSAs) in A3M format (*.a3m.tar files). The *.hhm.tar and *.a3m.tar files untar into thousands of separate files, so before unzipping and untarring, first create a directory for the database and untar the tar file within this directory. Under Linux, type

```
> mkdir scop70_1.75
> cd scop70_1.75
> tar -xzvf scop70_1.75.hhm.tar.gz
```

To generate an HHsearch database file, concatenate all *.hhm files:

```
> cat *.hhm > scop70_1.75.hhm
```

For the databases without asterisks, you can download the HMM models in HMMER format as *.hmm.tar files. Each *.hmm.tar file untars into a single concatenated HMME-formatted database file that can be read by HHsearch. For theses databases, unfortunately no alignments are publicly available. (Information to the contrary is welcome!)

The pdb70 and scop70 databases are built at the Gene Center using in-house scripts to select representatives and using three **(!!!!!! Michael, Joern: correct?!!!!!)** iterations of HHblits to generate MSAs for the representative sequences. They are distributed freely under the Lesser Gnu Public License. For the other databases, different copy right regulations may apply. Please refer to the databases' original web sites for the copy right notes and references to cite.

3 Brief tutorial to HH-suite tools

3.1 Overview of programs

```
hhblits         (Iteratively) search an HHblits database with a query sequence or MSA
hhsearch        Search an HHsearch database of HMMs with a query MSA or HMM
hhmake          Build an HMM from an input MSA
hhfilter        Filter an MSA by max sequence identity, coverage, and other criteria
hhalign         Calculate pairwise alignments, dot plots etc. for two HMMs/MSAs

reformat.pl     Reformat one or many MSAs
addss.pl        Add PSIPRED predicted secondary structure to an MSA or HHM file
hhmakemodel.pl  Generate MSAs or coarse 3D models from HHsearch or HHblits results
hhblitsdb.pl    Build HHblits database with prefiltering, packed MSA/HMM, and index files
multithread.pl  Run a command for many files in parallel using multiple threads
Align.pm        Utility package for local and global sequence-sequence alignment
HHPaths.pm      Configuration file with paths to the PDB, BLAST, PSIPRED etc.
```

Call a program without arguments or with -h to get a more detailed description of its syntax.

3.2 Searching databases of HMMs using HHsearch and HHblits

!!!!!!! Andy, does HHsearch find context_data.lib automatically at a hard-coded location? Does HHblits find cs219.lib and context_data.lib automatically at a hard-coded location? Is the .hhdefaults file removed and HHblits still works? !!!!!!!!!!!!

!!!!!!! Andy, we have to put files query.a3m, query.seq, and query.hhm into directory data. We can simply take these from pdb70/1tv9_A.a3m and 1tv9_A.hhm. For the .seq file, we can delete everything from .a3m except the sequence with name ¿1tv9_A. !!!!!!!!!!

We will use the MSA `query.a3m` in the `data/` subdirectory of the HH-suite as an example query. To search for sequences in the `scop70_1.75` database that are homologous to the query sequence or MSA in `query.a3m`, type

```
> hhsearch  -cpu 4 -i data/query.a3m -d hhsearch_dbs/scop70_1.75.hhm
```

If the input file is an MSA or a single sequence, HHsearch calculates an HMM from it and then aligns this query HMM to all HMMs in the `scop70_1.75` database using the Viterbi algorithm. You should see a dot printed for every twenty HMMs aligned. After the search, the most significant HMMs are realigned using the more accurate Maximum Accuracy (MAC) algorithm (subsection 3.5). After the realignment phase, the complete search results consisting of the summary hit list and the pairwise query-template alignments are written to the default output file, `query.hhr` (where the query file extension is replaced with `hhr`). The `hhr` result file format was designed to be human readable and easily parsable.

The `-cpu 4` option tells HHsearch to start four POSIX threads for searching and realignment. This will typically results in almost fourfold faster execution on computers with four or more cores. Since the management of the threads costs negligible overhead, this option could be given by default through an alias definition of hhsearch and hhblits (see section 2.1).

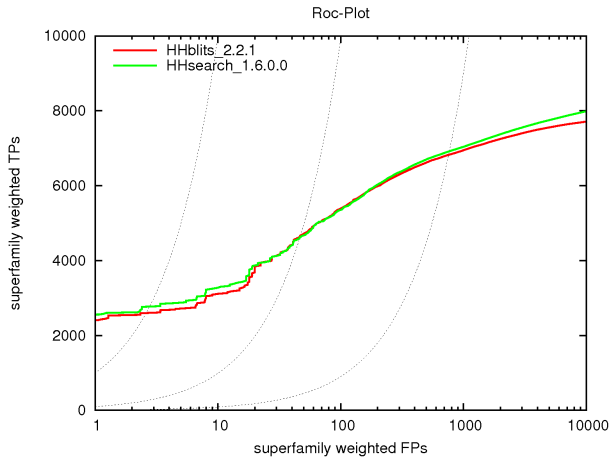

Figure 1: Benchmark of HHsearch and HHblits on a SCOP20 dataset.

The hhblits tool can be used in much the same way as hhsearch. Apart from the different database format, it takes the same input data and produces a results file in the same format as hhsearch. Most of the hhsearch options also work for hhblits, which has additional options associated with its extended functionality for iterative searches. Due to its fast prefilter, hhblits runs between 30 and 3000 times faster than HHsearch at the cost of only a few percent lower sensitivity (Fig. 1).

The same search as above is performed here using hhblits instead of hhsearch:

```
> hhblits  -cpu 4 -i data/query.a3m -d hhblits_dbs/scop70_1.75 -n 1
```

HHblits first scans the column state sequences in `scop70_1.75.cs219` with its fast prefilter. HMMs whose column state sequences pass the prefilter are read from the packed file `scop70_1.75_hhm_db` (using the index file `scop70_1.75_hhm_db.index`) and are aligned to the query HMM generated from `query.a3m` using the slow Viterbi HMM-HMM alignment algorithm. The search results are written to the default output file `query.hhr`. The option `-n 1` tells hhblits to perform a single search iteration. (The default is 2 iterations.)

3.3 Generating a multiple sequence alignment using HHblits

To generate an MSA for a sequence or initial MSA in query.a3m, the database to be searched should cover the entire sequence space, such as uniprot20 or nr20. The option `-oa3m <msa_file>` tells HHblits to generate an output MSA from the significant hits:

```
> hhblits -cpu 4 -i data/query.seq -d hhblits_dbs/uniprot20 -oa3m query.a3m -n 1
```

At the end of the search, HHblits reads from the packed database file containing the MSAs the sequences belonging to HMMs with E-value below the threshold. The E-value threshold for inclusion into the MSA can be specified using the `-e <E-value>` option. After the search, `query.a3m` will contain the MSA in A3M format.

We could do a second search iteration, starting with the MSA from the previous search, to add more sequences. Since the MSA generated after the previous search contains more information than the single sequence in `query.seq`, searching with this MSA will probably result in many more homologous database matches.

```
> hhblits -cpu 4 -i query.a3m -d hhblits_dbs/uniprot20 -oa3m query.a3m -n 1
```

Instead, we could start the search with `query.seq` and directly perform two search iterations, by adding the option `-n 2`:

```
> hhblits -cpu 4 -i data/query.seq -d hhblits_dbs/uniprot20 -oa3m query.a3m -n 2
```

In practice, it is recommended to use between 1 and 4 iterations for building MSAs, depending on the trade-off between reliability and specificity on one side and sensitivity for remotely homologous sequences on the other side. The more search iterations are done, the higher will be the risk of non-homologous sequences or sequence segments entering the MSA and recruiting more of their kind in subsequent iterations. This is particularly problematic when searching with sequences containing short repeats, regions with amino acid compositional bias and, although less dramatic, with multiple domains. Fortunately, this problem is much less pronounced in hhblits as compared to PSI-BLAST due to hhblits's lower number of iterations, its more robust Maximum Accuracy alignment algorithm, and the higher precision of its HMM-HMMs alignments.

The parameter `mact` (maximum accuracy threshold) lets you choose the trade-off between sensitivity and precision. With a low mact-value (e.g. `-mact 0.01`) very sensitive, but not so precise alignments are generated, whereas a search with a high mact-value (e.g. `-mact 0.9`) results in shorter but very precise alignments. The default value of mact in HHblits is 0.35 (changed from 0.5 in the beta version).

To avoid unnecessarily large and diverse MSAs, HHblits stops iterating when the diversity of the query MSA – measured as number of effective sequences, see section 5.1 – grows passed a threshold of 10.0. This threshold can be modified with the `--neffmax <float>` option. See subsection 5.2 for a description of how the number of effective sequences is calculated in HH-suite.

To avoid the final MSAs to grow unnecessarily large, by default the cluster MSAs are filtered with the option `-diff 1000` prior to merging them with the query MSA. (The `-diff 1000` option selects the most representative sequences from an MSA such that every regions is covered by at least 1000 sequences.) To turn off the filtering and obtain all sequences in the significantly similar uniprot20 clusters, use the `-nodiff` option:

```
> hhblits -cpu 4 -i data/query. -d hhblits_dbs/uniprot20 -oa3m query.a3m -nodiff
```

The A3M format uses small letters to mark inserts and capital letters to designate match and delete columns (see subsection 5.1), allowing to omit gaps aligned to insert columns. The A3M format therefore uses much less space for large alignments than FASTA but looks misaligned to the human eye. Use the `reformat.pl` script to reformat `query.a3m` to other formats, e.g. for reformatting the MSA to Clustal and FASTA format, type

```
> reformat.pl a3m clu query.a3m query.clu
> reformat.pl a3m fas query.a3m query.fas
```

Next, to add secondary structure information to the MSA we call the script `addss.pl`. For `addss.pl` to work, you have to make sure that the paths to BLAST and PSIPRED in the `scripts/HHPaths.pm` are correctly filled in. Then type

```
> addss.pl query.a3m
```

When the sequence has a SCOP or PDB identifier as first word in its name, the script tries to add the DSSP states as well. For this to work, the path to the pdb and dssp directories needs to be specified in the perl file `HHPaths.pm`. Open the `query.a3m` file and check out the two lines that have been added to the MSA. Now you can generate a hidden Markov model (HMM) from this MSA:

```
> hhmake -i query.a3m
```

The default output file is `query.hhm`. By default, the option `-M first` will be used. This means that exactly those columns of the MSAs which contain a residue in the query sequence will be assigned to Match / Delete states, the others will be assigned to Insert states. (The query sequence is the first sequence not containing secondary structure information.) Alternatively, you may want to apply the 50%-gap rule by typing `-M 50`, which assigns only those columns to Insert states which contain more than 50% gaps. The `-M first` option makes sense if your alignment can best be viewed as a seed sequence plus aligned homologs to reinforce it with evolutionary information. This is the case in the SCOP and PDB versions of our HMM databases, since here MSAs are built around a single seed sequence (the one with known structure). On the contrary, when your alignment represents an entire family of homologs and no sequence in particular, it is best to use the 50% gap rule. This is the case for Pfam or SMART MSAs, for instance. Despite its simplicity, the 50% gap rule has been shown to perform well in practice.

When calling hhmake, you may also apply several filters, such as maximum pairwise sequence identity (`-id <int>`), minimum sequence identity with query sequence (`-qid <int>`), or minimum

coverage with query (`-cov <int>`). But beware of reducing the diversity of your MSAs too much, as this will lower the sensitivity to detect remote homologs.

Previous versions of HH-suite (the 'HHsearch package') included a perl script `buildali.pl` to build MSAs for a query sequence using PSI-BLAST as its search engine. Because HHblits performs better than PSI-BLAST in all aspects that we have tested, we decided to remove this script from HH-suite. It can still be downloaded as part of HHsearch version 1.5.0.

3.4 Building customized databases

It is simple to build custom databases for HHsearch and HHblits using the same tools we use to build the standard HH-suite databases. An example application is to search for homologs among all proteins of an organism. To build your own HHsearch or HHblits database from a set of sequences, you first need to generate an MSA with predicted secondary structure for every sequence in the set. This can be conveniently done using the script `multithread.pl` in the HH-suite. This script runs a command for many files in parallel, distributing the individual jobs to multiple cores on your server. This will shorten the run-time roughly by the number of cores.

To build MSAs with HHblits from sequences `dbs/scop70_1.75/*.seq`, run

```
> multithread.pl 'dbs/scop70_1.75/*.seq' 'hhblits -i $file -d hhblits_dbs/uniprot20 -oa3m $na
```

The first argument is the file globbing expression, which selects the files with which to run the command given as second argument. In this command, the string `$file` is replaced by the actual, globbed file. You might also want to pipe the stdout and stderr streams of the command into a log-file: `'hhblits -i $file -d hhblits_dbs/uniprot20 1> stdout.log 2>stderr.log'`. The MSAs are written to the file `$name.seq`, which stands for the globbed file name without extension, (`$name`) plus an appended extension `.a3m`. The number of threads launched can be specified with option `-cpu <int>` (default value is 8). To be sure that everything went smoothly, check that the number of `*.a3m` files is the same as the number of `*.seq` files, and browse the file `'stderr.log'` for error messages. The number of HHblits search iterations and the HMM inclusion E-value threshold for hhblits can be changed from their default values (2 and 0.01, respectively) using the `'-n <int>'` and `'-e <float>'` options.

Now, add PSIPRED-predicted secondary structure to all MSAs:

```
> multithread.pl 'dbs/scop70_1.75/*.a3m' 'addss.pl $file' -cpu 16
```

Again, piping stdout and stderr into log files and inspecting the warnings and errors is recommended. From here on, the steps to build an HHblits database differ from those needed to build an HHsearch database.

HHblits databases

An HHblits database consists of eight files, as described in subsection 2.2. These files can be generated by a single call to `hhblitsdb.pl`:

```
> hhblitsdb.pl -o hhblits_dbs/scop70_1.75 -ia3m 'dbs/scop70_1.75' -cpu 16
```

In order to build the file containing the column state sequence for prefiltering, `hhblitsdb.pl` will generate a column state sequence for each MSAs. As this can take some time, the script calls `multithread.pl` to distribute the jobs to multiple cores. The script also needs to generate an

HHM files for each A3M MSA file, for which again `multithread.pl` is called. In the end, the `ffindex` utility is generates the packed files containing the A3M MSAs and HHM models with corresponding index files. The script will report the number of files of each category (column-state, A3M, HHM) and warn if the numbers differ. The option `-log err.log` pipes the stderr stream of each command executed into a log file. As with all perl scripts in the hh-suite, a list of additional options can be retrieved by calling the scripts without parameters.

Alternatively, if you have a set of HHM or HMMER model files, for example for Pfam, you can build an HHblits database with the command

```
> hhblitsdb.pl -o hhblits_dbs/pfamA_25 -ihhm 'dbs/pfamA_25' -hhmext hmm -cpu 16
```

The script will then build the column state sequences from the HMMER files instead of the A3M and will generate five database files. The three files referring to the A3M MSAs can not be built since no MSAs were supplied. However, these file are only needed to build output MSAs from merging the query MSA with the MSAs of significant hits and are therefore dispensable.

HHsearch databases

An HHsearch database consists simply of concatenated hhm files. If you already have an hhblits database, you can simply use the file `<dbname>_hhm_db` as HHsearch database. Otherwise, you may generate the `hhm` files using `multithread.pl`:

```
> multithread.pl 'dbs/scop70_1.75/*.a3m' 'hhmake -i $file' -cpu 8
```

The `hhm` files will have the same name but with a different extension as their `a3m` files. Now you can then concatenate your individual HMMs into your database:

```
> cat *.hhm > scop70_1.75_hhm
```

or, if the maximum command line buffer size is exceeded,

```
> find $tmpdir -name '*.seq219' -exec cat '{}' + >> scop70_1.75_hhm
```

3.5 Maximum Accuracy alignment algorithm

HHblits and HHsearch use a better alignment algorithm than the quick and standard Viterbi method to generate the final HMM-HMM alignments. Both realign all displayed alignments in a second stage using the more accurate Maximum Accuracy (MAC) algorithm [7, 8]. The Viterbi algorithm is employed for searching and ranking the matches. The realignment step is parallelized (`-cpu <int>`) and typically takes a few seconds only.

Please note: Using different alignment algorithms for scoring and aligning has the disadvantage that the pairwise alignments that are displayed are not always very similar to those that are used to calculate the scores. This can lead to confusing results where alignments of only one or a few residues length may have obtained significant E-values. In such cases, run the search again with the `-norealign` option, which will skip the MAC-realignment step. This will allow you to check if the Viterbi alignments are valid at all, which they will probably not be. The length of the MAC alignments can therefore give you additional information to decide if a match is valid. In order to avoid confusion for users of our HHpred server [4, 5], the `-norealign` option is the default there, whereas for you pros who dare to use the command line package, realigning is done by default.

The posterior probability threshold is controlled with the -mact [0,1[option. This parameter controls the alignment algorithm's greediness. More precisely, the MAC algorithm finds the alignment that maximizes the sum of posterior probabilities minus mact for each aligned pair. Global alignments are generated with -mact 0, whereas -mact 0.5 will produce quite conservative local alignments.

The -global and -local options now refer to both the Viterbi search stage as well as the MAC realignment stage. With -global (-local), the posterior probability matrix will be calculated for global (local) alignment. When -global is used in conjunction with -realign, the mact parameter is automatically set to 0 in order to produce global alignments. In other words, both following two commands will give global alignments:

```
> hhsearch -i <query> -d <db.hhm> -realign -mact 0
> hhsearch -i <query> -d <db.hhm> -realign -global
```

The first version uses *local* Viterbi to search and then uses MAC to realign the proteins globally (since mact is 0) on a *local* posterior probability matrix. The second version uses *global* Viterbi to search and then realigns globally (since mact is automatically set to 0) on a *global* posterior matrix. To detect and align remote homologs, for which sometimes only parts of the sequence are conserved, the first version is clearly better. It is also more robust. If you expect to find globally alignable sequence homologs, the second option might be preferable. In that case, it is recommended to run both versions and compare the results.

3.6 How can I verify if a database match is homologous?

Here is a list of things to check if a database match really is at least locally homologous.

- Check probability and E-value: HHsearch and HHblits can detect homologous relationships far beyond the twilight zone, i.e. below 20% sequence identity. Sequence identity is therefore not an appropriate measure of relatedness anymore. The estimated probability of the template to be (at least partly) homologous to your query sequence is the most important criterion to decide whether a template HMM is actually homologous or just a high-scoring chance hit. When it is larger than 95%, say, the homology is nearly certain. Roughly speaking, one should give a hit serious consideration (i.e. check the other points in this list) whenever (1) the hit has $> 50\%$ probability, or (2) it has $> 30\%$ probability and is among the top three hits. The E-value is an alternative measure of statistical significance. It tells you how many chance hits with a score better than this would be expected if the database contained only hits unrelated to the query. At E-values below one, matches start to get marginally significant. Contrary to the probability, when calculating the E-value HHsearch and HHblits do not take into account the secondary structure similarity. Therefore, the probability is a more sensitive measure than the E-value.
- Check if homology is biologically suggestive or at least reasonable: Does the database hit have a function you would expect also for your query? Does it come from an organism that is likely to contain a homolog of your query protein?
- Check secondary structure similarity: If the secondary structure of query and template is very different or you can't see how they could fit together in 3D, then this is a reason to distrust the hit. Note however that if the query alignment contains only a single sequence, the secondary structure prediction is quite unreliable and confidence values are overestimated.
- Check relationship among top hits: If several of the top hits are homologous to each other, (e.g. when they are members of the same SCOP superfamily), then this will considerably reduce

the chances of all of them being chance hits, especially if these related hits are themselves not very similar to each other. Searching the SCOP database is very useful precisely for this reason, since the SCOP family identifier (e.g. a.118.8.2) allows to tell immediately if two templates are likely homologs.

- Check for possible conserved motifs: Most homologous pairs of alignments will have at least one (semi-)conserved motif in common. You can identify such putative (semi-)conserved motifs by the agglomeration of three or more well-matching columns (marked with a '|' sign between the aligned HMMs) occurring within a few residues, as well as by matching consensus sequences. Some false positive hits have decent scores due to a similar amino acid composition of the template. In these cases, the alignments tend to be long and to lack conserved motifs.
- Check residues and role of conserved motifs: If you can identify possible conserved motifs, are the corresponding conserved template residues involved in binding or enzymatic function?
- Check query and template alignments: A corrupted query or template alignment is the main source of high-scoring false positives. The two most common sources of corruption in an alignment are (1) non-homologous sequences, especially repetitive or low-complexity sequences in the alignment, and (2) non-homologous fragments at the ends of the aligned database sequences. Check the query and template MSAs in an alignment viewer such as Jalview or ALNEDIT.
- Realign with other parameters: change the alignment parameters. Choose global instead of local mode, for instance, if you expect your query to be globally homologous to the putative homolog. Try to improve the probability by changing the values for minimum coverage or minimum sequence identity. You can also run the query HMM against other databases.
- Inspect the HHblits results after the first iteration and to include also hits above the E-value threshold of 0.001, based on biological plausibility, relatedness of the organism, reasonable looking alignment, or just based on guessing. Then jump-start HHblits with this manually enriched alignment.
- Try out servers for remote homology detection and structure prediction servers: A list of servers can be found in [6] and [9].
- Verify predictions experimentally: The ultimate confirmation of a homologous relationship or structural model is, of course, the experimental verification of some of its key predictions, such as validating the binding to certain ligands by binding assays, measuring biochemical activity, or comparing the knock-out phenotype with the one obtained when the putative functional residues are mutated.

4 HHsearch/HHblits output: hit list and pairwise alignments

4.1 Summary hit list

Let's do a search with the human PIP49/FAM69B protein, for which we generated an MSA in `query.a3m` with two iterations of HHblits in subsection 3.3:

```
Search results will be written to query.hhr
query.a3m is in A2M, A3M or FASTA format
Read query.a3m with 272 sequences
Alignment in query.a3m contains 431 match states
149 out of 270 sequences passed filter (up to 91% position-dependent max pairwise sequence identity)
Effective number of sequences exp(entropy) = 5.2
```

```
.............................................. 1000 HMMs searched
.............................................. 2000 HMMs searched
.............................................. 3000 HMMs searched
.............................................. 4000 HMMs searched
.............................................. 5000 HMMs searched
.............................................. 6000 HMMs searched
.............................................. 7000 HMMs searched
.............................................. 8000 HMMs searched
.............................................. 9000 HMMs searched
.............................................. 10000 HMMs searched
.............................................. 11000 HMMs searched
.............................................. 12000 HMMs searched
.............................................. 13000 HMMs searched
................................
Realigning 183 query-template alignments with maximum accuracy (MAC) algorithm ...

Query         sp|Q5VUD6|FA69B_HUMAN Protein FAM69B OS=Homo sapiens GN=FAM69B PE=2 SV=3
Match_columns 431
No_of_seqs    149 out of 272
Neff          5.2
Searched_HMMs 13730
Date          Wed Jan  4 17:44:24 2012
Command       /cluster/user/soeding/hh/src/hhsearch -i query.a3m -d /cluster/user/soeding/databases/scop.hhm -cpu 18

 No Hit                             Prob E-value P-value  Score    SS Cols Query HMM  Template HMM
  1 d1qpca_ d.144.1.7 (A:) Lymphoc  99.7 4.5E-17 3.2E-21  154.3  10.2   99  203-320    56-157 (272)
  2 d1jpaa_ d.144.1.7 (A:) ephb2 r  99.7 4.3E-17 3.1E-21  156.8   8.8   99  203-321    75-177 (299)
  3 d1uwha_ d.144.1.7 (A:) B-Raf k  99.7 5.1E-17 3.7E-21  154.8   7.7  100  203-322    52-154 (276)
  4 d1opja_ d.144.1.7 (A:) Abelson  99.7 6.2E-17 4.5E-21  154.8   8.3  100  203-321    61-164 (287)
  5 d1mp8a_ d.144.1.7 (A:) Focal a  99.6 9.9E-17 7.2E-21  151.3   8.6  100  203-322    56-158 (273)
  6 d1sm2a_ d.144.1.7 (A:) Tyrosin  99.6 1.2E-16 8.8E-21  150.3   8.8   99  203-321    48-150 (263)
  7 d1u59a_ d.144.1.7 (A:) Tyrosin  99.6 2.4E-16 1.7E-20  150.9   9.5   99  203-321    57-158 (285)
  8 d1xbba_ d.144.1.7 (A:) Tyrosin  99.6 2.2E-16 1.6E-20  150.2   8.6   97  203-320    56-155 (277)
  9 d1vjya_ d.144.1.7 (A:) Type I   99.6 2.6E-16 1.9E-20  151.3   8.8   98  204-320    46-156 (303)
 10 d1mqba_ d.144.1.7 (A:) epha2 r  99.6 4.4E-16 3.2E-20  148.0   8.7  193  203-422    57-272 (283)
...
 64 d1j7la_ d.144.1.6 (A:) Type II  97.3 0.00014   1E-08   65.0   6.3   33  292-324   184-216 (263)
 65 d1nd4a_ d.144.1.6 (A:) Aminogl  96.7  0.0012 8.5E-08   58.5   6.6   31  292-322   176-206 (255)
 66 d1nw1a_ d.144.1.8 (A:) Choline  96.6  0.0011 7.8E-08   63.9   5.8   37  203-239    92-128 (395)
 67 d2pula1 d.144.1.6 (A:5-396) Me  95.6  0.0071 5.2E-07   58.3   6.4   32  290-322   222-253 (392)
 68 d1a4pa_ a.39.1.2 (A:) Calcycli  91.7    0.12 8.9E-06   40.0   5.4   62  140-202    18-80  (92)
 69 d1ksoa_ a.39.1.2 (A:) Calcycli  91.2    0.17 1.2E-05   39.5   5.8   56  147-203    28-83  (93)
 70 d1e8aa_ a.39.1.2 (A:) Calcycli  90.5    0.23 1.7E-05   38.3   6.0   56  147-203    27-82  (87)
...
175 d1qxpa2 a.39.1.8 (A:515-702) C  23.7      29  0.0021   28.8   3.8   49  137-197    69-118 (188)
176 d1tuza_ a.39.1.7 (A:) Diacylgl  23.5      55   0.004   25.3   5.3   55  143-201    44-106 (118)
177 d1ggwa_ a.39.1.5 (A:) Cdc4p {F  23.1      26  0.0019   27.0   3.2   66  129-197    35-101 (140)
178 d1topa_ a.39.1.5 (A:) Troponin  22.8      72  0.0052   24.5   6.0   58  140-199    65-123 (162)
179 d1otfa_ d.80.1.1 (A:) 4-oxaloc  22.5      66  0.0048   21.5   5.0   40  267-306    12-53  (59)
180 d1oqpa_ a.39.1.5 (A:) Caltract  22.2      32  0.0023   24.0   3.2   32  165-197     3-34  (77)
181 d1df0a1 a.39.1.8 (A:515-700) C  21.7      43  0.0032   27.0   4.5   51  137-199    67-118 (186)
182 d1zfsa1 a.39.1.2 (A:1-93) Calc  21.1      41   0.003   24.6   3.8   30  170-199     8-38  (93)
183 d1snla_ a.39.1.7 (A:) Nucleobi  20.9      23  0.0016   26.2   2.2   24  174-197    18-41  (99)

Done
```

The summary hit list that is written to the screen shows the best hits from the database, ordered by the probability of being a true positive (column 4: 'Prob'). The meaning of the columns is the following:

Column 1 'No':	Index of hit
Column 2 'Hit':	First 30 characters of domain description (from name line of query sequence)

Column 3 'Prob':	Probability of target to be a true positive For the probability of being a true positive, the secondary structure score in column 7 is taken into account, together with the raw score in column 6 ('Score'). True positives are defined to be either globally homologous or they are at least homologous in parts, and thereby locally similar in structure. More precisely, the latter criterion demands that the MAXSUB score between query and hit is at least 0.1. In almost all cases the structural similarity will we be due to a global OR LOCAL homology between query and target.
Column 4 'E-value':	E-value and P-value are calculated without taking the secondary structure into account! The E-value gives the average number of false positives ('wrong hits') with a score better than the one for the target when scanning the database. It is a measure of reliability: E-values near to 0 signify a very reliable hit, an E-value of 10 means about 10 wrong hits are expected to be found in the database with a score at least this good.
Column 5 'P-value':	The P-value is just the E-value divided by the number of sequences in the database. It is the probability that in a PAIRWISE comparison a wrong hit will score at least this good.
Column 6 'Score':	Raw score, does not include the secondary structure score
Column 7 'SS':	Secondary structure score This score tells how well the PSIPRED-predicted (3-state) or actual DSSP-determined (8-state) secondary structure sequences agree with each other. PSIPRED confidence values are used in the scoring, low confidences getting less statistical weight.
Column 8 'Cols':	The number of aligned Match columns in the HMM-HMM alignment.
Columns 9,10:	Range of aligned match states from query HMM
Columns 11,12:	Range of aligned match states from target HMM
Column 14:	Number of match states in target HMM

4.2 HMM-HMM pairwise alignments

The output file d1bpya1.hhr contains the same hit list plus the pairwise HMM alignments. One example is give here:

```
No 68
>d1a4pa_ a.39.1.2 (A:) Calcyclin (S100) {Human (Homo sapiens), P11 s100a10, calpactin [TaxId: 9606]}
Probab=91.65  E-value=0.12  Score=40.00  Aligned_cols=62  Identities=16%  Similarity=0.149  Sum_probs=42.0

Q ss_pred             ccCCCCCCcHHHHHHHHHHHHHhhcccCccHHHHHHHHHhhhhccCCCCcCHHHHHH-HHHH
Q sp|Q5VUD6|FA69  140 FDKPTRGTSIKEFREMTLSFLKANLGDLPSLPALVGQVLLMADFNKDNRVSLAEAKSV-WALLQ  202 (431)
Q Consensus       140 ~d~p~~g~s~~eF~emv~~~i~~~lg~~~~l~~L~~~~~~~~d~nk~g~vs~~e~~sl-waLlq  202 (431)
                      ||+..-..|.+||.+++.......++.+.+ ...+..++..+|.|+||+|++.|...+ ..|..
T Consensus         18 yd~ddG~is~~El~~~l~~~~~~~~~~~~~-~~~v~~~~~~D~n~DG~I~F~EF~~li~~l~~   80 (92)
T d1a4pa_           18 FAGDKGYLTKEDLRVLMEKEFPGFLENQKD-PLAVDKIMKDLDQCRDGKVGFQSFFSLIAGLTI   80 (92)
T ss_dssp              HHGGGCSBCHHHHHHHHHHHCHHHHHSCC-TTHHHHHHHHHCTTSSSCBCHHHHHHHHHHHHH
T ss_pred              HcCCCCEEcHHHHHHHHHHhccccccccCC-HHHHHHHHHHHhCCCCCCcHHHHHHHHHHHH
Confidence             444433449999999998876655554332 2345666778999999999999997544 44443
```

This alignment shows an EF hand embedded in a kinase domain in PIP49/FAM69B. The pairwise alignment consists of one or more blocks with the following lines:

```
Q ss_dssp:      the query secondary structure as determined by DSSP (when available)
Q ss_pred:      the query secondary structure as predicted by PSIPRED (when available)
Q scop-id:      the query sequence
Q Consensus:    the query alignment consensus sequence
```

The predicted secondary structure states are shown in capital letters if the PSIPRED confidence value is between 0.7 and 1.0, for lower confidence values they are given in lower-case letters. With the option `'-ssconf'`, `'ss_conf'` lines can be added to the alignments which report the PSIPRED confidence values by numbers between 0 and 9 (as in versions up to 1.5).

The consensus sequence uses capital letters for well conserved columns and lower case for partially conserved columns. Unconserved columns are marked by a tilde `~`. Roughly speaking, amino acids that occur with >= 60% probability (before adding pseudocounts) are written as capital letters and amino acids that have >= 40% probability are written as lower case letters, where gaps are included in the fraction counts. More precisely, when the gap-corrected amino acid fraction

$$p_i(a) * N_{eff}(i)/(N_{eff} + 1)$$

is above 0.6 (0.4) an upper (lower) case letter is used for amino acid a. Here, $p_i(a)$ is the emission probability for a in column i, N_{eff} is the effective number of sequences in the entire multiple alignment (between 1 and 20) and $N_{eff}(i)$ is the effective number of sequences in the subalignment consisting of those sequences that do not have a gap in column i. These percentages increase approximately inversely proportionally with the fraction of gaps in the column, hence a column with only cysteines and 50% gaps gets a lower case letter.

The line in the middle shows the column score between the query and target amino acid distributions. It gives a valuable indication for the alignment quality.

```
= : column score below -1.5
- : column score between -1.5 and -0.5
. : column score between -0.5 and +0.5
+ : column score between +0.5 and +1.5
| : column score above   +1.5
```

A unit of column score corresponds approximately to 0.6 bits. From the column score line the excellent alignment around the conserved `'D.n.DG.i...E'` motif in the turn between two helices is evident. The alignment around the gap by contrast scores only a bit better than zero per residue and is therefore not very reliable.

After the template block, which consists of the following lines,

```
T Consensus:    the target alignment consensus sequence
T scop-id:      the target domain sequence
T ss_dssp:      the target secondary structure as determined by DSSP (when available)
T ss_pred:      the target secondary structure as predicted by PSIPRED (when available)
```

The last line in the block (`Confidence`) reports the reliability of the pairwise query-template alignment. The confidence values are obtained from the posterior probabilities calculated in the Forward-Backward algorithm. A value of 8 indicates a probability that this pair of HMM columns is correctly aligned between 0.8 and 0.9. The `Confidence` line is only displayed when the -realign option is active.

5 File formats

5.1 Input alignment formats

HMMs can be read by HHsearch/HHblits in its own .hhm format, as well as in HMMER format (.hmm). Performance is not as good for HMMER-format as for hhm format, so please use our hhm format if possible. HMMER's hmm format can be converted to hhm format simply with hhmake:

```
> hhmake -i test.hmm -o test.hhm
```

This works only for a single HMM per file, not for concatenated HMMs. A safer way to effect the conversion is to call hhmake with the original alignment file. Note: you may add predicted secondary structure to the hmm file with `addss.pl` before the conversion to hhm format.

Multiple alignments can be read in A2M, A3M, or aligned FASTA format. (Check the -M option for using an input format different from the default A3M). You can transform MSAs from Clustal or Stockholm format to A3M or aligned FASTA with the `reformat.pl` utility supplied in this package.

To reformat from Clustal format to A3M:

```
> reformat.pl test.aln test.a3m
```

or explicitly, if the formats can not be recognized from the extensions:

```
> reformat.pl clu a3m test.clustal test.a3m
```

To reformat from Stockholm to aligned FASTA:

```
> reformat.pl test.sto test.fas
```

Example for aligned FASTA format:

```
>d1a1x__ b.63.1.1 (-) p13-MTCP1 {Human (Homo sapiens)}
PPDHLWVHQEGIYRDEYQRTWVAVVEE--E--T--SF---------LR----------ARVQQIQVPLG-------DAARPSHLLTS-----QL
>gi|6678257|ref|NP_033363.1|:(7-103) T-cell lymphoma breakpoint 1 [Mus musculus]
HPNRLWIWEKHVYLDEFRRSWLPVVIK--S--N--EK---------FQ----------VILRQEDVTLG-------EAMSPSQLVPY-----EL
>gi|7305557|ref|NP_038800.1|:(8-103) T-cell leukemia/lymphoma 1B, 3 [Mus musculus]
PPRFLVCTRDDIYEDENGRQWVVAKVE--T--S--RSpygsrietcIT----------VHLQHMTTIPQ-------EPTPQQPINNN-----SL
>gi|11415028|ref|NP_068801.1|:(2-106) T-cell lymphoma-1; T-cell lymphoma-1A [Homo sapiens]
HPDRLWAWEKFVYLDEKQHAWLPLTIEikD--R--LQ---------LR----------VLLRREDVVLG-------RPMTPTQIGPS-----LL
>gi|7305561|ref|NP_038804.1|:(7-103) T-cell leukemia/lymphoma 1B, 5 [Mus musculus]
----------GIYEDEHHRVWIAVNVE--T--S--HS---------SHgnrietcvt-VHLQHMTTLPQ-------EPTPQQPINNN-----SL
>gi|7305553|ref|NP_038801.1|:(5-103) T-cell leukemia/lymphoma 1B, 1 [Mus musculus]
LPVYLVSVRLGIYEDEHHRVWIVANVE--TshS--SH---------GN----------RRRTHVTVHLW-------KLIPQQVIPFNplnydFL
>gi|27668591|ref|XP_234504.1|:(7-103) similar to Chain A, Crystal Structure Of Murine Tcl1
-PDRLWLWEKHVYLDEFRRSWLPIVIK--S--N--GK---------FQ----------VIMRQKDVILG-------DSMTPSQLVPY-----EL
>gi|27668589|ref|XP_234503.1|:(9-91) similar to T-cell leukemia/lymphoma 1B, 5;
-PHILTLRTHGIYEDEHHRLWVVLDLQ--A--ShlSF---------SN----------RLLIYLTVYLQqgvafplESTPPSPMNLN-----GL
>gi|7305559|ref|NP_038802.1|:(8-102) T-cell leukemia/lymphoma 1B, 4 [Mus musculus]
PPCFLVCTRDDIYEDEHGRQWVAAKVE--T--S--SH---------SPycskietcvtVHLWQMTTLFQ-------EPSPDSLKTFN-----FL
>gi|7305555|ref|NP_038803.1|:(9-102) T-cell leukemia/lymphoma 1B, 2 [Mus musculus]
---------PGFYEDEHHRLWMVAKLE--T--C--SH---------SPycnkietcvtVHLWQMTRYPQ-------EPAPYNPMNYN-----FL
```

The sequence name and its description must be contained in a single name line beginning with the > symbol and followed directly by the sequence name. The residue data is contained in one or more lines of arbitrary length following the name line. No empty lines should be used. In aligned FASTA the gaps are written with '-' and the n'th letter of each sequence (except newlines) is understood to build the n'th column of the multiple alignment.

The same alignment in A2M format looks like this:

```
>d1a1x__ b.63.1.1 (-) p13-MTCP1 {Human (Homo sapiens)}
PPDHLWVHQEGIYRDEYQRTWVAVVEE..E..T..SF.........LR..........ARVQQIQVPLG.......DAARPSHLLTS.....QL
>gi|6678257|ref|NP_033363.1|:(7-103) T-cell lymphoma breakpoint 1 [Mus musculus]
HPNRLWIWEKHVYLDEFRRSWLPVVIK..S..N..EK.........FQ..........VILRQEDVTLG.......EAMSPSQLVPY.....EL
>gi|7305557|ref|NP_038800.1|:(8-103) T-cell leukemia/lymphoma 1B, 3 [Mus musculus]
PPRFLVCTRDDIYEDENGRQWVVAKVE..T..S..RSpygsrietcIT..........VHLQHMTTIPQ.......EPTPQQPINNN.....SL
>gi|11415028|ref|NP_068801.1|:(2-106) T-cell lymphoma-1; T-cell lymphoma-1A [Homo sapiens]
HPDRLWAWEKFVYLDEKQHAWLPLTIEikD..R..LQ.........LR..........VLLRREDVVLG.......RPMTPTQIGPS.....LL
>gi|7305561|ref|NP_038804.1|:(7-103) T-cell leukemia/lymphoma 1B, 5 [Mus musculus]
----------GIYEDEHHRVWIAVNVE..T..S..HS.........SHgnrietcvt.VHLQHMTTLPQ.......EPTPQQPINNN.....SL
>gi|7305553|ref|NP_038801.1|:(5-103) T-cell leukemia/lymphoma 1B, 1 [Mus musculus]
LPVYLVSVRLGIYEDEHHRVWIVANVE..TshS..SH.........GN..........RRRTHVTVHLW.......KLIPQQVIPFNplnydFL
>gi|27668591|ref|XP_234504.1|:(7-103) similar to Chain A, Crystal Structure Of Murine Tcl1
-PDRLWLWEKHVYLDEFRRSWLPIVIK..S..N..GK.........FQ..........VIMRQKDVILG.......DSMTPSQLVPY.....EL
>gi|27668589|ref|XP_234503.1|:(9-91) similar to T-cell leukemia/lymphoma 1B, 5;
-PHILTLRTHGIYEDEHHRLWVVLDLQ..A..ShlSF.........SN..........RLLIYLTVYLQqgvafplESTPPSPMNLN.....GL
>gi|7305559|ref|NP_038802.1|:(8-102) T-cell leukemia/lymphoma 1B, 4 [Mus musculus]
PPCFLVCTRDDIYEDEHGRQWVAAKVE..T..S..SH.........SPycskietcvtVHLWQMTTLFQ.......EPSPDSLKTFN.....FL
>gi|7305555|ref|NP_038803.1|:(9-102) T-cell leukemia/lymphoma 1B, 2 [Mus musculus]
---------PGFYEDEHHRLWMVAKLE..T..C..SH.........SPycnkietcvtVHLWQMTRYPQ.......EPAPYNPMNYN.....FL
```

A2M format is derived from aligned FASTA format. It looks very similar, but it distinguishes between match/delete columns and insert columns. This information is important to uniquely specify how an alignment is transformed into an HMM. The match/delete columns use upper case letters for residues and the '-' symbol for deletions (gaps). The insert columns use lower case letters for the inserted residues. Gaps aligned to inserted residues are written as '.' Lines beginning with a hash # symbol will be treated as commentary lines in HHsearch/HHblits (see below).

The same alignment in A3M:

```
>d1a1x__ b.63.1.1 (-) p13-MTCP1 {Human (Homo sapiens)}
PPDHLWVHQEGIYRDEYQRTWVAVVEEETSFLRARVQQIQVPLGDAARPSHLLTSQL
>gi|6678257|ref|NP_033363.1|:(7-103) T-cell lymphoma breakpoint 1 [Mus musculus]
HPNRLWIWEKHVYLDEFRRSWLPVVIKSNEKFQVILRQEDVTLGEAMSPSQLVPYEL
>gi|7305557|ref|NP_038800.1|:(8-103) T-cell leukemia/lymphoma 1B, 3 [Mus musculus]
PPRFLVCTRDDIYEDENGRQWVVAKVETSRSpygsrietcITVHLQHMTTIPQEPTPQQPINNNSL
>gi|11415028|ref|NP_068801.1|:(2-106) T-cell lymphoma-1; T-cell lymphoma-1A [Homo sapiens]
HPDRLWAWEKFVYLDEKQHAWLPLTIEikDRLQLRVLLRREDVVLGRPMTPTQIGPSLL
>gi|7305561|ref|NP_038804.1|:(7-103) T-cell leukemia/lymphoma 1B, 5 [Mus musculus]
----------GIYEDEHHRVWIAVNVETSHSSHgnrietcvtVHLQHMTTLPQEPTPQQPINNNSL
>gi|7305553|ref|NP_038801.1|:(5-103) T-cell leukemia/lymphoma 1B, 1 [Mus musculus]
LPVYLVSVRLGIYEDEHHRVWIVANVETshSSHGNRRRTHVTVHLWKLIPQQVIPFNplnydFL
>gi|27668591|ref|XP_234504.1|:(7-103) similar to Chain A, Crystal Structure Of Murine Tcl1
-PDRLWLWEKHVYLDEFRRSWLPIVIKSNGKFQVIMRQKDVILGDSMTPSQLVPYEL
>gi|27668589|ref|XP_234503.1|:(9-91) similar to T-cell leukemia/lymphoma 1B, 5;
-PHILTLRTHGIYEDEHHRLWVVLDLQAShlSFSNRLLIYLTVYLQqgvafplESTPPSPMNLNGL
>gi|7305559|ref|NP_038802.1|:(8-102) T-cell leukemia/lymphoma 1B, 4 [Mus musculus]
PPCFLVCTRDDIYEDEHGRQWVAAKVETSSHSPycskietcvtVHLWQMTTLFQEPSPDSLKTFNFL
>gi|7305555|ref|NP_038803.1|:(9-102) T-cell leukemia/lymphoma 1B, 2 [Mus musculus]
---------PGFYEDEHHRLWMVAKLETCSHSPycnkietcvtVHLWQMTRYPQEPAPYNPMNYNFL
```

The A3M format is a condensed version of A2M format. It is obtained by omitting all '.' symbols from A2M format. Hence residues emitted by Match states of the HMM are in upper case, residues emitted by Insert states are in lower case and deletions are written '-'. A3M-formatted alignments can be reformatted to other formats like FASTA or A2M with the `reformat.pl` utility:

```
reformat.pl test.a3m test.a2m
```

Lines beginning with a hash # symbol will be treated as commentary lines in HHsearch/HHblits (see below). Please note that A3M, though very practical and space-efficient, is not a standard format, and the name A3M is our personal invention.

Secondary structure information in A3M/A2M or FASTA MSAs for HHsearch/HHblits

The alignments read in by HHblits, HHsearch or HHmake can also contain secondary structure information. This information can be included in sequences with special names, like in this A3M file:

```
>ss_dssp
CCSEEEEEETTEEEETTSCEEEEEEEECSSCEEEEEECCCCCCCSCCCHHHHTTCSSCSEEEEETTTEEEETTSC
>aa_dssp
PPDHLWVHQEGIYRDEYQRTWVAVVEEETSFLRARVQQIQVPLGDAARPSHLLTSQLPLMWQLYPEERYMDNNSR
>aa_pred
PPDHLWVHQEGIYRDEYQRTWVAVVEEETSFLRARVQQIQVPLGDAARPSHLLTSQLPLMWQLYPEERYMDNNSR
>ss_pred
CCCEEEEECCCEECCCCCEEEEEEEEECCCCCCEEEEEEECCCCCCCCCCCCCCCCCCCEEEECCCCCEECCCCC
>ss_conf
98768996187010458707899997057864013215310378878877777442461478721770203563 1
>d1a1x__ b.63.1.1 (-) p13-MTCP1 {Human (Homo sapiens)}
PPDHLWVHQEGIYRDEYQRTWVAVVEEETSFLRARVQQIQVPLGDAARPSHLLTSQLPLMWQLYPEERYMDNNSR
>gi|6678257|ref|NP_033363.1|:(7-103) T-cell lymphoma breakpoint 1 [Mus musculus]
HPNRLWIWEKHVYLDEFRRSWLPVVIKSNEKFQVILRQEDVTLGEAMSPSQLVPYELPLMWQLYPKDRYRSCDSM
>gi|7305557|ref|NP_038800.1|:(8-103) T-cell leukemia/lymphoma 1B, 3 [Mus musculus]
PPRFLVCTRDDIYEDENGRQWVVAKVETSRSpygsrietcITVHLQHMTTIPQEPTPQQPINNNSLPTMWRLESMNTYTGTDGT
>gi|11415028|ref|NP_068801.1|:(2-106) T-cell lymphoma-1; T-cell lymphoma-1A [Homo sapiens]
HPDRLWAWEKFVYLDEKQHAWLPLTIEikDRLQLRVLLRREDVVLGRPMTPTQIGPSLLPIMWQLYPDGRYRSSDSS
```

The sequence with name `>ss_dssp` contains the 8-state DSSP-determined secondary structure. `>aa_dssp` and `>aa_pred` contain the same residues as the query sequence (`>d1a1x__` in this case). They are optional and used merely to check whether the secondary structure states have correctly been assigned to the alignment. `>ss_pred` contains the 3-state secondary structure predicted by PSIPRED, and `>ss_conf` contains the corresponding confidence values. The query sequence is the first sequence that does not start with a special name. It is not marked explicitly.

Name lines in alignments

If you would like to create HMMs from alignments with a specified name which differ from the name of the first sequence, you can do so by adding name lines to your FASTA, A2M, or A3M alignment:

```
#PF02043 Bac_chlorC:  Bacteriochlorophyll C binding protein
>ss_pred
CCCCHHHHHHHHHHHHHHHHHHHHHHHHHHHHHHHHHHHHHHHHHCCCCCCCCCCCCCCCCCCCCCCCCCCCCHHHHHHHCC
>ss_conf
98632346588876777777500199999999988788864034458866667756555766787776606676330 39
>CSMA_CHLAU/1-79
ATRGWFSESSAQVAQIGDIMFQGHWQWVSNALQATAAAVDNINRNAYPGVSRSGSGEGAFSSSPSNGFRPKRIRSRFNR
>CSMA_CHLPH/2-51
NGGGVFTDILAASGRIFEVMVEGHWATVGYLFDSLGKGVSRINQNAYGNM------------------------------
...
```

When creating an HMM from an A3M file with hhmake, the first word of the name line is used as the name and file name of the HMM (PF02043 in this case). The following is an optional description. The descriptions will appear in the hit list and alignment section of the search results. The name lines can be arbitrarily long and there can be any number of name/description lines included, marked by a '#' as the first character in the line. Note that name lines are read by HHmake but are not a part of the standard definition of the FASTA or A2M format.

5.2 HHsearch/HHblits model format (hhm-format)

HHsearch/HHblits uses a format HMM that is unchanged since HHsearch version 1.5. This is the example of an HHM model file produced by HHmake:

```
HHsearch 1.5
NAME  d1mvfd_ b.129.1.1 (D:) MazE {Escherichia coli}
FAM   b.129.1.1
FILE  d1mvfd_
COM   hhmake1 -i d1mvfd_.a3m -o test.hhm
DATE  Wed May 14 10:41:06 2011
LENG  44 match states, 44 columns in multiple alignment
FILT  32 out of 35 sequences passed filter (-id 90 -cov 0 -qid 0 -qsc -20.00 -diff 100)
NEFF  4.0
SEQ
>ss_dssp
CBCEEETTEEEEECCHHHHHHTTCCTTCBEEEEEETTEEEEEEC
>ss_pred
CCCCCCCCCCCCCCHHHHHHHHCCCCCCEEEEEEECCEEEEEEC
>ss_conf
93233467666600578899808998986889874993798739
>Consensus
sxIxKWGNSxAvRlPaxlxxxlxlxxgdxixxxxxxxxivlxPv
>d1mvfd_ b.129.1.1 (D:) MazE {Escherichia coli}
SSVKRWGNSPAVRIPATLMQALNLNIDDEVKIDLVDGKLIIEPV
>gi|10176344|dbj|BAB07439.1|:(1-43) suppressor of ppGpp-regulated growth inhibitor [Bacillus halodurans]
TTIQKWGNSLAVRIPNHYAKHINVTQGSEIELSLgSDQTIILKP-
>gi|50120611|ref|YP_049778.1|:(3-43) suppressor of growth inhibitory protein ChpA [Erwinia carotovora]
-TVKKWGNSPAIRLSSSVMQAFDMTFNDSFDMEIRETEIALIP-
>gi|44064461|gb|EAG93225.1|:(2-42) unknown [environmental sequence]
-SVVKWGSYLAVRLPAELVLELGLKEGDEIDLVKDDGPVRVR--
>gi|31442758|gb|AAP55635.1|:(1-44) PemI-like protein [Pediococcus acidilactici]
TRLAKWGNSKAARIPSQIIKQLKLDDNQDMTITIENGSIVLTPI
>gi|44419085|gb|EAJ13619.1|:(3-43) unknown [environmental sequence]
SAIQKWGNSAAVRLPAVLLEQIDASVGSSLNADVRPDGVLLSP-
>gi|24376549|gb|AAN57947.1|:(3-44) putative cell growth regulatory protein [Streptococcus mutans UA159]
SAINKWGNSSAIRLPKQLVQELQLQTNDVLDYKVSGNKIILEKV
>gi|11344928|gb|AAG34554.1|:(1-44) MazE [Photobacterium profundum]
TQIRKIGNSLGSIIPATFIRQLELAEGAEIDVKTVDGKIVIEPI
>gi|45681193|ref|ZP_00192636.1|:(2-44) COG2336: Growth regulator [Mesorhizobium sp. BNC1]
-TIRKIGNSEGVILPKELLDRHNLKTGDALAIVEEGSDLVLKPV
#
NULL  3706 5728 4211 4064 4839 3729 4763 4308 4069  3323  5509 4640 4464 4937 4285 4423 3815 3783 6325 4665
HMM   A    C    D    E    F    G    H    I    K     L     M    N    P    Q    R    S    T    V    W    Y
      M->M M->I M->D I->M I->I D->M D->D Neff NeffI NeffD
      0    *    *    0    *    0    *    *    *     *
S 1   *    *    *    *    *    *    *    *    *     *     *    *    *    *    *    1012 988  *    *    *   1
      0    *    *    *    *    *    *    2817 0     0

S 2   2307 *    *    *    *    *    *    *    *     *     *    *    *    3178 3009 2179 1546 *    *    *   2
      0    *    *    *    *    *    *    3447 0     0

V 3   *    *    *    *    *    *    *    917  *     3009  *    *    *    *    *    *    *    1530 *    *   3
      0    *    *    *    *    *    *    3447 0     0
  .
  .
  .
V 44  *    *    *    *    *    *    *    1309 *     *     *    *    *    *    *    *    *    745  *    *   44
      0    *    *    0    *    *    *    2533 0     0

//
```

The first line (`HHsearch 1.5`) gives the format version, which corresponds to the HHsearch version for which this format was first introduced. Newer versions of HHsearch/HHblits may use previous format versions. The `NAME` line gives the name of the HMM and an optional description. The first 30 characters of this field are used in the summary hit list of the search results in hhr format, the full name line is given above the query-template alignments of the search results. The `FAM` line contains the family if the sequence is from SCOP of PFAM (used for calibration). `COM` is the

command that was used to generate the file. `NEFF` is the diversity of the alignment, calculated as exp of the negative entropy averaged over all columns of the alignment.

The `SEQ` section contains a number of aligned, representative (pseudo) sequences in A3M format and is terminated with a line containing only a `#`. The first sequence represents the DSSP secondary structure (if available, i.e. if contained in the A3M or FASTA alignment from which the HMM model was built), the second and third sequences contain the predicted secondary structure and the corresponding confidence values in the range 0–9 (if available). The fourth sequence is the consensus annotation sequence that is shown in the pairwise query-template alignments in the hhsearch output. The first *real* sequence after the pseudo sequences is the *seed* or *master* sequence from which the alignment was built (`>d1mvfd_`, in our example). If the alignment does not represent a single master sequence but an entire family, as in the case of PFAM alignments for example, the first real sequence may be a consensus sequence calculated for the entire alignment. This master sequence is shown in the pairwise query-template alignments in the hhsearch output.

The next line specifies the null model frequencies, which are extracted from the selected substitution matrix used to add pseudocounts. Each of the positive integers is equal to 1000 times the negative logarithm of the amino acid frequency (which is between 0 and 1):

$$-1000 \times \log_2(frequency) \tag{1}$$

After the two annotation lines that specify the order of columns for the emission and transition probabilities that follow, there is a line which is not currently read by HHsearch and that lists the transition frequencies from the begin state to the first Match state, Insert state and Delete state.

The last block contains two lines for each column of the HMM. The first line starts with the amino acid in the master sequence at that column in the HMM and the column number. Following are 20 positive integers representing the match state amino acid emission frequencies (see eq. 1). Asterisks `*` stand for a frequency of 0 (which would otherwise be represented by 99999). Please note that, unlike in HMMER format, *the emission frequencies do not contain pseudo-counts* in the HHsearch model format. The second line contains the seven transition frequencies (eq. 1) and three local diversities, `Neff_M`, `Neff_I`, and `Neff_D` (see next paragraph). The end of the model is indicated by a line containing only `\\`.

Calculation of the local number of effective sequences

`Neff_M(i)` quantifies the local diversity of the alignment at a position i. More precisely, it measures the diversity of subalignment $Ali_M(i)$ that contains all sequences that have a residue at column i of the full alignment. The subalignment contains all columns for which at least 90% of these sequences have no end gap. End gaps are gaps to the left of the first residue or to the right of the last residue. The latter condition ensures that the sequences in the subalignment $Ali_M(i)$ cover most of the columns in it. The number of effective sequences in the subalignment $Ali_M(i)$ is exp of the average sequence entropy over all columns of the subalignment. Hence, `Neff_M` is bounded by 0 from below and 20 from above. (In practive, it is bounded by the entropy of a column with background amino acid distribution f_a: $N_{eff} < \sum_{a=1}^{20} f_a log f_a \approx 16$.) Similarly, `Neff_I(i)` gives the diversity of the subalignment $Ali_I(i)$ of all sequences that have an insert at position i, and `Neff_D(i)` refers to the diversity of subaligment $Ali_D(i)$ of all sequences that have a Delete (a gap) at position i of the full alignment. The number of effective sequences of the full alignment, which appears as NEFF in the header of each hhm file, is the average of `Neff_M(i)` over all alignment positions i.

6 Summary of command-line parameters

This is just a brief summary of command line parameters for the various binaries and perl scripts as they are displayed by the programs when calling them without command line parameters. On the help pages of our HHpred/HHblits web servers

http://toolkit.tuebingen.mpg.de or http://toolkit.genzentrum.lmu.de

you can find more detailed explanations about some of the input parameters ('Parameters' section) and about how to interpret the output ('Results' section). The FAQ section contains valuable practical hints on topics such as how to validate marginally significant database matches or how to avoid high-scoring false positives.

6.1 `hhblits` – HMM-HMM-based lighting-fast iterative sequence search

HHblits is a sensitive, general-purpose, iterative sequence search tool that represents both query and database sequences by HMMs. You can search HHblits databases starting with a single query sequence, a multiple sequence alignment, or an HMM. HHblits prints out a ranked list of database HMMs/alignments and can also generate a multiple sequence alignment from the significant HMMs/alignments.

```
Usage: hhblits -i query [options]

 -i <file>      input query (single FASTA-sequence, A3M- or FASTA-alignment, HMM-file)

Options:
 -d    <base>   database basename (default=)
 -n     [1,8]   number of iterations (default=2)
 -e     [0,1]   E-value cutoff for inclusion in result alignment (def=0.001)

Input alignment format:
 -M a2m         use A2M/A3M (default): upper case = Match; lower case = Insert;
               ' -' = Delete; '.' = gaps aligned to inserts (may be omitted)
 -M first       use FASTA: columns with residue in 1st sequence are match states
 -M [0,100]     use FASTA: columns with fewer than X% gaps are match states

Output options:
 -ofas <file>   write multiple alignment of significant matches in FASTA format
                Analogous for output in a2m, a3m, hhm format (e.g. -ohhm, -Oa3m)
 -o <file>      write results in standard format to file (default=<infile.hhr>)
 -oa3m <file>   write multiple alignment of significant matches in a3m format
 -opsi <file>   write multiple alignment of significant matches in PSI format
 -ohhm <file>   write HHM file for multiple alignment of significant matches
 -oalis <base>  write multiple alignments in A3M format after each iteration
 -Ofas <file>   write pairwise alignments of significant matches in FASTA format

HMM-HMM alignment options:
 -norealign     do NOT realign displayed hits with MAC algorithm (default=realign)
 -mact [0,1[    posterior probability threshold for MAC re-alignment (def=0.350)
                Parameter controls alignment greediness: 0:global >0.1:local
 -glob/-loc     use global/local Viterbi alignment for searching/ranking (def=local)

Other options:
 -v <int>       verbose mode: 0:no screen output  1:only warings  2: verbose (def=2)
 -cpu <int>     number of CPUs to use (for shared memory SMPs) (default=2)
```

```
An extended list of options can be obtained by using '--help all' as parameter

Example: hhblits -i query.fas -oa3m query.a3m -n 2
```

6.2 hhsearch – search a database of HMMs with a query MSA or HMM

```
Usage: hhsearch -i query -d database [options]

 -i <file>     input query alignment (A2M, A3M, FASTA) or HMM
 -d <file>     HMM database of concatenated HMMs in hhm, HMMER, or A3M format,
               OR, if file has extension pal, list of HMM file names, one per
               line. Multiple dbs, HMMs, or pal files with -d '<db1> <db2>...'

Output options:
 -o <file>     write results in standard format to file (default=<infile.hhr>)
 -Ofas <file>  write pairwise alignments of significant matches in FASTA format
 -ofas <file>  write multiple alignment of significant matches in FASTA format
               Analogous for output in a2m, a3m, hhm format (e.g. -ohhm, -Oa3m)
 -e [0,1]      E-value cutoff for inclusion in multiple alignment (def=0.001)
 -v <int>      verbose mode: 0:no screen output  1:only warings  2: verbose
 -seq <int>    max. number of query/template sequences displayed (def=1)
 -nocons       don't show consensus sequence in alignments (default=show)
 -nopred       don't show predicted 2ndary structure in alignments (default=show)
 -nodssp       don't show DSSP 2ndary structure in alignments (default=show)
 -ssconf       show confidences for predicted 2ndary structure in alignments
 -aliw <int>   number of columns per line in alignment list (def=80)
 -p <float>    minimum probability in summary and alignment list (def=20)
 -E <float>    maximum E-value in summary and alignment list (def=1E+06)
 -Z <int>      maximum number of lines in summary hit list (def=500)
 -z <int>      minimum number of lines in summary hit list (def=10)
 -B <int>      maximum number of alignments in alignment list (def=500)
 -b <int>      minimum number of alignments in alignment list (def=10)
               Remark: you may use 'stdin' and 'stdout' instead of file names

Filter input alignment (options can be combined):
 -id   [0,100] maximum pairwise sequence identity (%) (def=90)
 -diff [0,inf[ filter most diverse set of sequences, keeping at least this
               many sequences in each block of >50 columns (def=100)
 -cov  [0,100] minimum coverage with query (%) (def=0)
 -qid  [0,100] minimum sequence identity with query (%) (def=0)
 -qsc  [0,100] minimum score per column with query  (def=-20.0)

Input alignment format:
 -M a2m        use A2M/A3M (default): upper case = Match; lower case = Insert;
               '-' = Delete; '.' = gaps aligned to inserts (may be omitted)
 -M first      use FASTA: columns with residue in 1st sequence are match states
 -M [0,100]    use FASTA: columns with fewer than X% gaps are match states

HMM-HMM alignment options:
 -realign      realign displayed hits with max. accuracy (MAC) algorithm
 -norealign    do NOT realign displayed hits with MAC algorithm (def=realign)
 -mact [0,1[   posterior probability threshold for MAC re-alignment (def=0.350)
               Parameter controls alignment greediness: 0:global >0.1:local
 -glob/-loc    use global/local alignment mode for searching/ranking (def=local)
 -alt <int>    show up to this many significant alternative alignments(def=2)
 -excl <range> exclude query positions from the alignment, e.g. '1-33,97-168'
 -shift [-1,1] score offset (def=-0.03)
```

```
 -corr [0,1]   weight of term for pair correlations (def=0.10)
 -ssm  0-4     0:   no ss scoring
               1,2: ss scoring after or during alignment  [default=2]
               3,4: ss scoring after or during alignment, predicted vs. predicted
 -ssw [0,1]    weight of ss score  (def=0.11)

Other options:
 -cpu <int>    number of CPUs to use (for shared memory SMPs) (default=1)

An extended list of options can be obtained by using '--help all' as parameter

Example: hhsearch -i a.1.1.1.a3m -d scop70_1.71.hhm
```

6.3 hhmake – build an HMM from an input MSA

Build an HMM from an input alignment in A2M, A3M, or FASTA format or convert between HMMER format (.hmm) and HHsearch format (.hhm). A database file is generated by simply concatenating these HMM files.

```
Usage: hhmake -i file [options]
 -i <file>     query alignment (A2M, A3M, or FASTA), or query HMM

Output options:
 -o <file>     HMM file to be written to  (default=<infile.hhm>)
 -a <file>     HMM file to be appended to
 -v <int>      verbose mode: 0:no screen output  1:only warings  2: verbose
 -seq <int>    max. number of query/template sequences displayed (def=10)
               Beware of overflows! All these sequences are stored in memory.
 -cons         insert consensus as main representative sequence of HMM
 -name <name>  use this name for HMM (default: use name of first sequence)

Filter input alignment (options can be combined):
 -id   [0,100] maximum pairwise sequence identity (%) (def=90)
 -diff [0,inf[ filter most diverse set of sequences, keeping at least this
               many sequences in each block of >50 columns (def=100)
 -cov  [0,100] minimum coverage with query (%) (def=0)
 -qid  [0,100] minimum sequence identity with query (%) (def=0)
 -neff [1,inf] target diversity of alignment (default=off)
 -qsc  [0,100] minimum score per column with query  (def=-20.0)

Input alignment format:
 -M a2m        use A2M/A3M (default): upper case = Match; lower case = Insert;
               '-' = Delete; '.' = gaps aligned to inserts (may be omitted)
 -M first      use FASTA: columns with residue in 1st sequence are match states
 -M [0,100]    use FASTA: columns with fewer than X% gaps are match states

Other options:

Example: hhmake -i test.a3m
```

6.4 hhfilter – filter an MSA

Filter an alignment by maximum pairwise sequence identity, minimum coverage, minimum sequence identity, or score per column to the first (seed) sequence etc.

```
Usage: hhfilter -i infile -o outfile [options]
 -i <file>     read input file in A3M/A2M or FASTA format
 -o <file>     write to output file in A3M format
 -a <file>     append to output file in A3M format

Options:
 -v <int>      verbose mode: 0:no screen output  1:only warings  2: verbose
 -id   [0,100] maximum pairwise sequence identity (%) (def=90)
 -diff [0,inf[ filter most diverse set of sequences, keeping at least this
               many sequences in each block of >50 columns (def=0)
 -cov  [0,100] minimum coverage with query (%) (def=0)
 -qid  [0,100] minimum sequence identity with query (%) (def=0)
 -qsc  [0,100] minimum score per column with query  (def=-20.0)
 -neff [1,inf] target diversity of alignment (default=off)

Input alignment format:
 -M a2m        use A2M/A3M (default): upper case = Match; lower case = Insert;
               '-' = Delete; '.' = gaps aligned to inserts (may be omitted)
 -M first      use FASTA: columns with residue in 1st sequence are match states
 -M [0,100]    use FASTA: columns with fewer than X% gaps are match states

Example: hhfilter -id 50 -i d1mvfd_.a2m -o d1mvfd_.fil.a2m
```

6.5 hhalign – Align a query MSA/HMM to a template MSA/HMM

Align a query alignment/HMM to a template alignment/HMM by HMM-HMM alignment. If only one alignment/HMM is given it is compared to itself and the best off-diagonal alignment plus all further non-overlapping alignments above significance threshold are shown. The command also allows to sample alignments randomly, to generate png-files with dot plots showing alignments or to print out a list of indices of aligned residue pairs.

```
Usage: hhalign -i query [-t template] [options]

 -i <file>     input query alignment  (fasta/a2m/a3m) or HMM file (.hhm)
 -t <file>     input template alignment (fasta/a2m/a3m) or HMM file (.hhm)
 -png <file>   write dotplot into PNG-file (default=none)

Output options:
 -o <file>     write output alignment to file
 -ofas <file>  write alignments in FASTA, A2M (-oa2m) or A3M (-oa3m) format
 -Oa3m <file>  write query alignment in a3m format to file (default=none)
 -Aa3m <file>  append query alignment in a3m format to file (default=none)
 -atab <file>  write alignment as a table (with posteriors) to file (default=none)
 -index <file> use given alignment to calculate Viterbi score (default=none)
 -v <int>      verbose mode: 0:no screen output  1:only warings  2: verbose
 -seq  [1,inf[ max. number of query/template sequences displayed  (def=1)
 -nocons       don't show consensus sequence in alignments (default=show)
 -nopred       don't show predicted 2ndary structure in alignments (default=show)
 -nodssp       don't show DSSP 2ndary structure in alignments (default=show)
 -ssconf       show confidences for predicted 2ndary structure in alignments
 -aliw int     number of columns per line in alignment list (def=80)
 -P <float>    for self-comparison: max p-value of alignments (def=0.001
 -p <float>    minimum probability in summary and alignment list (def=0)
 -E <float>    maximum E-value in summary and alignment list (def=1E+06)
 -Z <int>      maximum number of lines in summary hit list (def=100)
 -z <int>      minimum number of lines in summary hit list (def=1)
```

```
 -B <int>      maximum number of alignments in alignment list (def=100)
 -b <int>      minimum number of alignments in alignment list (def=1)
 -rank int     specify rank of alignment to write with -Oa3m or -Aa3m option (def=1)

Dotplot options:
 -dthr <float> probability/score threshold for dotplot (default=0.50)
 -dsca <int>   if value <= 20: size of dot plot unit box in pixels
               if value > 20: maximum dot plot size in pixels (default=600)
 -dwin <int>   average score over window [i-W..i+W] (for -norealign) (def=10)
 -dali <list>  show alignments with indices in <list> in dot plot
               <list> = <index1> ... <indexN>  or  <list> = all

Filter input alignment (options can be combined):
 -id   [0,100] maximum pairwise sequence identity (%) (def=90)
 -diff [0,inf[ filter most diverse set of sequences, keeping at least this
               many sequences in each block of >50 columns (def=100)
 -cov  [0,100] minimum coverage with query (%) (def=0)
 -qid  [0,100] minimum sequence identity with query (%) (def=0)
 -qsc  [0,100] minimum score per column with query  (def=-20.0)

Input alignment format:
 -M a2m        use A2M/A3M (default): upper case = Match; lower case = Insert;
               '-' = Delete; '.' = gaps aligned to inserts (may be omitted)
 -M first      use FASTA: columns with residue in 1st sequence are match states
 -M [0,100]    use FASTA: columns with fewer than X% gaps are match states

HMM-HMM alignment options:
 -glob/-loc    global or local alignment mode (def=local)
 -alt <int>    show up to this number of alternative alignments (def=1)
 -realign      realign displayed hits with max. accuracy (MAC) algorithm
 -norealign    do NOT realign displayed hits with MAC algorithm (def=realign)
 -mact [0,1[   posterior probability threshold for MAC alignment (def=0.300)
               A threshold value of 0.0 yields global alignments.
 -sto <int>    use global stochastic sampling algorithm to sample this many alignments
 -excl <range> exclude query positions from the alignment, e.g. '1-33,97-168'
 -shift [-1,1] score offset (def=-0.010)
 -corr [0,1]   weight of term for pair correlations (def=0.10)
 -ssm  0-4     0:no ss scoring [default=2]
               1:ss scoring after alignment
               2:ss scoring during alignment
 -ssw  [0,1]   weight of ss score  (def=0.11)

 -def          read default options from ./.hhdefaults or <home>/.hhdefault.

Example: hhalign -i T0187.a3m -t d1hz4a_.hhm -png T0187pdb.png
```

6.6 `reformat.pl` – reformat one or many alignments

Read one or many multiple alignments in one format and write them in another format

```
Usage: reformat.pl [informat] [outformat] infile outfile [options]
  or   reformat.pl [informat] [outformat] 'fileglob' .ext [options]

Available input formats:
   fas:     aligned fasta; lower and upper case equivalent, '.' and '-' equivalent
   a2m:     aligned fasta; inserts: lower case, matches: upper case, deletes: '-',
            gaps aligned to inserts: '.'
```

```
   a3m:     like a2m, but gaps aligned to inserts MAY be omitted
   sto:     Stockholm format; sequences in several blocks with sequence name at
            beginning of line (HMMER output)
   psi:     format as read by PSI-BLAST using the -B option (like sto with -M first -r)
   clu:     Clustal format; sequences in several blocks with sequence name at beginning
            of line
Available output formats:
   fas:     aligned fasta; all gaps '-'
   a2m:     aligned fasta; inserts: lower case, matches: upper case, deletes: '-',
            gaps aligned to inserts: '.'
   a3m:     like a2m, but gaps aligned to inserts are omitted
   sto:     Stockholm format; sequences in just one block, one line per sequence
   psi:     format as read by PSI-BLAST using the -B option
   clu:     Clustal format
If no input or output format is given the file extension is interpreted as format
specification ('aln' as 'clu')

Options:
  -v int    verbose mode (0:off, 1:on)
  -num      add number prefix to sequence names: 'name', '1:name' '2:name' etc
  -noss     remove secondary structure sequences (beginning with >ss_)
  -sa       do not remove solvent accessibility sequences (beginning with >sa_)
  -M first  make all columns with residue in first seuqence match columns
            (default for output format a2m or a3m)
  -M int    make all columns with less than X% gaps match columns
            (for output format a2m or a3m)
  -r        remove all lower case residues (insert states)
            (AFTER -M option has been processed)
  -r int    remove all lower case columns with more than X% gaps
  -g ''     suppress all gaps
  -g '-'    write all gaps as '-'
  -uc       write all residues in upper case (AFTER other options have been processed)
  -lc       write all residues in lower case (AFTER other options have been processed)
  -l        number of residues per line (for Clustal, FASTA, A2M, A3M formats)
            (default=100)
  -d        maximum number of characers in nameline (default=1000)

Examples: reformat.pl 1hjra.a3m 1hjra.a2m
          (same as reformat.pl a3m a2m 1hjra.a3m 1hjra.a2m)
          reformat.pl test.a3m test.fas -num -r 90
          reformat.pl fas sto '*.fasta' .stockholm
```

6.7 `addss.pl` – add predicted secondary structure to an MSA or HMM

Add PSIPRED secondary structure prediction (and DSSP annotation) to a multiple sequence alignment (MSA) or HMMER (multi-)model file.

If the input file is an MSA, the predicted secondary structure and confidence values are added as special annotation sequences with names `>ss_pred`, `>ss_conf`, and `>ss_dssp` to the top of the output A3M alignment. If no output file is given, the output file will have the same name as the input file, except for the extension being replaced by `'.a3m'`. Allowed input formats are A2M/FASTA (default), A3M (-a3m), CLUSTAL (-clu), STOCKHOLM (-sto), HMMER (-hmm).

If the input file contains HMMER models, records SSPRD and SSCON containing predicted secondary structure and confidence values are added to each model. In this case the output file name is obligatory and must be different from the input file name.

```
Usage: perl addss.pl <ali file> [<outfile>] [-fas|-a3m|-clu|-sto]
  or   perl addss.pl <hmm file> <outfile> -hmm
```

6.8 hhmakemodel.pl – generate MSAs or coarse 3D models from HHsearch results file

From the top hits in an hhsearch output file (hhr), you can

- generate a MSA (multiple sequence alignment) containing all representative template sequences from all selected alignments (options -fas, -a2m, -a3m, -pir)
- generate several concatenated pairwise alignments in AL format (option -al)
- generate several concatenated coarse 3D models in PDB format (option -ts)

In PIR, PDB and AL format, the pdb files are required in order to read the pdb residue numbers and ATOM records. The PIR formatted file can be used directly as input to the MODELLER homology modeling package.

```
Usage: hhmakemodel.pl [-i] file.hhr [options]

Options:
 -i   <file.hhr>        results file from hhsearch with hit list and alignments
 -fas <file.fas>        write a FASTA-formatted multiple alignment to file.fas
 -a2m <file.a2m>        write an A2M-formatted multiple alignment to file.a2m
 -a3m <file.a3m>        write an A3M-formatted multiple alignment to file.a3m
 -m   <int> [<int> ...] pick hits with specified indices  (default='-m 1')
 -p   <probability>     minimum probability threshold
 -e   <E-value>         maximum E-value threshold
 -q   <query_ali>       use the full-length query sequence in the alignment
                        (not only the aligned part);
                        the query alignment file must be in HHM, FASTA, A2M,
                        or A3M format.
 -N                     use query name from hhr filename (default: use same
                        name as in hhr file)
 -first                 include only first Q or T sequence of each hit in MSA
 -v                     verbose mode

Options when database matches in hhr file are PDB or SCOP sequences
 -pir <file.pir>        write a PIR-formatted multiple alignment to file.pir
 -ts  <file.pdb>        write the PDB-formatted models based on *pairwise*
                        alignments into file.pdb
 -al  <file.al>         write the AL-formatted *pairwise* alignments into file.al
 -d   <pdbdirs>         directories containing the pdb files (for PDB, SCOP, or DALI
                        sequences)
 -s   <int>             shift the residue indices up/down by an integer
 -CASP                  formatting for CASP (for -ts, -al options)
                        (default: LIVEBENCH formatting)
```

6.9 hhblitsdb.pl – Build an HHblits database

Builds the HHblits database files from MSA and HMM files

```
Usage: hhblitsdb.pl -o <db_name> [-ia3m <a3m_dir>] [-ihhm <hhm_dir>] [-ics <cs_dir>]
```

```
                                                                    [more_options]

Depending on the input directories, the following HHblits database files are generated:
 <db_name>.cs219               column-state sequences, one for each MSA/HMM (for prefilter)
 <db_name>.cs219.sizes         number of sequences and characters in <db_name>.cs219
 <db_name>_a3m_db              packed file containing A3M alignments read from <a3m_dir>
 <db_name>_a3m_db.index        index file for packed A3M file
 <db_name>_a3m.db.index.sizes number of lines in <db_name>_a3m_db.index
 <db_name>_hhm_db              packed file containing HHM-formatted HMMs read from <hhm_dir>
 <db_name>_hhm_db.index        index file for packed HHM file
 <db_name>_hhm_db.index.sizes number of lines in <db_name>_hhm_db.index

Options:
 -o <db_name>    name of database
 -ia3m <a3m_dir> input directory (or glob of directories) with A3M-formatted files
 -ihhm <hhm_dir> input directory (or glob of directories) with HHM (or HMMER) files
                 (WARNING! HMMER format results in decreased performance over HHM format)
 -ics  <cs_dir>  input directory (or glob of directories) with column state sequences
 -log <logfile>  log file recording stderr stream of cstranslate and hhmake commands

 -csext          extension of column state sequences (default: $csext)
 -a3mext         extension of A3M-formatted files (default: $a3mext)
 -hhmext         extension of HHM- or HMMER-formatted files (default: $hhmext)
 -append         if the packed db files exists, append input A3M/HHM files (def: overwrite)
 -v [1-3]        verbose mode (default: $v)
 -cpu <int>      numbers of threads for generating cs219 and hhm files (default = $cpu)


Example 1: only -ia3m given; cs sequences and hhm files are generated from a3m files
   perl hhblitsdb.pl -o hhblits_dbs/mydb -ia3m mydb/a3ms

Example 2: only -ihhm given; cs sequences are generated from hhm files, but no a3m db file
   perl hhblitsdb.pl -o hhblits_dbs/mydb -ihhm mydb/hhms

Example 3: -ia3m and -ihhm given; cs sequences are generated from a3m files
   perl hhblitsdb.pl -o hhblits_dbs/mydb -ia3m mydb/a3ms -ihhm mydb/hhms

Example 4: -ics, -ia3m, and -ihhm given; all db files are created
   perl hhblitsdb.pl -o hhblits_dbs/mydb -ia3m mydb/a3ms -ihhm mydb/hhms -ics mydb/cs

Example 5: using glob expression to specify several input databases
   perl hhblitsdb.pl -o hhblits_dbs/mydb -ihhm 'mydbs*/hhms'
```

6.10 multithread.pl – Run a command for many files in parallel using multiple threads

```
 Usage: multithread.pl '<fileglob>' '<command>' [-cpu <int>] [-v {0,1,2}]

 <command> can include symbol
    $file for the full filename, e.g. /tmp/hh/1c1g_A.a3m,
    $name the filename without extension, e.g. /tmp/hh/1c1g_A, and
    $base for the filename without extension and path, e.g. 1c1g_A.

  -cpu <int>  number of threads to launch (default = 8)
  -v {0,1,2}  verbose mode (default = 1)

 Example: multithread.pl '*.a3m' 'hhmake -i $file 1>$name.log 2>>error.log' -cpu 16
```

7 Changes from previous versions

7.1 2.0.0 (January 2012)

- The iterative HMM-HMM search method HHblits has been added and the entire package is now called HH-suite. HHblits brings the power of HMM-HMM comparison to mainstream, general-purpose sequence searching and sequence analysis.
- The speed of HHsearch was further increase through the use of SSE3 instructions.
- An option `-atab` for writing alignment as a table (with posteriors) to file was introduced
- HHsearch is now able to read HMMER3 profiles (but should not be used due to a loss of sensitivity).
- A local amino acid compositional bias correction was introduced. Improvements are slight (<= 1%) on a standard SCOP single domain benchmark. However, the improvement for more realistic sequences containing multiple domains, repeats, and regions of strong compositional bias will is probably more pronounced. The score shift parameter has been set to -0.03 bits and the mact parameter to 0.30.

7.2 1.6.0 (November 2010)

- A new procedure for estimation of P- and E-values has been implemented that circumvents the need to calibrate HMMs. Calibration can still be done if desired. By default, however, HHsearch now estimates the lambda and mu parameters of the extreme value distribution (EVD) for each pair of query and database HMMs from the lengths of both HMMs and the diversities of their underlying alignments. Apart from saving the time for calibration, this procedure is more reliable and noise-resistant. This change only applies to the default local search mode. For global searches, nothing has changed. Note that E-values in global search mode are unreliable and that sensitivity is reduced.
  Old calibrations can still be used:
  -calm 0 : use empirical query HMM calibration (old default)
  -calm 1 : use empirical db HMM calibration
  -calm 2 : use both query and db HMM calibration
  -calm 3 : use neural network calibration (new default)
- Previous versions of HHsearch sometimes showed non-homologous hits with high probabilities by matching long stretches of secondary structure states, in particular long helices, in the absence of any similarity in the amino acid profiles. Capping the SS score by a linear function of the profile score now effectively suppresses these spurious high-scoring false positives.
- The output format for the query-template alignments has slightly changed. A 'Confidence' line at the bottom of each alignment block now reports the posterior probabilities for each alignment column when the -realign option is active (which it is by default). These probabilities are calculated in the Forward-Backward algorithm that is needed as input for the Maximum ACcuracy alignment algorithm. Also, the lines 'ss_conf' with the confidence values for the secondary structure prediction are omitted by default. (They can be displayed with option '-showssconf'). To compensate, secondary structure predictions with confidence values between 7 and 9 are given in capital letters, while for the predictions with values between 0 and 6 lower-case letters are used.

- In the hhsearch output file in the header lines before each query-database alignment, the substitution matrix score (without gap penalties) of the query with the database sequence is now reported in bits per column. Also, the sum of probabilities for each pair of aligned residues from the MAC algorithm is reported here (0 if no MAC alignment is performed).
- The buildali.pl script now uses context-specific iterative BLAST (CSI-BLAST) instead of PSI-BLAST. This considerably increases the sensitivity of buildali.pl/HHsearch.
- Removed a bug which produced a segfault for input alignments with more than 15000 match columns. Now, the HHsearch binaries will issue a warning and will transform only the first 15000 match columns into an HMM.
- Removed a bug in the multi-threading code that could lead to occasional hang-ups (race condition) in situations where slow file access was impeding program execution and inter-thread signaling was unreliable.
- Removed a memory leak and optimized memory management.
- Removed a bug in hhalign that could lead to unreasonably significant E-values and probabilities due to calibration problems.
- HHsearch now performs realign with MAC-alignment only around Viterbi-hit.

7.3 1.5.0 (August 2007)

- By default, HHsearch realigns all displayed alignments in a second stage using the more accurate Maximum Accuracy (MAC) alignment algorithm (Durbin, Eddy, Krough, Mitchison: Biological sequence analysis, page 95; HMM-HMM version: J. Söding, unpublished). As before, the Viterbi algorithm is employed for searching and ranking the matches. The realignment step is parallelized (`-cpu <int>`) and typically takes a few seconds only. You can switch off the MAC realignment with the -norealign option. The posterior probability threshold is controlled with the -mact [0,1[option. This parameter controls the alignment algorithm's greediness. More precisely, the MAC algorithm finds the alignment that maximizes the sum of posterior probabilities minus mact for each aligned pair. Global alignments are generated with -mact 0, whereas -mact 0.5 will produce quite conservative local alignments. Default value is -mact 0.35, which produces alignments of roughly the same length as the Viterbi algorithm. The -global and -local (default) option now refer to both the Viterbi search stage as well as the MAC realignment stage. With -global (-local), the posterior probability matrix will be calculated for global (local) alignment. Note that '-local -mact 0' will produce global alignments from a local posterior probability matrix (which is not at all unreasonable).
- An amino acid compositional bias correction is now performed by default. This increases the sensitivity by 25% at 0.01 errors per query and by 5% at 0.1 errors per query. By recalibrating the Probabilities, the increased selectivity of this new version allows to give higher probabilities for the same P-values. Also, the score offset could be increased from -0.1 bits to 0 as a consequence.
- The algorithm that filters the set of the most diverse sequences (option -diff) has been improved. Before, it determined the set of the N most diverse sequences. In the case of multi-domain alignments, this could lead to severely underrepresented regions. E.g. when the first domain is only covered by a few fairly similar sequences and the second by hundreds of very diverse ones, most or all of the similar ones were removed. The '-diff N' option now filters the most diverse set of sequences, keeping at least N sequences in each block of ¿50 columns. This generally leads to a total number of sequences that is larger than N. Speed is similar.

The default is '-diff 100' for hhmake and hhsearch. Speed is similar. Use -diff 0 to switch this filter off.

- The sensitivity for the -global alignment option has been significantly increased by a more robust statistical treatment. The sensitivity in -global mode is now only 0-10% lower than for the default -local option on a SCOP benchmark, i.e. when the query or the templates represent single structural domains. The E-values are now more realistic, although still not as reliable as for -local. The Probabilities were recalibrated.
- A new binary hhalign has been added. It is similar to hhsearch, but performs only pairwise comparisons. It can produce dot plots, tables of aligned residues, and it can sample alternative alignments stochastically. It uses the MAC algorithm by default.
- HHsearch and hhalign can generate query-template multiple alignments in FASTA, A2M, or A3M format with the -ofas, -oa2m, -oa3m options
- Returned error values were changed to comply with convention that 0 means no errors:
  1. Finished successfully
  2. Format error in input files
  3. File access error
  4. Out of memory
  5. Syntax error on command line
  6. Internal logic error (please report)
  7. Internal numeric error (please report)
  8. Other
- Added script `buildali.pl <file>` to automatically build PSI-BLAST multiple sequence alignments, including predicted and DSSP secondary structure. `buildali.pl` is much more robust to alignment corruption by non-homologous fragment by pruning sequences individually from both ends as necessary (J. Söding, unpublished).
- Added script `hhmakemodel.pl <file.hhr>` that parses hhsearch results files and can generate FASTA or PIR multiple alignments or build rough 3D models.
- Moved memory allocation from stack to heap to avoid segmentation faults under some Windows systems.
- Removed a bug due to which pseudocounts where added to HMMer HMMs (which already have their own pseudocounts added). This bug reduced sensitivity for HMMs read in HMMer format.
- Removed a bug due to which the query-template alignments where not displayed on some platforms when output was directed to stdout
- Removed a bug that caused occasional segfaults under SunOS when reading HMMer files
- Added multi-threading (`-cpu <int>`) for Windows x86 platform
- Cleaned up output formatting of summary list for Windows x86
- Stopped support for the Alpha/DEC platform

Is anyone still interested in Mac OSX/PPC or SunOS support?

8 License

The HHsearch/HHblits software package is distributed under Gnu Public License, Version 3.

This means that the HH-suite 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.

You should have received a copy of the GNU General Public License along with this program in the file `LICENSE`. If not, see http://www.gnu.org/licenses/.

References

[1] D. T. Jones, Protein secondary structure prediction based on position-specific scoring matrices, J. Mol. Biol. 292 (1999) 195–202.

[2] D. Marks, L. Colwell, R. Sheridan, R. Hopf, A. Pagnani, R. Zecchina, S. C., Protein 3D structure computed from evolutionary sequence variation, PLoS ONE 24 (2011) 807–814.

[3] M. Remmert, A. Biegert, A. Hauser, J. Söding, Hhblits: Lightning-fast iterative protein sequence searching by hmm-hmm alignment, Nat. MethodsEpub Dec. 25, doi: 10.1038/nmeth.1818.

[4] J. Söding, A. Biegert, A. N. Lupas, The HHpred interactive server for protein homology detection and structure prediction, Nucleic Acids Res. 33 (2005) W244–W248.

[5] A. Hildebrand, M. Remmert, A. Biegert, J. Söding, Fast and accurate automatic structure prediction with HHpred, Proteins 77 (2009) 128–132.

[6] V. Mariani, F. Kiefer, T. Schmidt, J. Haas, T. Schwede, Assessment of template based protein structure predictions in CASP9, Proteins 79(Suppl 10) (1) (2011) 3758.

[7] R. Durbin, S. Eddy, A. Krogh, G. Mitchison, Biological sequence analysis: Probabilistic models of proteins and nucleic acids, Cambridge Univ. Press, Cambridge, 2008.

[8] A. Biegert, J. Söding, De novo identification of highly diverged protein repeats by probabilistic consistency, Bioinformatics 24 (2008) 807–814.

[9] J. N. Battey, J. Kopp, L. Bordoli, R. J. Read, N. D. Clarke, T. Schwede, Automated server predictions in CASP7, Proteins 69 (2007) 68–82.

Good luck with your work!

Johannes Söding, Michael Remmert, Andy Hauser

Gene Center Munich
Ludwig-Maximilians-Universität München
Feodor-Lynen-Strasse 25
81377 Munich
http://www.soeding.genzentrum.lmu.de
soeding@genzentrum.lmu.de