Haystack

Instruction manual and practical hints

1 What can haystack do?

Haystack computes molecular superpositions. A possible large molecule (target) is scanned for the occurrence of a given molecular motif (model) within tolerances. The target may be a large protein. Both model and target are represented by all non-hydrogen atoms.

1.1 What is fed into haystack?

The atom sets of both the model and target must be in a PDB-like format where only the lines starting with `ATOM` and `HETATM` are used.

It is called PDB-like format because not all items legal to PDB will be interpreted by haystack. Prior to be an PDB file becomes input to haystack it should represent a single conformation of a molecule, e. g.

- not contain alternate positions for the same atom, and
- not contain different models.

In fact, haystack is able to select the first model or the first alternative position for a single atom. But in PDB there are more complex alternatives for atom positions which haystack is not able to handle. In any case it is better to check first.

A pair of such PDB-like files is all that is necessary for a first run of haystack:

```
haystack Model.pdb Target.pdb
```

Besides haystack can take many individual parameters, a parameter file, or output files of another run of haystack.

1.2 What is the result?

Haystack puts information onto the screen: parameters, input files, atom numbers, possibly error messages. Of particular interest are the lines informing that output files are saved:

```
Match found with RMS: 0.834166 (0)  saved in out1.pdb
```

The computed superposition is output in a PDB-like format for easy visual control with molecular viewers like rasmol. The REMARK section contains information about the program run, the parameters, the score, and the Cartesian transformation.

The transformed model is the first chain with the letter `m` of the file. As opposed to the PDB format it also contains the atom number of the corresponding target atom in columns 66-71 (or the word `skip` if there is no corresponding atom) and the deviation (in Å) in columns 73-80.

The second chain (after a line TER) is a copy of the target atoms (which can be suppressed by the `-bp` parameter).

1.3 Getting help

A technical help text can be obtained with

```
haystack -h
```

It contains complete information about all parameters and also shows the current setting.

1.4 Version

```
haystack -v
haystack version 1.6.2
  (restricted length/4 word floats/6 word precision floats)
  no checks done, no warnings done
  Compiler: Reading specs from /usr/lib/gcc-lib/i386-redhat-linux/2.96/specs
    gcc version 2.96 20000731
  Compiler flags: -DMAININFO  -Wall -Wno-uninitialized -D_GNU_SOURCE -pipe
    -mpentiumpro -march=pentiumpro -DDOUBLEFLOAT -DLONGDOUBLES -DNOMALLINFO
    -O3 -ffast-math
  Inlining: yes
  Compiled: Fre Apr 6 11:43:34 CEST 2001
```

2 What if? — questions and answers

2.1 Haystack does not finish

Haystack performs two passes. The first pass searches for superpositions, the second pass improves them. Finishing a pass results in a message to the screen. It is easy to see which pass should be accelerated.

2.1.1 Too many anchor matches

One reason for a long first pass runtime can be that there are too many anchor matches. Several million anchor matches are useually O.K. but billions may be too many. First, three atoms of the model are selected as anchor. An anchor match is a set of three atoms in the target which form a similar triangle. The similarity is measured by the differences of the respective side lengths. The differences must not be greater than the number given by the `-ct` option[1]. The number of anchor matches reduces drastically if the value of this parameter is reduced. On the other hand the likelihood to miss good superpositions increases.

2.1.2 Too much time for a single anchor match

For every anchor match the algorithm begins to assign the transformed model atoms with the target atoms. To increase the speed there are several parameters involved.

The maximal distance for a single model atom to its assigned target atom is controlled by the `-md` option.

The average distance for the first m assigned atom pairs must not be larger than $mA + E$, where A is given by the option `-ad` and E is given by the option `-ade`. Again speed can be bought by an increased likelihood to miss good superpositions.

Also the order of the atoms in the model influences the assignment time. Putting the most distant atoms first has a similar effect as reducing the parameter `-ade`. The lines of the model file can be swapped using a text editor with a copy-and-paste function. It is no problem that the normal order of atoms given by the PDB file is destroyed.

Reducing the number of skips (controlled by the parameter `-as`) also increases speed. Equivalently the skip penalty (controlled by the parameter `-sp`) can be enlarged which is only effective if several skips are allowed and the maximal distance (`-md`) is low[2].

2.2 Haystack finishes but finds no superpositions

The parameters of the previous section sould be changed in the opposite direction.

[1] The tolerances for the three side lengths can also be given in separately. Then the `'-ct' option must be followed by three numbers.`

[2] Normally the skip penalty is equal to the maximal distance.

2.3 Too many superpositions

The number of superpositions can be restricted by decreasing the values of `-ct`, `-md`, `-ad`, `-ade`, `-as`, or by increasing `-sp` as described in section 2.1.

The usual course of the program is to store the solutions internally until the end when they are sorted and the best superpositions are written to files. If there are many results but the program does not finish it is helpful to store the superpositions as they are computed, with the parameter `-rt 2`. The results can be read in again with the `-R` option for the second pass if they are output in the first pass.

2.4 The second pass does not finish

The parameter `-bh` controls the number of superpositions which the first pass passes to the second pass. Lowering this number reduces the runtime. On the other hand the chance to miss good superpositions increases slightly.

A single optimization should be very rapid. If this is not the case the parameters `-it`, `-im`, and `-itm` can be increased.

The first and the second pass can be separated into different program runs. Setting `-it 2` executes the first pass only, setting `-1` executes the second pass only. Superpositions computed from the first program run are read in with the `-R` parameter. The output files do not completely contain the input files — model and target have to be given in both runs.

For this purpose there is a compressed binary format instead of the usual PDB-like format of the output files. It is written if the option `+so` is set. These binary file are read in with the `-S` option like PDB-like output files are read in with the `-R` option.

A typical session looks like (using bash)

```
haystack -I Params.ini model.pdb target.pdb -it +so
for file in *.sup; do
  haystack -I Params.ini model.pdb target.pdb -1 -S $file;
done;
```

2.5 Can the result be further improved?

The default parameters are a trade-off between computing time and quality. If the computation is rapid we can increase the accuracy of the optimization as follows.

We can lower the value of the parameter `-it`. It is the minimal improvement of a single optimization step. The unit is Angstroms. It must be non-negative. Setting it to zero might lead to a very long chain of very slight improvements which might only represent floating point peculiarities of the CPU. The gradient method has another parameter — the minimal improvement for a single numeric step `-itm`. To make sense it must be lower than the value of the `-it` parameter.

3 Special features

3.1 One-one-uniqueness

Usually one target atom must not be assigned to more than one model atoms. If it makes sense for the structural question this prerequisite can be dropped.

Two parameters have to be changed `-mm` which controls the first pass and `-im` which controls the second pass. The default matching techniques assumed `-mm 3 -im 11` must be given.

3.2 More than one anchor

Sometimes a unlucky choice of the anchor leads to bad superpositions. The program can also be ordered to try an number given by the option `-na` of anchors. The running time will be approximately multiplied by the same number with a increased likelihood of good superpositions. This is particularly important the higher the number of skips (option `-as`) is.

3.3 Setting the anchor manually

If we are suspicious that the choice of the anchor avoided finding good superpositions we can demand a certain anchor. The atom numbers of the model can be given in the parameter `-c`. However, guessing good anchors requires much handwork. It is easier to try several anchors with the `-na` option.

In the PDB-like output file there is a line starting with

```
The stabilizing points of the model ...
```

This is the anchor given as atom numbers in the model.

3.4 Heuristics for the automatic choice of the anchor

When `-fm 0` is given, the following method is used. First, we look for two points with the longest distance from each other. Second, we look for a third point such that the height of the resulting triangle is maximal.

When `-fm 1` is given, two more parameters have to be taken care of: `-fp` is a suggested length for the distance of two points (in Angstrom) and `-fs` is the suggested length for the height of the triangle.

4 Features for larger searches

4.1 Time dependency

It is possible to set a time boundary with the parameter `-to` (in seconds). When the time runs out during the first pass the program stops searching for new superpositions. However, the superpositions computed already are used for the second pass. The second pass may use the same amount of time (which is rarely used). After that the superpositions are written to files as usual.

4.2 Search agenda

Normally, the first pass searches depth-first. This saves temporary system memory. It is also possible to restrict the number of anchor matches with the parameter `-nc`. In this case all possible anchor matches are collected before any superpositions are computed. The anchor matches are sorted by the measure of congruence of the anchor and the anchor match triangle. This is done efficiently by putting them into a hash-list while they are collected.

Then the best anchor matches are used to compute superpositions. It is done in the order of the anchor match accuracy. This allows for a much better runtime control than by the rather unpredictable result when changing the parameter `-ct`.

The main drawback is the temporary memory consumption and the time to control the hash-list. It is not advisable to give `-nc` values larger than 10000. So the usability of this option is restricted to the case of very large models or rough and fast searches in large databases.

4.3 Parameter files

There is an convenient alternative way to enter parameters—with a parameter file. The easiest way to produce a parameter file is to call haystack with the option `-hi`. The output is a commented parameter file with the actual parameters. If simply `haystack -hi` it is a file with the default parameters. If `-hi` is the last in a row of options all the command line options are transformed to paramters of a parameter file.

The option to read in a parameter file is `-I <file>`. It is possible to overwrite some of the settings in the command line. These options must follow the `-I` option.

Parameter files are simple ASCII-text files and can be viewed and edited with a simple text editor. The format is very simple: a parameter line starts with the parameter name followed by the value. A coment line starts with a `#`.

4.4 Predefined parameter files

For an easy beginning we provide the parameter files `Params1.ini` ... `Params7.ini`. They are numbered such that lower numbers stand for faster and higher numbers stand for more thorough program runs.

5 Computers and operating systems

Haystack is available for Pentium compatible machines under the operating system LINUX. It is a static executable and has no dependencies on installed libraries.

The last executable can be found on `http://www.charite.de/bioinf/haystack`.