User Guide to SCANPS - Scan Protein Sequence Database Version 2.3.9 - July 2002

Geoffrey J. Barton

School of Life Sciences
University of Dundee
Dow St.
Dundee DD1 5EH
Scotland, UK.

Tel: (44) 1382-345860
e-mail: geoff@compbio.dundee.ac.uk

1 What is SCANPS?

SCANPS is a program for comparing a protein sequence to a database of sequences. It has the following features:

1. Full Smith-Waterman style searching with a single sequence.
2. Multiple domain matches found against each database sequence.
3. Iterative profile searching similar in concept to PSI-BLAST, but with full dynamic programming on each cycle for additional sensitivity.
4. Significance of matches calculated "on the fly" for each search.
5. Efficient implementation on Intel CPUs by using MMX and SSE instructions.
6. Output of each search as pairwise alignments and multiple alignments.

2 Citing SCANPS

A paper is in preparation that describes the program, its statistics and benchmarking. In the meantime, please cite,

Barton, G. J. (2002) "SCANPS Version 2.3.9 User guide", University of Dundee, UK.

One underlying algorithm used in SCANPS is described in:

Barton, G. J. (1993) CABIOS, 9, 729-734.

3 Disclaimer

4 Development History and Acknowledgements

The bulk of SCANPS was written between 1989 and 1997 while I was a Royal Society University Research Fellow in the Laboratory of Molecular Biophysics at the University of Oxford, UK. Parallel code for the SGI Challenge and Origin was developed during 1996 on the SGI Challenge at the Wellcome Trust Centre for Human Genetics, University of Oxford. I acknowledge the technical assistance from Silicon Graphics, in particular Nick Camp and Pam Bremer, in helping me get good parallel performance from the SGI shared memory machines.

Development of iterative searching was done at EMBL-EBI during 1999. Development of MMX/SSE code and statistics were done in collaboration with Steve Searle and Caleb Webber. Other developments in progress include an MPI parallel version for Linux and a version for OpenMP.

5 Installing SCANPS

Until publication of the manuscript describing SCANPS, the program is available in binary form for Linux computers and compiled for Intel PIII/IV and Athlon XP processors.

5.1 Computer requirements

Version 2.3.9 of SCANPS is available for Linux running on X86 hardware (e.g. Pentium III/IV or Athlon XP). It has been tested on Debian, RedHat and SuSE flavours of Linux with 2.4 kernels. The current distribution is compiled with the gcc 3.1 compiler.

SCANPS reads the entire sequence database into memory, so you must have enough physical memory (RAM) on the computer to store the database.

5.2 Step-by-step installation

1. Unpack the scanps distribution into a directory. Assuming we do this in /usr/local then:

   ```
   tar zxf scanps_2.3.9.tar.gz
   ```

   will create a directory structure like this:

```
/usr/local/scanps_2.3.9
/usr/local/scanps_2.3.9/doc      (Documentation)
/usr/local/scanps_2.3.9/bin      (Executables for scanps)
/usr/local/scanps_2.3.9/dat      (Configuration and data files for scanps)
/usr/local/scanps_2.3.9/mat      (Pairscore matrix files - e.g. BLOSUM)
/usr/local/scanps_2.3.9/examples (Example output discussed in manual)
/usr/local/scanps_2.3.9/db       (Example swissprot database files)
```

the binary directory contains a README file that explains what the different binaries are for. You should provide a link from your normal bin directory to the appropriate binary for your system. For example if you normally put local binaries in /usr/local/bin:

```
cd /usr/local/bin
ln -s /usr/local/scanps_2.3.9/bin/xpscanps_16K scanps
```

will put the standard MMX and SSE scanps for Linux in your path.

2. Configure the dat/scanps_defaults.dat file. Find the following lines in the file:

```
MATRIX_DIR  /usr/local/scanps/mat/
GENERAL_DIR /usr/local/scanps/dat/
DB_DIR      /usr/local/scanps/db/
```

edit the lines MATRIX_DIR and GENERAL_DIR to specify the "mat" and "dat" subdirectories of the scanps distribution. edit the DB_DIR line to point at the directory where you will put scanps database files that you will search against. There is an example SWISS-PROT database in the directory scanps/db for testing, but you should build your own databases before using scanps for real scans. For speed, the database directory should be on a disk local to to the computer on which you will run scanps.

3. Set the SCANPSDIR environment variable to point at the scanps/dat directory. If you use the c-shell then type:

```
setenv SCANPSDIR /usr/local/scanps/dat/
```

if you use the bash shell, then do:

```
set SCANPSDIR=/usr/local/scanps/dat/
```

4. Type /usr/local/bin/scanps (or whatever the path to scanps is that you have defined) and you should see something like:

```
/usr/local/bin/scanps
---------------------------------------------------------------------
SCANPS Version: 2.3.9 (Wed Jul 17 14:02:27 BST 2002)
Authors: GJ Barton, SMJ Searle, C. Webber:  University of Dundee and EMBL-EBI
---------------------------------------------------------------------
See scanps_manual for FULL information.
Normal scans:
MODE  200:  Protein vs protein with iteration and simple gaps.
MODE  202:  Protein vs protein with iteration and affine gaps.

Experimental methods:
MODE  210:  Protein vs protein with iteration and variable simple gaps.
MODE  212:  Protein vs protein with iteration and variable affine gaps.

MODE 99:  Build a scanps database - see scanps_manual.

Assuming SCANPS has been installed correctly, the following should work...

MODE 200 scan with FASTA format file query.seq against SwissProt
scanps -s query.seq -qseq_format 1 -bdb swissprot > logfile

To scan a fasta format database in any mode change -bdb swissprot to:
-d swissprot.fa -db_format 1

Common options - for full list, see scanps_manual
Change pairscore matrix from default to another matrix, e.g. PAM100
Add:  -m PAM100 to command line
Change gap penalties - e.g. to length dependent of 9 and creation of 13
Add:  -ugd 0 -ld_pen 9 -li_pen 13

---------------------------------------------------------------------
SCANPS Version: 2.3.9 (Wed Jul 17 14:02:27 BST 2002)
Authors: GJ Barton, SMJ Searle, C. Webber:  University of Dundee and EMBL-EBI
---------------------------------------------------------------------
```

Assuming you have set the DB_DIR line in scanps/dat/scanps_defaults.dat to point at the scanps/db directory, then you should now be able to run a test of SCANPS.

```
cd /usr/local/scanps/examples  (or wherever you have put scanps)
/usr/local/bin/scanps -s test.fa -bdb sprot > mytest.log
```

This will take all the default values for a search, output options etc and put the results in the file 'mytest.log'. You can look at this output file and compare it to the file 'test.log' in the same directory. As a quick check of your SCANPS installation, you can 'diff' the two files There should only be minor differences similar to those shown below:

```
cd /usr/local/scanps/examples
diff test.log mytest.log
1c1
< # SCANPS Scan:    Tue Jul 23 10:43:05 2002
---
> # SCANPS Scan:    Tue Jul 23 10:14:46 2002
1283c1283
< # Scan Completed at: Tue Jul 23 10:43:05 2002
---
> # Scan Completed at: Tue Jul 23 10:14:47 2002
1287,1291c1287,1291
< # Load database:                      0.6     0.6     0.7     0.7
< # Scan database:                      9.3     9.9     9.3    10.0
< # Output scores:                      0.5    10.3     0.5    10.4
< # Generate and write alignments:      0.3    10.7     0.3    10.8
< # Millions of cell updates/sec in scan (CPU): 323.68 (Elapsed): 323.68
---
> # Load database:                      1.2     1.2     3.5     3.5
> # Scan database:                      9.7    10.8    10.8    14.3
> # Output scores:                      0.8    11.6     1.0    15.3
> # Generate and write alignments:      0.4    12.0     0.6    15.9
> # Millions of cell updates/sec in scan (CPU): 310.33 (Elapsed): 278.72
```

If you get the above output or similar, then all is happiness. Now lets go through some of the different scanps options, examine the output in more detail and how to control what you see in the output file.

6 Building SCANPS Databases

Although SCANPS will read a normal FASTA formatted sequence file as the database to search, it runs more efficiently if you first build its own binary database files. This is particularly important for the MMX/SSE versions of the program. SCANPS is used to build its own databases. For example:

```
scanps -mode 99 -d trembl.fa -bdb trembl
```

will take a FASTA formatted sequence database file called "trembl.fa" and create the SCANPS binary database and index files "trembl.bix" and "trembl.bsq" in the directory /db/scanps (or whatever you specified in the scanps_defaults.dat file above.

Once you have your binary database file and a sequence you want to scan you specify the two on the command line - e.g.:

```
scanps -s test.fa -bdb trembl > test_trembl.log
```

the output of the scan is written to standard output which in this case is redirected to the file test_trembl.log.

7 Running SCANPS

SCANPS is controlled by the defaults file (scanps_defaults.dat) that we have already met above, and/or command line options. The details of the defaults file and command line options are explained in later sections, but for now, lets just go through some examples, look at the output and find out the most common control options.

7.1 Standard SCANPS search

SCANPS has different modes, that implement different types of search. In the current distribution, there are four MODES which have the numbers - 200, 202, 212 and 210. We will concentrate on MODES 200 and 202 which are the most commonly used. MODE 200 searches a protein sequence against a protein sequence database, either with or without iteration and with simple, length-dependent gap-penalties. MODE 202 does the same, but with a more complicated gap-penalty model that has a creation and extension penalty. MODE 200 is the fastest, MODE 202 will typically be more sensitive and give longer alignments.

The command:

```
scanps -mode 200 -s test.fa -probcut 30 -bdb sprot.fas
```

produced the output in the file: test_200_probcut30.log. The output has a number of sections, the *preamble* the *score list*, the *pairwise alignments*, the *multiple alignment* and the *trailer*. The sections are exaplained below:

7.1.1 Output preamble

```
# SCANPS Scan:    Thu Jul 18 12:55:08 2002
# SCANPS Version: 2.3.9 (Wed Jul 17 14:02:27 BST 2002)
# Authors:        G. J. Barton, S. M. J. Searle, C. Webber
#                 University of Dundee and EMBL-EBI
# Query file:     test.fa
# Query ID:       SCANPS_TEST
```

```
# Query Title:    - Annexin domain test seq for SCANPS
# Query Length:   74
# SCANPS MODE:    200
# Matrix file:    /homes/geoff/c/scanps/gjscanps/mat//BLOSUM50
# Matrix title:   NCBI Matrix - see file for type
# LD penalty:   6.00
# Database name:                  sprot.fas
# Created on:                     Mon Jun 24 22:43:18 2002
# Number of sequences:            110788
# Number of residues/bases in database: 40678337
# Eval Cutoff:                    30
# FIT_A: -1.42464e+06 1.4246e+06 -163342
# FIT_B: 0 0.781179
# Eval Cutoff for profile:      0.1
# Number of iterations:           1
# Profile Weight Method:          1
# Pos Weight Factor:              1.500000
# Pid Threshold:                  0.970000
#
# Intel MMX and SSE instructions supported by chip.
#
```

and contains information about the scan that was done, and which database was searched with which parameters. The output is intended to be easy to parse with a perl script. Lines starting with a # are comments, values follow a colon on the line, and are identified by a string between the # and the colon, or when there are multiple values on a line, between the last value and a colon.

7.1.2 Score list

The next part of the output contains the score list, ranked by Evalue. This is bounded by the key words "Start_Scores" and "End_Scores". Only the first 17 and last 6 scores are shown here:

```
# Start_Scores: ----------------------------------------------------
   1  450 56.95  2.05e-20 ANX3_RAT        (P14669) Annexin III (Lipocort
   2  439 55.40  9.61e-20 ANX3_MOUSE      (O35639) Annexin III (Lipocort
   3  392 48.78  7.26e-17 ANX3_HUMAN      (P12429) Annexin III (Lipocort
   4  255 29.48  1.75e-08 ANX5_MOUSE      (P48036) Annexin V (Lipocortin
   5  245 28.07  7.15e-08 ANX5_RAT        (P14668) Annexin V (Lipocortin
   6  245 28.07  7.18e-08 ANX5_HUMAN      (P08758) Annexin V (Lipocortin
   7  245 28.06  7.2e-08  ANX5_BOVIN      (P81287) Annexin V (Lipocortin
   8  247 27.78  9.52e-08 ANXB_HUMAN      (P50995) Annexin A11 (Annexin
   9  239 27.19  1.72e-07 ANX8_MOUSE      (O35640) Annexin A8 (Annexin V
  10  239 27.19  1.72e-07 ANX8_HUMAN      (P13928) Annexin A8 (Annexin V
  11  243 26.97  2.15e-07 ANX6_BOVIN      (P79134) Annexin VI (Lipocorti
  12  236 26.79  2.57e-07 ANX5_CHICK      (P17153) Annexin V (Lipocortin
  13  228 26.77  2.61e-07 ANX1_CHICK      (Q92108) Annexin I (Lipocortin
  14  240 26.44  3.63e-07 ANX6_HUMAN      (P08133) Annexin VI (Lipocorti
  15  237 26.38  3.88e-07 ANXB_RABIT      (P33477) Annexin A11 (Annexin
```

```
  16  237 26.38  3.88e-07 ANXB_MOUSE       (P97384) Annexin A11 (Annexin
  17  237 26.38  3.88e-07 ANXB_BOVIN       (P27214) Annexin A11 (Annexin

  lines deleted here.

  55  189 20.10  0.000207 ANX9_HUMAN       (O76027) Annexin A9 (Annexin 3
  56  187 19.91  0.000251 ANX4_FRAAN       (P51074) Annexin-like protein
  57  186 19.66  0.000319 AN11_COLLI       (P14950) Annexin I, isoform P3
  58  184 19.39  0.000419 ANX9_MOUSE       (Q9JHQ0) Annexin A9 (Annexin 3
  59  116 12.21  0.551    ANX2_PIG         (P19620) Annexin II (Lipocorti
  60  100 8.82   16.3     Y069_ARCFU       (O30167) Hypothetical protein
# End_Scores: -----------------------------------------------------
```

The columns are: 1: Rank, 2: Raw Score, 3: (Intermediary score - ignore, this will be deleted from a future release of SCANPS) 4: Evalue, 5: ID of database sequence, 6: Title of database sequence.

How many scores you see is controlled by the **probcut**, **nptt**, and **max_nout** command line options. **nptt** means "numbers, print to threshold" and is a toggle. For example, to print all results down to an Evalue of 100 you would add:

```
-aptt 1 -probcut 100
```

on the command line (the default is -aptt 1 -probcut 10). If you want to print all scores down to a rank, e.g. the top 50, rather than controlled by Evalue, then do the following:

```
-aptt 0 -max_nout 50
```

This is not recommended, unless you are trying to suppress a huge output. It is better to control output through the -probcut option.

7.1.3 Pairwise Alignments

By default, SCANPS will output a single paiwise alignment (two-sequence alignment) between the query sequence and each database sequence in the score list. For example:

```
# Start_Alignments_Rank: 1
# Query: SCANPS_TEST Target: ANX3_RAT Number_of_Alignments: 1
# Target_Title: (P14669) Annexin III (Lipocort
# N: 1 Raw_Score: 450 Query_Length: 74 Target_Length: 324 Evalue: 1.9793e-20
        *****************.*************************. ****
      1 YPGFNPSVDAEAIRKAIRGIGTDEKTLINILTERSNAQRQLIVKQYQEAY      50
     16 YPGFNPSVDAEAIRKAIKGIGTDEKTLINILTERSNAQRQLIVKHIQEAY      65
```

```
        ************************
     51 EQALKADLKGDLSGHFEHVMVALI      74
     66 EQALKADLKGDLSGHFEHVMVALI      89

# End_N: 1
# End_Alignments_Rank: 1

# Start_Alignments_Rank: 2
# Query: SCANPS_TEST Target: ANX3_MOUSE Number_of_Alignments: 1
# Target_Title: (O35639) Annexin III (Lipocort
# N: 1 Raw_Score: 439 Query_Length: 74 Target_Length: 323 Evalue: 9.29946e-20
        ****.**************.*************************** **
      1 YPGFNPSVDAEAIRKAIRGIGTDEKTLINILTERSNAQRQLIVKQYQEAY      50
     15 YPGFSPSVDAEAIRKAIRGLGTDEKTLINILTERSNAQRQLIVKQYQAAY      64

        ** ** *****************.
     51 EQALKADLKGDLSGHFEHVMVALI      74
     65 EQELKDDLKGDLSGHFEHVMVALV      88

# End_N: 1
# End_Alignments_Rank: 2
```

Each alignment is contained within a pair of "Start_Alignments" and "End_Alignments" labels. The rank of the alignment is also shown. The line beginning "# N: 1" idenifies this as the first alignment for this pair of sequences. SCANPS can output more than one alignment for each pair of sequences, so this allows these to be separated and parsed in the output. The pairwise alignment is output in a fairly conventional way with the "-" character used to denote gaps. The line above the alignment shows "*" for identities and "." for pairs of amino acids that score positive in the pairscore matrix that was used in the search.

There are various ways to control this pairwise output. For example, the number of residues per line can be modified with the -output_len command line option.

You can adjust how many alignments you want to see by using the -aptt and -max_aout commands. These work in a similar way to the -nptt and -max_nout described above. For example to only output the first 20 alignments rather than alignments for every pair shown in the score list do:

```
-aptt 0 -max_aout 20
```

Often one want so suppress pairwise alignments entirely. You can do this with:

```
-aptt 0 -max_aout 0
```

At the moment, there is no way to set a probablility threshold on pairwise alignments that is separate from the -probcut threshold.

7.1.4 Multiple Alignment Output

The multiple alignment output is a "pseudo" multiple alignment that has the same length as the query sequence. It shows the query sequence as the first row of the alignment, with the database sequences below. Insertions in database sequences are simply discarded in this output, so it should NOT be used for full multiple alignment analysis (but see below for a way to do this).

The alignment is output in blocks of 50 by default, though this can be changed with the MULTIPLE_OUTPUT_LENGTH command line option.

```
# Multiple alignment Start for iteration: 0
# Format: Simple
# Residues_per_line: 50
#
# ID            Evalue   Start   End   Len
#                                          1
SCANPS_TEST     0            1    74    74: YPGFNPSVDAEAIRKAIRGIGTDEKTLINILTERSNAQR
ANX3_RAT        1.98e-20    16    89   324: YPGFNPSVDAEAIRKAIKGIGTDEKTLINILTERSNAQR
ANX3_MOUSE      9.3e-20     15    88   323: YPGFSPSVDAEAIRKAIRGLGTDEKTLINILTERSNAQR
ANX3_HUMAN      7.02e-17    15    88   323: YPDFSPSVDAEAIQKAIRGIGTDEKMLISILTERSNAQR
ANX5_MOUSE      1.7e-08     10    83   319: FPGFDGRADAEVLRKAMKGLGTDEDSILNLLTSRSNAQR
ANX5_RAT        6.92e-08     9    82   318: FSGFDGRADAEVLRKAMKGLGTDEDSILNLLTARSNAQR
ANX5_HUMAN      6.94e-08    11    84   319: FPGFDERADAETLRKAMKGLGTDEESILTLLTSRSNAQR
ANX5_BOVIN      6.97e-08    11    84   320: FPGFDERADAETLRKAMKGLGTDEESILTLLTSRSNAQR
ANXB_HUMAN      9.21e-08   198   270   505: -PGFDPLRDAEVLRKAMKGFGTDEQAIIDCLGSRSNKQR
ANX8_MOUSE      1.67e-07    21    91   327: ---FNPDPDAETLYKAMKGIGTNEQAIIDVLTKRSNVQR
ANX8_HUMAN      1.67e-07    21    91   327: ---FNPDPDAETLYKAMKGIGTNEQAIIDVLTKRSNTQR
ANX6_BOVIN      2.08e-07   304   378   618: -PGFNPDADAKALRKAMKGLGTDEDTIIDIITHRSNAQR
ANX6_BOVIN      5.57e-07    38   107   618: -----PAADAKEIKDAISGIGTDEKCLIEILASRTNEQH
ANX5_CHICK      2.49e-07    14    85   321: -P-FDARADAEALRKAMKGMGTDEETILKILTSRNNAQR
ANX1_CHICK      2.53e-07    35   107   130: -PNFDPSADVSALDKAITVKGVDEATIIDILTKRTNAQR
ANX6_HUMAN      3.51e-07    16    89   672: FPGFDPNQDAEALYTAMKGFGSDKEAILDIITSRSNRQR
ANX6_HUMAN      8.19e-07    92   161   672: -----PACDAKEIKDAISGIGTDEKCLIEILASRTNEQH

lines deleted


ANX7_DICDI      0.000126   166   231   462: QIKREFSAKYSKDLIQDIKSETSGNFEKCLVALL
ANX2_XENLA      0.000152    33   103   339: DIAFAFHRRTKKDLPSALKGALSGNLETVMLGLI
ANX2_XENLA      0.00126    107   174   339: LDIQNYRELFKTELEKDIMSDTSGDFRKLMVAL-
ANX1_RODSP      0.000155    38   111   345: HLKAVYQETGE-PLDETLKKALTGHIQELLLAMI
ANXD_HUMAN      0.00016     10    83   315: QIKQKYKATYGKELEEVLKSELSGNFEKTALALL
ANXD_HUMAN      0.000184    86   155   315: IAIKEYQRLFDRSLESDVKGDTSGNLKKILVSLL
ANX9_HUMAN      0.0002      31   104   338: LISRNFQERTQQDLMKSLQAALSGNLERIVMALL
ANX4_FRAAN      0.000243     8    76   314: EIRAAYEQLYQEDLLKPLESELSGDFEKAV----
AN11_COLLI      0.000309    37   107   341: RIKAAYHKAKGKSLEEAMKRVLKSHLEDVVVALL
ANX9_MOUSE      0.000405    35   104   338: LISRAFQERTKQDLLKSLQAALSGNLEKIVVALL
#
# Multiple alignment End for iteration: 0
#
```

The first two columns should be self-explanatory. The start and end refer to the first and last residue position within the database sequence, while

"Len" refers to the database sequence length so that you can see if there are any pathologically short sequences in the alignment. This alignment is used in iteration to build a profile for subsequent searches, which sequences are included in the alignment is controlled by the **probcut2** command. By default **probcut2** is set to 0.1.

You can obtain a FASTA formatted sequence file that contains the complete sequence fragments found in the database search by adding

```
-pff 1 -frag_file_out frags.fa
```

to the command line. This will create a file called "frags.fa" that contains the fragments between Start and End in each line of the multiple alignmnent, but without any internal deletions. You can feed this file to clustal or another multiple alignment program for further analysis.

7.1.5 Trailer

This is just some timing information at the moment - this can be useful to diagnose performance problems on a computer.

```
# Scan Completed at: Tue Jul 23 10:14:47 2002
#
# Times:                              (Seconds)
# Load matrix, query and index:        0.0     0.0     0.0     0.0
# Load database:                       1.2     1.2     3.5     3.5
# Scan database:                       9.7    10.8    10.8    14.3
# Output scores:                       0.8    11.6     1.0    15.3
# Generate and write alignments:       0.4    12.0     0.6    15.9
# Millions of cell updates/sec in scan (CPU): 310.33 (Elapsed): 278.72
```

The above times were obtained on a 1GHz PIII processor for the test.fa sequence against sprot. The important numbers to look at when making comparisons are the first column and the last row. For example, the time taken to scan the database was 9.7 seconds, which corresponds to 310 Million Cell updates/second. If you see big numbers by the "Load database" row, then you have a disk access problem or you do not have enough RAM to store the database. On most machines the swissprot database should load in under 2 seconds.

7.2 Obtaining more than one alignment per sequence pair

SCANPS can output more than one alternative alignment for each sequence in the score list. You turn this option on by adding:

```
-top_only 0
```

to the command line. The output in the file test_200_probcut30_toponly0.log is an example of output produced with this command. The test.fa query is an Annexin domain, and many Annexins contain multiple copies of this domain. SCANPS finds these and outputs them both as pairwise alignements, and as the multiple alignment.

KNOWN BUG - July 2002: If you set -top_only 1, you should only see one alignment per pair of sequences. However, some sequence pairs will show more than one. This is a known bug and will get fixed in due course.

7.3 SCANS with Affine Gaps

All the above applies, just set -mode 202 instead of -mode 200.

7.4 Iterative Searching

Iterative searching will normally be able to find more remote similarities to the query sequence than a single sequence search. This is illustrated in an example below.

Iterative searching is enabled by adding the -niter command on the command line. For example:

```
scanps -s hahu.fa -mode 200 -niter 5 -bdb sprot > niter_test.log
```

will run SCANPS with 5 iterations. Examples of scans with 5 iterations, that use the human alpha haemoglobin sequence as a query are shown in the files with "niter5" in their name.

An iterative search starts just the same as a non-iterative search, the query sequence is compared to the database and the score list, pairwise and multiple alignment outputs are reported. The multiple alignment is then used to create a query "profile" that contains information about the types of amino acid seen at each position in the alignment. This profile is then searched against the database, a score list, pairwise and multiple alignments are output and the process is then repeated. The iterations will stop either when the number of iterations has been reached, or if two successive iterations find exactly the same sequences.

The key parameter that controls iterative searching is **probcut2**. This controls which sequences from a search will be included in the profile with which the next search is done

The file: hahu_202_probcut30_niter5.log shows an iterative search with human alpha haemoglobin. This file includes pairwise output, but normally one would switch this off with -aptt 0 -max_aout 0 on the command line in order to minimise the output file. In the scan, probcut2 was set to 0.1 by default and in Iteration 0, there are 695 sequences that score above the probcut2 value:

```
deleted lines

674  144 22.25  2.4e-05  MYG_PHOSI        (P30562) Myoglobin
675  143 22.05  2.94e-05 MYG_MOUSE        (P04247) Myoglobin
676  142 21.85  3.6e-05  MYG_ELEMA        (P02186) Myoglobin
677  141 21.76  3.94e-05 GLB3_MYXGL       (P02209) Globin III
678  140 21.53  4.93e-05 GLBA_SCAIN       (P14821) Globin II, A chain (H
679  135 20.56  0.00013  GLP2_GLYDI       (P21659) Globin, polymeric com
680  136 20.53  0.000135 GLBC_CAUAR       (P80018) Globin C, coelomic
681  117 19.79  0.000281 HBE_MACEU        (P81042) Hemoglobin epsilon ch
682  129 19.35  0.00044  GLB_NASMU        (P31331) Globin (Myoglobin)
683  128 19.06  0.000586 GLB_CERRH        (P02215) Globin (Myoglobin)
684  124 18.33  0.00121  GLP1_GLYDI       (P23216) Globin, major polymer
685  121 17.72  0.00223  GLB_BUSCA        (P02214) Globin (Myoglobin)
686  120 17.69  0.00231  Y211_AQUAE       (O66586) Hypothetical globin-l
687  121 17.66  0.00237  GLBA_ANATR       (P14395) Globin I alpha chain
688  100 16.52  0.00742  HBB_PAPAN        (Q9TSP1) Hemoglobin beta chain
689  100 16.52  0.00742  HBB_COLGU        (Q9TT33) Hemoglobin beta chain
690   98 15.70  0.0168   HBO_MACEU        (P81041) Hemoglobin omega chai
691  111 15.29  0.0254   GLB3_LUMTE       (P11069) Globin III, extracell
692  107 14.79  0.0419   GLBP_CHITH       (P11582) Globin CTT-E/E' precu
693  106 14.73  0.0443   GLBB_RIFPA       (P80592) Giant hemoglobins B c
694   90 14.47  0.0578   HBB_PONPY        (Q9TT34) Hemoglobin beta chain
695  105 14.40  0.0617   GLBB_SCAIN       (P14822) Globin II, B chain (H
696  102 13.64  0.132    GLBY_CHITP       (P18968) Globin CTT-Y precurso
697   99 13.31  0.184    GLB_APLJU        (P14393) Globin (Myoglobin)

more deleted lines
```

the next iteration (Iteration 1) reports 777 sequences to be above the probcut2 threshold. At the end of the score list, is a report on which new sequences are found, and which (if any) sequences now fall below the threshold. As shown here:

```
# End_Scores: ---------------------------------------------------
#
# Reported in iteration 0 but below the threshold in this iteration (1)
#
# Reported in this iteration (1) but not in the previous iteration (0)
#
 689  133 30.75  4.9e-09  GLBB_ANATR (P04251) Globin I beta chain
 690  128 29.35  1.99e-08 GLB1_SCAIN (P02213) Globin I (Dimeric hemoglobin) (HBI)
 692  127 29.02  2.75e-08 GLB4_LUMTE (P13579) Globin IV, extracellular (Erythrocruori
 693  124 28.21  6.18e-08 GLB_APLKU (P02211) Globin (Myoglobin)
 694  124 28.20  6.28e-08 GLB1_ANABR (P02212) Globin I
 695  122 27.62  1.12e-07 GLB2_ANATR (P14394) Globin IIB
 696  122 27.62  1.12e-07 GLB1_ARTSX (P19363) Globin E1, extracellular
 698  120 27.01  2.06e-07 GLBM_ANATR (P25165) Globin, minor
 699  119 26.78  2.6e-07  GLB_APLJU (P14393) Globin (Myoglobin)
 700  119 26.75  2.69e-07 GLB3_TYLHE (P13578) Globin IIB, extracellular (Erythrocruor
 701  119 26.64  2.97e-07 GLBH_CHITP (P29242) Globin CTT-VIIB-7 precursor
 702  129 26.51  3.41e-07 HMPA_ALCEU (P39662) Flavohemoprotein (Hemoglobin-like prote
 703  118 26.44  3.63e-07 GLB2_LUCPE (P41261) Hemoglobin II (Hb II)
 704  118 26.36  3.96e-07 GLBH_CHITH (P12550) Globin CTT-VIIB-7 precursor
 706  117 26.19  4.7e-07  GLB_DOLAU (P09965) Globin (Myoglobin)
 707  116 25.77  7.09e-07 GLBV_CHITP (P29243) Globin CTT-V precursor (HBV)
 708  114 25.32  1.12e-06 GLP3_GLYDI (P21660) Globin, polymeric component P3
 711  111 24.46  2.64e-06 GLB_APLLI (P02210) Globin (Myoglobin)
 712  111 24.34  2.99e-06 GLBZ_CHITH (Q23761) Globin CTT-Z precursor (HBZ)
 714  109 23.76  5.31e-06 GLBZ_CHITP (P29245) Globin CTT-Z precursor (HBZ)
 715  118 23.64  6.02e-06 HMPA_VIBCH (Q9KMY3) Flavohemoprotein (Hemoglobin-like prote
 716  118 23.60  6.27e-06 HMPA_BACSU (P49852) Flavohemoprotein (Hemoglobin-like prote

 lines deleted...
```

The next iteration (2) finds 801 sequences above the probcut2 threshold, the third iteration pushes this up to 802, but Iteration 4 does not change the output.

8 Getting more from SCANPS

The sections above provide an introduction to running SCANPS and give most of the commonly used options. What follows is a more detailed explanation of how SCANPS is controlled and a complete options list. Once you have got used to the basics of SCANPS it would be worth reading through these sections to find out other useful features or options for customisation.

8.1 The Basics

The interface to SCANPS follows a fairly standard Unix command style. If you are used to using Unix, then this should be easy. By default, output goes to stdout and so can be piped to other programs for processing.

If you are not happy with the Unix command line, then it should be easy for someone to hide this interface behind a WWW form, or other windowing interface. This has been done at the European Bioinformatics Institute (http://www.ebi.ac.uk). We also run a server at Dundee (http://www.compbio.dundee.ac.uk).

SCANPS is controlled by a set of keyword, value commands. The commands may either be specified in a defaults file (scanps_defaults.dat), or on the command line. Values specified on the command line override the defaults set in the scanps_defaults.dat file. The scanps_defaults.dat, scanps_alias.dat and scanps_gapdefs.dat files must all reside in the directory pointed to by the environment variable SCANPSDIR. If SCANPSDIR is not defined, then the program assumes that these files are in the user's current directory. Thus, an installation can have a central set of defaults which may be overridden by individual users who may copy and modify their own copies of the scanps_defaults.dat, scanps_alias.dat and scanps_gapdefs.dat files.

8.2 The scanps_defaults.dat file

The scanps_defaults.dat contains settings for valid commands. When you first install scanps, you will have to modify this file to define the correct locations for your directories. For example:

```
DB_DIR      /gjb/delly/databases/scanps/
MATRIX_DIR  /gjb/delly/gjb/c/scanps/gjscanps/mat/
GENERAL_DIR /gjb/delly/gjb/c/scanps/gjscanps/dat/
MATRIX_FILE PAM250
MAX_NSEQ 5000000
CODON_FILE codon.dat
MATRIX_FILE nmd.mat
MAX_NSEQ 500000
MAX_SEQ_LEN 400000
LD_PEN 1
LI_PEN 8
DB_FORMAT 0
```

This defines the directories for various files. See the specific command summary below for further details.

Commands in the scanps_defaults.dat file may either be the full command name or an alias as defined in the scanps_alias.dat file (see below). Usually, it is best to use the full command name in the scanps_defaults.dat file. Aliases are there to ease typing commands on the command line. Note that ALL commands and their aliases are case sensitive.

8.2.1 File names

File names in the scanps_defaults.dat file MUST NOT be fully qualified. Thus,

```
MATRIX_FILE nmd.mat
```

will look in the users' current directory for the file. If the matrix file was in /data/local/scanps/matrices you would have to define:

```
MATRIX_DIR /data/local/scanps/matrices/
```

as well.

This is also true of the binary database files. The directory for these may be defined by DB_DIR. e.g.

```
DB_DIR /data/local/databases/scanps/
BDB_ROOT pir66
```

would look for the files pir66.bix and pir66.bsq in the directory /data/local/databases/scanps.

The only other directory that can be specified in this way is GENERAL_DIR, this holds "other" files needed by the program. At the moment, the only file that is put in GENERAL_DIR is the CODON_FILE for use with DNA vs protein comparisons. More of that later...

8.3 The scanps_alias.dat file

You will not normally modify this file. For the sake of completeness, the format of the file is described here.

The scanps_alias.dat file allows aliases to be defined for any of the valid commands. For example, here is an excerpt from scanps_alias.dat.

```
MAX_NSEQ        max_nseq        #Maximum number of sequences allowed
TIME            time            #Set to 1 to record CPU times
MODE            mode            #Type [scanps HELP modes] to see available modes
QSEQ_FORMAT     QSEQ_F          qseq_format     qseq_f   #0 = PIR format, 1 = FASTA format
DB_FORMAT       DB_F            db_format       db_f     #As for QSEQ_FORMAT
MAX_SEQ_LEN     max_seq_len     #Max allowed length for an amino acid sequence
```

Each command name is followed by a list of aliases, then a # followed by some optional descriptive text.

8.4 Command line switches

All the commands in scanps_alias.dat are available for use on the command line. You can either specify the command or its alias, with or without a preceding - symbol. For example,

```
QSEQ_FILE hahu.seq
-qseq_file hahu.seq
-s hahu.seq
```

all do the same thing on the command line assuming the standard scanps_alias.dat file has not been modified.

A typical scan might be started by:

```
scanps -s hahu.seq -bdb swall -ugd 0 -ld_pen 8 -mode 0 > hahu.out
```

This would scan the sequence in file "hahu.seq" against the binary database called "swall" using a length dependent gap penalty of 8 with whatever default matrix file was specified in the scanps_defaults.dat file. The ugd 0 turns off the use of default gap penalty combinations stored in the file "scanps_gapdefs.dat".

More examples are given below.

8.5 Controlling the scan

SCANPS has a series of different MODEs. Each mode does a different job. Note that not all of the modes shown here are in Version 2.3.9, but are discussed for the sake of completeness (they may come back in future releases).

```
MODE 0 Scan protein sequence against protein sequence database
with simple gap penalty.  Default and fastest method.
MODE 2 As for MODE 1, but with Affine gaps.
MODE 20 DNA vs protein database with frameshifts (simple gaps).
MODE 22 DNA vs protein database scan with frameshifts (affine gaps).
MODE 99 Build scanps binary database files from FASTA or PIR format files.
MODE 100 Extract sequences from database (reads output of earlier scan).

In Version 2.3.2 two new modes are introduced:

MODE 200 is like MODE 0 but does iterative searching.
MODE 202 is like MODE 2 but does iterative searching.

MODES 200 and 202 will replace MODE 0 and 2 in a future release of SCANPS.
```

There are a variety of controlling commands that affect the scan and the output. By default, all results are printed to stdout. You can override this by using the STDOUT command to specify a different file for output. STDERR can also be redefined.

Commands affecting the scan - look in the file scanps_alias.dat to see alternative names for these commands and see APPENDIX II for a full list of commands:

```
QSEQ_FILE   is the query sequence file
qseq_format  defines the format (0 for pir, 1 for fasta)
bdb_root defines the root name of the binary database to search
ld_pen length dependent gap-penalty
li_pen length independent gap-penalty
matrix_file pairscore matrix file (e.g. Dayhoff, BLOSUM etc )
use_gapdefs set to 1 to enable the gap defaults in scanps_gapdefs.dat
                set to 0 to allow the values of ld_pen and li_pen to work.
```

See the full list in APPENDIX II for other valid commands.

8.6 Commands to help find problems

VERBOSE Setting this to something > 0 will give messages to stderr as the program executes. Setting it to a big number will give more messages. VERBOSE 100 should output all messages. NOTE: If you set VERBOSE and you still don't get useful messages, try setting it first to 1, then to be absolutely sure of getting all messages, set VERBOSE in the scanps_defaults.dat file rather than on the command line.

TIME Setting this to a number > 0 will output various timings to stderr. Again, bigger numbers should give more messages.

8.7 The scanps_gapdefs.dat file

This file sets default gap penalty combinations for each matrix type and mode. If USE_GAPDEFS is set to 1, then scanps will take the default gap penalty combinations from the scanps_defaults.dat file for the given matrix and MODE. These values will OVERRIDE any penalties specified in the scanps_defaults.dat file, or on the command line. If you wish to specify non-default gap penalty combinations, then set USE_GAPDEFS (ugd) to 0.

For example:

If the scanps_gapdefs.dat file has the following entries:

```
#Format is:
# mode:matrix_name:ld_pen:li_pen:fs_pen:fse_pen
#
0:nmd.mat:8:0:0:0
2:nmd.mat:4:12:0:0
0:md.mat:8:0:0:0
2:md.mat:0.2:12:0:0
#
```

and USE_GAPDEFS = 1, then

scanps -s hahu.seq -mode 200 -m nmd.mat -bdb swissprot

will do a scan of the swissprot database with the nmd.mat matrix and a penalty of 8

scanps -s hahu.seq -mode 202 -m md.mat -bdb swissprot

would do an affine scan with the penalties of ld_pen 0.2, li_pen 12.

scanps -s hahu.seq -mode 2 -m md.mat -bdb swissprot -ugd 0 -li_pen 12 -ld_pen 4

would do the affine scan with penalties of 12, 4 rather than 12, 0.2.

WARNING: With the exception of the values for BLOSUM50 the scanps_gapdefs.dat file contains numbers that I guessed might be appropriate for the given matrix and mode. They are almost certainly not the optimum choices.

8.8 Scanning Protein with DNA sequence

Note: This is not availble in Version 2.4.x, but may come back in a later version.

Version 2.3 allows a DNA sequence to be compared to a protein database (MODE 22). Scanps first translates the DNA in three forward reading frames to create a pseudo sequence of amino acids where every base is an amino acid residue or STOP. This sequence is then compared to each protein sequence using a dynamic programming algorithm. The result is an alignment of the DNA with a protein that can include frameshifts.

Additional adjustable parameters are:

FS_PEN and STOP_WEIGHT. These are defined in the command summary APPENDIX II.

Work is in progress to establish suitable values for these parameters. I have found that a large and negative value for STOP_WEIGHT is appropriate. FS_PEN should be set larger than LI_PEN and LD_PEN.

Here is an example of the comparison of the primary transcript of human alpha haemoglobin versus the corresponding protein sequence. The DNA has been modified to introduce a single frameshift error, an additional G at position 356.

Here is the command line:

```
scanps -s hahunucerror.fasta -d hahu.fasta -db_format 1 -mode 22 \
-ugd 0 -m PAM250 -ld_pen 0.2 -li_pen 12 -fs_pen 24 -qseq_format 1
```

we are scanning a database of only one sequence (-d hahu.fasta), overriding the gap-penalty defaults for this MODE and matrix (-ugd 0) and supplying our own gap-penalties (-ld_pen, -li_pen, -fs_pen). The query sequence is in FASTA format, but we have set the default sequence format to 0 (PIR) in the scanps_defaults.dat file. Accordingly, we need to override the default on the command line (-qseq_format).

Here is the output:

```
# SCANPS Scan: Mon Aug 11 15:36:04 1997
# SCANPS Version: 2.3 (Fri Jul  4 16:16:41 BST 1997)
# Author:  G. J. Barton,  University of Oxford, UK
# Query file: hahunucerror.fasta
# Query ID: HSHBA4
# Query Title: Dummy title inserted by gseq_fasta
# Query Length: 835
# SCANPS MODE: 22
# Matrix file: /gjb/delly/gjb/c/scanps/gjscanps/mat//PAM250
# Matrix title: NCBI Matrix - see file for type
# LD penalty:   0.20
# LI penalty:  12.00
# FS  penalty:  24.00
# MAX number of scores output: 1
# MAX number of alignments output: 10000
# MIN score to output: 769.70
# MIN score of alignments to  output: 769.70
# Number of residues/bases in database: 141
# Score type: ln() scores
# Start_Scores: ----------------------------------------------------
   1 769 HAHU Hemoglobin alpha chain - Human and chimpanzees
# End_Scores: ------------------------------------------------------
# ------------------------------------------------------------------
# Start_Alignments_Rank: 1
# Query: HSHBA4 Target: HAHU Number_of_Alignments: 1
# Target_Title: Hemoglobin alpha chain - Human and chimpanzees
# N: 1 Raw_Score: 566 Query_Length: 835 Target_Length: 141 ln()_Score: 769
     41 gctcggaaagaggtgagggcgggtggggcgatgtccttgcgttcgcacac
        ttcccaacataccggatgcacgaagcactagggctcgcacgccccgcagc
        ggttccgcccgccgtgccgctcgttggcgggacccccccgccgccgccga     190
     41 VLSPADKTNVKAAWGKVGAHAGEYGAEALER!GSLPCSDPGSSPARTHRP      90
        *******************************
      1 VLSPADKTNVKAAWGKVGAHAGEYGAEALER                        31

    191 ctagcgcgcacacctgtccaatcttcaaaattcctgcacgtgcgagcgaa
        cccttccacacccaccccggtttctcccacatcatatgagccatagagaa
```

```
       caccgcgcaccctctttccggcgcccccgcccgcccgccctcgtgcccgg     340
    91 PSTVLAPDPNPTPHSASPRRMFLSFPTTKTYFPHFDLSHGSAQVKGHGKK     140
                           ******************************
    32                     MFLSFPTTKTYFPHFDLSHGSAQVKGHGKK      61

   341 ggggcGaagggcgggacagctgcagccgcaccggcgataGtgggcgctgt
       tcact cactcataatcactcctgatacaatgtactata gccggagcgc
       gcccg cccggcgccgccggccgccgcgcgtggcgcccg aggcggatgg     486
   141 VADAL^TNAVAHVDDMPNALSALSDLHAHKLRVDPVNFK^!AAGRERSGS     188
       ***** *********************************
    62 VADAL^TNAVAHVDDMPNALSALSDLHAHKLRVDPVNFK^                99

   487 aggagcttcgagtcgtcggtcaccccgtggcacttgcccactccgacggc
       ggatccccagggcggtgataggggtgcgcctcttccattgagtttctcca
       gcgggtctgcaaacgggggcgggggcgcagccctagcacccgggcgccc     636
   189 RGEMAPSSQGRGSRGLREV!RRRRLRAWAALTLFSAQLLSHCLLVTLAAH     238
                                            *************
   100                                      LLSHCLLVTLAAH     112

   637 ccggtacggcgtcgatcgtgaagcatatc
       tccatccctacctaattcctgcttccaag
       cccgcctggcccgcgcgttgccggccact     723
   239 LPAEFTPAVHASLDKFLASVSTVLTSKYR     267
       *****************************
   113 LPAEFTPAVHASLDKFLASVSTVLTSKYR     141

# End_N: 1
# End_Alignments_Rank: 1
# -------------------------------------------------------
# Scan Completed at: Mon Aug 11 15:36:04 1997
#
# Times:                              (Seconds)
# Load matrix, query and index:        0.0     0.0     0.0     0.0
# Load database:                       0.0     0.0     0.0     0.0
# Scan database:                       0.1     0.1     0.9     0.9
# Sort results :                       0.0     0.1     0.0     0.9
# Output scores:                       0.0     0.1     0.0     0.9
# Generate and write alignments:       0.1     0.2     0.1     1.0
# Millions of cell updates/sec in scan (CPU):   1.18 (Elapsed):   0.13
```

The output shows the base triplets arranged VERTICALLY above the corresponding amino acid residue. The protein sequence is shown conventionally below this. Frameshifts are indicated by a caret symbol "^" and the bases involved in the frameshift are shown in uppercase. You may see odd effects around frameshifted gaps in alignments. My advice is to look carefully at any frameshifted gaps to see if there might be alternative alignments by taking a different reading frame in the region, or shifting the frameshift gap a base or so to either side. With any dynamic programming method, there may be more than one, equally valid alignment in a region, but the program only reports one solution. There are more possibilities for such alternatives in DNA vs protein comparisons than for DNA v DNA or protein v protein.

As well as MODE 22, there is MODE 20. MODE 20 does not have affine gap penalties and as a consequence it is about a factor fo 3 faster than MODE 22. Penalties for frameshifts are length dependent in MODES 20 and 22, thus if FS_PEN is 8, then a single frameshift costs 8 and a double costs 16.

9 APPENDIX I - Revision notes

```
2.1

First full parallel version with searching for protein vs protein database.
Includes code to build and use binary database for fast loading.
Includes code for sorting the results.
ln() scores.
alignment output options.
All within the same program.

2.2

Fix get_fasta.c so that it will read a FASTA file that is missing a title.
Fix DB_DIR to permit missing / at end of directory name.
Add MATRIX_DIR command.

2.2.1

Add option to read NCBI format matrix files.  MATRIX_TYPE command.

2.2.2

Various odds and ends.  Added EXTRACT options to enable the sequence
of high scoring database hits to be fished out of the database.

2.2.3

Add routines to compare DNA to protein, with frameshifts.
Fix mysterious looking  bug when generating alignments with affine gaps.
Tidy up timing routines.
Add option to print full length titles.

First released version?  Nope.

2.2.4

Small changes not worth mentioning

2.2.5

Add MODE 20 for fast frameshifting DNA vs Protein comparisons.  Add
HQUERY option at complile time to allow VERY big query sequences
(e.g. 2 Megabases).  Add COMPLEMENT_QUERY option to allow scanning with
complement of the query.  Modify MODE 22 code to eliminate FSE_PEN and replace
with simple length-dependent penalty for frameshift gaps.
Add MODE 100 to allow sequences to be extracted from the database following
```

```
a scan that produced no alignments.

2.2.6

Add statistical estimates based on extreme value distribution.  This is based on the
statistics used in the programs FASTA and SSEARCH3 though the implementation is
different.  No statistics in alignment output as yet,
just the score list.

2.3

Small bug fixes. Add the licensing routines.  Tidy up the distribution.


2.3.1

Small changes to the way in which the probcut and max_nout options interact.
The program now allows max_nout to control the number of sequences output
when probcut >1.  Bug removed for probcut ==1 case.

2.3.2

Add new statistical routines with on-the-fly EVD fitting.  Add iterative searching methods.
Replace MODES 0 and 2 with code from MODE 200 and MODE 202.

2.3.9

Re-write the manual to include description of iterative searching and standard
protein and protein profile searching methods.
```

10 Plans for additions - in no special order

Extend new statistics to the DNA options - reactivate the DNA searching options. Option to read multiple alignment files as query. Option to read "profiles" as query. Add option to read and apply HMMs. Option to scan a database of alignments. Allow comparison of protein sequence to DNA database.

11 APPENDIX II - Alphabetical list of scanps commands

This section is simply a sorted and extended version of the scanps_alias.dat file. New commands may from time to time get added to that file, so look there if something does not makes sense.

```
APPLY_INDEX     apply_index

Flag when creating database in mode 99 if =1 then the characters used
to represent the amino acids in the binary sequence database are
converted to allow fast indexing of the pairscore matrix.  This is the
default.  If you use a lot of different pairscore matrices with
different index strings, then set this to zero.

APRINT_TO_THRESHOLD aprint_to_threshold aptt
```

```
Flag to specify if alignments will be printed down to the probability
threshold defined by PROBCUT.

AUTO_CORNER auto_corner # Obsolete

BDB_ROOT bdb_root bdb

Name of root name for binary database files - e.g. sprot would mean
there is a file called sprot.bix and sprot.bsq.

BLOCK_FILE_OUT block_file_out  bfo

Name for a file to contain multiple alignment output in AMPS block file format.

CALEB_LPLBUCKETWIDTH buckwidth  # logprodlen bucket width
CALEB_MINBUCK   minbuck         # minimum number needed for a stats bucket.
CALEB_MINRANGE  minrange        # minimum number of buckets needed for fitting.
CALEB_PRINT     caleb_print     # print out the caleb stats files.  1 = yes, 0 = no.

Options related to the Webber and Barton on-the-fly statistics.

CODON_FILE codon_file #Translation table for DNA to Protein

COMPLEMENT_QUERY comp_query cq # scan with complement of DNA query

CUT_CONSTANT cut_constant # Obsolete
CUT_CORNERS cut_corners # Obsolete

DB_DIR db_dir

Directory for storing database files.

DB_FILE db_file  d

Sequence database filename - name for file that is not a binary database file.

DB_FORMAT DB_F db_format dbf

As for QSEQ_FORMAT - the format of the database sequence.

DO_SORT       do_sort         # Obsolete.

DO_STATS do_stats

This should always be set to 2.

DUMP_STATS      dump_stats      # Obsolete.

EPB_PEN epb_pen # Obsolete.
EPL_PEN epl_pen # Obsolete.
EPR_PEN epr_pen # Obsolete.
EPT_PEN ept_pen # Obsolete.

EXTRACT_FILE extract_file efile # Obsolete.
EXTRACT_SEQ extract_seq eseq # Obsolete.
EXTRACT_SEQ_FORMAT extract_seq_format # Obsolete.
```

```
FAST           fast           # Obsolete.
FIND_REPEATS find_repeats # Obsolete.

FIT_FILE fit_file # Obsolete.
FIT_TYPE       fit_type       # Obsolete.

FRAG_FILE_FORMAT frag_file_format fff
```

Format of the frag file optionally output for a profile alignment 0 for PIR 1 for FASTA.

```
FRAG_FILE_OUT frag_file_out  ffo # the name of the frag file to output

FSE_PEN fse_pen # Obsolete.
FS_PEN fs_pen FSC_PEN fsc_pen
```

Penalty for frame shift creation in DNA vs Protein modes.

```
GAP_CHARACTER GAP_CHAR gap_character gap_char
```

Character used to show a gap.

```
GENERAL_DIR gen_dir
```

Directory to store miscellaneous information in - e.g. the codon.dat file.

```
LD_PEN ld_pen e_pen E_PEN
```

Length dependent (extension) gap penalty

```
LICENSE_DIR     license_dir     # Obsolete.

LI_PEN li_pen c_pen C_PEN
```

Length independent (creation) gap penalty

```
LOG_SCORES log_scores lns # Obsolete.

MATRIX_DIR matrix_dir md
```

Directory that contains pairscore matrices.

```
MATRIX_FILE matrix_file m
```

Name of pairscore matrix file.

```
MATRIX_FORMAT matrix_format mf
```

Format of matrix file =0 for PIR =1 for NCBI/BLAST

```
MAX_AOUT max_aout maout
```

Max number of alignments to output - works in conjunction with -aptt option.

```
MAX_BLOC_SEQ max_bloc_seq max_blc_seq # Not used.

MAX_ID_LEN max_id_len
```

Maximum length for sequence identifiers

MAX_NOUT max_nout mnout

Max number of sequences to output in score list (works with -nptt).

MAX_NSEQ max_nseq

Maximum number of sequences allowed in program - this must be bigger than the database size.

MAX_SEQ_LEN max_seq_len

Max allowed length for an amino acid sequence. In most compiled programs this has a precompiled limit. If the program is compiled without MMX/SSE support or parallel processing support, then this variable can be set.

MAX_TITLE_LEN max_title_len #Maximum length for sequence titles

MIN_ASCORE min_ascore # Obsolete.

MIN_LEN min_len # Minimum length allowed for an alignment

MIN_NSTATS min_nstats # Obsolete.

MIN_SCORE min_score # Minimum score required for score list to be output

MIXED_MODE mixed_mode mm #

When set to 1 and mode 202 is active, first iteration is mode 202
others are mode 200.

MM_LD_PEN mm_ld_pen mmpen

Length dependent penalty to use in MIXED_MODE on iterations 1-N

MODE mode

Specify type of scan or processing - see manual above.

MULTIPLE_OUTPUT_LENGTH multiple_output_length mol

Output length for multiple alignments

NCHUNK nchunk

MPI chunk size - number of sequences to send to each processor in each
parallel chunk.

NITER niter

Number of iterations to do.

NPRINT_TO_THRESHOLD nprint_to_threshold nptt

If 1 then output is down to probcut threshold if 0 then MAX_NOUT is used

NRANS nrans # Obsolete.

OSEQ_FILE oseq_file # Obsolete.

OUTPUT_LENGTH output_length out_len OUT_LEN

Output length for pairwise alignments.

PCUT pcut # Obsolete.

PEN_FACTOR pen_factor penf

Factor to modify observed gap penalties in modes 210/212

PIDTHRESH pidthresh pidt

percentage identity threshold for inclusion in profile.

POSWGTFAC poswgtfac pwf # profile position specific weighting factor

PRECISION precision precis PRECIS

Precision - all floats multiplied by this number. Usually 100.

PRED_FILE pred_file # Obsolete.

PRINT_ALIGN print_align # Obsolete.

PRINT_BLOCK_FILE print_block_file pbf

If 1 then will print an AMPS block file for the multiple alignment.

PRINT_FRAG_FILE print_frag_file pff

If 1 then will print a file of seq fragments in format. See manual.

PRINT_FRQ_TABS print_frq_tabs

Print out the frequency tables in an iterated search.

PRINT_PROFILES print_profiles

Print out the lookup tables in iterated search.

PROBCUT   probcut   eval

Probability or eval cutoff for do_stats 1 or 2.

PROBCUT2 probcut2   eval2

Probability or eval cutoff for inclusion in profile.

PROB_TYPE prob_type

prob type 0=probs 1 =oldevals 2 =newevals - only prob_type 2 is tested.

```
PROFILE_WEIGHT_METHOD profile_weight_method pwm

0 for original weighting 1 for HH.

QBLC_FILE qblc_file QBLC # Obsolete.

QBLC_FORMAT QBLC_F qblc_format # Obsolete.

QSEQ_FILE qseq_file QSEQ qseq s S

Query sequence file.

QSEQ_FORMAT QSEQ_F qseq_format qsf

0 = PIR format, 1 = FASTA format

READ_PRED read_pred  # Obsolete.
ROBSIM robsim  # Obsolete.
RUN_SW_MIN run_sw_min # Threshold required before NALL algorithm runs

SAVE_PROFILES save_profiles # Obsolete.
SCAN scan # Obsolete.
SEC_FILE sec_file # Obsolete.
SHOW_BLURB      show_blurb

If set to 0 then no info is printed to output file.

SHOW_IDENT_WIDTH show_ident_width siw

The width of the ident string to output.

SHOW_LEN show_len lens # Obsolete.

SHOW_PMATRIX  show_pmatrix spm

Print pairscore matrix to output file (requires SHOW_BLURB).

SHOW_TITLES show_titles titles

Print titles in output.

SHOW_TITLES_WIDTH show_titles_width stw

The width of the title string to output.

SPECIAL special # Obsolete.

STDERR stderr # redefinition of stderr
STDIN stdin # redefinition of stdin
STDOUT stdout # redefinition of stdout

STOP_WEIGHT stop_weight sw  #

Weight for matching amino acid to STOP codon (MODE 22).

TEST #Test - no alias
```

```
TIME time # Set to 1 to record CPU times

TOP_CUTFRAC     top_cutfrac     # Obsolete.
TOP_ONLY top_only

1 for only top scoring alignment in each pair. 0 for all alignments
down to probcut.

USE_GAPDEFS     use_gapdefs     ugd

if = 1 then use scanps_Gapdefs.dat values if posssible

VERBOSE verbose

If >=1 then print various messages as program runs.

VINGRON_FILE vingron_file # Obsolete.
```