User and Developer's guide for

(Physical-chemical property based motif analyzer)
Version 2.0

 Bin Zhou, Venkatarajan S. Mathura & Prof. Werner Braun

UTMB, Galveston

CONTENTS

1 INTRODUCTION

PCPMer

(Physical-chemical property based motif analyzer)

Version 2.0

Bin Zhou, Venkatarajan S. Mathura & Prof.Werner Braun
UTMB, Galveston

PCPMer is a software package that can be applied to identify important sequence regions that are evolutionarily conserved in terms of their physical-chemical properties. A multi-dimensional analysis of 237 relevant physical-chemical properties of amino acids revealed that 5 dimensional representations are possible (PCP descriptors or vectors E1-E5)[1]. This five-dimensional property space can be constructed such that the amino acids are in a similar spatial distribution to that in the original high-dimensional property space. Properties that correlate well with the five major components were hydrophobicity, size, preferences for amino acids to occur in alpha-helices, number of degenerate triplet codons and the frequency of occurrence of amino acid residues in beta-strands. Distances computed for pairs of amino acids in the five-dimensional property space are highly correlated with corresponding scores from similarity matrices derived from sequence and 3D structure comparison. PCPMer calculates conservation of these five vectors using a multiple alignment. It calculates relative entropy of distribution of five vectors in equally spaced five bins between the protein family sequence of interest and random occurrence of amino acids (natural frequency). The relative entropy (R) cutoff is used to filter out insignificant regions in the protein sequence. In order to group significantly conserved positions, empirical parameters like G and L-cutoff are used [2]. G-cutoff restricts the number of insignificant positions between two significant positions

in a motif and L-cutoff excludes motifs smaller than the L significant members. A bayesian based scoring scheme that uses distribution of scores in the true positive and the database of interest helps in identifying related protein sequences that shares similar motifs. Thus PCPMer can be used for both identifying motifs in a protein and data mine for related members in sequence database. Currently the following functions are included in PCPMer:

1. **Create Motifs**
2. **Search for motifs and related sequence in a database**
3. **Search for the highest scoring motifs in a set of sequence**
4. **Create motifs and search database**
5. **Create motifs and score set of sequences**
6. **Create macro file of MOLMOL**

REFERENCES

[1]
Venkatarajan, M.S., Braun W., 2001, "New quantitative descriptors for amino acids based on multidimensional scaling of a large number of physical-chemical properties", J Mol Modeling 7:445-453

[2]
Venkatarajan, M.S., Schein, C. H., Braun W., 2003, "Identifying physical chemical property based sequence motifs in protein families and superfamilies: Application to DNase I related endonuclease", Bioinformatics vol. 19:1381-1390, 2003.

[3]
Venkatarajan, M. S. 2002 "Automated generation of sequence motifs and 3D models for proteins and their applications" Doctoral Disseration, University of Texas Medical Branch, Galveston.

2 DISTRIBUTION

PCPMer can be obtained freely for academic research purpose. This package is not allowed to be modified or redistributed without the knowledge of the authors. Separate commercial license is available. To obtain the software please send an email or contact:

Prof. Werner Braun

Sealy Center for Structural Biology,
University of Texas Medical Branch,
301 University Blvd.
Galveston, TX 77555.
email: werner@newton.utmb.edu
phone: 409-7476810
fax: 409-7476850

3 VERSION AND HISTORY

Version 1.0 (2003)

Version 2.0 (2004) parallel code, optimization of entropy values & molmol display of motifs

and more details will be added those section in next releases.

4 COPYRIGHT AND LIABILITY

COPYRIGHT:

PCPMer is copyrighted to Dr. Bin Zhou, Dr. Venkatarajan S. Mathura and Werner Braun at the University of Texas Medical Branch, Galveston a component of The University of Texas System. This program is not covered under public license and hence several restrictions apply.

- ?? The authors have exclusive rights to determine appropriate users and usage.
- ?? The package must be requested using the 'request form' and should be used only by the person for the purpose it was requested.
- ?? Any intention of modifying the software must be informed to the authors in written and must be approved by the authors.
- ?? Redistribution of this software in any form is prohibited.
- ?? This software (PCPMer) and its components may not be used for commercial purpose unless written approval is granted in writing from the authors.
- ?? It is the intention of the authors to make this program available freely to academic institution for research purpose only.
- ?? Appropriate citations of this program and related publications must be made in cases where the program is used.

LIABILITY:

IN NO EVENT SHALL THE AUTHORS OR ANY INSTITUTIONS IN WHICH THEY WORK (INCLUDING, BUT NOT LIMITED TO, UNIVERSITY OF TEXAS SYSTEM) BE LIABLE TO ANY PARTY FOR DIRECT, INDIRECT, SPECIAL, INCIDENTAL, OR CONSEQUENTIAL DAMAGES, INCLUDING LOST PROFITS, ARISING OUT OF THE USE OF THIS SOFTWARE AND ITS DOCUMENTATION, EVEN IF THE

AUTHORS HAVE BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGE. THE AUTHORS SPECIFICALLY DISCLAIMS ANY WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE. THE SOFTWARE PROVIDED HEREUNDER IS ON AN "AS IS" BASIS, AND THE AUTHOR HAS NO OBLIGATIONS TO PROVIDE MAINTENANCE, SUPPORT, UPDATES, ENHANCEMENTS, OR MODIFICATIONS.

5 INSTALLATION AND REQUIREMENTS

Hardware requirement:

- Atleast Pentium III processor or higher.
- RAM atleast 128MB.

Software requirement:

- Operating systems: LINUX, UNIX, IRIX.
- PERL 5.0 or higher versions
- C shell preferred

INSTRUCTIONS:

```
To unzip and untar the distribution, type the following.
 $unixprompt> gunzip PCPMerpack2.0.tar.gz
and then type
 $unixprompt> tar -xvf PCPMerpack2.0.tar

This will create a directory PCPMerpack. Change current
directory to PCPMerpack.
```

INSTALL:

```
Basically, to build and install PCPMer from sources, you enter three
commands:

      $ ./configure
      $ make
      $ make install

The `configure' shell script attempts to guess correct values for
various system-dependent variables used during compilation.  It uses
those values to create a `Makefile' in each directory of the PCPMer
package.  It also creates some `.h' files containing system-dependent
definitions.  Finally, it creates a shell script `config.status' that
you can run in the future to recreate the current configuration, a
file `config.cache' that saves the results of its tests to speed up
reconfiguring, and a file `config.log' containing compiler output
(useful mainly for debugging `configure').

If you need to do unusual things to compile the PCPMer package, please
try to figure out how `configure' could check whether to do them, and
mail us diffs or instructions to so they can be considered for the next
release.  If at some point `config.cache' contains results you don't
want to keep, you may remove or edit it.

The file `configure.in' is used to create `configure' by a program
```

called `autoconf'.  You only need `configure.in' if you want to change it or regenerate `configure' using a newer version of `autoconf'.

1. `cd' to the directory containing the PCPMER package source code type

   ```
   $ ./configure
   ```

   to configure PCPMER for your system. If you're using `csh' on an old version of System V, you might need to type

   ```
   % sh ./configure
   ```

   instead to prevent `csh' from trying to execute `configure' itself.

   If you're building PCPMER on Windows using CYGWIN, type

   ```
   $ bash ./configure
   ```

   instead.

   Running `configure' takes awhile. While running, it prints some messages telling which features it is checking for.

   If `configure' reports an error or some bad result, check the files `config.log' for diagnostics.

2. Check the `Makefile', `PCPMer', and `config.h' files generated by `configure'. Most settings should be guessed correctly by the `configure' program.  You may, however, wish to edit the settings, or re-run `configure' with special options.

3. Now type

   ```
   $ make
   ```

   to build PCPMER. Any modern MAKE flavors should do, but for incremental reconstruction, GNU MAKE is required on most systems.

4. Check the `make' output for compiler errors and warnings.

   If you see any compiler errors or warnings, please see the sections `Warnings during build' and `Errors during build', below.

5. Type

   ```
   $ make install
   ```

   This installs
   - the PCPMER executable `PCPMer' in some public place (usually in `/usr/local/bin/')

6. You can remove the program binaries, libraries and object files

from the source directory by typing

```
$ make clean
```

7. You can remove PCPMER from your system by typing

```
$ make uninstall
```

This undoes all effects of a previous `make install'.

By default, `make install' will install the PCPMER files in `/usr/local/bin', `/usr/local/man', etc. You can specify an installation prefix other than `/usr/local' by giving `configure' the option `--prefix=PATH'.

You can specify separate installation prefixes for architecture-specific files and architecture-independent files. If you give `configure' the option `--exec-prefix=PATH', the package will use PATH as the prefix for installing programs and libraries. Documentation and other data files will still use the regular prefix.

TO EXECUTE:

If you installed as root:

```
$unixprompt > PCPMer
```

If you installed as non-root:

Either include the PCPMerpack in your path or every time call the program with absoulte location.

```
$unixprompt >/PCPMerpackDir/PCPMer
```

If you cannot gain root permission, you still install PCPMer and include the PCPMerpack directory manually in the path or call PCPMer executable using absolute path i.e:

```
$unixprompt > /PCPMerpackDir/PCPMer
```

6. FILES PRESENT IN THE DISTRIBUTION

FILE NAME	DESCRIPTION
0COPYRIGHT	Copyright information and terms of use
0README	Provides brief summary of PCPMer
0INSTALL	Installation instructions
PCPMer	PCPMer executable created when installation script is executed
PCPMer.pl	PCPMer front engine
PCPmotifmaker.pm	Module containing routines for Motif detection
PCPmotifminer.pm	Module containing routines for Motif search
errormsg.pm	Module containing error messages
install.sh	Installation script
multialign.pm	Module containing routines to manipulate multiple alignment
vectorlib.pm	Parameters for amino acids (PCP descriptors)
/examples	Example directory (one example in ex0 directory)
/doc	Document directory

6 TERMS AND DEFINITIONS

PCP descriptors:

Five dimensional vectors that were adequate to represent distribution of natural amino acids in 237 dimensions.

Relative entropy:

A measure of dissimilarity of distribution. Here used for defining the significance of conservation.

G-cutoff:

An empirical parameter that defines number of insignificant positions in motif between two significant positions.

L-cutoff:

An empirical parameter that defines the minimum number of significant residue position in a motif.

Motif:

A set of consecutive or closely occurring residue position that are conserved in terms of their physical-chemical property in the evolution of the protein family.

Sequence string:

Residues in one letter code

Motif score:

Simple Lorentzian based score between a motif profile and window.

Effective score:

Score for each highest scoring motif window in a sequence using bayesian statistics that utilizes motif score distribution in the true positive and the database

Combined score:

Addition of effective scores for each motif for each sequence after applying score filter.

Score filter:

Filter applied to calculate combined score for each sequence. One can use raw scoring, or include those motifs that score above average scores in the database or above a cutoff.

7 INPUT/OUTPUT

7.1 INPUT FILES:

7.1.1 Multiple alignment file

Must be generated with CLUTALW. Currently the program accepts only the ALN format. Sequence of your interest must always on the top in the multiple alignment. Spaces in the sequence name are not allowed. Codes for amino acids other than the uppercase single letter code of 20 amino acids are not allowed. Gaps must be indicated by "-" and not by any other symbol.

```
CLUSTAL W (1.82) multiple sequence alignment


APE_H._sapiens                  ALYEDPPDHKTSPSGKPATLKICS-------------------
APE_M._musculus                 VLYEDPPDQKTSPSGKSATLKICS-------------------
APE_R._norvegicus               VLYEDPPDQKTSASGKSATLKICS-------------------
APE_B._taurus                   VLYEDPPDQKTSPSGKSATLKICS-------------------
APE_C._griseus                  FLYEDAPDNKTSPGGKLATLKICS-------------------
APE_D._melanogaster             TTVTLDKDAFALPADKEFNLKICS-------------------
EXO_C._elegans                  --------------NNQKSWKFVC-------------------
APE_S._pombe                    -------------------MRILS-------------------
ARP_M._graminicola              -------------------MRLTT-------------------
APE_D._discoideum               ASVSIAIDNLDEPKVEENQMKIIS-------------------
```

Example of multiple alignment generated using APE sequences with CLUSTALW.

7.1.2 Sequence Database file

Text file containing sequences in the FASTA format is allowed. Pre-processing of sequence data like removing redundant sequence above a percentage identity will result in higher speed. Use caution while using sequence database or more than 5000.

7.2 OUTPUT FILES:

Each output file will have unique extension. The output file type can be identified using this extension.

7.2.1 Log file (*.PCPlog)

The log file records the entire session when the program operates. It also records user input parameters, progress in the program and output. The log file summarizes PCPMer run.

```
PCPMer - Physical-chemical property based motif analyzer
                  Version 2.0
  COPYRIGHT (2003) Bin Zhou, Venkat Mathura & Werner Braun
     University of Texas Medical Branch, Galveston
                   LOG FILE
          Thu Jan 29 16:30:49 2004


Your selection is 4. This will :Create and search for motifs in a
database.
Reading multiple alignment file : APEALIGN.aln
Relative-entropy user defined = 1.25
G-value user defined = 2
L-value user defined = 4
Creating global profile file :EXAMPLE.PCPgprf
Creating motif Nlist file    :EXAMPLE.PCPNlist
Creating motif profile file  :EXAMPLE.PCPprf
Creating motif list file  :EXAMPLE.PCPSlist


************************************
MOTIF DETAILS:

#PARAM R_G_L:1.25:2:4:
#MOTIF :  0:   36*42*71*72*103*125*177*181*184*216*253*274
#MOTIF :  1:   20 LKICSWNVDGLRA 32
#MOTIF :  2:   47 PDILCLQETK 56
#MOTIF :  3:   84 EGYSGVGLLSR 94
#MOTIF :  4:  110 DQEGRVI 116
#MOTIF :  5:  129 YVPNA 133
#MOTIF :  6:  139 RLEYRQRW 146
#MOTIF :  7:  163 LVLCGDLNVAH 173
#MOTIF :  8:  189 GFTPQER 195
#MOTIF :  9:  205 VPLADSFR 212
#MOTIF : 10:  222 YTFWTY 227
#MOTIF : 11:  235 NVGWRLDYFLLSHSL 249
#MOTIF : 12:  264 GSDHCPI 270
************************************

  >><< Starting PCPMotifMiner
  >><< Using the sequences from database : ASTRAL40v1.55.txt
Score filter option : 1
Number of top scorers : 30
Reading sequence database file : ASTRAL40v1.55.txt
Scoring true positive sequence
Scoring profile against sequence database
PCPMer - Physical-chemical property based motif analyzer
                  Version 2.0
           COPYRIGHT University of Texas System
```

```
                LOG FILE
          Thu Jan 29 16:31:25 2004


Your selection is 2. This will :Search for motifs in a database..
Reading multiple alignment file : APEALIGN.aln  >><< Starting
PCPMotifMiner
  >><< Using the sequences from database : ASTRAL40v1.55.txt
Score filter option : 1
Number of top scorers : 30
Reading sequence database file : ASTRAL40v1.55.txt
Scoring true positive sequence
Scoring profile against sequence database
```

Example of log file for running example. (*.PCPlog)

7.2.2 Profile files (*.PCPgprf, *.PCPprf)

Profile files contains the average vector values, standard deviation and relative entropy for each of the five PCP vectors. There are two types of profile file. The global profile file (*.PCPgprf) consists of profile for the entire position of the first sequence in the alignment. The motif profile file (*.PCPprf) contains profiles for those residues that are considered to be significant within each motif. The motif profile is used for scoring. Additional information like motif number, parameters used to generate the motifs, motif size is also provided in the motif profile file. Details of columns. Column 1: Position in the mutiple alignment file. Column 2: Total number of sequence in the multiple alignment. Column 3:Number of sequence without gaps. Column 4: Residue position in terms of sequence one. Column 5: Single letter code of correponding residue in sequence 1. Column 6-10: Average magnitude of E1-E5 vectors for the particular column. Column 11-15: Standard deviation of vectors E1-E5. Column 16-20: Relative entropy of E1-E5.

```
  1  42  42   1 A   2.1316   0.9833  -3.5787  -5.6635   0.5030   5.7621
2.2974   2.4349   4.3591   2.2130   0.0698   0.0520   0.0185   0.0319
0.0459
  2  42  42   2 L   1.4334   1.6069  -4.0075  -6.5620   0.7728   6.8598
1.9251   1.8334   2.8840   2.1110   0.0305   0.0535   0.0416   0.0227
0.0337
  3  42  42   3 Y   2.2581   0.9920  -2.3657  -5.9903   0.7704   6.2562
3.3562   4.6703   3.7801   2.0724   0.1275   0.0195   0.0869   0.0100
0.0356
  4  42  42   4 E   4.3238   0.5675  -3.4094  -5.7059   2.0372   5.0647
4.5315   3.8774   4.2581   1.9599   0.0252   0.0487   0.0714   0.0646
0.1536
  5  42  42   5 D   4.0360   0.8095  -3.2601  -5.4695   1.4397   6.7050
3.4520   2.4164   4.5896   1.6056   0.0218   0.0349   0.0182   0.0552
0.0479
  6  42  42   6 P   3.9853   1.6002  -2.7805  -6.1360   1.4948   4.6721
3.2306   4.5917   3.6892   2.0349   0.0501   0.0436   0.0687   0.0116
0.0228
  7  42  42   7 P   4.4693   1.3749  -1.4142  -6.7572   1.7136   4.4467
```

```
4.3337   5.1467   2.6438   2.6900   0.1180   0.1136   0.1207   0.0772
0.1249
  8  42  42   8 D   5.0968  -0.3543  -3.2120  -5.2469   1.3351   5.6783
2.9303   2.0935   5.0441   1.7962   0.0252   0.0864   0.0201   0.1607
0.0483
  9  42  42   9 H   4.8131  -0.2342  -2.9610  -5.4953   0.3903   3.3371
3.8035   2.6546   4.4071   2.1908   0.0544   0.0204   0.0250   0.0416
0.0344
 10  42  42  10 K   3.8608   0.1445  -3.2554  -6.2259   1.2728   5.9000
4.7665   3.1700   3.4804   1.4801   0.0163   0.0301   0.0081   0.0200
0.1032
 11  42  42  11 T   4.3427   1.4346  -3.3660  -5.9298   1.0972   3.7897
3.1090   3.9625   3.6214   2.2892   0.0792   0.0261   0.0738   0.0372
0.0344
 12  42  42  12 S   5.2948   2.2648  -3.4202  -5.5683   0.6996   3.7026
4.4693   2.5940   3.8969   1.9257   0.0785   0.0944   0.0431   0.0663
0.0674
 13  42  42  13 P   4.4071   2.0568  -1.8880  -5.7627   2.0983   4.4960
3.3113   5.4769   3.9562   2.2909   0.0757   0.0877   0.1269   0.0337
0.1198
 14  42  42  14 S   3.6709   1.3236  -2.4649  -5.9119   1.1296   6.4030
4.5854   3.6159   3.5406   2.2291   0.0141   0.0728   0.0250   0.0346
0.0201
 15  42  42  15 G   4.0401   2.8086  -2.8861  -4.8461   1.4245   6.7738
5.0773   2.9848   4.7681   1.6433   0.0379   0.1048   0.0065   0.0562
0.0895
 16  42  42  16 K   5.3148  -0.8701  -2.6754  -6.0158   0.9589   5.4151
5.4909   2.9130   3.6389   1.8109   0.0795   0.1186   0.0200   0.0557
0.1748
 17  42  42  17 P   5.2621   1.0791  -3.0819  -5.0524   0.9358   4.3459
5.4974   2.9284   4.3693   2.4657   0.0719   0.1023   0.0324   0.0291
0.0499
```

Example of global profile (*.PCPgprf) file for running example. Column details are provided in motif profile file

```
#PARAM R_G_L:1.25:2:4:
#COLUMNLABELS POS[1] TOTSEQ[2] SEQNOGAP[3] RESINSEQ1[4] RESCODE[5]
AVG[6-10] STD[11-15] R[16-20]
#MOTIF NUM_SIGCOUNT:1:12:
 20  42  42  20 L  -9.9080  -2.4175  -2.9473   1.4906   1.3928   3.9493
2.2939   4.3159   4.4517   1.7708   1.3314   0.9949   1.3882   1.0177
0.8044
 21  42  42  21 K  10.3465  -9.3774  -0.6542  -4.8372  -0.9833   1.2128
1.2620   1.6056   1.8376   2.1148   1.0070   2.3470   0.7038   1.1539
1.9612
 22  42  42  22 I -13.7147   1.2190  -2.4491  -2.8755   0.3873   3.0840
1.3489   2.7701   1.5973   2.8321   1.6113   0.7192   0.2814   0.6418
0.5728
 24  42  42  24 S   6.5840   5.9627  -0.3962  -1.1844  -3.4568   4.3137
1.1441   0.6862   1.7360   0.6662   0.6605   1.6833   0.9523   1.3122
1.4733
 72  42  42  25 W -12.7756  -3.8550   7.0216   0.8871   4.0103   2.6279
2.2727   4.1455   0.8396   1.3585   1.4848   1.9765   2.3691   1.4633
1.5575
 73  42  42  26 N  11.3446   1.0035   2.6455   2.0054  -0.7805   0.3241
0.4066   0.4639   0.2712   0.5025   1.4747   1.1389   2.5052   2.4972
2.7883
 74  42  42  27 V -12.8005   3.1101  -3.8295  -2.9882  -3.4964   2.6127
1.3750   1.5247   1.3598   1.8304   1.6352   1.1389   0.8154   1.1275
1.2106
 75  42  42  28 D  10.3101   0.7308   0.3345   1.9640   0.2624   3.7533
1.9583   4.5586   1.1618   1.7077   1.1711   0.6504   1.2672   1.4840
1.5454
 76  42  42  29 G   9.2890  13.5515  -0.4820   0.1304   0.7101   1.3330
4.5881   0.2387   0.9018   1.8544   1.3570   2.8161   1.2220   1.4156
0.8256
 77  42  42  30 L -12.3006   1.3533  -4.7719  -3.9590   1.6422   3.7944
2.4322   2.1142   1.0874   2.4966   1.5906   1.0391   0.8424   1.1321
0.6807
 78  42  42  31 R   7.9243  -8.5414   2.0479  -3.3916  -4.7265   1.4395
3.8697   1.6814   2.1794   2.2269   1.6267   1.6775   2.0496   0.8338
1.6327
 79  42  42  32 A   2.6729   4.5949  -7.0829  -0.8425   1.1711   5.3310
3.9800   5.1964   1.2295   2.5651   1.3792   0.6356   1.5147   0.9360
0.5189
#MOTIF NUM_SIGCOUNT:2:9:
103  42  42  47 P   3.4587   5.8426   3.6896  -2.8414   3.9114   7.3243
3.5553   8.2736   1.3101   3.1952   0.8843   1.2275   1.6573   0.6072
1.0153
104  42  42  48 D  12.6439  -1.3801  -0.2780   3.5361   2.3254   4.3079
1.2675   0.3047   0.9286   0.8505   1.3192   2.0187   1.0949   2.9772
1.3478
105  42  42  49 I -12.7783   3.0987  -3.4262  -3.0911  -2.5641   4.4391
2.2877   1.9295   0.8738   2.4396   1.4670   1.0391   0.4629   1.0403
0.8494
```

Example of motif profile (*.PCPprf) file for running example. Column details are provided under #COLUMNLABELS and #MOTIF provides the motif number and length (significant members)

7.2.3 List files (*.PCPNlist, *PCPSlist)

The list files contains the motif list with parameters used to derive it. The list file is the summary of motifs identified in the protein family. It provides motif blocks with the starting and ending residue position number along with the string of residues that occur. Within the program there are two lists 1]Numbered list 2] Stringed list. Numbered list consists of motifs and the residues are indicated by a number rather than string. The output list file contains the stringed list.

```
#PARAM R_G_L:1.25:2:4:
20*21*22*24*25*26*27*28*29*30*31*32
47*48*49*50*51*52*53*54*56
84*85*86*87*88*89*92*94
110*112*113*114*116
129*131*132*133
139*142*143*146
163*164*165*166*167*168*169*170*171*172*173
189*190*193*194*195
205*207*209*211*212
222*224*225*227
235*237*238*239*240*241*242*244*246*249
264*265*266*267*268*269*270
36*42*71*72*103*125*177*181*184*216*253*274
```

Example of Numbered list (*.PCPNlist). The numbered list consists of significant residues of each motif. The first line #PARAM records all the parameters used to derive motifs. Each line is a motif. The last line is the stray motifs, where these significant positions cannot be adjusted to any other motif in the list. Subroutine &adjustmotif can reduce the number of residues in the stray motif list.

```
#PARAM R_G_L:1.25:2:4:
#MOTIF :  0:   36*42*71*72*103*125*177*181*184*216*253*274
#MOTIF :  1:   20 LKICSWNVDGLRA 32
#MOTIF :  2:   47 PDILCLQETK 56
#MOTIF :  3:   84 EGYSGVGLLSR 94
#MOTIF :  4:  110 DQEGRVI 116
#MOTIF :  5:  129 YVPNA 133
#MOTIF :  6:  139 RLEYRQRW 146
#MOTIF :  7:  163 LVLCGDLNVAH 173
#MOTIF :  8:  189 GFTPQER 195
#MOTIF :  9:  205 VPLADSFR 212
#MOTIF : 10:  222 YTFWTY 227
#MOTIF : 11:  235 NVGWRLDYFLLSHSL 249
#MOTIF : 12:  264 GSDHCPI 270
```

Example of Stringed list (*.PCPSlist). The stringed list consists of blocks of residues of each motif. #MOTIF is the motif with numbers in ::. Motif number zero is stray motif. For each motif (except stray) starting and ending position as given in the multiple alignment is provided.

7.2.4 Score file (*.PCPscore)

This file contains score from all the highest scoring windows in a sequence for each motif sorted, with their effective score (bayesian score) and motif score. It also contains the information about the starting position of the window, motif number and sequence number (assigned as per the order in which sequences are read from the database).

```
#column 1-eff_score 2-score 3-start_res. 4-WinString 5-
Motifnum 6-Seqnumber 7-seq
       131.8132580   0.9077   19 LKICSWNVDGLRA    1 3053
>d1hd7a_ d.151.1.1 4.2.99.18  (A:) DNA repair endonuclease
Hap1 {Human (Homo sapiens)}
       130.2158736   0.8967    1 MKFVSFNINGLRA    1 3052
>d1ako__ d.151.1.1 3.1.11.2  (-) DNA-repair enzyme
exonuclease III {Escherichia coli}
       112.8334267   0.7770  204 AIGSTFNVNGVRA    1 1559
>d1f74a_ c.1.10.1 4.1.3.3  (A:) N-acetylneuraminate lyase
{Haemophilus influenzae}
```

Example of score file (*.PCPscore). Each entry is a highest scoring window for a single motif in a given sequence. Column #1 is the effective score in bits. Column #2 is lorentzian score for the motif window with the motif profile. #3 is the starting position. #4 is the string window #5 is the motif profile used #6 is the sequence index #7 is the sequence name.

7.2.5 Result file (*.PCPres)

The result file contains the highest scoring sequences for all the motifs sorted using the filter options. The filter options include raw scoring, scoring motifs that score above mean of average scores and score motifs that score above a cutoff. The result file has the combined bit scores for the top scoring sequences in a descending order and sequence name.

```
2420.03 *>d1hd7a_*d.151.1.1*4.2.99.18*(A:)*DNA*repair*endonuclease*Hap1*{Human*(Homo*sapiens)}
 2354.21 *>d1ako__*d.151.1.1*3.1.11.2*(-)*DNA-repair*enzyme*exonuclease*III*{Escherichia*coli}
 1396.81 *>c1i9ya_*d.151.1.2*0.0.0.0*phosphatidylinositol*phosphate*{addedbyvenkat}
 1250.49 *>d2dnja_*d.151.1.1*0.0.0.0*(A:)*Deoxyribonuclease*I*{Cow*(Bos*taurus)}
 1231.10 *>d1ekma1*b.30.2.1*1.4.3.6*(A:237-672)*Copper*amine*oxidase,*domain*3*(catalytic)*{Yeast*(Hansenula*polymorpha)}
 1135.37
*>d1dp4a_*c.93.1.1*4.6.1.2*(A:)*Hormone*binding*domain*of*the*atrial*natriuretic*peptide*receptor*{Rat*(Rattus*norvegicus)}
```

Example of result file (*.PCPres). Combined effective scores are expressed in bits. Highest scoring sequences are listed.

7.3 Input Parameters

User defined:

Fixed Relative cut-off:

A high relative cutoff means significantly conserved position. When dealing with protein families that are not sufficiently diverged one can use higher relative entropy cutoff (range 0.75-2.5)

Variable Relative cut-off:

These parameters are defined by a range (minimum Relative entropy and maximum relative entropy) and a scan step to find the local PCP motifs. A high relative cutoff means significantly conserved position. When dealing with protein families that are not sufficiently diverged one can use higher relative entropy cutoff (range 0.75-2.5)

G - cutoff:
To define all conserved residues in a motif one can specify G-cutoff to be zero. Increased G-cutoff will result in longer motifs that may not be meaningful.

L - cutoff:
Motifs are defined by presence of more number of significant positions. Hence one should use higher L-cutoff. Lower L-cutoff will result in shorter and too many motifs.

Default parameters:

Standard deviation weight:
This is a multiplication factor added to the denominator for the lorentzian based motif scoring. This is set to 1.5 in the subroutine (motifminer::&scoreprofilestring) as a variable $sdwt.

Relative entropy cutoff:
Only those vectors that score above R-value will be scored for each motif. This values is set to 1.25 in the subroutine (motifminer::&scoreprofilestring) as a variable $entropy

Shift factor:
To prevent overflow due to standard deviation zero (for absolutely conserved positions or vectors) a shift factor is added to the denominator. This value is set to 0.001 in the subroutine (motifminer::&scoreprofilestring) as a variable $epsilon.

8 USING PCPMer

Example 1: To identify motifs in APE protein family and use it to data mine related proteins in the ASTRAL database:

To start the program type PCPMer on the unix prompt and follow the questions. For this example you will use R=1.25, G=2, L=4. A multiple alignment of 42 APE protein sequences are available in the file 'APEALIGN.ALN'. The sequence database is the ASTRAL 40 version 1.55 is provided in the text file 'ASTRAL40v1.55.txt'. In the original installation all the output from this run will be available under /example/ex0 directory. Please read README for more details.

Type on the unix prompt:

```
$unixprompt > ./PCPMer
```

The run session is shown below (user entered options are shown in bold italics):

```
|**********************************************************|
| PCPMer - Physical-chemical property based motif analyzer |
|                    Version 2.0                           |
|    (C) 2004  Bin Zhou, Venkatarajan S. Mathura           |
|               & Prof.Werner Braun                        |
|       Sealy Center for Structural Biology, HBC&G         |
|                UTMB, Galveston                           |
|                                                          |
|                                                          |
|                 Sun May 25 14:19:57 2003                 |
|                                                          |
|                                                          |
|                     MAIN MENU                            |
|      1. Create Motifs                                    |
|      2. Search for motifs and related sequence in a      |
|         database                                         |
|      3. Search for the highest scoring motifs in a set of|
|         sequence                                         |
|      4. Create motifs and search database                |
|      5. Create motifs and score set of sequences         |
|      6. Create macro file of MOLMOL                      |
|      7. Help                                             |
|      8. Exit                                             |
|__________________________________________________________|

  >> Enter your selection :  4
  >> Please enter the multiple alignment file name : APEALIGN.aln
  >><< All output files can be identified using a prefix
  >> Enter a name for the project that will be used as prefix: EXAMPLE
  >><< To identify significantly conserved regions relative entropy is
calculated.
  >><< Use [2.0-3.0] for highly conserved sequences
  >><< or [0.75-2.0] for moderately conserved family
  >> Please enter Relative entropy cut-off : 1.25
```

```
  >><< Motifs are defined by blocks of conserved positions
  >><< Gap cutoff value limits stretches of non-conserved positions in a motif
  >><< Use [0 or 1] for highly conserved sequence
  >><< or [2-4] for moderately conserved family
  >> Please enter Gap cut-off : 2
  >><< Short motifs are not desirable
  >><< L-cutoff limits motifs with less significant positions.
  >><< Use [4-7] for highly conserved sequence
  >><< and [2-4] for moderately conserved family.
  >> Please enter minimum Length cut-off : 4
  >><< Creating global profile file :EXAMPLE.PCPgprf
  >><< Creating motif Nlist file    :EXAMPLE.PCPNlist
  >><< Creating motif profile file  :EXAMPLE.PCPprf
  >><< Creating motif list file     :EXAMPLE.PCPSlist


  >><< ***********************************
  >><< MOTIF DETAILS:

#PARAM R_G_L:1.25:2:4:
#MOTIF :  0:   8*36*42*63*71*72*177*181*184*216*253*274
#MOTIF :  1:   20 LKICSWNVDGLRA 32
#MOTIF :  2:   47 PDILCLQETK 56
#MOTIF :  3:   83 KEGYSGVGLLSRQCP 97
#MOTIF :  4:  103 GIGDEEHDQEGRVIVAEFDSFVL 125
#MOTIF :  5:  129 YVPNA 133
#MOTIF :  6:  139 RLEYRQRW 146
#MOTIF :  7:  162 PLVLCGDLNVAH 173
#MOTIF :  8:  189 GFTPQERQGFGEL 201
#MOTIF :  9:  205 VPLADSFR 212
#MOTIF : 10:  222 YTFWTYM 228
#MOTIF : 11:  232 RSKNVGWRLDYFLLSHSL 249
#MOTIF : 12:  264 GSDHCPI 270
***********************************
  >> Enter the name of the database sequence file (.seq) : ASTRAL40v1.55.txt
  >> Enter combined score filter [0 - raw; 1 - >= mean; 2 - >= cutoff] : 1
  >> Number of effective top scoring sequences required : 30
  >> Reading sequence database file........
  >> Scoring true positive sequence
  Processing true positives for motif number :
  : 1 : : 2 : : 3 : : 4 :  : 5 : : 6 :
  : 7 : : 8 : : 9 :  :10 : : 11: : 12:

Program ended
```

List of output files:

```
EXAMPLE.PCPNlist  [Numbered list file]
EXAMPLE.PCPSlist  [Stringed list file]
EXAMPLE.PCPgprf   [Global profile file]
EXAMPLE.PCPlog    [Log file]
EXAMPLE.PCPprf    [Profile file]
EXAMPLE.PCPres    [Result file]
EXAMPLE.PCPscore  [Score file]
EXAMPLE.PCPavg    [Average scores for motifs and database file]
EXAMPLE.PCPexcl   [Sequence numbers eliminated during scoring as they are
short]
```

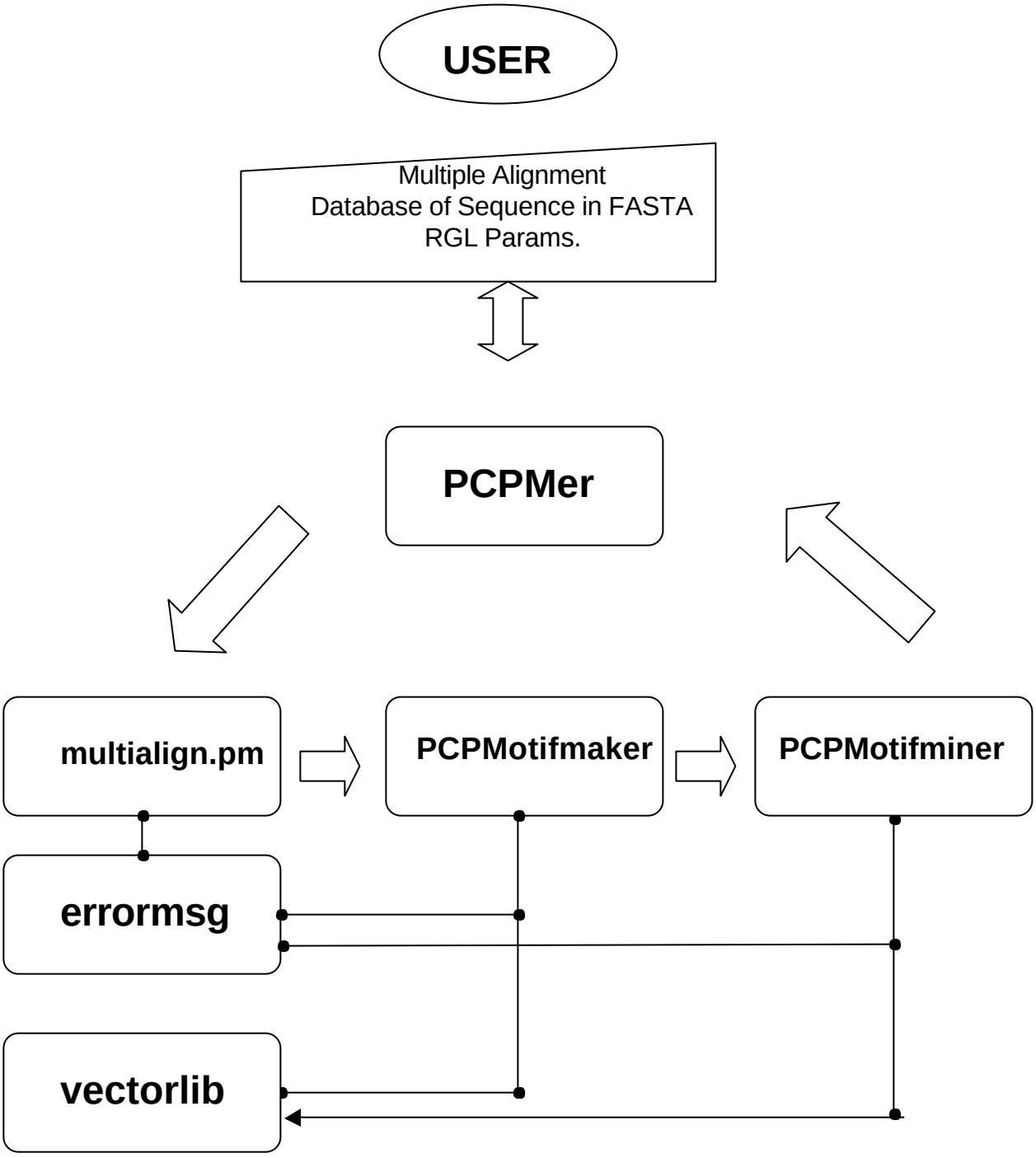
9 PROGRAMER GUIDE(Version 1.0)*

* Software design document of version 2.0 will be provided soon.

9.1 MODULES AND ROUTINES

PROGRAM	MODULE	SUBROUTINE
PCPMer		
	motifmaker.pm	
		makeprofile
		fullprofile2motif
		adjuststraymotif
		motiflist2profile
		evalmotiflist
		autoevalparam
		convert
		mean
		standard_dev
		calcentropy
	motifscorer.pm	
		readprofile
		readfastatostrings
		readmultipletostrings
		parseprofile
		scoreprofiledatabase
		scoreprofilemultialign
		finalscoreout
		sortDindex
		mean
		standard_dev
		scoreprofilestring
		findmax
	multialign.pm	
		readmalign
		filtermalign
		pairid
		formatmalign
	vectorlib.pm	
	errormsg.pm	

9.2 FLOWCHART OF PROGRAM OPERATION



9.3 MODULE AND SUBROUTINE DETAILS

PROGRAM	'PCPMer'	
FUNCTION	Front engine and simple user interface	
INPUT	Multiple alignment, Database file, RGL parameters.	
OUTPUT	FILE EXT.	DESCRIPTION
	.PCPlog	Log file
	.PCPgprf	Global profile
	.PCPlist	Stringed list of motifs
	.PCPprf	Motif profile
	.PCPscore	Highest scoring windows of each motif
	.PCPres	Top x scoring sequences in the database
VARIABLES	NAME	DESCRIPTION
	@files	List of PCPM files
	$response_maln	Multiple alignment filename
	@m_aln_data	Multiple alignment data in array
	$response	Name for the project
	$user_R_value	User defined relative entropy cutoff
	$user_L_value	User defined L-value cutoff
	$user_G_value	User defined G-value cutoff
	$global_prf	Reference to global profile
	$motif_N_list	Reference to numbered motif list
	$motif_S_list	Reference to stringed motif list
	$motif_prf	Reference to motif profile
	$temR	Response to continue motifminer
	$response_dbase	Database file name
	$filter_opt	Filter option (0 - raw, 1- average, 2- above cutoff)
	$cutoff_opt	User specified score cutoff
	$seqnamedb	Reference to sequence name
	$seqstringdb	Reference to sequence strings
	$TrueAvg	Average scores of all motifs in true positives
	$TrueStd	Standard deviation of all true positives
	$DataAvg	Averages scores of motifs in all database sequence.
	$FinalOut	Top x sequences ranked according to the filter.

VERSION HISTORY	
Current	Version 1.0
Release date	01MAR03
Written by	Venkatarajan S. Mathura
Details of modification (date/author)	

MODULE	'errormsg.pm'
FUNCTION	To detail error and warning messages when caught
INPUT/OUTPUT	Package 'errormsg' should be called to use any subroutine
SUBROUTINE	‘errmsg’
VERSION HISTORY	
Current	Version 1.0
Release date	01MAR03
Written by	Venkatarajan S. Mathura
Details of modification (date/author)	

SUBROUTINE	‘errmsg’
FUNCTION	Provides suitable error messages upon external call using a number.
INPUT/OUTPUT	Input is a three-digit number. First number indicates the type of error. 1XX errors in input 2XX errors in output 3XX overflow errors. 4XX unknown errors.
PSEUDOCODE	Uses a hash that stores error messages for a three number key.
VARIABLES	'%errormsg' hash that contains error messages
VERSION HISTORY	
Current	Version 1.0
Release date	01MAR03
Written by	Venkatarajan S. Mathura
Details of modification (date/author)	

MODULE	'vectorlib.pm'		
FUNCTION	Contains amino acid parameters and supplies these params upon call.		
INPUT/OUTPUT	Package 'vectorlib' should be called to use any subroutine		
VARIABLES	NAME	A	DESCRIPTION
	%vectorX	E	PCP descriptor X for all 20 AA. Single letter AA is KEY. X = 1..5
	%natfreq{}[]	E	Provides magnitude of the vector in different bin for naturally occurring AA. Example PCP vector 2 magnitude in bin 1 is $natfreq{2}[1];
	%range{}[]	E	Provides range of vector magnitude for binning. Example PCP vector 2 in the Ist bin is $range{2}[1] to range{2}[0]
	@natfreq	E	Natural frequency of amino acid occurrence. 20 in order.

VERSION HISTORY

Current	Version 1.0
Release date	01MAR03
Written by	Venkatarajan S. Mathura
Details of modification (date/author)	

MODULE	'multialign.pm'		
FUNCTION	To read and format multiple alignment in clustalw format		
INPUT/OUTPUT	Package 'multialign' should be called to use any subroutine		
SUBROUTINES	NAME	A	DESCRIPTION
	&readmalign	E	Reads in an array reference containing clustalw file
	&filtermalign	E	Filters multiple alignment based on %id cutoff with respect to first sequence.
	&pairid	I	Calculates pairid between two strings excluding gaps
	&formatmaligm	I	Reformats multiple alignment after removing redundant seq.

VERSION HISTORY

Current	Version 1.0
Release date	01MAR03
Written by	Venkatarajan S. Mathura
Details of modification (date/author)	

SUBROUTINE	'readmalign'	
FUNCTION	To read in a multiple alignment in clustalw format.	
INPUT	Reference to an array containing clustalw file	
OUTPUT	Reference to arrays containing 1. sequence name 2. sequence string	
PSEUDOCODE	Read in the array reference for multiple alignment Check for clustalw format Split data lines into sequence name and sequence string. Append all sequence string of single sequence identified by name. If there is 1 or 2 sequence only than call error If the sequence string are of different lengths call error If the sequence string contains non-standard amino acid codes than raise error Return array references for sequence name and string.	
VARIABLES	NAME	DESCRIPTION
	$input	Reference to input multiple alignment
	$i	Increment counter for the sequence
	@dataline	Each line of the multiple alignment
	@seqname	Array of sequence names
	@seqstring	Array of sequence strings corresponding to sequenenames
	%seqdat	Hash that has sequence name as key and sequence strings as value

VERSION HISTORY

Current	Version 1.0
Release date	01MAR03
Written by	Venkatarajan S. Mathura
Details of modification (date/author)	

SUBROUTINE	'filtermalign'	
FUNCTION	To filter those sequences that have higher percentage identity above cutoff specified	
INPUT	Reference to an array containing clustalw, percentage id cutoff and internal subroutine calls	
OUTPUT	Array references to list of sequence names and sequence strings after reformat	
PSEUDOCODE	Read the mutliple alignment Split the data into sequence names and sequence strings Calculate pairwise identity scores excluding gaps for all sequences Remove sequences that exceed above the specified cutoff using first sequence as reference. Reformat the sequence strings (like removing '-' if exist in all sequences) Return references to new arrays containing sequence name and sequence strings.	
VARIABLES	NAME	DESCRIPTION
	$input	Reference to input multiple alignment
	$idcut	Identity cut off
	$seqname	Reference to array containing sequence names
	$seqstring	Reference to array containing sequence strings
	&readmalign	Subroutine that reads multiple alignment
	&pairid	Subroutine to calculate pairwise sequence id
	%remove	Flag those sequences that have sequence id greater than id cutoff
	$count1	Sequence counter
	$count2	Sequence counter
	@seqname1	Array containing unflagged sequence name
	@seqstring1	Array containing unflagged sequence string
	$seqname2	Reference to array containing sequence name after reformat
	$seqstring2	Reference to array containing strings after reformat
	&formatmalign	Reformats strings (removes '-')
	&errmsg	Raise an error message

VERSION HISTORY

Current	Version 1.0
Release date	01MAR03
Written by	Venkatarajan S. Mathura
Details of modification (date/author)	

SUBROUTINE	'pairid'	
FUNCTION	Calculates pair-wise identity	
INPUT	Input two strings	
OUTPUT	%id	
PSEUDOCODE	??Read in two sequences	
	??Apply checks to make sure lengths of the string are equal	
	??Identify matching alphabets at different positions	
	??Divide identical positions/(number of non '-' positions)	
VARIABLES	NAME	DESCRIPTION
	$string1	Variable containing first string
	$string2	Variable containing second string
	$count1	Position counter of each string
	$idcount	Count identical residues in both string
	$strlengthminus	Count number of gaps in any sequence

VERSION HISTORY

Current	Version 1.0
Release date	01MAR03
Written by	Venkatarajan S. Mathura
Details of modification (date/author)	

SUBROUTINE	'formatmalign'	
FUNCTION	Excludes gaps that are common in all position and reformats multiple alignment	
INPUT	Reference to array containing sequence names and strings	
OUTPUT	Reformatted array reference containing sequence names and strings	
PSEUDOCODE	??Read sequence strings	
	??Run the position counter	
	??For each position extract one letter from each sequence string	
	??If all the position in the strings are gaps than exclude from new sequence strings	
VARIABLES	NAME	DESCRIPTION
	$inputname	Array reference to sequence names
	$inputstring	Array reference to sequence strings
	$count1	Sequence name counter
	$count2	Sequence name counter
	$i	Original sequence position counter
	$temp	Temporary string (column from maln) holder
	$gap	Holds a gap string no_of_seq x "-"
	@tempstr[A][B]	Holds the residues at position B of seq A
	@splitstr	Array contains all positions of maln
	$j	New position incrementer
	$k	New position counter
	$tempstr	Concatenates new position for a sequence
	@seqname2	Name of the sequence
	@seqstring2	New strings after eliminating

VERSION HISTORY	
Current	Version 1.0
Release date	01MAR03
Written by	Venkatarajan S. Mathura
Details of modification (date/author)	

MODULE	'PCPmotifmaker'		
FUNCTION	To create motif from multiple alignment using empirical parameter		
INPUT/OUTPUT	Package 'PCPmotifmaker' should be called to use any subroutine		
SUBROUTINES	NAME	A	DESCRIPTION
	&makeprofile	E	Makes global profile with multiple alignment
	&fullprofile2motif	E	Uses empirical parameters RGL to make motif from global profile
	&motiflist2profile	E	Uses a list to extract profile from global profile
	&autoevalparam	E	Evaluates different param combination to select optimal RGL
	&adjuststraymotif	E	Adjusts stray motifs defined using RGL by redistributing residues to the defined motifs that are lesser than d specified
	&evalmotiflist	I	Evaluates whether the current motif list has motifs greater than 30 residues
	&convert	I	Converts AA one letter code into vector and computes average, std. and rel. entropy
	&mean	I	Calculates mean of the array
	&standard_dev	I	Calculates standard deviation of the array
	&calcentropy	I	Calculates relative entropy
	&samp_mean	I	Calculates average using N-1
	&sam_standard_dev	I	Calculates sample standard deviation

VERSION HISTORY

Current	Version 1.0
Release date	01MAR03
Written by	Venkatarajan S. Mathura
Details of modification (date/author)	

SUBROUTINE	'makeprofile'	
FUNCTION	To make a global profile of a multiple	
INPUT	Array reference to sequence name and sequence strings	
OUTPUT	Reference to global profiles that has headers.	
PSEUDOCODE	Read sequence string and name Iterate each position of the multiple alignment and obtain AA in a column Convert each of the AA code in to PCP descriptors or vectors 1-5 Calculate average, standard deviation and relative entropy Calc. the number of gaps, position and residue index of the 1 res Print profile [GLOBAL]	
VARIABLES	NAME	DESCRIPTION
	$inputname	Array reference to sequence names
	$inputstring	Array reference to sequence strings
	$i	Position index multiple alignment
	$j	Residue index of first sequence
	@temp	Non '-' column string for each residue index of the first sequence.
	$gap	Number of gaps in a particular column corresponding to residue index of first sequence.
	$vectorX	Magnitude of vector X averages correponding to column string. X varies from 1 ..5
	$vectorsdX	Standard deviation of vector X correponding to column string. X varies from 1 ..5
	$rentropyX	Relative entropy of vector X correponding to column string. X varies from 1 .. 5
	&convert	Subroutine that converts string correponding to a column (no '-') in to average, std and rel. entropy
	$temps	Formatted string that has the following format. 1] position 2]maxseq 3]nongaps 4] pos.no.of1stseq 5]resname 6] avg1 7]avg2 8]avg3 9]avg4 10]avg5 11-15]std 16-20]relativeentropy
	@profile	Array of global profile

VERSION HISTORY	
Current	Version 1.0
Release date	01MAR03
Written by	Venkatarajan S. Mathura
Details of modification (date/author)	

SUBROUTINE	'fullprofile2motif'
FUNCTION	Convert global profile into motif list using RGL
INPUT	Array reference to global profile, R cutoff, G cutoff, L cutoff
OUTPUT	Array reference to Motif list
PSEUDOCODE	Read input global profile, R, G and L. Mark all singificant datalines{profile} that have relative entropy greater than or equal to R Calculate residue index difference [4th column] for two sequential singificant profiles If the difference is less than G then concatenate to the motif. If the difference is greater than G then start a new motif. Calculate the size of each motif (end res. position - start res position). If it is lesser than L Then add to stray motif list

VARIABLES

NAME	DESCRIPTION
$inputarray	Array reference to global profile input
$R	R cutoff
$G	G cutoff
$L	L cutoff
@elements	Array containing elements of a profile
@signif	Array containing profile positions that are significant
$i	Array index for significant position
$motifstring	String of profile index that consititutes a motif
@motif	Stores motif strings
$k	Index for motifs
@elementcount	Number of significant position in the motif
@motiflist	Array containing list of motifs
$straymotif	String of loose significantly conserved sites

VERSION HISTORY

Current	Version 1.0
Release date	01MAR03
Written by	Venkatarajan S. Mathura
Details of modification (date/author)	

SUBROUTINE	'adjuststraymotif'	
FUNCTION	Adjusts stray motifs that were not included as a motif in the first pass	
INPUT	Reference to motif list, New G cut off that can be allowed	
OUTPUT	Modified motif list	
PSEUDOCODE	Read in the motif list and new G Calculate difference of residue between first and last residues of each motif and a member of stray motiflist If the difference is less than G then add to motif (either first or last) where the difference is less Return reference to new motiflist	
VARIABLES	NAME	DESCRIPTION
	$inputlist	Array reference to input motif list
	$allowgap	New G-cutoff value
	@strayelem	Array containing stray elements
	$i	Array index for stray elements
	$min	Arbitray number set to 1000 to reduce and find min
	$j	Array index for motif list
	@storeA	Stores all residue positions for each motif
	$fit	Absolute difference between end or first residue and stray element
	$point	String containing a flag to indicate whether the stray element is close to the start (0) or end (1)
	@tempA	Array containing element of flag and stray element and motif
	$stray	New stray motif string after adjusting
	@newlist	Array of new motif list

VERSION HISTORY

Current	Version 1.0
Release date	01MAR03
Written by	Venkatarajan S. Mathura
Details of modification (date/author)	

SUBROUTINE	'motiflist2profile'	
FUNCTION	Converts motif list into motif profile	
INPUT	Reference to array containing global profile and a list OUTPUT	
OUTPUT	Reference to array containing motif profile and an AA list.	
PSEUDOCODE	Read array reference to the input profile and list	
	Extract residues in the list from global profile	
	For every motif include the details of motif number and residue	
VARIABLES	NAME	DESCRIPTION
	$inputprofile	Array reference to the profile
	$inputlist	Array reference to the input motif list
	@storeA	Array reference to the input motif list
	@storeB	Array to store profile
	$i	Index for motifs
	@temp	Array containing elements of each motif
	$tempS	Temporary string that holds residue name corresponding to each element
	$j	Index for motif elements
	@tempt	Array containing individual element of a profile
	$tempX	String containing profile of each motif

VERSION HISTORY

Current	Version 1.0
Release date	01MAR03
Written by	Venkatarajan S. Mathura
Details of modification (date/author)	

SUBROUTINE	'evalmotiflist'	
FUNCTION	Evaluates whether a motif list has length greater than 30 residues	
INPUT	Array reference to motif list.	
OUTPUT	Flag indicating Motif is not too long.	
PSEUDOCODE	Read input list Check the length of motif (last residue position - 1st residue position) is less than 30	
VARIABLES	NAME	DESCRIPTION
	$inputlist	Array reference to input list
	$i	Index for motif
	@temp	Array containing elements of each motif
	$flag	Flag to indicate whether length is greater than 30 residues (-1)

VERSION HISTORY	
Current	Version 1.0
Release date	01MAR03
Written by	Venkatarajan S. Mathura
Details of modification (date/author)	

SUBROUTINE	'autoevalparam'
FUNCTION	To find optimal RGL combinations to define motifs automatically
INPUT	Array reference to motif profile
OUTPUT	Optimal RGL parameters
PSEUDOCODE	Iterate through different combinations of RGL For each combination evaluate whether all motifs are less than 30 residue

VARIABLES	NAME	DESCRIPTION
	$inputprofile	Array reference to input profile
	@Rvalues	Array containing list of R values
	@Gvalues	Array containing list of G values
	@Lvalues	Array containing list of L values
	$tempR	R value holder
	$tempG	G value holder
	$tempL	L value holder
	$templist	Reference to motif list
	&fullprofile2motif	Subroutine to convert global profile into motif
	$tempres	Flag to indicate accept (0) or reject (-1) current RGL param
	&evalmotiflist	Subroutine to evaluate the fitness of the motif list

VERSION HISTORY

Current	Version 1.0
Release date	01MAR03
Written by	Venkatarajan S. Mathura
Details of modification (date/author)	

SUBROUTINE	'convert'
FUNCTION	To convert an string of AA code into different vector average, std. dev. and relative entropy
INPUT	String of amino acid and vector number
OUTPUT	Average, standard deviation and relative entropy
PSEUDOCODE	Obtain string and parse it into single letter Use vector library to convert it in to number Calculate average, standard deviation, relative entropy for each vector

VARIABLES

NAME	DESCRIPTION
$inputprofile	Array reference to input profile
$inputvec	Vector number
$temp	String that can be evaluated to obtain the correct vector
@newarray	Contains vector values
%vectorX{}	Vector values from module `vectorlib.pm' for X=1..5
&mean	Subroutine to calculate mean
&standard_dev	Subroutine to calculate standard deviation
&calcentropy	Subroutine to calculate relative entropy

VERSION HISTORY

Current	Version 1.0
Release date	01MAR03
Written by	Venkatarajan S. Mathura
Details of modification (date/author)	

SUBROUTINE	'mean' 'samp_mean'	
FUNCTION	To calculate average of an array of numbers (or N-1 for samp_mean)	
INPUT	Array reference containing numbers	
OUTPUT	Average value	
PSEUDOCODE	Obtain numbers in the array Add them and calculate average	
VARIABLES	NAME	DESCRIPTION
	$arrayref	Array reference to input number array
	$result	Summed value of all array

VERSION HISTORY	
Current	Version 1.0
Release date	01MAR03
Written by	Venkatarajan S. Mathura
Details of modification (date/author)	

SUBROUTINE	'standard_dev' 'samp_standard_dev'	
FUNCTION	To calculate standard deviation of an array of numbers (NOT SAMPLE)	
INPUT	Array reference containing numbers	
OUTPUT	Standard deviation	
PSEUDOCODE	Obtain numbers in the array Add them and calculate average	
VARIABLES	NAME	DESCRIPTION
	$arrayref	Array reference to input number array
	$result	Summed value of all array
	$deviation	Standard deviation

VERSION HISTORY	
Current	Version 1.0
Release date	01MAR03
Written by	Venkatarajan S. Mathura
Details of modification (date/author)	

SUBROUTINE	'calcentropy'
FUNCTION	To calculate relative entropy of an array of vector values
INPUT	Array reference containing numbers
OUTPUT	Relative entropy
PSEUDOCODE	Obtain numbers in the array and vector number Calculate background value using natural frequency distribution of amino acid in a range Calculate relative entropy using standard formula over the five bins

VARIABLES

NAME	DESCRIPTION
$arrayref	Array reference to input number array
$vectorval	Summed value of all array
$i	Index for original array of numbers
$range{A}[]	Boundary of magnitude for vector A that defines bins. Value defined in 'vectorlib.pm'
$counteX	Counts number of occurrence of AA vector values for bins X = 1..5
@obsf	Array of observed frequency within five bins
$j	Index for bins
$rentropy	Relative entropy

VERSION HISTORY

Current	Version 1.0
Release date	01MAR03
Written by	Venkatarajan S. Mathura
Details of modification (date/author)	

MODULE	'PCPmotifminerr'		
FUNCTION	To score motifs against a database and select top hits using filter		
INPUT/OUTPUT	Package 'PCPmotifminerr' should be called to use any subroutine		
SUBROUTINES	NAME	A	DESCRIPTION
	&readprofile	E	To read motif profile
	&readfastatostrings	E	To read fasta files and convert each sequence into a linear string
	&readmultipletostrings	E	To read multiple alignment and convert into strings
	&parseprofile	E	To parse individual profile that can be used for searching
	scoreprofiledatabase	E	To score profile against a defined database
	&scoreprofilemultialign	E	To score profile against TRUE sequences
	&finalscoreout	E	To score each sequence in the database using a combined score
	&sortDindex	I	To sort highest score
	&mean	I	To calculate average
	&standard_dev	I	To calculate standard deviation
	&scoreprofilestring	E	To score profile against a string of AA alphabets
	&findmax	I	To find the index of an array of number that is maximum

VERSION HISTORY

Current	Version 1.0
Release date	01MAR03
Written by	Venkatarajan S. Mathura
Details of modification (date/author)	

SUBROUTINE	'readprofile'	
FUNCTION	To read motif profile file	
INPUT	Array reference to profile data	
OUTPUT	Array reference to profilelengths and individual profile	
PSEUDOCODE	Read motif profile	
	Split individual profile using motif separator	
VARIABLES	NAME	DESCRIPTION
	$dataprofs	Array reference to motif profile
	@allsplt	Array containing elements of motif profile
	$profnum	Motif number
	$maxprof	Maximum motif number
	@proflength	Length {number of significant residues} of a motif
	$cce	Index for the residues in a motif
	%profile{A}[B]	Hash containing motif number A and profile index B

VERSION HISTORY

Current	Version 1.0
Release date	01MAR03
Written by	Venkatarajan S. Mathura
Details of modification (date/author)	

SUBROUTINE	'readfastatostrings'	
FUNCTION	To read FASTA file	
INPUT	Array reference to fasta data	
OUTPUT	Array reference to sequence names and sequence strings	
PSEUDOCODE	Read FASTA file	
	Concatenate sequence strings and parse all sequence names	
VARIABLES	NAME	DESCRIPTION
	$data	Array reference to fasta data
	$i	Sequence index
	$seq	String containing the sequence
	@sequen	Array containing all sequence strings
	@name	Array containing all names

VERSION HISTORY	
Current	Version 1.0
Release date	01MAR03
Written by	Venkatarajan S. Mathura
Details of modification (date/author)	

SUBROUTINE	'readmultipletostrings'	
FUNCTION	To read multiple alignment file and convert in to array	
INPUT	Array reference to multiple alignment	
OUTPUT	Array reference to sequence names and sequence strings	
PSEUDOCODE	Read multiple alignment file	
	Concatenate sequence strings and parse all sequence names	
VARIABLES	NAME	DESCRIPTION
	$input	Array reference to multiple alignment data
	@checks	Array containing elements of each row of multiple alignment
	%seqdat	Hash containing sequence string as a value and sequence name as a key
	$i	Sequence index
	@seqstring	Array containing all sequence strings
	@seqname	Array containing all names

VERSION HISTORY

Current	Version 1.0
Release date	01MAR03
Written by	Venkatarajan S. Mathura
Details of modification (date/author)	

SUBROUTINE	'parseprofile'	
FUNCTION	To parse motif profile in to individual file	
INPUT	Array reference to motif profiles, Array reference to motif lenghts, Motif number	
OUTPUT	Array reference to single motif profile	
PSEUDOCODE	Read motif profiles,motif lenghts,motif number Parse motif profiles and identify profiles that belong to a single motif required	
VARIABLES	NAME	DESCRIPTION
	$motifprofileref	Array reference to motif profiles
	$motiflengthref	Array reference to motif lengths
	$motif	Motif number
	@motifdata	Array containing motif profile of motif $motif

VERSION HISTORY

Current	Version 1.0
Release date	01MAR03
Written by	Venkatarajan S. Mathura
Details of modification (date/author)	

SUBROUTINE	'scoreprofiledatabase'	
FUNCTION	To score a motif profile against sequences in a database	
INPUT	Array reference to motif profile, Array reference to sequence data, Array reference to true positive average, Array reference to true positive standard deviation	
OUTPUT	List of scores for highest scoring window for each profile for every sequence	
PSEUDOCODE	Read motif profiles and sequences For each sequence obtain string of window size equivalent to that of a motif For each motif and window score using the scheme developed Record the highest scoring window in a sequence	
VARIABLES	NAME	DESCRIPTION
	$dataarray1	Array reference to motif profiles
	$dataarray2	Array reference to motif lengths
	$TRUavg	Array reference to true average of all motifs
	$TRUstd	Array reference to true std.dev of all motifs
	$temp5A	Array reference to motif profile
	$temp5B	Array reference to motif list
	$temp6A	Array reference to sequence name
	$temp6B	Array reference to sequence strings
	$motifn	Index for motifs
	$temp7	Array reference to single motif profile
	$seqn	Index for database sequence
	$temp8	Highest scoring window details for a string and profile {SCORE POSITION STRING}
	$tempstring	String formatter [# {SCORE POSITION STRING} MOTIFNUM SEQNUM SEQNAME #] {} elements returned by '&scoreprofilestring'
	@motifRES1	Array to store highest scoring window of all sequences for a motif
	@tempdat	Array of elements for each in motifRES1
	@scoredata	Array of highest scores for a particular motif
	$average	Average score for a motif for a database
	$Effectivescore1	Combined score (Sx) for each motif
	$temp	String to hold effective score.
	&readprofile	Subroutine to read profile
	&readfastatostrings	Subroutine to read fasta file
	&parseprofile	Subroutine to parse profile
	&scoreprofilestring	Subroutine to score a profile against string
	&mean	Subroutine to calculate average

VERSION HISTORY

Current	Version 1.0
Release date	01MAR03
Written by	Venkatarajan S. Mathura
Details of modification (date/author)	

SUBROUTINE	'scoreprofilemultialign'
FUNCTION	To score motif profiles against sequences in the mutliple alignment
INPUT	Array reference to motif profiles, Array reference to multiple alignment
OUTPUT	Array reference to average and standard deviation of True positive
PSEUDOCODE	Read motif profiles and multiple alignment Convert each multiple alignment into sequence strings Score each profile against strings and record highest scores For each motif calculate average and standard deviation.

VARIABLES

NAME	DESCRIPTION
$dataarray1	Array reference to motif profiles
$dataarray2	Array reference to motif lengths
$TRUavg	Array reference to true average of all motifs
$TRUstd	Array reference to true std.dev of all motifs
$temp5A	Array reference to motif profile
$temp5B	Array reference to motif list
$temp6A	Array reference to sequence name
$temp6B	Array reference to sequence strings
$motifn	Index for motifs
$temp7	Array reference to single motif profile
$seqn	Index for database sequence
$temp8	Highest scoring window details for a string and profile {SCORE POSITION STRING}
$tempstring	String formatter [# {SCORE POSITION STRING} MOTIFNUM SEQNUM SEQNAME #] {} elements returned by '&scoreprofilestring'
@motifRES1	Array to store highest scoring window of all sequences for a motif

@tempdat	Array of elements for each in motifRES1
@scoredata	Array of highest scores for a particular motif
$average	Average score for a motif for a database
$Effectivescore1	Combined score (Sx) for each motif
$temp	String to hold effective score.
&readprofile	Subroutine to read profile
&readfastatostrings	Subroutine to read fasta file
&parseprofile	Subroutine to parse profile
&scoreprofilestring	Subroutine to score a profile against string
&mean	Subroutine to calculate average

VERSION HISTORY

Current	Version 1.0
Release date	01MAR03
Written by	Venkatarajan S. Mathura
Details of modification (date/author)	

SUBROUTINE	'finalscoreout'	
FUNCTION	To output the highest combined scoring sequences using filters	
INPUT	Name of score file, highest number of x sequences required, Filter option, Cutoff (for Filter option 2), Average TP, Average DB.	
OUTPUT	Array reference to top scoring sequences	
PSEUDOCODE	Read motif profiles,motif lenghts,motif number Parse motif profiles and identify profiles that belong to a single motif required	
VARIABLES	NAME	DESCRIPTION
	$dataName	Score data file name
	@dataA	Array containing data
	@temp	Elements of each row of data red
	$vv	String containing sequence information
	$k	Index for data element
	@addscore	Array containing combined score for each sequence
	@namedetails	Array containing sequence details
	$newID	Array reference to index of array that is sorted in decreasing order
	$i	Iterator for top x sequences
	&sortDindex	Subroutine to sort an array according to numerical decreasing order
	$Option	Type of filter to apply 0) Raw ranking 1)Include motifs score greater than mean of Averages 2) Motif scores above a cutoff
	$ScoreCutoff	Cutoff score above which you want to include as a motif hit
	$Truemean	Array reference to average scores for motifs in true positive sequence
	$Datamean	Array reference to average scores for motifs in the database

VERSION HISTORY

Current	Version 1.0
Release date	01MAR03
Written by	Venkatarajan S. Mathura
Details of modification (date/author)	

SUBROUTINE	'sortDindex'	
FUNCTION	To output the index of an array after sorting the elements in numerical descending order	
INPUT	Array reference to an array containing scores.	
OUTPUT	Index of sorted scores in decreasing order	
PSEUDOCODE	Read array that contains scores Sort the array numerically	
VARIABLES	NAME	DESCRIPTION
	$data	Array reference to an array containing scores
	@newIndex	Index in the sorted order

VERSION HISTORY

Current	Version 1.0
Release date	01MAR03
Written by	Venkatarajan S. Mathura
Details of modification (date/author)	

SUBROUTINE	'scoreprofilestring'	
FUNCTION	To score a motif profile against a sequence of string and report the highest scoring window.	
INPUT	Array reference to motif profile and the sequence string.	
OUTPUT	Highest score, window position, window string.	
PSEUDOCODE	Read the profile and the string Determine the length of the profile that should be used as the window Obtain chunks of string from each scanning window and score profile against it Determine the highest scoring window	
VARIABLES	NAME	DESCRIPTION
	$profiledata	Array reference to a motif profile data
	$fastastring	Fasta sequence string
	$entropy	Entropy cutoff
	$sdwt	Standard deviation weight
	@temp	Profile elements
	$startvalue	Staring residue position of a motif
	$finalvalue	Last residue position of a motif
	$length	Length of a motif
	$seq	Position iterator for the fasta sequence
	$seqstring	Sequence string chunk defined by scanning window
	$s1	Score for each significant vector additively
	$s1max	Maximum score possible for each vector
	$epsilon	Shift factor
	$i	Iterator for motif profile data
	@val	Elements in each row of motif profile data
	$profileshift	Shift in the current residue position in a motif
	$aa	Residue at a particular position in one letter code
	$zfract	Fractional z score
	$vectorx{}	Vector values defined in the 'vectorlib.pm' x= 1..5
	$temp	String containing score, position and string information
	@alldata	Collects all $temp
	@sscore	Array of scores
	$max	Maximum score
	$maxindex	Index of maximum score
	&findmax	Subroutine to find the maximum score

VERSION HISTORY

Current	Version 1.0
Release date	01MAR03
Written by	Venkatarajan S. Mathura
Details of modification (date/author)	

SUBROUTINE	'findmax'	
FUNCTION	To the index and highest score of an array {simple implementation}	
INPUT	Array reference to an array containing scores	
OUTPUT	Highest score and its index	
PSEUDOCODE	Read array that contains scores Sort the array numerically	
VARIABLES	NAME	DESCRIPTION
	$scoredata	Array containing scores
	$max	Maximum value
	$maxindex	Index of maximum value

VERSION HISTORY

Current	Version 1.0
Release date	01MAR03
Written by	Venkatarajan S. Mathura
Details of modification (date/author)	