```
#######################################################

          Protein Structure Analysis and Design Lab
          Michigan State University
          Sep 18, 2014

#######################################################
```

MSU ProFlex version 5.2

This file has three parts:

(i)   Installation notes
(ii)   Input pre-processing and how to run ProFlex
(iii)   Interpreting and using the ProFlex output

```
          Part - I        Installation Notes
          ===============================
```

ProFlex runs on various Unix systems (Linux with kernel 2.4.18 or higher,
Solaris 9 (and probably newer versions), and SGI Octane running Irix64.

Before installing and compiling ProFlex, the tar file has to be placed
in the directory that will contain the root directory of the global ProFlex
installation (e.g. /usr/soft/).

ProFlex has been build-tested with the GNU GCC compiler, v. 4.1.2 and higher.
The Fortran component builds of ProFlex have been tested with both g77 and
gfortran.

To install ProFlex:

1. Unzip and untar the proflexv5.2.tgz using the command:

        tar zxfv proflexv5.2.tgz

The untarred directory name will be proflexv5.2. It should be renamed to
proflex, e.g.,

        mv proflexv5.2 proflex

Before you do this, please remember to back up any existing versions of
ProFlex you might have in the directory.

2. Set the environment variable PROFLEX_HOME to the prog subdirectory of the
proflex directory just created above, using the full, explicit path. This is
the directory in which you performed the tar command, followed by its
subdirectory proflex/prog, e.g.,

        setenv PROFLEX_HOME /usr/soft/proflex/prog (tcsh shell), or
        export PROFLEX_HOME=/usr/soft/proflex/prog (bash shell)

3. If you have g77 installed in your system path, you will not need to set
the F77 environmental variable. However, if you do not, you will need to
assign an appropriate fortran compiler to the F77 variable, e.g.,

```
        setenv F77 gfortran (tcsh shell), or
        export F77=gfortran (bash shell)
```

4. Then change the current working directory to the proflex directory and
 run the make command to complete installation, e.g.,

```
        cd /usr/soft/proflex

        make
```

This creates a bin directory with a link to the proflex executable, e.g.,

```
        /usr/soft/proflex/bin/proflex
```

NOTE: make requires the gcc, g++, and g77/gfortran, the GNU C, C++, and FORTRA
N77
compilers, respectively, to be reachable through your Unix PATH, which is
usually set in your .cshrc or other similar shell initialization script.
These compilers can be downloaded from the GNU website:

```
        http://www.gcc.gnu.org
```

        Part - II   How to pre-process input files and run ProFlex
        ==

Notes on preparing PDB files as input to ProFlex:

i) ProFlex requires the input file to be named with an extension ".pdb",
e.g., 1ahb.pdb (not 1ahb.ent)

ii) Include only those heteroatoms (atoms that are not proteinaceous in
origin) and ligands that you would like to be included in the flexibility
analysis and which are essentially a part of the protein. See section XXX
of the User Guide for details on how to process heteroatoms so their
covalent and non-covalent bonds will be correctly interpreted. Note that for
the accuracy of flexibility predictions, it is recommended to not include
water molecules in the PDB input file, except for those water molecules that
are entirely buried in the protein (which may be assessed by PROACT or
another tool) or which form essential interactions between the protein and
another molecule (e.g., a protein-water-metal bond network).

iii) The input file is expected to have appropriately protonated polar atoms
(e.g., Lys NZ has three protons at pH 7; a hydroxyl group has one, as does
the backbone amide N). ProFlex does not add Hâ s to the input file and will
give erroneous results on a PDB file that does not contain polar hydrogen
atoms. Please use a tool such as WhatIf, Reduce, InsightII, CHARMM, or AMBER
to do this.

For a more detailed description of the ProFlex options including the usage
format and the purpose of each option, please refer to Section 3 of the
included ProFlex user guide:

            $PROFLEX_HOME/../docs/ProFlex_User_Guide.pdf

To run ProFlex:

1) Add ProFlex's bin directory to your shell's PATH variable, e.g.,

```
        setenv PATH ${PATH}:/usr/soft/proflex/bin
```

2) The basic command to run proflex on an input file is:

```
        proflex -h <input file name.pdb>
```

   Running proflex without any options will display the proflex "help" menu

with a list of valid options and the format of acceptable input arguments.
Users who wish to run proflex in an automated (noninteractive) fashion
should use the "-non" option (see the "help" menu).

                    Part - III   How to interpret ProFlex output
                    ===

ProFlex outputs the results of its analysis into text files as well as
scripts that facilitate graphical visualization of the protein based on the
contents of those text files. Here is a list of the various output files and
a brief description of their purpose:

Text files:

NOTE: In the file names below, (1) <protein> represents the prefix that
precedes â .pdbâ in the input filename, e.g., 1bck.pdb -> <protein> = 1bck;
and (2) â xxxxâ represents the run number to differentiate the output files

generated over multiple runs of ProFlex in the same directory. For example,
the user may choose to analyze the flexibility of a protein without ligand
followed by another run on the protein with ligand bound. The user can
selectively add the bonds between the ligand atoms and the protein atoms by
adding only those bonds to the â proflexdatasetâ file and running ProFlex
using the â -pâ option to read-in the input from the â proflexdatasetâ fil
e
that contains the modified bond network information rather than the input
PDB file. This allows the user to avoid reprocessing the entire bond network
from scratch.

1) <protein>_proflexdataset

        - A ProFlex-generated text file that contains all the ATOM and HETATM
records from the input PDB file followed by the information about all the
covalent and non-covalent bonds (e.g., H-bonds, hydrophobic interactions).
For each covalent bond, the pair of atoms that participate is listed. For
each H-bond, the acceptor, the donor, and the hydrogen atom involved are
identified. All the H-bonds and the hydrophobic tethers are assigned a
unique index.

2) <protein>_allbond.xxxx

        - The _allbonds file contains a list of all the covalent and non-
covalent bonds. For every pair of bonded atoms, this file associates a bond
weight (see Jacobs et al. (2001) Proteins 44, 150-165) that represents the
relative flexibility of that bond on a scale of -1 (maximum rigidity) to +1
(maximum flexibility). Based on these weights, ProFlex partitions the bond
set into independently rigid or flexible clusters and assigns a cluster
label accordingly, i.e., positive if flexible, negative if rigid, and zero
to dangling ends (e.g., side chains not participating in hydrogen bonds or
hydrophobic interactions) whose motions are not coupled with other groups in
the protein.

3) <protein>_analysis.log

        - A list of various filtering options selected by the user as well as a brief summary of the ProFlex analysis.

4) <protein>_flex_xxxx.pdb

        - A replica of the input PDB file with the b-value column replaced by each atomâ s flexibility index value and the atom number column is   replaced

by an alternate index to help identify rigid and flexible   clusters easily

in the visualization scripts.

5) <protein>_h-bonds_SEfilt.xxxx, <protein>_h-phobs_SEfilt.xxxx

        - These two text files contain lists of indices of H-bonds and tethers, respectively, filtered based on stereochemical and energy filters.

6) preacptr_info

        - A text file that lists the pre-acceptor atom numbers corresponding to the H-bonds listed in the â h-bondsâ file.

7) decomp_list

        - This file contains rigid cluster decomposition information corresponding to each H-bond broken, which is then processed to generate a postscript file that graphically represents the cluster decomposition.

Scripts:

ProFlex output facilitates visualization of an input protein's flexibility through scripts designed to be run in the PyMol molecular graphics program. PyMol is a biomolecular visualization software tool available at:

                        http://pymol.sourceforge.net/

One version of PyMol is free for academics, and the fully-functional versions are available at modest cost.

ProFlex outputs three PyMol scripts (with extension â .pmlâ) that display: (i) the rigid cluster decomposition of the protein, with each mutually rigid group of atoms with at least 7 atoms given a unique rigid cluster index (e.g., RC1, RC2, etc.), (ii) flexible clusters in which the atoms are coupled through the bond network and can move collectively (also given unique cluster indices, in this case FC1, FC2, etc.), and (iii) the flexibility index of each of the identified rigid or flexible clusters, as measured by the number of remaining degrees of freedom, in terms of single bond rotations, divided by the number of bonds in that region (see Jacobs et al. (2001) Proteins 44, 150-165). +1 represents maximal flexibility and -1 represents maximum rigidity. To map bond flexibility values onto atoms, main-chain atoms are assigned the index of the most rigid bond of its N-CA or CA-C bonds and for side chain atoms, the index is assigned based on the most rigid of all its incident bonds. This information is complementary to that provided by the flexible or rigid cluster decomposition; in that case, coupling or independence of motion is shown, and in the case of flexibility index, relative flexibility or rigidity is shown (where 0 represents

isostatic, or just barely rigid).

The three PyMol scripts for automatically coloring and visualizing the protein flexibility based on flexibility index values, rigid cluster decomposition, and flexible cluster decomposition (collective motions) are output by ProFlex as: <protein>_flex_xxxx.pml, <protein>_RC_xxxx.pml, and <protein>_FC_xxxx.pml, respectively.

Each of the pml scripts can be loaded into PyMol by clicking the "Run" command under the "File" drop-down menu and choosing the script name in the pop-up window that appears. The examples directory under proflex ("$PROFLEX_HOME/../examples/") has examples of each of the above scripts along with a detailed description of the contents of each script and a screen dump of how the corresponding results should appear in PyMol for two proteins.

A fourth way of analyzing the data is presented in a hydrogen-bond dilution profile (a ProFlex run-time option) representing the gradual thermal denaturation of the hydrogen-bond and salt bridge network of the protein with increasing temperature (see A. J. Rader, B. M. Hespenheide, L. A. Kuhn, and M. F. Thorpe (2002)â Protein Unfolding: Rigidity Lostâ , PNAS 99, 3540-3545 and B. M. Hespenheide, A. J. Rader, M. F. Thorpe, and L. A. Kuhn (2002) â Observing the Evolution of Flexible Regions During Unfoldingâ , J. Molec. Graphics and Modelling 21, 195-207).

When hydrogen-bond dilution analysis is performed via the run-time option, the following file is created:

                   <protein>_h-bonds.ps

This is a postscript file that displays the rigid cluster decomposition after each H-bond whose dilution affects the overall cluster set. (It can be conveniently changed to a pdf file by the ps2pdf command available on many Unix systems.) Residue indices (including insertions and missing residues) are displayed along the top line to help identify each rigid cluster distinctly. For each H-bond broken, the index of that bond (a unique identifier assigned in the proflexdataset file, the energy threshold for that bond, and the residues to which the donor and the acceptor atoms belong, are indicated on the right margin of the plot, as well as by carets above the corresponding residues in the graphical output. For more information, see B. M. Hespenheide, A. J. Rader, M. F. Thorpe, and L. A. Kuhn (2002) â Observing the Evolution of Flexible Regions During Unfoldingâ ,

J. Molec. Graphics and Modelling 21, 195-207.

For an exhaustive listing and description of all the output files, please refer to the User Guide ($PROFLEX_HOME/../docs/ProFlex_User_Guide.pdf). Examples of ProFlex runs with input and output files along with snapshots of the visualization scripts when viewed in PyMol are present in the â $PROFLEX_HOME/../examples/â directory for two proteins: 1bck and 1dif.

Happy usage!

Thank you,

The ProFlex Team

Protein Structural Analysis and Design Lab
Department of Biochemistry & Molecular Biology

Michigan State University
E-mail: proflex@sol.bch.msu.edu