SimRNP (version 0.60) User Manual

1. SimRNP: Functionality

SimRNP is a tool for simulations of RNA conformational dynamics (folding, unfolding, multiple chain complex formation etc.), and its applications include RNA 3D structure prediction. SimRNP can be initiated with input files that include either the RNA sequence (or sequences) in a single line (similar to the Vienna format) or in the form of a structure written in PDB format. The PDB format should be simply the structure of the RNA and the protein with no heteroatoms or unusual names. For RNA, the current version can only read the residue names A, C, G and U only (i.e. no modified residues are supported as of yet). For proteins, the residue names must be in the three letter format: ALA, ARG, ASN, ASP, CYS, GLN, GLU, GLY, HIS, ILE, LEU, LYS, MET, PHE, PRO, SER, THR, TRP, TYR, and VAL.

The file outputs from SimRNP consist of the trajectory (or set of trajectories) of the RNA simulation trace (with file extension `.trafl`), a file containing bonding information (with file extension "`.bonds`"), and a PDB file containing a reduced representation of the initial structure at the beginning of the simulation. All these files can be used (or reused) in various tasks of post-processing. SimRNP also generates output during the simulation specifying the total energy at a particular step of the simulation.

Additional tools that accompany this distribution are explained in the Section 6 titled **Additional Tools**.

2. SimRNP: Additional requirements

The distribution comes with a directory "`data`" that contains files with information about the energy function and important settings for the statistical potentials.

- The user **must** have the directory (or symbolic link to some other location) named 'data' inside of the working directory where SimRNP is running.
- The '`data`' directory (or link) contains the energy function files and is therefore essential to run SimRNP.
- The data stored inside these files strongly affect results the user obtains. Therefore, the user should not make changes inside these files unless the user understands the purpose and function of the files; or at least the user should make a backup copy of the data before changing it.
- If this directory is missing in the working directory where SimRNP is called or the program (for some reason) can't see it, the user will usually encounter an error message indicating that this directory could not be found in the current path.

3. SimRNP: Usage

SimRNP is fairly flexible on command line formats requirement; however, the following syntax is required to successfully run SimRNP:

```
> SimRNP -p input_file_PDB
```

or

```
> SimRNP -P input_file_PDB
```

Comments:
♦ Please pay attention to the format.
♦ For the "-p" or "-P" option, in the current SimRNP distribution, the input PDB file **must** contain everything: all the RNA chains and all the protein chains. The order does not matter technically, but it is probably best to arrange the chains such that the proteins are with the proteins and the RNA are with the RNA. Moreover, the residues must be complete, namely they should contain at least the atoms that are used in SimRNP representation: P, C4', N (N1, N9), C2 and C4 (pyrimidines) or C6 (purines) and basically all the heavy atoms for the protein. The names of the atoms may be renamed in the protein structure when translated into the reduced representation.

`-p input_file_PDB`

Note the *lowercase* p. Here, the input file is a structure given in PDB format. In this PDB file, the user submits structures containing one or more RNA chains, RNA chains and protein chains, and/or protein chains. Because the residue names are unique, the program will distinguish if the structure is a protein or an RNA molecule.

`-P input_file_PDB`

Note the *uppercase* P. Again, the input file is a structure given in PDB format. As above, the PDB file can contain one or more RNA chains, RNA chains and protein chains, and/or protein chains. When the uppercase P is used, SimRNP assumes that the PDB file contains restraint pinning information and therefore will read the occupancy column and B-factor column of the PDB file to determine pinning or freezing constraints. In such cases

- If the occupancy is equal to 0.00, then the current atom is fixed: its position is not changed during entire simulation.
- If the occupancy is between 0.00 and 1.00 (but neither 0.00 nor 1.00), then the B-factor value is treated as a radius of unrestricted movement. Hence, a B-factor of 2.00 is 2.0 Å, 4.30 would be 4.3 Å and so forth.
- If the occupancy is equal 1.00, then the atom is not restricted, while the other atoms can be.
- The PDB formatted file can also contain more than one chain. Subsequent chains are recognized only by the chain id (one character). The chains have to be in one piece; they cannot be interlaced. The PDB keyword "TER" is not mandatory.

Comment:

♦ The format of the PDB file has some very strict requirements: i.e., it should be a simple structure

- no heteroatoms
- no non-standard base notations: **the readable residues are A, C, G and U and standard three letter amino acid codes only** (i.e. no modified residues are supported as of yet).
- *Every* 5′-most residue *must contain a P*! This can be added by some software tools. In a pinch, if the P of the 5′-most residue is missing *from any chain* in the PDB file, then the user can rename the O5′ atom of the 5′-most residue to P in that particular chain and the SimRNP will treat that atom as a P. Likewise, the amino acid chain should contain the C_α atom of all relevant residues in the input file.
- Residue names are restricted to A, C, G and U for RNA and the three letter amino acid code for proteins (ALA, ARG, ASN, ASP, CYS, GLN, GLU, GLY, HIS, ILE, LEU, LYS, MET, PHE, PRO, SER, THR, TRP, TYR, and VAL).

♦ If in the case of multichain task, initiated from a pdb file, the atom numbers have to be unique within the whole file. Because of the reduced representation, SimRNP uses "CONNECT" in the output PDB file, where "CONNECT" references the atom numbers. If the atom numbers are not unique, it is likely to result in considerable ambiguity for the display tools. Nevertheless, it doesn't affect simulation itself.

♦ If a simulation is initiated just from a sequence, the residues and atoms are numbered consecutively starting from 1. In case of multichain tasks, subsequent chains are named: A, B, C …Z, a, b, c …z.

3.1 Additional command line options:

`-c simulation_config_file`

The config file contains a variety of important settings and is the recommended way to start a simulation: see section 3.5 for details.

`-o output_files_basename`

If this is not specified, then the program will use pdb or sequence file name as a base name for output.

`-E int_number_of_Replicas`

Request to use the replica exchange Monte Carlo method (REMC), where the program initiates a specified number of relica and perform a REMC simulation.

`-R int_number`

This allows the user to choose, generate or provide a specified random number seed for doing calculations. This is mainly useful when trying to reproduce a simulation. The user can always check the initial random seed in the SimRNP standard output, regardless of whether

the random seed was specified by the user or generated by default (using the clock).

`-l`

The program generates a list of PDB files (each trajectory frame is written to a PDB file in addition to the trajectory file. This is generally not necessary and does take up a lot of space when running a long simulation, so it should not be asked for frivolously. (It is recommended that the user extract all desired PDB formatted files using SimRNP_trafl2pdb, see Section 5.)

`-n number_of_iterarions`

This specifies the number of iterations to be done in the simulation. The argument is the number of iterations in one temperature cycle of simulated annealing. (It is recommended that the user use the config file instead of this command.) This option overrides "NUMBER_OF_ITERATIONS" option specified in config file (see Section 3.5).

3.2 Format of the config file

`-c simulation_config_file`

The argument is a file containing simulation configuration parameters. With this option, the user can configure the simulation in a more advanced way.

The configuration file is text file with one parameter specified on each line. There can be blank lines, each line must contain a command or must be empty.

Config file options

These are some of the options that can be included in a typical config file.

`NUMBER_OF_ITERATIONS N`

This specifies the number of unified iterations in a simulated annealing simulation. This can be overridden by –n option.

`TRA_WRITES_IN_EVERY_N_ITERATIONS N`

This specifies how many iterations should occur before the conformation is appended to the trajectory file. In general, there probably should be at least 10000 iterations of SimRNP before a write is done to generate a meaningful structural change. However, if the user desires to have a chain of quite similar structures, this value can be low.

`INIT_TEMP float`

This specifies the starting temperature of the simulation

`FINAL_STEP float`

Likewise, specifies the final temperature of simulation.

Comments:

♦ !!! The final temperature can be lower or higher that initial temperature. In the second case, the system is being gradually heated. !!!

♦ !!! The final temperature can be also equal the initial temperature. In such a case, the program maintains same temperature during entire simulation !!!

♦ The order of parameters is not important.

♦ !!! The config file lines must be in upper case only

The format of the file up to this point should be as follows

```
NUMBER_OF_ITERATIONS  1000
TRA_WRITE_IN_EVERY_N_ITERATIONS 20
INIT_TEMP        1.35
FINAL_TEMP       0.90
```

There are some additional short-range energy terms that can be set up the use in the configuration file. These options include the following:

```
BONDS_WEIGHT       1.0
ANGLES_WEIGHT      1.0
TORS_ANGLES_WEIGHT 0.0
ETA_THETA_WEIGHT   0.4
```

The BONDS_WEIGHT defines the strength of the bonds holding the structure together. A value zero would mean that there are no bonds and the atoms will all behave as though the RNA is a gas. The ANGLES_WEIGHT define the response of the flat angles to bending. In the current version of SimRNP, the TORS_ANGLES_WEIGHT is ignored, this is intended to add additional helical orientation to the RNA strand. In the place of TORS_ANGLES_WEIGHT, the option ETA_THETA_WEIGHT has been introduced and accomplishes most of the same properties.

Comment:

♦ The latter terms are aimed at RNA parameters. Parameters associated with the protein interactions are currently hardwired.

The user will be notified about redefinition of those variables in the SimRNP output.

4. SimRNP: Output

The main output of a simulation using SimRNP is the trajectory file (`.trafl`). At the beginning of the simulation, the program generates a PDB file (in reduced representation) of the initial structure (.pdb).

4.1 Trajectory format: trafl

The SimRNP output is a trajectory file "`.trafl`". If a single simulation is launched, then one trafl file is generated. If the replica exchange Monte Carlo (REMC) method is requested, the number on each trafl file corresponds to one replica. It is worth noting that, in the case of the REMC method, each trafl file corresponds to the replica, not to the temperature shelf. Hence, each trafl file maintains the conformational continuity (the current frame is always obtained from series of modifications of the previous one). However, the temperature is changed according to the way that the current replica is assigned to a particular temperature shelf. The tracking of the temperature field allows the user to trace back the movement of the replica through the different temperature shelves.

The SimRNP trafl format is a simple text file. Each frame consists of two lines:
– a 5 fields header: (int) (int) (float) (float) (float)
– a line with the atomic coordinates: (a string of floats corresponding to the position of the atoms in each base.)

The format of the header (across the line) is:

```
consec_write_number
replica_number
energy_value_plus_restraints_score
energy_value
current_temperature
```

where

- `consec_write_number:` is just the order in which the file is written.
- `replica_number:` useful for tracing the replica when the files from the simulation are concatenated.
- `energy_value_plus_restraints_score:` indicates the energy including the restraints.
- `energy_value:` indicates the energy without the restraints or just the energy if no restraints are used (see Section 3.4)
- `current_temperature:` denotes current temperature, in replica exchange mode it denotes particular temperature shelf

The format of coordinates line is just:

```
x1 y1 z1 x2 y2 z2 … xN yN zN
```

The coordinates of the subsequent points corresponds to the following order of the atoms: P, C4′, N(N1 or N9 for pyrimidine or purine, respectively), B1 (C2), B2 (C4 or C6 for pyrimidine or purine, respectively). In general, the coordinate line will contain 5*numberOfNucleotides points, so

3*5*numberOfNucleotides coordinate items (in 3D space: 3 coordinates per atom, 5 atoms per residue; hence 15 coordinates per residue). However, for the protein residues, this easy pattern cannot be preserved, and therefore the exact relation between the various elements in the trafl file in SimRNP can only be discerned by studying the bonds file. This requires the development of specialized scripts that read the RNA and amino acid sequences from the PDB file and know how to translate this information into reduced representation.

Comments:

♦ The trajectories can be concatenated. An easy way is using the Unix command

```
> cat file1.trafl file2.trafl … > newfile.trafl.
```

When SimRNP is run in several instances (which is recommended), all files can be combined into a single trafl file and then can be subjected to further processing (e.g., clustering). It also is useful for outputs from simulations in replica exchange mode: especially when several instances are initiated (also recommended). The user will receive many trajectory files. The necessary requirements for trajectory files to be concatenated is that they should have the same number of atoms in coordinate lines, so basically they should originate from simulations of the same sequence. For some types of processing, an additional requirement is that the trajectories should originate from simulations under the same conditions (e.g., same energy function, similar temperature, similar restraints, or other possibilities).

♦ The trajectory may contain only one conformation. In this case it will be just a two line file.

♦ The trajectory file doesn't contain any information about the sequence: chain names, atom/residue numberings, etc Some stages of trajectory processing can be done on just trajectories, while the other stages (especially conversion to pdb) require additional information that are not in trajectory itself.
+ When displaying the files using traflView, the initial ".bonds" file is needed (see Section 6.4).
+ When the user desires to convert some of the trajectory frames into PDB files (Section 6.1), the missing information must be provided by the user as a PDB file (which will only be used as a template, where the actual coordinates will be provided from the given trajectory frame or frames of interest). *It is the user's responsibility is to provide the proper data.*

♦ The 5th field of the header (the temperature field) allows for binning the conformations into specific temperature ranges or assigning them to particular temperature shelves.

5. Examples

When starting SimRNP, please make sure that there is a link to the data file in the working directory (See Section 2). This can be accessed by linking the directory where the distribution is located to the current working directory.

```
> ln -s ${path_to_SimRNP_directory}/data data
```

It is strongly recommended that the configuration file be used with the simulation

```
> cat configSA.dat
NUMBER_OF_ITERATIONS 16000000
TRA_WRITE_IN_EVERY_N_ITERATIONS 16000

INIT_TEMP  1.35
FINAL_TEMP 0.90

BONDS_WEIGHT 1.0
ANGLES_WEIGHT 1.0
TORS_ANGLES_WEIGHT 0.0
ETA_THETA_WEIGHT 0.40
```

Comment:

♦ This config file is set to a fairly long simulated annealing run where the intermittent sampling iterations per frame involve 16000 steps. (The duration of the simulation is commensurate with the total number of iterations and the number of residues.) The user will receive 1001 frames (1000+1: the first is just the starting conformation). When SimRNP is called with the option "-E `number_of_replicas`" (replica exchange), then instead of a simulated annealing run, the replica exchange protocol will be initiated. Replica swaps will be attempted with the same timing as writing into trajectory files. In this case the user will receive N trajectories (where N is the number of replicas), each file of them will contain 1001 frames.

♦ The number of structures you will obtain from a single thread is always:
`NUMBER_OF_ITERATIONS/TRA_WRITE_IN_EVERY_N_ITERATIONS + 1`

Presently, both the protein and the RNA chain much be included in the same PDB file. Presently, SimRNP cannot start from unfolded sequences.

To run SimRNP, the following syntax is then recommended

```
> SimRNP -p mytest.pdb -c configSA.dat -o mytest1 >& mytest1.log &
```

and the output files will be the following

```
mytest1.bonds
mytest1.trafl
mytest1-000001.pdb
```

If there are places in the PDB file where some of the structure should be frozen, then a uppercase P is used with as above with the following syntax

```
> SimRNP -P mytest.pdb -c configSA.dat -o mytest2 >& mytest2.log &
```

with the name of the output files being the same as above for the "-p" option.

To carry out replica exchange Monte Carlo simulations (with 10 replica), the following command line

is sufficient

```
> SimRNP –s mytest.pdb –c configSA.dat –E 10 –o mytestE >& mytestE.log &
```

To run a simulation with a specific random seed, the following syntax is required

```
> SimRNP –s mytest.pdb –c configSA.dat –R 17801 –o mytest1 >& mytest1.log &
```

Where, in this example, the random seed was set to 17801.

Comments:

♦ The –o option is recommended if one attempts more than one calculation in the same directory, because the default output tag (if nothing is specified) is mytest.pdb and in any subsequent calls to SimRNP, SimRNP will overwrite the user's preciously obtained calculations of the same name!!!

♦ Requesting the log file is recommended because it is possible to check the configurations and settings that were used to start the simulation. Since the output it to stderr, the "`>&`" notation should be used. The "`&`" at the end of the line is recommended so that the terminal is free for other activities such as checking the status of the calculations.

♦ The random seed variable for the Monte Carlo simulation is initiated from the clock. This information can be found in the log file after the simulation, in case it is desired to exactly reproduce the same calculation.

6. Additional Tools

Additional tools are also included with the distribution.

6.1 SimRNP_trafl2pdbs – trajectory converter

SimRNP_trafl2pdbs transforms an output from the SimRNP trajectory file into a PDB formatted file. It also makes all–atom reconstruction (optional). The SimRNP_trafl2pdbs converter requires a reference PDB structure (this can be the initial PDB structure generated at the beginning of the run), and the trajectory data (extension "`.trafl`") and a specification of which structure(s) is desired.

To run this application, simply type the reference PDB file, the trafl file, and the list of structures (or ranges) to be converted that is/are desired.

```
> SimRNP_trafl2pdbs structure.pdb trajectory.trafl {list}
```

where `{list}` can consist of the following types of formats

{list}	Output
1	Converts the first frame of an input trajectory file (trafl) to a PDB file (input myfile.trafl: output myfile-000001.pdb).
1 10 35	Converts frame numbers 1, 10 and 35 (of an input trajectory file) into the PDB files with the respective indices (input myfile.trafl: output myfile-000001.pdb, myfile-000010.pdb, etc.).
1 10 35 52:64	Converts frame numbers 1, 10, 35, and the range of frames 52 to 64 to PDB files with the respective indices
:	Converts all frames of an input trajectory file. However, please see warnings in comments (below).

Comments:

♦ The SimRNP_trafl2pdbs converter also requires presence of 'data' directory (or link to it), this is because it has to read in all-atom backbone conformers, as well as all-atom base representations

♦ Be forewarned that frivolous usage of the option ":" will mean that a 10000 frame trajectory file will generate 10000 independent PDB files. If this is expected by the user, fine, but be sure that is what is desired.

6.2 trafl_extract_lowestE_frame.py

This is a simple python script that reads in the trajectory file, finds the lowest energy frame, and outputs it as a single frame trafl file.

Usage:

```
> trafl extract_lowestE_frame.py trajectory.trafl
```

The output (for name above) will be:

trajectory_minE.trafl

This single frame trajectory can be converted into a PDB file using the SimRNP_trafl2pdbs tool.

The user is encouraged to examine this script file if additional tools for trajectory processing are needed or desired.

Comments:

♦ Whereas obtaining the lowest energy is one way (the simplest way) of processing the trajectory files, benchmarks demonstrated that better results were generally obtained by using clustering.

♦ The current version of the tool assumes that the energy was calculated without restraints (the fourth element in the header for each frame, see Section 4.1 for the format of the trafl file). This information is read at line 30 of the script file. When the full energy (with restraints) is to be considered, this line of the script should to be changed to the third item.

6.3 Clustering

Clustering is a tool for processing trajectory file by finding and grouping similar structures together into a single group etc.. Benchmarks demonstrated that clustering provides better results in terms of structure prediction.

The input for clustering is just the trajectory.trafl file. It can be the result of concatenation of several trajectory files. Typically those trafl files originate from multi-instance runs of replica exchange methods. The output of clustering is a set of trafl files corresponding to subsequent clusters.

Usage:

```
> clustering trajectory.trafl fraction_of_lowE_frames_to_cluster
rmsd_thrs_1 [rmsd_thrs_2] …
```

Example usage:

```
> clustering mytest.trafl 0.01 3.5 5.0 >& mytest_clust.log
```

where:
mytest.trafl is an input trafl file,
0.01 indicates that 1% of the lowest energy frames of an input trafl will be subjected for clustering (where the remaining frames will be ignored)
3.5 is the RMSD threshold in the set for first pass of clustering
5.0 is RMSD threshold in the set for a second pass of clustering

NOTE: those two clustering passes (it can be more) are just independent clustering instances but are using the same RMSD all_vs_all matrix. The reason for that is because thecalculation of the RMSD for the all_vs_all matrix is the most computational demanding stage of the clustering procedure (it may take a lot of time).

As a result (assuming mytest.trafl input), the user will obtain:
mytest_thrs3.5A_clust01.trafl
mytest_thrs3.5A_clust02.trafl
mytest_ther3.5A_clust03.trafl
…
…
mytest_thrs5.0A_clust01.trafl
mytest_thrs5.0A_clust02.trafl
mytest_thrs5.0A_clust03.trafl
…
….

Each of the files listed above is a trajectory format file (it is not a trajectory by definition) containing

elements of a given cluster. The frames of each cluster in the trafl files are sorted where the first frame corresponds for the structure closest to the center of density. Thus, conversion to a PDB file can be applied to just the first frame. For example,

```
> SimRNP_trafl2pdbs mytest-000001.pdb mytest_thrs3.5A_clust01.trafl 1 AA
```

This command converts the first frame of the first cluster (obtained with a 3.5A RMSD threshold) to a pdb file (with all-atom reconstruction). Of course other (or all) frames of the cluster can be converted into pdbs as well.

There is one further consideration. The trafl file that is generated as a result of clustering has a different meaning in 5th fileld of the header. For a clustering output, this holds the RMSD value in relation to the center of the density (instead of current temperature)

NOTE: the current implementation of the clustering tool doesn't provide any cut-off for a number of clusters. Actually it identifies subsequent clusters until it exhausts the input data. In such cases, only the first few clusters are meaningful: the remaining clusters are usually just the sparse remains. The user should inspect the clustering output to identify the percentage of structures that are contained in the 1st, 2nd, 3rd, and so on, set of clusters.

6.4 calc_rmsd_to_1st_frame

This tool allows for calculation of the RMSD value in relation to the first frame for the entire trajectory. The input is just the trajectory (trafl) file while the output is a two column file: RMSD vs Energy. The RMSD value is calculated, while the energy value is just read from trajectory file. The calculated RMSD value is in relation to the first frame, so the RMSD in the first line will be always 0.0.

This tool is typically used for generating data for RMSD vs Energy plots, when the native structure is known. It allows for showing the projection of conformational space, rooted in the native conformation. It is worth noting that this tool allows for calculation of the RMSD for the shortest, two frame, trajectory, when each frame can be any structure. The minimum requirement is that number of atoms in each frame must be the same. The RMSD calculator assumes a 1 to 1 correspondence between the atoms. Such calculations only make sense when done for the same sequence.

The input trajectory for the tool can be any conformation; however, typically the user will want to put reference frame at the beginning. It can be done by concatenating trajectories:

```
> cat reference_frame.trafl trajectory1.trafl … > sum_trajectory_with_first_reference.trafl
```

where,

- the `reference_frame.trafl` file is one frame (one conformation) trajectory,
- it can be the native structure obtained by running SimRNP in zero interations mode (see below).

Assuming file names as above:

```
> calc_rmsd_to_1st_frame sum_trajectory_with_first_reference.trafl output_name.rmsd_e
```

The output file becomes a two column file which can be plotted in separate plotting program, like gnuplot. (see figure 3 in the main text)

6.5 TraflView

TraflView allows the user to browse the trajectory files from SimRNP directly. This program was first developed for protein structure manipulation using REFINER in the Kolinski lab[1]. The file with extension "`.bonds`" is required to view the "`.trafl`" files. This tool is still under development; nevertheless, it is a very handy tool for viewing or even monitoring the progress of a SimRNP simulation, so we decided to add it to SimRNP suite.

To browse the trajectories (run this application), simply type output SimRNP "`.trafl`" file and "`.bonds`" file.

```
traflView SimRNPoutput.trafl SimRNPoutput.bonds
```

where *`SimRNPoutput`* is output file generated by SimRNP in the given run.

TraflView can also be used to visualize a set of trajectories given in pdb format. The "`.bonds`" information is require for files with reduced representation that have bond lengths much greater than 1.5 Å. Standard PDB files serving as trajectories can be read directly without the additional information from bonds.

Comment:
♦ As a piece of browsing software,TraflView has some very specific requirements on unix libraries. If TraflView fails to run on the particular Linux system, then it is necessary to install the graphics package mesa. In general, if the package mesa is available and installable, then this can be installed using

```
> sudo aptitude install mesa
```

on a Linux operating system. It is also recommended that a relatively new version of Linux Ubuntu for Fedora Core operating system be installed to guarantee compatibility.

When the user launches TraflView (for longer trajectories it may take some time), a black graphic window will be initiated. The window can be enlarged by pulling with the mouse cursor. In the current implementation, it doesn't maintain the same horizontal/vertical proportions (that will have to be the job of the user).

The user interface is accomplished by several keys and mouse movements (keys are case sensitive):

- c (on/off): colors the structures, rainbow style, when 5' end is blue, and 3' is red
- e: (on/off): show energy plot, green line in bottom part of the window

[1] Software Developed by Michal Boniecki during his PhD studies, initially dedicated for viewing Refiner and CABS trajectories.

— Esc: quitting the viewer

Selecting different frames (moving forward and backward) is accomplished by digit keys:

— 1 and 3: one step backward or forward, respectively.
— 7 and 9: ten steps backward or forward, respectively.
— 4 and 6: a hundred steps backward or forward, respectively,

NOTE: TraflView only reads the digit keys. Therefore, to have this function, it is recommended that the user set the keypad such that the NumLock is switched on.

Rotating, zooming (in/out), and moving the current conformation is accomplished by using the mouse:

— Left button + mouse movement (left/right) and (up/down): rotation of a structure,
— Middle button + mouse movement (only left/right): zooming (in/out) of a structure,
— Right button + mouse movement (only left/right): moving of a structure.

It is worth noting that TraflView also outputs data that is displayed in the terminal where the application is called. In the output, there is the RMSD value to the reference frame. By default, the reference frame is first frame in the trafl file; however, it can be any frame. A new reference structure can be selected by pressing key ‘r’ when the desired structure is currently displayed. The current display can be also dumped into bmp file by pressing key ‘S’.