RedMDStream Quick Start

Filip Leonarski, Joanna Trylska

http://cent.uw.edu.pl/Software/RedMD

The purpose of this document is to provide quick and practical introduction on how to use RedMDStream. For a thorough description of the program please refer to the Manual.

Please send questions or comments to F.Leonarski@cent.uw.edu.pl or joanna@cent.uw.edu.pl.

Start

To use RedMDStream you need access to a Linux or UNIX running computer, with the recent C/C++ compiler.

Please make sure that you have installed the libxml2 library. Besides the library itself, you need headers that allow compilation of the applications using libxml2. In *Debian/Ubuntu* distributions you should install *libxml2-dev* package and dependencies. In *RedHat/Fedora/SuSE* the required package is *libxml2-devel* and its dependencies.

RedMDStream is distributed as a source code. After downloading the code from the web page unpack the file, run the *configure* script, and compile the source code using *make* as displayed below:

```
test@ws-joanna9:~> tar xzf redmdstream-1.0.tar.gz
test@ws-joanna9:~> cd redmdstream-1.0/
test@ws-joanna9:~/redmdstream-1.0> ./configure

(...)

test@ws-joanna9:~/redmdstream-1.0> make
make  all-recursive
make[1]: Entering directory `/home/test/redmdstream-1.0'

(...)

make[1]: Leaving directory `/home/test/redmdstream-1.0'
test@ws-joanna9:~/redmdstream-1.0>
```

To install RedMDStream in the system directories run *make install*, however this requires the “superuser” privileges to execute. Without installing, RedMDStream executable is accessible in the *src/* subdirectory. Please ensure that you can execute the application before going further with this tutorial:

```
test@ws-joanna9:~/redmdstream-1.0> src/RedMDStream
RedMDStream v.1.0 (OpenMP) running on 8 cores
Usage:  RedMDStream inputfile
   inputfile: RedMDStream optimization or protocol XML file
```

The above output shows which version you are currently running, what kind of parallelization is used (serial, OpenMP or MPI) and in the case of parallel execution how many cores are used. By default all available cores are used, however, in the case of OpenMP one may adjust this value with the OMP_NUM_THREADS environment variable:

```
test@ws-joanna9:~/redmdstream-1.0> export OMP_NUM_THREADS=4
test@ws-joanna9:~/redmdstream-1.0> src/RedMDStream
RedMDStream v.1.0 (OpenMP) running on 4 cores
Usage:  RedMDStream inputfile
   inputfile: RedMDStream optimization or protocol XML file
```

OpenMP should be used on desktop machines. For large optimization runs on high performance computing machines, MPI should be the preferred choice. For more information on parallelization consult the Manual, section 1.3.

To run this tutorial a program for visualization is needed. Examples are presented with PyMol, however you may use any other visualization program.

How to proceed?

If you would like to **design a new coarse-grained model** for proteins or nucleic acids, RedMDStream will help you to find the best parameter set for this model. RedMDStream will also help you find out how these numerical parameters of the model, e.g. equilibrium distances or force constants, affect the molecular dynamics simulation of the chosen test biomolecules. For these you need to create:

1. Topology XML file that describes how biomolecules are represented in the model and defines the potential energy function of the system.
2. Protocol XML file that describes the steps to prepare the system (files containing the structure, additional secondary or tertiary structure data), the molecular dynamics protocol (the time step and number of steps, temperature etc.) and analysis methods of the trajectory.

After the model is specified (mapping, potential energy function terms, molecular dynamics protocol), you can use **RedMDStream to optimize the coarse-grained molecular dynamics models** using the evolutionary algorithm, particle swarm optimization or other methods.

Defining topology

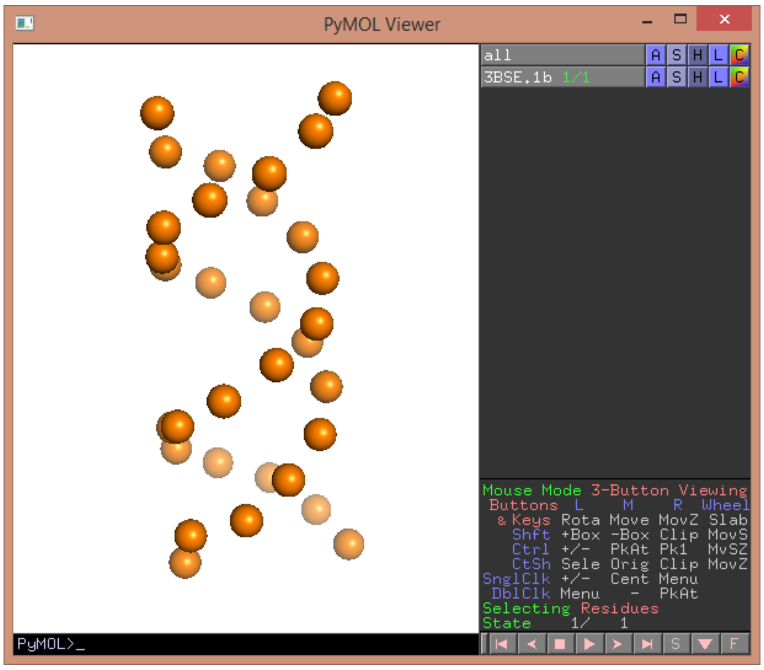
The first step is to tell RedMDStream what kind of a coarse-grained representation you would like to use, i.e., **the bead choice and their mapping, their connectivity network, and potential energy function.** This is achieved by creating a topology XML file. Consult Section 2.2 in the Manual for details on how to define the topology and Examples in Section 3.1 to obtain some test files.

RedMDStream contains also the *Examples/QuickStart* subfolder. Run the *protocol.xml* file with RedMDStream:

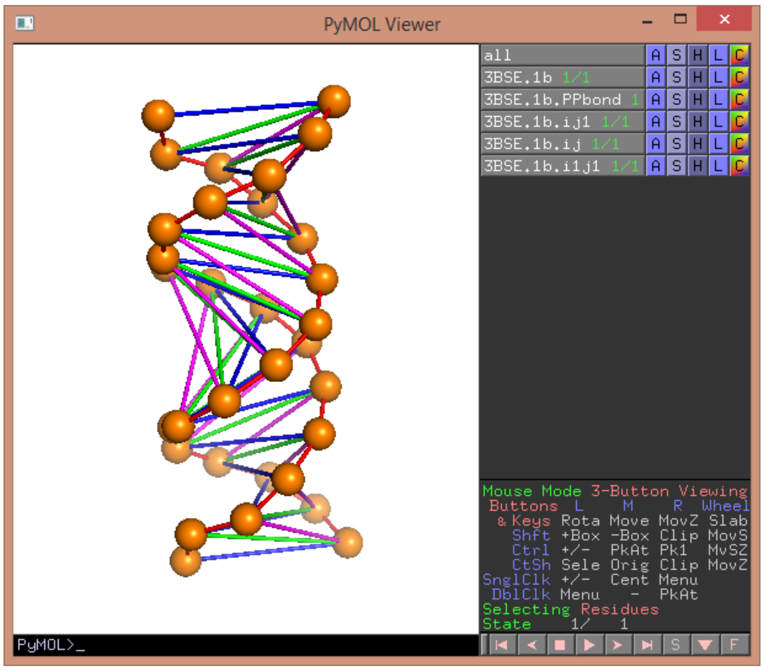
```
test@ws-joanna9:~/redmdstream-1.0> cd Examples/QuickStart
test@ws-joanna9:QuickStart> ls
3BSE.xml  optimize.xml protocol.xml  topology.xml
test@ws-joanna9:QuickStart> ../../src/RedMDStream protocol.xml
RedMDStream v.1.0 (OpenMP) running on 4 cores
test@ws-joanna9:QuickStart> ls
3BSE.1b.i1j1.pdb  3BSE.1b.PPbond.pdb  3BSE.1b.sxml.tab2  protocol.xml
3BSE.1b.ij1.pdb   3BSE.1b.sxml        3BSE.1b.sxml.tab3  topology.xml
3BSE.1b.ij.pdb    3BSE.1b.sxml.tab0   3BSE.xml
3BSE.1b.pdb       3BSE.1b.sxml.tab1   optimize.xml
test@ws-joanna9:QuickStart>
```

Note, *../../src/RedMDStream* path – since RedMDStream was not installed with *make install*, one should always use the relative path to the executable. If *make install* was executed before, the plain *RedMDStream* command should suffice, instead of*../../src/RedMDStream*.

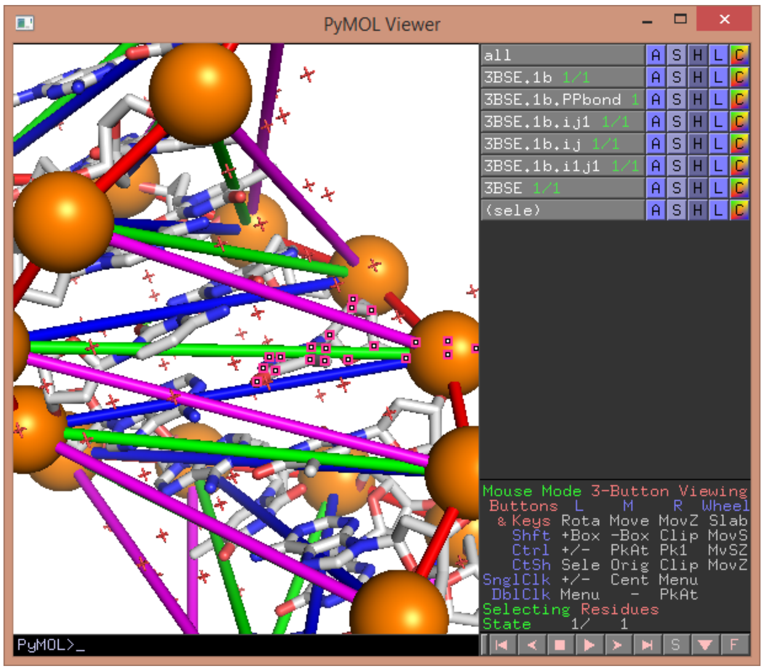
As a result new files are created. Please open the *3BSE.1b.pdb* file with PyMol and pick the sphere representation (find the letter S next to 3BSE.1b, press it, and select "spheres"). You will see the non-connected beads on the screen:



Next you may load the other .pdb files (*3BSE.1b.PPbond.pdb*, *3BSE.1b.ij.pdb*, *3BSE.1b.ij1.pdb*, *3BSE.1b.i1j1.pdb*). Use the stick representation for the other files (click S and "sticks") and use a different color for each one (click C next to the file name and choose the color from the list):



In this representation each rule is visualized. One may also compare the structure with the X-ray original file. Write "fetch 3BSE" in the prompt below the structure. Now you may compare the full-atomistic and coarse-grained models:



Do not close PyMol – these settings will be necessary for the next points.

Running a simulation

To run a simulation with the model in the example above please add the following to the *protocol*.xml file:

```
<SIMULATION mol="3BSE" mode="langevin" gamma="5" dt="10 fs"
dcd="3BSE.dcd" temp="310">
        <MINIMIZE/>
        <STEP time="10 ns" saveFreq="100"/>
</SIMULATION>
```

between the last <OUTPUT ...> and </PROTOCOL> tags in the edited file. These additions will minimize the structure and run **Langevin** dynamics for **10 ns**, with a **5 ps^{-1}** dumping constant (gamma), **10 fs** time step, **310 K** temperature. The trajectory will be saved to the *3BSE.dcd* file every **100** steps (100 x 10 fs = 1 ps). Now you can run the modified *protocol2*.xml with RedMDStream:

```
test@ws-joanna9:QuickStart> ../../src/RedMDStream protocol.xml
RedMDStream v.1.0 (OpenMP) running on 4 cores
test@ws-joanna9:QuickStart>
```

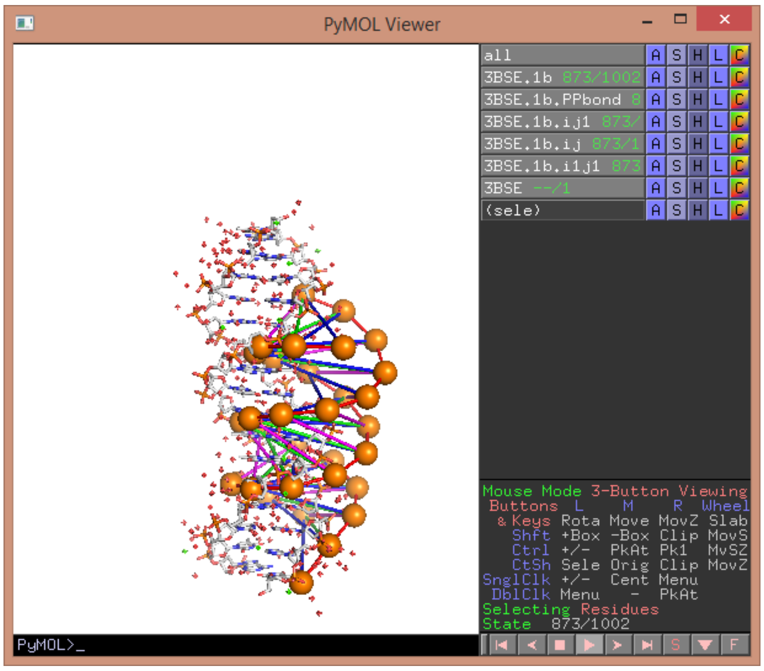
A new file, *3BSE.dcd* appears in the directory. To visualized the dynamics in PyMol please type the following command (use PyMol settings and molecules loaded in the previous point):

```
load_traj 3BSE.dcd, 3BSE.1b
```

For the best visual effect you can also load the trajectory for each bond type:

```
load_traj 3BSE.dcd, 3BSE.1b.PPbond
load_traj 3BSE.dcd, 3BSE.1b.ij
load_traj 3BSE.dcd, 3BSE.1b.ij1
load_traj 3BSE.dcd, 3BSE.1b.i1j1
```

Now press the play button in the lower right corner of the Pymol screen:



By visual inspection you can see that the DNA molecule becomes vertically “squeezed” early in the dynamics, as compared to the reference X-ray structure. We will optimize the force field parameters and model in the next section of this QuickStart. You can modify the SIMULATION section of the *protocol2.xml* file to better investigate this effect:

```
<SIMULATION mol="3BSE" mode="langevin" gamma="5" dt="10 fs"
dcd="3BSE.dcd" temp="310">
        <MINIMIZE/>
        <STEP time="10 ns" saveFreq="100"/>
        <ANALYZE>
            <RMSD var="rmsd" />
            <DISTRIBUTION min="0" max= "100" bins="300" >
                <TXTFILE save="all.txt" />
            </DISTRIBUTION>
        </ANALYZE>
</SIMULATION>
<PRINT var="rmsd"/>
```

For details on what could be found in the protocol XML file consult Sec. 2.3 of the Manual with the examples described therein. If executed, RedMDStream will print the average root mean square deviation value, as well as output the distribution of all the P-P distances in the structure in a separate file *all.txt*:

```
test@ws-joanna9:QuickStart> ../../src/RedMDStream protocol.xml
RedMDStream v.1.0 (OpenMP) running on 4 cores
rmsd : 5.20474
test@ws-joanna9:QuickStart>
```

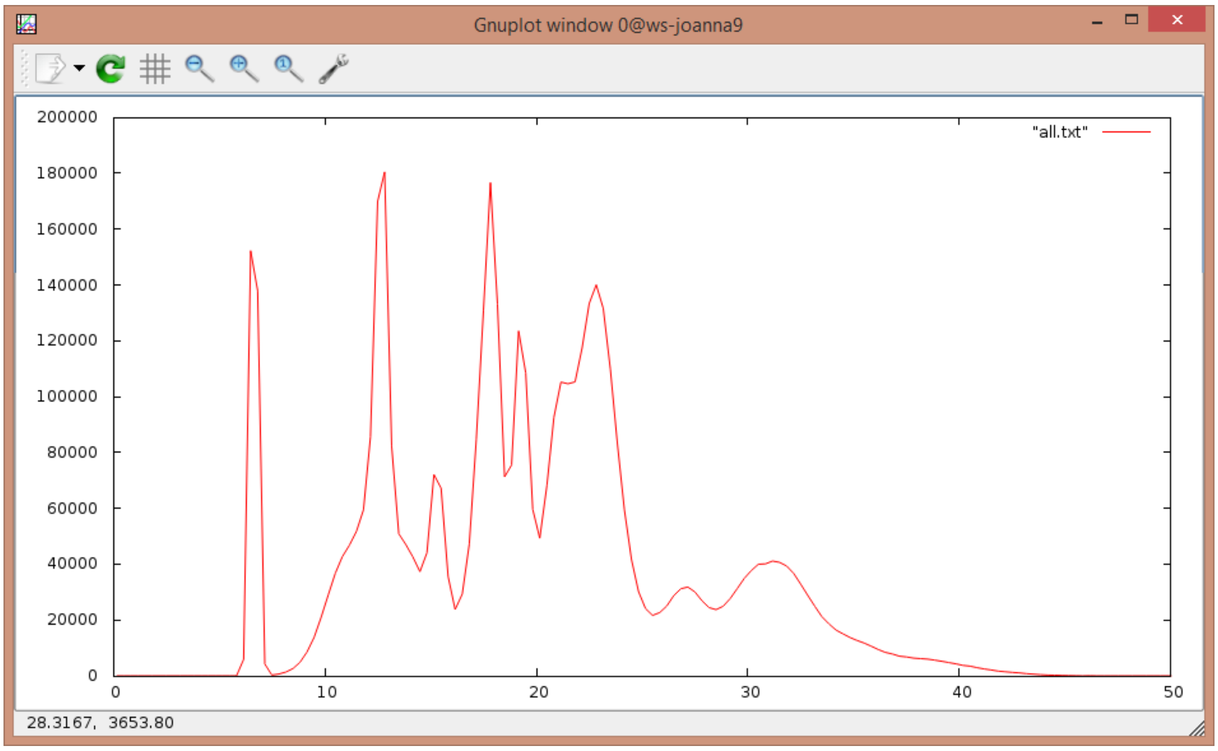
Distributions can be plotted using e.g., Gnuplot:

```
test@ws-joanna9:QuickStart> gnuplot
        G N U P L O T
        Version 4.6 patchlevel 3    last modified 2013-04-12
        Build System: Linux x86_64

        Copyright (C) 1986-1993, 1998, 2004, 2007-2013
        Thomas Williams, Colin Kelley and many others

        gnuplot home:     http://www.gnuplot.info
        faq, bugs, etc:   type "help FAQ"
        immediate help:   type "help"  (plot window: hit 'h')

Terminal type set to 'qt'
gnuplot> plot [0:50] "all.txt" w lines
gnuplot>
```

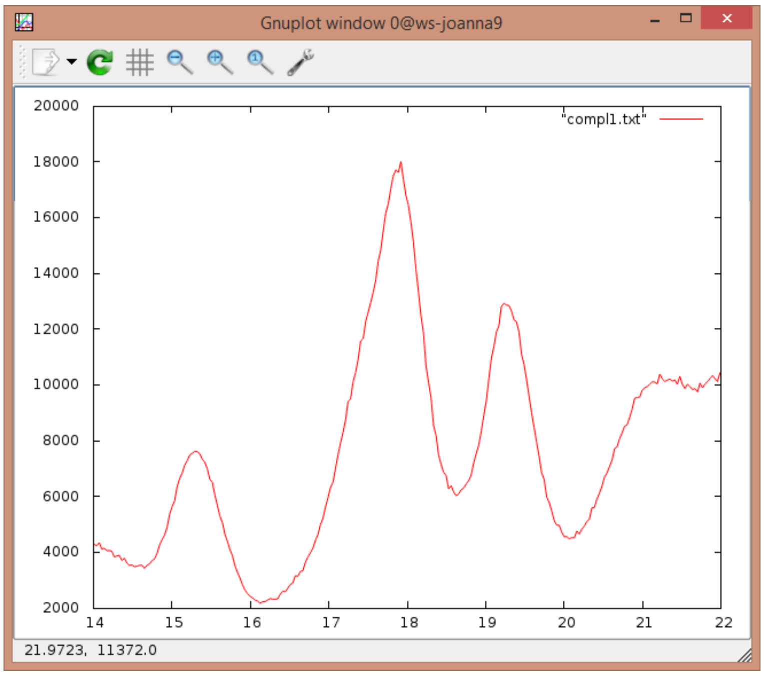


This distribution can be compared with the reference data, e.g. from the ribosome X-ray structures or full-atomistic Amber ff99 molecular dynamics simulation – please see Sec. 3.2.1 in the Manual and the example in the *Examples/Protocol/Distribution* directory. This distribution contains all possible P-P distances. However, it might be more practical to restrict this choice for a particular type of interactions, e.g. bead distances between complementary pairs. Use the following ANALYZE section (inside SIMULATION):

```
<ANALYZE>
      <RMSD var="rmsd" />
      <DISTRIBUTION min="0" max= "100" bins="300" >
          <TXTFILE save="all.txt" />
      </DISTRIBUTION>
      <DISTRIBUTION min="14.0" max= "22.0" bins="250" bondRule="compl1" >
          <TXTFILE save="compl1.txt" />
      </DISTRIBUTION>
</ANALYZE>
```

This disctribution can be also plotted with Gnuplot:

```
gnuplot> plot [14:22] "compl1.txt" w lines
gnuplot>
```



Restricting the distance distribution to a particular set of beads provides a picture that is easier to analyze and compare.

You are now welcome to modify the *topology.xml* file to change the force field parameters and see how the distributions and RMSD are affected.

When manually optimizing the model, one should bear in mind that every trajectory starts from a random velocity ensemble. In the Langevin setup, often used to mimic the effects of solvent, there is also a stochastic part in the equation of motions taking into account random collisions of molecules. Random effects do affect the simulation and repeating the simulation multiple times might give a different result each time. Any results of the model optimization will be meaningful if, and only if one knows how the random effects affect the simulation outcome. If any numeric parameters are used to compare trajectories, their difference is meaningful only if it is higher than variations of these parameters between subsequent simulations of the same system.

Optimizing a model

Next, try to **automatically optimize the 1-bead DNA model** for the system already used in the previous points. First, you need to choose the parameters for this automatic optimization. We propose to use here the r0 and r1 parameters of the nonbonded interactions and r0 and r1 parameters of the i:j interactions between the complementary pairs. However, you are welcome to use any of the available parameters. Second, please make a copy of *topology.xml* file to *topology2*.xml. Third, modify *topology2.xml* changing all the optimized parameters from the numbers to variables:

```
(…)

<NONBONDED>
    <BOND_CUST name="repulsive" formula="E*((1-exp(-alpha*(r-r0)))^2-c)*(1+Enb2*(1-
tanh(l2*(r-r2))))*0.5*Enb1*(1-tanh(l1*(r-r1)))+A*Enb1*(1-tanh(l1*(r-r1)))">
        <VAR name="r0">nonb_r0</VAR>
        <VAR name="E">75.0</VAR>
        <VAR name="alpha">0.014</VAR>
        <VAR name="r1">nonb_r0+nonb_r1</VAR>
        <VAR name="r2">16.5</VAR>
        <VAR name="l1">1</VAR>
        <VAR name="l2">3</VAR>
        <VAR name="Enb1">1.0</VAR>
        <VAR name="Enb2">0.77</VAR>
        <VAR name="c">0.007</VAR>
        <VAR name="A">0.4</VAR>
    </BOND_CUST>
</NONBONDED>

(…)

<BOND_CUST name="compl1" formula="E*((1-exp(-alpha*(r-r0)))^2-c)*Enb1*(1-tanh(l1*(r-
r1)))">
    <BONDRULE save="compl1" >
        <INTERACTION name="WC" pos1="1" pos2="2"/>
    </BONDRULE>
    <VAR name="r0">ij_r0</VAR>
    <VAR name="E">2.8</VAR>
    <VAR name="alpha">0.8</VAR>
    <VAR name="r1">ij_r0+ij_r1</VAR>
    <VAR name="l1">3.7</VAR>
    <VAR name="Enb1">1.0</VAR>
    <VAR name="c">0.4</VAR>
  </BOND_CUST>

(…)
```

Since the topology file was copied, remember to change in the *protocol2.xml* file the link to the topology file (from *topology.xml* to *topology2.xml*). It is also better to remove **dcd="3BSE.dcd"** from the SIMULATION section in *protocol*.xml. The trajectory will be then saved in memory, not in a file, which is more convenient in the automatic optimization.

Next, an optimization XML file *optimize.xml* has to be created, with the following content:

```
<OPTIMIZATION method="pso">
  <DATA>
    <PARAMETER name="nonb_r0" min="10.0" max="20.0"/>
    <PARAMETER name="nonb_r1" min="1.0" max="5.0"/>
    <PARAMETER name="ij_r0" min="10.0" max="20.0"/>
    <PARAMETER name="ij_r1" min="1.0" max="5.0"/>
    <SCORE name="rmsd" val="rmsd" max="100"/>
  </DATA>
  <PROTOCOL runs="5" file="protocol.xml"/>
  <OPTIONS>
    <ITERATION size="10" number="8"/>
    <NORMALIZE_TOTAL />
    <FILE saveTxt="pso1.csv" separator=","/>
  </OPTIONS>
</OPTIMIZATION>
```

This file describes how to perform the optimization with RedMDStream. Section enclosed between <DATA> and </DATA> tags describes what parameters of the force field are varied (in our case it is *nonb_r0*, *nonb_r1*, *ij_r0* and *ij_r1*) with acceptable range of the variation and the way how the quality of the force will be evaluated – the score. In this example we use a root mean square deviation to measure, how much the structure differs during the dynamics from the starting conformation, however RedMDStream allows also usage of distance distribution comparison and atoms mobility measured with a root mean square fluctuation. Please note that the optimize XML file needs to be consistent with respective protocol and topology XML files. Parameter names found in the optimize.xml file are also used in the *topology.xml* file and a score variable name (**val=”rmsd”**) is present in RMSD calculation line in the *protocol.xml*.

<PROTOCOL> tag decides which file will be used for the MD protocol – in this case *protocol.xml* file. It will be re-executed **5** times to account for uncertainty of the score caused by stochastic nature of the Langevin dynamics.

Finally the <OPTION>…</OPTION> section defines parameters of the optimization itself: the optimization method (Particle Swarm Optimization), the number of the iterations (**8**), the size of each iteration (**10**) and score normalization to a 0.0-1.0 range (NORMALIZE_TOTAL tag). Details can be found in Sec. 2.4 of the Manual and in the examples in Sec. 3.3.

To run the optimization please write:

```
test@ws-joanna9:QuickStart> ../../src/RedMDStream optimize.xml
RedMDStream v.1.0 (OpenMP) running on 4 cores
```

The optimization, even in such a small case, can take a significant amount of time (a few hours). Please be patient. The outcome is saved in the *pso1.csv* file and looks as follows:

```
Iter,Memb,nonb_r0,nonb_r1,ij_r0,ij_r1,Total,TotalStd,rmsd,rmsdStd,

0,0,15.375,3.04291,18.0699,4.79628,0.037057,0.00222244,3.7057,0.222244,
0,8,12.0661,4.45549,17.2033,1.97478,0.0638561,0.000207455,6.38561,0.0207455,
0,3,10.8927,2.26881,16.3426,1.80455,0.0797185,0.000679319,7.97185,0.0679319,
0,6,13.5176,2.80378,16.2402,1.93565,0.0614269,0.00132735,6.14269,0.132735,
0,1,17.9739,3.18139,19.4228,4.54002,0.0696849,0.00080522,6.96849,0.080522,
0,4,16.9879,4.61107,14.6393,3.77537,0.144045,0.00309899,14.4045,0.309899,
0,7,16.1935,2.2891,16.182,2.92617,0.12699,0.00531181,12.699,0.531181,
0,9,14.8815,2.48051,11.7505,2.74755,0.0650491,0.00376665,6.50491,0.376665,
0,2,13.1813,3.92159,17.8842,2.40089,0.0476022,0.000502308,4.76022,0.0502308,
0,5,16.6183,3.15758,14.4455,2.22925,0.103281,0.00586589,10.3281,0.586589,
```

The last two columns list the value of the root mean square deviation for each parameter set. Results are reported as an average of 5 runs (because of the **runs="5"** parameter in the <PROTOCOL … > tag from *optimize.xml*), followed by standard deviation of these 5 executions. The file can be also opened with Microsoft Excel or LibreOffice Calc for further analysis. For details on how to analyze this output see Sec. 2.4.8 and 2.4.9 of the Manual, as well as the examples in Sec. 3.3.6.