Last Update April 2020

User Manual for TS2CG

Backmapping Triangulated Surface to Coarse-Grained Model

Weria Pezeshkian, Melanie König, Tsjerk A. Wassenaar and Siewert J. Marrink

Groningen Biomolecular Sciences and Biotechnology Institute and Zernike Institute for Advanced Materials University of Groningen, Groningen, The Netherlands

Nat Commun **11,** 2296 (2020).

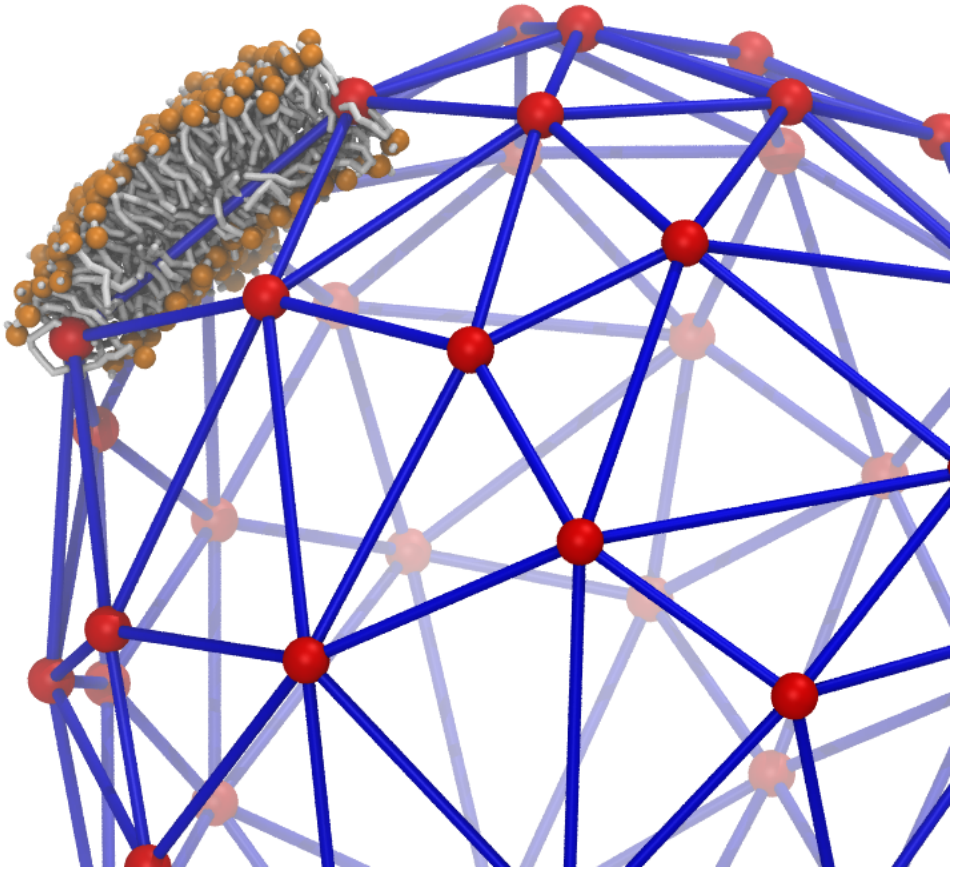

Table of Contents

Introduction

TS2CG is an algorithm to backmap a triangulated surface (TS) structure to a corresponding coarse-grained (CG) model.

TS2CG is implemented in C++ and includes two separate scripts. *Pointillism* and *CG Membrane Builder*. Both of these scripts are available on the Martini website.

Compiling

For compiling, gcc version 8.3.0 or above is needed.
In the source code folder, execute the script "*compiling.sh*" as

./compiling.sh

In this folder, two binary files will be generated **PLM** and **PCG.** PLM preform pointillism (Step 1 and 2) and PCG preforms Membrane builder (Step 3 and 4).

Pointillism

Read a TS file (or a dynamical triangulated surfaces simulation trajectory file) and extend the TS file based on the given rescaling factor and an approximated area per lipid. The output is two folders, 1) a folder contains few visualization files 2) a folder that can be read by *CG Membrane Builder* script (See next for more information).
Note: the approximated area per lipid does not need to be precise, it will be modified during the later processes.

Input files

Triangulated surfaces file: A triangulated surface file that can be read by this script should have an extension of ".q" or ".dat".

The *.q file: is formatted as shown below.

```
50   50   50
1840
0       21.4    33.8     32.7    0
1       38.1    26.1     32.3    0
2       40.9    24.2     19.9     0
.
.
.
1839    31.2   323.2    23       0
0
3680
0   75    776     1043     1
1   796   1821    752      1
2   995   1027    279      1
3   662   1162    56       1
4   167   38      391      1
.
.
```

Line 1: Box information (3 double numbers).
Line 2: Number of vertices (1 integer number; Let's call it NV).
Line 3 to NV+2: Vertex ID and coordinate (1 integer number and 3 double numbers).
Line NV+3: Number of triangles (1 integer number; Let's call it NT).
Line NV+4 to NV+NT+3: Triangle ID and ID of its vertices (4 integer number).

The *.dat files, are DTS simulation trajectory outputs. It contains information about vertices, triangles and protein positions. Many of the information is not used by PLM and it is heavy. *.dat at minimum must contains 5 sections and will end with word "**END**".

Note: This file will be replaced by a different format soon.

Section 1: System information

```
Step 1
Box 100.00000000 100.00000000 100.00000000
32450 64896 194688 402
```

Line 1: is DTS output step. Not used by **PLM.** Can be identical for all the files.
Line 2: Information about the system box.
Line 3: numbers of vertices, triangles, links and inclusions respectively in the system.

Section 2: Vertices

```
Vertex 0
Position 47.33860010 56.94659980 63.58480010
Network 0
Inclusion 0 0
NeighbourLinks 5
12899 12900 12916 70840 70852
NeighbourTriangle 5
4299 4300 4305 23613 23617
NeighbourVertex 5
9305 15614 15613 15623 9304
Kappa 20.00
```

Line 1: vertex id
Line 2: vertex position
Line 3: Network id (not used by **PLM**). Can be identical for all the vertices.
Line 4: If the vertex contains an inclusion, the first number is the inclusion type id (will be used in **PCG** to place the right protein) and the second number is inclusion id.
Line 5: Number of the links connected to the vertex.
Line 6: The id of all the links connected to the vertex
Line 7: Number of the triangles connected to the vertex.
Line 8: The id of all the triangles connected to the vertex.
Line 9: Number of the neighboring vertices of the vertex.
Line 10: The id of the neighboring vertices of the vertex.
Line 11: Rigidity of the vertex. Not used in **PLM**. Can be identical for all the vertices.

Section 3: Triangles

```
Triangle 0
Visualization 1
Nodes 18903 18904 18905
```

Line 1: Triangle id.
Line 2: Visualization state. 0, will remove it in ParaView file format.
Line 3: id of the triangle vertices.

Section 3: Links

```
Link 0
Vertices 362 8114 8116
Triangle 0
Neighborlinks  1 2
Visualization 1
MirorLink 15
```

Line 1: id of the link.
Line 2: the first two number is the link vertices and the third one is third vertex in the triangle that the link belong to.
Line 3: id of the triangle that the link belongs to. (Considering anti-clock wise system)
Line 4: Visualization state. 0, will remove it in vmd file format.
Line 5: The id of a link that is antiparallel to the link and build by same vertices.

Section 3: Inclusions

```
Inclusion 401
Type Pro 1
m_NSymmetry 2
LocalDirection 0.93341443 0.35880009 0.00000000
Kappa 10.00000000 10.00000000 0.00000000
Curvature 0.00000000 0.50000000 0.00000000
Vertex 2028
```

Line 1: inclusion id.
Line 2: inclusion type (later will be used by **PCG** to place the right protein).
Line 3: Can be identical for all inclusions. (Not used by **PLM**).
Line 4: Can be identical for all inclusions. (Not used by **PLM**)
Line 5: Can be identical for all inclusions. (Not used by **PLM**)
Line 6: Id of the vertex that inclusion belong to.

Output files

Output files will be separated into two folders. 1) A folder contains few visualization files. 2) a folder that can be read by *CG Membrane Builder* script.

Command line option and examples Options

```
./PLM  -TSfile TS.q -r PLM  -Mashno 2  -rescalefactor 3  -o $path/data
```

```
-rescalefactor          rescaling factor
-bilayerThickness       bilayer thickness
-r                      function (PLM/check)
-o                      name of the output folder
-monolayer              to generate monolayer instead (1/-1)
-shape                  fixed geometry (flat)
-TSfile                 TS file name
-Mashno                 number of consecutive Pointillism operations
-AlgType                algorithm type for Mosaicing
-ap                     an approximation of lipid AP
-h                      help
```

For a flat bilayer use `-shape flat` option in the command line (for this, no TS file is required).

CG Membrane builder

It generates a Gromacs based topology and coordinate file using the Pointillism output files. If the system contains protein, or if the user wants to add proteins, a Gromacs coordinate file (.gro file format) of the protein structure should be provided. By default, Martini forcefield lipids will be used. To backmap to another CG forcefield a lipid structure library should be provided.

Input file str

The input file should be formatted as

```
include  protein.gro
[Lipids List]
; lipidname ratioup ratiodown  ap
POPC 0.5 0.5 0.63
POPE 0.5 0.5 0.64
End
[Protein List]
;proteinname inclusionID surface_coverage 0 0 z-position
STxB 1  0.5   0  0  -1.1
End Protein
```

First line is to include gromacs formatted coordinate file of a protein. Note, if more than one protein type is needed, add a new line for each protein type.
In the [Lipid List] section, each entry defines a different lipid type and the ratio of it in the upper and lower leaflet of the bilayer. The last entry defines the area per lipid (in nm^{-2}).
The last section [Protein List] contains one entry per protein type. Depending on which TS file format is provided, only one of the first two number will be used: If a *.dat file is provided as input for the pointillism procedure, the protein ID (first number) has to correspond to the inclusion ID in the .dat file and the protein inclusions will be placed at the positions from the DTS simulation output. If a *.q file is used instead, the first number will be ignored and the second number will be used as the protein coverage ratio. In this case, the proteins will be placed randomly until the defined surface coverage is reached (If it is possible!!). It is also possible to place proteins at desired positions, which will be explained in another tutorial. The following two numbers, which define the rotation in plane and tilt angle, will be implemented soon. The last number defines the protein position along the membrane normal.

Lipid library file

Example of lipid library file is as below

```
Description Martini Map CG
Version Martini 2
 [ DOPC ]
1 NC3 0 0    1
2 PO4 0 0    0
3 GL1 0 0   -1
4 GL2 0 0.5 -1
5 C1A 0 0   -2
6 D2A 0 0   -3
7 C3A 0 0   -4
8 C4A 0 0   -5
9 C1B 0 1   -2
10 C2B 0 1  -3
11 C3B 0 1  -4
12 C4B 0 1  -5
```

Command line option and examples Options

```
./PCG -str input.str -dts data -seed 34 -LLIB Martini2.LIB -Bondlength 0.15
```

-dts	Pointillism output folder address
-str	Input file name
-defout	output files prefix
-Bondlength	Initial bond guess;
-LLIB	a CG lipid library file name
-renorm	renormalized the lipid molar ratio
-iter	iter*number of the point try to cover the surfaces
-seed	seed for random number generator

Output files

The output file of this command is a Gromacs coordinate file and topology file of the full system.

Manuscript Example Files

The required files to perform the examples in the manuscript are available in a folder called tutorial/required_files_for_manuscript_examples:

For the vesicle growth example we provide dumbbell.q, input.str file and Martini2.LIB (the lipid structure library).
For the STxB bud formation example we provide STxBBud.dat, input.str, Martini2.LIB and STxB.gro (gromacs coordinate file for STxB with the bound Gb3 lipids in a DOPC lipid bilayer). The STxB-Gb3 complex gromacs topology file (*.itp) is also provided.

The mitochondria example includes a *.q and *.str file for each bilayer and Martini2.LIB.

Please note: The simulation procedures of the manuscript examples are complicated and computationally expensive. Therefore, we recommend you to first follow the simple tutorial below.

Mixed Bilayer Tutorial

In this tutorial we are going to explain how to use TS2CG for creating a mixture of DOPC POPC lipid on a twisted vesicle. For this, first download the source code. Then

```
cd source_code
./compile.sh
```

This will create two binary files, **PLM** and **PCG**.

You can find all the files needed for this tutorial here:

```
cd tutorial/mixedbilayer
```

We need a triangulated surface (TS) of the considered shape. A TS file with the name twisted.q is provided in this folder.

To see how does this TS surface look like we use PLM binary to generate vmd and paraview readable files by below command.

```
path/PLM -bilayerThickness 0 -rescalefactor 1 -Mashno 0 -o ini -TSfile twisted.q
```

This command generates a folder with name inivisualization_data. Enter this folder and use vmd or paraview to have a look in the structure of the initial triangulated surface.

Next we use PLM to generate input files for membrane builder script from twisted.q TS.

```
path/PLM -bilayerThickness 4 -rescalefactor 1.5 -o point -TSfile
twisted.q
```

This command will generate two folders, *point* and *pointvisualization_data.* Enter *pointvisualization_data* folder and make sure that different part of surfaces does not penetrate into each other. The *point* folder will be used for membrane structure generation.

Next, we will use PCG binary to generate CG membrane structure for DOPC POPC mixture. For this, we need an input file and lipid structure library file.

To make an input file, write below text into a file with str extension (input.str).

```
[Lipids List]
POPC 0.7 0.7 0.65
```

```
DOPC 0.3 0.3 0.65
```

This file indicates that your system should contain POPC and DOPC lipids. 70% of the upper and lower monolayers lipids should be POPC and 30% should be DOPC. There is also a required field for APL of each lipids.

The lipid library should contain both DOPC and POPC lipids (Here Martini2.LIB). Then, using below command, you can generate two files, one coordinate file (mixed.gro) and one a topology file (mixed.top).

```
Path/PCG -dts point -str input.str -seed 39 -Bondlength 0.15
-LLIB Martini2.LIB -defout mixed
```

Note, the topology file does not contain any force field or molecule definition files. Include the force field header in the mixed.top and run simulations using Gromacs.

Instead of above procedure, you can just run easyrun.sh script in the tutorial folder. This script, in addition to the above procedures, It will run, an energy minimization, equilibration and a short md run.

How to place proteins at specific positions

*.q triangulated surfaces file format does not include position of the proteins (inclusions) and *.dat file is very complicated an impractical. However, TS2CG allows for protein placement. For random placement, it is easy and the user can define proteins in the str file and place them randomly. For specific placement, with the current version, it requires an extra afford. Below, we explain a considerably simple procedure to do this.

PCG script, reads inclusion file “IncData.dat” in the PLM output folder and checks if this file is existing or empty. In both cases, it will then produce random distribution of inclusions based on the converge defined in the input file (*.str). Therefore, to enforce PCG to place proteins in a specific position is to manually or using script create a IncData.dat file in the PLM output folder.

An example of IncData.dat is as following

```
< Inclusion NoInc      2   >
< id typeid pointid lx ly lz  >
    0    1  460   0.923  -0.212   0.322
    1    1  427   0.891  -0.138  -0.433
```

The first line tells, how many inclusions are in this file.
Second should be kept the same.
Third and fourth lines are line each defining one protein. The first number is an id, the second line is protein type id which must match the protein id in the *.str (PCG input file) file. Third line is the id of a vertex from the original triangulated surface that you want to place the protein there. The last 3 numbers representing a vector in 3D space that shows the orientation of the proteins. But do not worry, if you do not know what would be the orientation of the protein at that point just use (1,0,0). This will be

converted to a vector in the plane of the membrane. By creating this file and placing it in the PLM output folder, you can just normally run PCG and place the proteins in the position that you prefer.