User Manual

Compiling the source code

To compile the OPENDTS source code just execute “compile.sh” script.

```
./compile.sh
```

This will create three binary files: **DTS**, **CNV**, and **GEN**. The **DTS** is for running simulations; **GEN** is for generating triangulated files (TS) and **CNV** is for converting different file formats.

How to run a simulation using OPENDTS

To run a simulation using **OPENDTS**, we need to feed at least two files to the **DTS** binary. An input file (with *dts* extension) that includes all the simulation inputs (see the “Input File” section) and a topology file that contains information about the topology of the TS mesh. The topology file could be in *top* or *tsi* format (see the Topology File sections).

With these two files a simulation can be started as.

```
$PATH/DTS -in input.dts -top top.top
```

Some of the parameters that are defined in the input file could be overwritten in the command line. Note: running a restart simulation is only possible by providing the restart file name in the command line as: `-restart restart_file_name.res`

Simulation Output files

There are a number of output files for each DTS simulations that are listed below.
vtu files: This is a list of files that shows the evolution of the system and can be opened by paraview.
tsi files: A list of files that shows the evolution of the system. These files can be converted to different file format using the Convert Script.
restart file: A file to restart a simulation. This file has “res” extension.
log file: Stores some information of about the current simulation. This file has “*log”* extension.
energy file: This file end with *“-en.xvg*” and contains system the bending energy, box size, volume and etc., at each step.

Command line options

All the option that can be defined in the command line. Also by executing `./DTS -h` you can see this list.

Identifier	Type	Default value	Description
-in	string	Input.dts	input file name
-top	string	topology.top	Topology file name
-b	int	1	initial time step
-e	int	100	final time step
-seed	int	36723	Random number seed
-defout	string	output	A general string for label a specific run
-ndx	string	Index.inx	Index file name
-restart	string	NO	restart file name
-angle	double	-0.5	minimum of cos of the angle between two faces
-minDist	double	1	Square of minimum distance between two vertices
-maxDist	double	3	Square of maximum link length

Input file option and format

The input file must have a dts extension. Below, you can find the most common options defined in this file. However, this file could contain more information, including inclusions and their interactions. Other options are defined in their corresponding sections.

```
Integrator                      = MC
MC_Moves                        = 1 1 1
Initial_Step                    = 1
Final_Step                      = 1000000
```

```
Display_periodic                = 1000
OutPutEnergy_periodic           = 1000
Restart_periodic                = 10000
Kappa                           = 15 0
Seed                            = 37382
OutPutTRJ_TSI                   = 1000  10  TrjTSI
GeneralOutputFilename           = output
Cell_Size                       = 2.5 2.6 2.5
```

Integrator: The algorithm for updating the configuration of the system. Currently only "MC" available.
MC_Moves: An option for activating or disactivating the MC moves. First one is vertex move, second is the link flip and the last one is the inclusion moves. 1 1 1 means all the moves are active.
Initial_Step: Initial step of the simulation, usually it should be one, unless something else is intended.
Final_Step: The final step of the simulations
Display_periodic: Frequency of coordinate file in vtu file format.
OutPutEnergy_periodic: Frequency of energy file. Energy file contains several other information depending on global constrains applied on the system.
Restart_periodic: Frequency of restart file recording.
Kappa: Membrane bending and gaussian rigidity.
Seed: Random number seed.
OutPutTRJ_TSI: Frequency for writing trajectory frames in tsi file, the precision of the coordinates and the name of the folder that the files will be stored in.
GeneralOutputFilename: A general string to label all the generated files of a specific run with that string.
Cell_Size: The size of the unit cells for domain decompositions, should be larger than 1

Fixed frame tension simulations: For doing this, one needs to add below line to the input file. The last two numbers are the frame tension in the unit of [k_BT/a^2] and inverse frequency of applying the box size check algorithm to maintain the fixed membrane tension respectively. Note: this algorithm only makes sense for predoc membranes.

```
Frame_Tension                   = on Position_Rescale 0 5
```

Osmotic pressure or fixed volume simulation.

```
;Volume_Constraint              = on eqtime   DP    K     gamma
Volume_Constraint               = on 100     -0.1   0     0
```

Applying harmonic potential on the surface global curvature.

```
;CouplingtoFixedGlobalCurvature = state K gC0
CouplingtoFixedGlobalCurvature = on 60 0.1
```

Membranes in confined spaces: OPENDTS also allows simulating membranes while they are in confined regions. If the initial membrane is not within the defined region, bthe surface is forced to enter this region by defining an equilibrium time. However, if the eq time is short, it may fail to do so. There are four types of confided space defined in OPENDTS: TwoFlatParallelWall, Cuboid, Ellipsoid and EllipsoidalShell. To apply one of these confinements, one of the below commands must be added to the input file.

```
CoupleToRigidWalls              = on TwoFlatParallelWall  10000  2
```

```
CoupleToRigidWalls              = on Cuboid  10000  10 10 10
```

```
CoupleToRigidWalls              = on Ellipsoid  10000  10 10 10
```

```
CoupleToRigidWalls              = on EllipsoidalShell  10000  10 10 10  0.01
```

Harmonic potential between two groups of vertices: To apply a harmonic potential between two groups of vertices, one need to add below command to the input file.

```
; HarmonicPotentialBetweenTwoGroups = on K L0 eqstep  Group1 Group2 nx ny nz
HarmonicPotentialBetweenTwoGroups = on 10 100 2000000 Group1 Group2 0 0 1
```

You also need to provide an index file for the group definition. This file should have *"inx"* extension. An example of index file is as below:

```
Group1 1
24
Group2 3
12 16 33
```

Adding inclusions: To have inclusion in the simulations, we need to define different type of inclusions. Inclusion type must be defined at the end of the input file as

```
INCLUSION
Define 4 Inclusions
SRotation Type   K   KG  KP  KL  C0  C0P C0L
3         Pro1   10  0   0   0   0   0   0
3         Pro2   20  0   0   0   0   1   0
3         Pro3   20  5   0   0   0   0   0
2         Pro4   20  5   0   0   0   0   0
```

The Inclusion-inclusion interactions should be always placed at the end of the file as

```
Inclusion-Inclusion-Int
 1    1    1   2    0.0     0.0
```

```
Type 1  n  A  B
```

$$e_{i,j} = -A_{i,j} + B_{i,j}(1 + \cos[n\Theta])$$

Defining inclusions type inside the input file does not means that they are present in the simulation. To create inclusion in the simulations, we need to define them either inside the "*tsi*" file or tell the input file to generate them. Currently we can only generate random distribution of inclusions from the input file. However, some can manually or use a script to add such inclusions to the tsi file.

```
GenerateInclusions
Selection_Type Random
TypeID     1     2     3
Density    0.0   0     0
```

Topology file format

Topology file provides information about the position of all the vertices and how they are linked. There are two types of file that you can provide as a topology file; *top* and *tsi* file format.

***top* file:** This file has a "top" extension and contains a list of TS files in q (TS file) file format. An advantage of this topology format is that it allows us to feed multiple TS file into the system. A disadvantage is that it does not include inclusions.

***tsi* file**: tsi topology is single file that is also single frames of OPENDTS trajectories. An advantage of this topology file is that it includes the inclusions information.

TS file

Triangulated surface files that can be read by OPENDTS either has "q" format or "tsi" format. The Generate script can generate files in both formats (see Generate script section).

"tsi" format files:
The following shows a part of a .tsi file with all necessary keywords highlighted in bold. Every .tsi file starts with a line calling version 1.1. The next line defines the box size (x, y, and z) of the system in nm. The next three sections describe the TS mesh. Each section starts with a keyword (vertex,

triangle and inclusion) and their corresponding number. Here, we have 130 vertices (the numbering starts from 0). Each vertex has an index and a position in x, y and z (in nm). The 130 vertices are connected via 256 triangles. Again, every triangle has an index (starting from 0) and is defined by the vertices the triangle connects, i.e. triangle 0 connects vertices 11, 55 and 43. Furthermore, a .tsi file can have a (protein) inclusion section. Here, there are three inclusions from two different types. Again, each inclusion has an index. The index is followed by the inclusion type (here: type 1 for inclusions 0 and 1, type 2 for inclusion 2) and the corresponding vertex index. The last two (floating point) numbers describe a unit two-dimensional vector (sum of both numbers must be one!) which defines the orientation of the inclusion with respect to the bilayer normal.

```
version 1.1
box   50.0000000000     50.0000000000     50.0000000000
vertex         130
0     21.1606233083     25.4394806652     25.5960855271
1     27.0284995400     23.2012757654     21.6715285158
2     26.9921761232     25.5136587223     28.0195776981
3     23.3273229896     26.2315165676     28.0075875808
4     26.2722773116     26.3271061222     28.1420707299
5     22.0396876425     23.6080597437     26.8858740866
.
.
.
125   21.5556280860     25.5595098219     26.5363425272
126   23.2182025326     26.8060871266     21.5195141902
127   25.3199303865     24.3519379911     20.6752314764
128   28.0093200458     22.6356946990     23.4685318698
129   21.4000741257     26.5841316766     25.2761757772
triangle       256
0     11     55     43
1     94     75     14
2     64      3     91
3     59     52     40
.
.
.
253   33    109     44
254   53     69     47
255   85      6     74
inclusion        3
0      1     22      0     1
1      1      5      0     1
2      2     30      0     1
```

"q" format files: The previous tsi file in q format can be seen below.
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).

```
50.0000000000     50.0000000000     50.0000000000
130
0     21.1606233083     25.4394806652     25.5960855271
1     27.0284995400     23.2012757654     21.6715285158
2     26.9921761232     25.5136587223     28.0195776981
3     23.3273229896     26.2315165676     28.0075875808
4     26.2722773116     26.3271061222     28.1420707299
5     22.0396876425     23.6080597437     26.8858740866
.
.
.
125   21.5556280860     25.5595098219     26.5363425272
126   23.2182025326     26.8060871266     21.5195141902
127   25.3199303865     24.3519379911     20.6752314764
128   28.0093200458     22.6356946990     23.4685318698
129   21.4000741257     26.5841316766     25.2761757772
256
0     11     55     43
1     94     75     14
2     64      3     91
3     59     52     40
.
```

```
.
.
253   33    109     44
254   53     69     47
255   85      6     74
```

Generate Script

Generate (**GEN**) bindery allows you to create triangulated surface files with different topologies in two different file formats, *"q"* or *"tsi"*. Three options exist, flat bilayer periodic in x and y directions, cylinder periodic in the x-direction and a closed sphere. To see all the options, execute `$path/GEN -h`

Some example:

To generate flat bilayers

```
$path/GEN -box 50 50 30 -type flat -o topol.q
```

Generating closed membranes

```
$PATH/GEN -box 50 50 50 -type tetrahedron -N 20 -o topol.q
```

Note: You can force this structure to become sphere by using Ellipspoidalshell command and a short simulation.

```
CoupleToRigidWalls              = on EllipsoidalShell  10000  10 10 10  0.01
```

Convert Script

The convert (**CNV**) bindery allows for converting "*tsi"* and "*q*" files to each other and several other different file format such as "*vtu"* and "*gro*". To see all the options, execute `$path/GEN -h`

Tutorials

T1: Framed membranes

To perform simulation on a membrane in a PBC box, we have to first generate a flat TS file. This can be done by using **GEN** binary. Using this you can generate a TS file in a q file format using below command line.

```
$path/GEN -box 50 50 100 -type flat -o topol.q
```

This command creates a TS file in *q* file format with a box size of 50*50*100. Next, this file name should be added to a topology file name with top extension (here it is named top.top). And we should write below line in the top file.

```
topol.q 10
```

The number in front of the file gives an id to the entire mesh (the value is not important but need to be unique of multiple files are fed). To run a simulation, we need also an input file to define the simulation parameters (see input file format section).

To make a membrane tensionless, we need to add below command to the input file (such the example in the input file section).

```
Frame_Tension                   = on Position_Rescale 0 5
```

Using these two files we are now able to run DTS simulations as:

```
$PATH/DTS -in input.dts -top top.top -seed 76532
```

In the dts_tutorials/T1-FramedMembrane folder run the *./run.sh* script to perform this tutorial.

T2: Adding inclusions

By adding below section to the end of the input file from the previous tutorial, we can add inclusions to the system (For more information, see the Input file section in the User Manual).

```
INCLUSION
Define 4 Inclusions
SRotation Type   K   KG  KP  KL  C0  C0P C0L
0         Pro1   10  0   0   0   0.4       0   0
0         Pro2   20  0   0   0   -0.4      0   0
GenerateInclusions
Selection_Type Random
TypeID     1     2     3
Density    0.3   0.1     0
Inclusion-Inclusion-Int
1    1    1   2     2     0.0
1    2    1   2     0     0.0
2    2    1   2     2     0.0
```

In the dts_tutorials/*T2-Membranes_Inclusions* folder run the *./run.sh* script to perform this tutorial.

T2: Vesicle simulations

To start a simulation of a vesicle, we first need to create a vesicle structure. For this, we use **GEN** script to generate a tetrahedron and run a short simulation with EllipsoidalShell confinement to shape the tetrahedron into a vesicle. To do this, we first run below command to create a TS file in the shape of tetrahedron.

```
$PATH/GEN -box 50 50 50 -type tetrahedron -N 20 -o topol.q
```

Next we add the generated TS file to the topology file and run a normal simulation with below command added in the input file.

```
CoupleToRigidWalls              = on EllipsoidalShell  10000  10 10 10  0.01
```

After about 100K steps, you should get a spherical TS file. You can convert the last tsi file from the trajectory file to create a new q file and start a simulation of a vesicle with different setups. For example, you can add inclusions, or apply osmotic pressure or membrane curvature.

In the dts_tutorials/*T3-Vesicle* folder run the *./run.sh* script to perform this tutorial.

T3: Pulling a membrane nanotube

To pull a nanotube from a membrane, we need to have a membrane under tension (see previous tutorials for such a simulations). Then we can apply a harmonic potential between two groups, one group can be the single vertex while the other group can be all the other vertices. Using a corresponding index file (See the index file section) and adding below command to the input file we have everything we need to pull a nanotube from a flat membrane.

```
HarmonicPotentialBetweenTwoGroups = on 10 100 2000000 Group1 Group2 0 0 1
```

Next, just run below command and wait for the nanotube to form.

```
$PATH/DTS -in input.dts -top top.top -seed 76532 -ndx index.inx
```

In the dts_tutorials/*T4-tether_pulling* folder run the *./run.sh* script to perform this tutorial.

OpenDTS: Mesoscopic simulation of Biomembranes using Dynamically triangulated surfaces

Weria Pezeshkian[1]* and John H. Ipsen[2]

[1]Niels Bohr International Academy, Niels Bohr Institute, University of Copenhagen.
[2] MEMPHYS/PhyLife, Department of Physics, Chemistry and Pharmacy (FKF), University of Southern Denmark, Campusvej 55, 5230 Odense M, Denmark.

*corresponding author: weria.pezeshkian@nbi.ku.dk

Abstract

We present OpenDTS software to perform computational research on biomembranes at messocpic length-scale. In this mode a membrane is represented by a dynamically triangulated surface equipped with vertex-based inclusions to integrate the effects of integral and peripheral membrane proteins. The model parameters of these inclusions can be calibrated using finer scale simulation techniques e.g., all atom and coarse grain molecular dynamics or through a top down approach through experimental data. Several algorithms are included into the software that allow for simulation of framed membrane with constant tension, vesicles with various fixed volume or constant pressure difference, confined membranes into the fixed region of the space, constant fixed global curvature and application for external forces on regions of the membrane. In addition, the software allows one to turn off the shape evolution of the membrane and only explore inclusions organization. This allows to take realistic membrane shapes obtained from Cryo-ET and obtain heterogeneous organization of biomolecules which can be backmapped to finer simulations models. In addition to many biomembrane exploration. this software brings us a step closer to simulate realistic biomembranes with molecular resolution. Here we show several interesting show cases of the power of the software and the detail information of how to use the software is included in the associated documents.

Introduction

Many biological processes involve large scale changes in lateral chemical organization and geometrical shapes of biological membranes. The modeling of this processes with molecular based simulations is very expensive, not feasible or at least many interesting features would be lost. Therefore, macroscopic modes in which the molecular details are ignored altogether and the membrane is typically represented by a continuous surface and the proteins, as one or few interacting particles, are better suited and a pragmatic choice.

Here, we present OpenDTS for computational investigation of biomembranes at messocpic length-scale. In OpenDTS a membrane is represented by a dynamically triangulated surface equipped with vertex-based inclusions to integrate the effects of integral and peripheral membrane proteins. The model parameters of these inclusions can be calibrated using finer scale simulation techniques e.g., all atom and coarse grain molecular dynamics or through a top down approach through experimental data.

Several algorithms are implemented into OpenDTS to performed complex membrane simulations in conditions mimicking biophysical exploration of complex membranes in wet labs. For instance, it is possible to simulate framed membrane with constant tension and pull a nanotube (membrane tethers). Also, closed membranes, e.g., vesicles, simulations can be performed with various fixed volume, constant pressure difference or/and constant fixed global curvature. Membranes also can be confined into the fixed region of the space to explore the effect of the environment on the membrane shape and fluctuations. In addition, OpenDTS allows one to turn off the shape evolution of the membrane and only explore inclusions organization. This allows to take realistic membrane shapes obtained for instance from cryo-electron tomography and obtain heterogeneous organization of biomolecules which can be backmapped to finer simulations models. This feature with a help of backmapping software e.g., TS2CG, brings us a step closer to simulate realistic biomembranes with molecular resolution. Here we show several interesting show cases of the power of the software and the detail information of how to use the software is included in the associated documents.

Mesoscopic Membranes

At mesoscale the molecular detail and interaction can be ignored all together and the system can be described by an energy function that captures membrane bending and the energy associated with mesoscale lateral organization of its chemical constituents. For the membrane deformation the energy

of the bending can be well described by Helfrich Hamiltonia that is quadratic in the extrinsic curvature of the surface.

$$E = \int \left\{ \frac{\kappa}{2}(2H - C_0)^2 - \kappa_g K \right\} da \quad (1)$$

The validity of this for large scale membrane conformation changes has been tested well even some md shows it is good up to ….

However, with a simple argument this could be obtained…. since area is an extensive variable …

For numerical integration, a continuous membrane is discretized by a dynamical triangulated surface (DTS) containing N_v vertices, NT triangles, NL links which together form an irregular planer triangulated network (Figure 1A).

To mimic this model to a realistic system, a vertex should be seen as a segment of a bilayer containing hundreds of lipids, this means that the resolution of the model is limited to the length-scales above few nanometers.

Using a set of discretized geometrical operations, each vertex is furthermore assigned with a normal vector Nˆu, surface area Au (one third of the area of its neighboring triangles), principal curvatures (c1u, c2u) and principal directions (X1(u), X2(u)) (Ramakrishnan et al., 2010) (Figure 1A). This suffices to construct an elastic energy function associated with membrane bending that allows us to obtain the surface equilibrium configurations using numerical update algorithms.

Length scale definition:
Although this are suggested length scale definition, the code does not care about it and ….

System evolution

The difference between dynamical and static triangulation is that the mutual link between two neighboring triangles can flip (Alexander moves). This allows to sample through all possible triangulations for a given N_v, N_T, N_L. Link flipping and positional updates of the vertices gives the fluid character with full translational invariance in the plane of the surface. To ensure self-avoidance of the surface each vertex is equipped with a spherical bead. In this work, we have employed the Metropolis Monte Carlo algorithm

(Ramakrishnan et al., 2010; Bahrami et al., 2012; van der Wel et al., 2016), but many other updating schemes are possible (Noguchi and Takasu, 2001; Cooke et al., 2005; Noguchi and Gompper, 2006; Peng et al., 2013; Mauer et al., 2018).

About the code

Software

To run the software, you need a triangulated mesh in q or tsi file format.
Restart with the res file format. Is a binary file
Trajectory is in both tsi and bts file format.

Visualization

DTS produces a set of file names as conf(i).vtu that can all be loaded to Paraview for visualization and evaluation of the system. Tsi file format can be visualized using vmd.

Performance

Moves and Step
Time for one accepted step

Results

1) Framed membranes

Flat membranes are very common model of segments of biological membranes as the curvature of typical cellular or model membranes are very large and can be locally considered flat. OPENDTS allows for simulations of flat membranes within a periodic box.

Framed membrane under tension and tensionless

Such membrane shows a specific undulation mode that follows

$$\langle u(q)u(-q)\rangle = \frac{1}{\kappa q^4 + \tau q^2}$$

Figure 1 shows the results for such spectrum obtained from the DTS simulations for two situations in which one is tensionless membrane and the other is under tension. For more details on how to perform such a simulation, see SI-section XX.

Sandwiching tensionless membranes

Confining a tension less membrane between two rigid walls change the fluctuation spectrum. Such effect can also be investigated using OPENDTS.

Figure XX shows

How area and projected area changes as function of wall thickness
How fluctuation spectrum changes

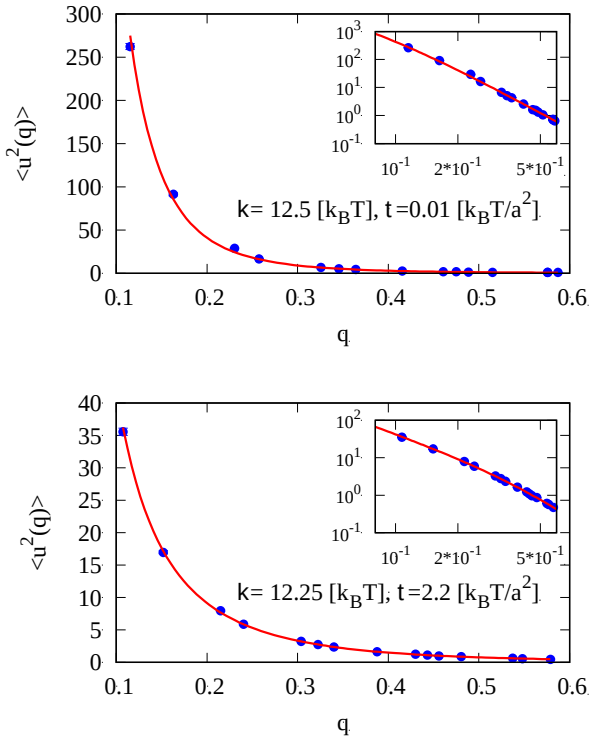

Figure 1: Membrane undulation spectrum obtained from DTS simulation, (A) tension less membrane (B) membrane under tension of …

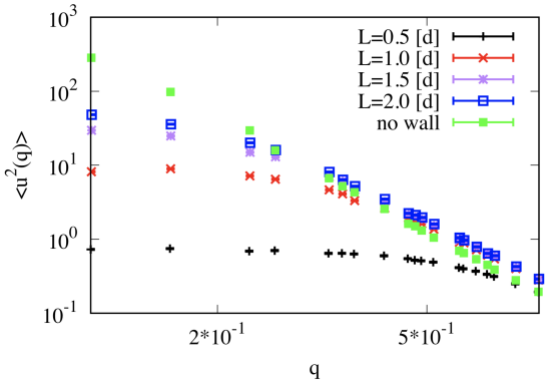

Figure 2: Membrane undulation spectrum obtained from DTS simulation for surfaces confined between to rigid wall.

Proteins, membrane shape deformation

In the current version, proteins are modelling as in-plane inclusions. These inclusions are vertex based, meaning each protein is mapped into one point. Therefore, only limited choice of this proteins is possible.
180 degree-symmetric inclusions can locally bend the membrane differently in different directions. Such inclusions can thus be given an orientation in the plane of the vertex. Since at each point of a smooth surface only two independent direction could exist, these proteins also can induce different curvature in different direction one parallel to their in-plane orientation $C_{||0}$ and one perpendicular to their inplane direction $C_{\perp 0}$.

The membrane curvature in these directions can easily be obtained by Eulers curvature formula

$$C_{||} = c_1 \cos^2\theta + c_2 \sin^2\theta$$

And

$$C_{\perp} = c_1 \sin^2\theta + c_2 \cos^2\theta$$

where θ is the angle between the orientation of the inclusion and the direction of the main principal curvature of the membrane. Such inclusion will give rise to an additional local contribution to the total elastic energy in Equation (1),

$$e = \frac{k_1}{2}\left(C_{||} - C_{||0}\right)^2 + \frac{k_2}{2}\left(C_{\perp} - C_{\perp 0}\right)^2$$

where k_1 and k_2 are the directional bending rigidities imposed by the inclusion on the membrane.

$$e_{i,j} = -A_{i,j} + B_{i,j}(1 + \cos[n\Theta])$$