



Science and  
Technology  
Facilities Council

# Guide to DPD exercises

Michael Seaton  
UKRI STFC Daresbury Laboratory  
[michael.seaton@stfc.ac.uk](mailto:michael.seaton@stfc.ac.uk)

# Mesoscale exercises

## Put together practical exercises on DPD and LBE

- Available in Jupyter notebooks from the workshop folder on GitLab, designed to run in (non-GPU) STFC Cloud Environment:

```
git clone https://gitlab.com/ccp5/workshop.git WORKSHOP
```

- DPD Exercises available in [WORKSHOP/Day\\_9Meso](#)
- LBE Exercises available in [WORKSHOP/Day\\_10Meso](#) and [WORKSHOP/Day\\_11Meso](#)
- Information also available via main CCP5 Summer School exercises page:

[ccp5.gitlab.io/summerschool](https://ccp5.gitlab.io/summerschool)

- Points to notebooks for each exercise
- Additional background information on exercises, theory and brief C programming guide (needed for LBE exercises)
- Download link to these slides on the DPD exercises
- Possible to run notebooks on your own laptops: see CCP5 Summer School exercises page for details (including required Python modules)

# Mesoscale exercises

The screenshot shows a web browser window displaying the CCP5 Summer School 4.0.0 documentation. The page title is "CCP5 Summer School: Methods in Molecular Simulation". The left sidebar contains a table of contents with the following items:

- CCP5 Summer School 4.0.0 documentation
- Introduction to Work Environment
- Day 1-2: Introduction to Modern Fortran
- Day 1-2: Introduction to Python
- Day 3: Statistical Mechanics Problems
- Day 4 Monte Carlo integration
- Day 4 Monte Carlo at Constant Pressure
- Day 5 Introduction to Molecular Dynamics
- Day 5 Phase equilibria
- Day 6 Stability and Accuracy of Molecular Dynamics
- Day 6 Force-fields, potentials and optimisation methods
- Day 6 Constraints: Multiple timesteps and RATTLE
- Day 8 MD Ensembles: NVE and NVT
- Day 9: Setup environment for advanced courses
- Day 9-11 Mesoscale: Introduction
- Day 9 Mesoscale: DPD Tutorial Exercise 1: Simple DPD systems
- Day 9 Mesoscale: DPD Tutorial Exercise 2: Applying DPD to molecular systems
- Day 10 Mesoscale: LBE Tutorial Exercise 1: Single component LBE simulations
- Day 11 Mesoscale: LBE Tutorial Exercise 2: Multi-Component LBE simulations
- Day 9-11: First principles: CRYSTAL Tutorials
- Day 9-11: First Principles: Castep Simulation
- Day 9-11: First Principles: Phonons and Spectroscopy Tutorial
- Day 10: Band Structure and DOS Calculation of ZnS

The main content area shows a "Contents:" section with a list of links. A blue box highlights the following items:

- Day 9: Setup environment for advanced courses
- Day 9-11 Mesoscale: Introduction
- Day 9 Mesoscale: DPD Tutorial Exercise 1: Simple DPD systems
- Day 9 Mesoscale: DPD Tutorial Exercise 2: Applying DPD to molecular systems
- Day 10 Mesoscale: LBE Tutorial Exercise 1: Single component LBE simulations
- Day 11 Mesoscale: LBE Tutorial Exercise 2: Multi-Component LBE simulations

Another blue box highlights the following items:

- Appendix B: The Boltzmann Equation - an outline derivation
- Appendix C: Outline Notes on Fluid Mechanics

Text annotations on the right side of the screenshot:

- "Pages on setting up connection to SCARF, DPD and LBE Tutorials" (pointing to the highlighted items in the first blue box)
- "Additional theoretical information on LBE" (pointing to the highlighted items in the second blue box)

# DPD Jupyter notebooks

training.jupyter.stfc.ac.uk

File Edit View Run Kernel Tabs Settings Help

Filter files by name

Name	Last Modified
d_l_meso	last year
DPD1Ex2	2 days ago
DPD2Ex1	2 days ago
DPD2Ex2	2 days ago
images	2 days ago
Day9DPDTutorial1Ex1.ipynb	12 seconds ago
Day9DPDTutorial1Ex2.ipynb	2 days ago
Day9DPDTutorial2Ex1.ipynb	22 hours ago
Day9DPDTutorial2Ex2.ipynb	2 days ago

dlmhistoryread.py 2 days ago  
dlmoutputread.py 2 days ago  
dpd\_solution.F90 2 days ago  
dpd.F90 2 days ago  
flory\_huggins.py yesterday  
floryhuggins-rho-3.000.dat 2 days ago  
history\_dlm\_to\_vtf.py 2 days ago  
INPUT 2 days ago  
launchdlmeso.py 2 days ago  
plotcorrel.py 2 days ago  
plotfloryhuggins.py 2 days ago  
plotisovolume.py 2 days ago  
plotstats.py 2 days ago

Folder with codes, Python scripts, DL\_MESO etc.

## DPD Tutorial Exercise 1: Simple DPD systems

### Ex1. DPD fundamentals and code hacking

Take a look at [dpd.F90](#). You should be able to make out various sections and subroutines:

- reading an input file (to come)
- allocations of arrays for positions, velocities and forces of particles
- initial simulation setup (allocate particles and assign velocities)

## Notebooks for DPD exercises: launchable in (non-GPU) cloud environment

- calculating forces on pairs of particles within the cutoff distance
- applying stage 2 of Velocity Verlet
- writing to output files and to the screen at specified intervals
- calculations and printing of final properties
- closing of files and the program itself

Try the following command to compile the code:

```
!gfortran -o DPD dpd.F90
```

You should find that the code does not compile and throws up error messages for lines 603 and 609: *this is expected behaviour*, so please do not panic! These two lines are the ones meant to calculate the dissipative and random forces for a given pair of particles, but at the moment they are incomplete. Your main task for this exercise is to complete these lines and try out the code for a single-component DPD simulation.

Some hints on completing those lines:

- The dissipative force parameter  $\gamma$  is available as `gamma` in the code.
- The variable `sigma` is equal to  $\sqrt{\frac{2\gamma k_B T}{\Delta t}} = \frac{\sigma}{\sqrt{\Delta t}}$ , i.e. the random force parameter divided by the square root of the timestep size. (This reduces the number of divisions the code has to make.)
- The *simplest option* for the screening function for random forces  $w^R$  would be the equivalent of that for conservative forces, given as `wrr`, and a Gaussian random number  $\zeta$  is already calculated for you, available as `zeta`.
- The screening function for dissipative forces *must* equal  $(w^R)^2$ , and the dot product of the vector between the particles and their relative velocity,  $\mathbf{r}_{ij} \cdot \mathbf{v}_{ij}$ , is available as `rdv`.
- The forces `fdr` and `frr` need to be divided by `rrr` (`rrr`) as these will then be multiplied by the actual vector between the particles: see lines 545-550. Note that you may have to divide `fdr` by `rrr` twice, and `rij2` is available as `rsq`.

Once you are done, try compiling the code again:

Simple 0 3 Python 3 (ipykernel) | Idle Mode: Command Ln 1, Col 1 Day9DPDTutorial1Ex1.ipynb 1

# DPD practical exercises

## Designed to show DPD's capabilities as a mesoscale modelling method

- Two tutorials, each with two exercises (a notebook for each exercise):
  1. Simple DPD systems
    1. DPD fundamentals and code hacking
    2. Parameterising DPD
  2. Applying DPD to molecular systems
    1. DPD mesophases
    2. Lipid bilayers, micelles and vesicles
- Tutorial 1, Exercise 1 uses a simple custom-written code you will need to 'hack' (modify)
- All other exercises use DL\_MESO's DPD code (DL\_MESO\_DPD)
- Python scripts supplied to launch DL\_MESO\_DPD, convert outputs for visualisation and plot results

# Tutorial 1, Exercise 1

## Constructing and running a basic DPD code

- Notebook: [Day\\_9Meso/Day9DPDTutorial1Ex1.ipynb](#)
- Starts with a simple DPD code in Fortran90: `dpd.F90`
  - Includes a compile-time option to modify how bead pairs are found for force calculations (see later)
- Models a simple fluid – one bead species, no bonds between beads – with a DPD thermostat
- However ... the lines calculating dissipative and random forces are incomplete, so the code will not compile
- Your task: complete the lines to calculate dissipative and random forces to get the DPD thermostat to work
  - Two lines in a subroutine (`calculate_dpd_forces`) that takes  $r_{ij}$ ,  $r_{ij}^2$  and  $\mathbf{r}_{ij} \cdot \mathbf{v}_{ij}$  as inputs
- Code can be used to explore the properties of a single species

# Tutorial 1, Exercise 1

## calculate\_dpd\_forces subroutine (lines 587–611 of dpd.F90)

```
IF (rrr>1.0e-10_dp) THEN          ←———— Avoids division-by-zero problems

!   screening function (for conservative force)

      wrr = MAX ((bij - rrr), 0.0_dp) ←———— Screening function for conservative forces

!   potential calculation

      pot = 0.5_dp * aij * wrr * wrr

!   conservative interaction force

      fcr = aij * wrr / rrr        ←———— Magnitude of conservative force: note division by rrr ( $r_{ij}$ )

!   dissipative force (rdv = dot product of vector and relative velocity) ←———— Variable rrv input into
      fdr =                        ←———— Magnitude of dissipative force (missing)                               subroutine

!   random force (zeta = gaussian random number)

      CALL RANDOM_NUMBER (zeta)
      zeta = fac12 * (zeta - 0.5_dp) ←———— Approximation of Gaussian random number  $\xi_{ij}$ 
      frr =                          ←———— Magnitude of random force (missing)

END IF
```

# Tutorial 1, Exercise 1

## Hints to complete dissipative and random forces

- Variables available for force parameters:

- gamma**  $\gamma$

- sigma** Equal to  $\sqrt{\frac{2\gamma k_B T}{\Delta t}} = \frac{\sigma}{\sqrt{\Delta t}}$

- Screening function for random forces  $w^R$ : we suggest using the same as for conservative forces ( $w^R$ ), but you can choose something else

- Screening function for dissipative forces:

$$w^D(r_{ij}) = [w^R(r_{ij})]^2$$

- The forces coming out of the `calculate_dpd_forces` subroutine should be divided by  $r_{ij}$ : multiplying these by the actual vector between particles  $\mathbf{r}_{ij}$  (e.g. lines 545–550) gives

$$\frac{F_{ij}}{r_{ij}} \times \mathbf{r}_{ij} \equiv F_{ij} \hat{\mathbf{r}}_{ij} = \mathbf{F}_{ij}$$



# Tutorial 1, Exercise 1

## Compiling code

```
gfortran -o DPD dpd.F90
```

## Running code

```
./DPD < INPUT
```

- **INPUT** is a text file with the simulation properties

```
10000      number of time steps
1000       number of equilibration steps
10         sampling interval for properties
900        particle population
300.0      system volume (reduced units)
1.0        temperature (reduced units)
1.0        energy parameters
1.0        particle diameters
4.5        drag coefficient
0.04       time step (LJ units)
.true.     trajectory file option
.true.     write positions (T) or velocities (F)
```

- Commands can be invoked using cells in notebook
- Script available to plot results (e.g. energy, temperature, pressure), VMD can open simulation trajectory for visualisation and some analysis

# Tutorial 1, Exercise 1

## Speeding up the code

- By default, code uses 'brute force' approach to search over all possible particle pairs, check distances and calculate forces if within cutoff
- Code also includes implementation of linked-cell lists, assigning particles to lists for each cell (size of at least the cutoff) and uses the lists to search for particle pairs within each cell and its nearest neighbours
- To invoke linked-cell list version, recompile using additional compiler flag:

```
gfortran -o DPD-FAST -DFAST dpd.F90
```

- Calculation times are measured in both versions of the code so you can compare them
- How does the calculation time vary with the number of beads for each version?

# DL\_MESO

## General purpose mesoscopic modelling software package

- Includes Lattice Boltzmann (LBE) and DPD codes (serial and parallel with MPI and/or OpenMP), Java GUI (optional)
- Created for CCP5 at UKRI STFC Daresbury Laboratory in 2004
  - Now developed for UKCOMES (EPSRC High-End Computing consortium)
- Free for academic use (licence on annual subscription for commercial use)
- Described in Molecular Simulation articles:
  - Seaton *et al.*, *Mol Sim* **39** (10), 796–821 (2013)
  - Seaton, *Mol Sim* **47** (2–3), 228–247 (2021)
- Copy of current version (2.7) included with course materials, available for you to take home with you
  - Small additions included for exercises (makefiles in working directory, additional Python scripts)

# Tutorial 1, Exercise 2

## Now expanding to two immiscible species

- Conservative force parameter between species higher than those for like-like interactions: particles should separate out

## Making use of DPD code in DL\_MESO (DL\_MESO\_DPD)

- Notebook: [Day\\_9Meso/Day9DPDTutorial1Ex2.ipynb](#)
- Unpack and compile serial version and utilities (using makefiles)
- Run DL\_MESO\_DPD in directory ([Day\\_9Meso/DPD1Ex2](#)) with input files (CONTROL, FIELD)
  - Script available with notebook to launch calculation in background and display progress bar while DL\_MESO\_DPD runs

# Tutorial 1, Exercise 2

## Input files for DL\_MESO\_DPD

```
DL_MESO phase separation example
```

```
volume 1000.0  
temperature 1.0  
cutoff 1.25
```

```
timestep 0.01  
steps 20000  
equilibration steps 0  
scale temperature every 10  
trajectory 0 100  
stats every 100  
stack size 100  
print every 10  
job time 3600.0  
close time 200.0
```

```
ensemble nvt mdvv
```

```
l_scr
```

```
finish
```

**CONTROL**

```
DL_MESO phase separation example
```

```
SPECIES 2  
A 1.0 0.0 1500  
B 1.0 0.0 1500
```

Numbers of unbonded beads

```
INTERACTIONS 3  
A A dpd 25.0 1.0 1.0  
A B dpd 100.0 1.0 1.0  
B B dpd 25.0 1.0 1.0
```

Interaction lengths  $r_{c,ij}$

Dissipative force parameters  $\gamma_{ij}$

```
CLOSE
```

Conservative force parameters  $A_{ij}$

**FIELD**

# Tutorial 1, Exercise 2

## Output files

- **OUTPUT**            General output with simulation summaries, error/warning messages and statistical properties (instantaneous and averaged) during simulation
- **export**            Simulation restart file (instantaneous particle configuration)
- **REVIVE**            Simulation restart file (statistical accumulators and random number states)
- **HISTORY**          Simulation trajectory data (in binary)
- **CORREL**          Tabulated statistical properties in plottable file
- Plotting trajectory data requires utility to convert into formats for visualisation software
  - Utilities available with DL\_MESO to do this and carry out analyses
  - Also have a Python script with notebook to create a VTF file for visualisation with VMD
- Statistical properties can be plotted using e.g. Excel, gnuplot or Python script (`dmlresultviewer.py`)
  - Python script for plotting also available in notebook

# Tutorial 1, Exercise 2

## Effect of conservative force parameter on separation

- Flory-Huggins relationship

$$\chi^{AB} \propto (A_{ij}^{AB} - A_{ij}^{AA})$$

- Assumes beads for both components have same sizes and like-like interactions
- Proportionality constant changes with particle density  $\rho$
- Can measure  $\chi^{AB}$  from simulations of separating particles, using volume fraction of one species in phase-separated regions (e.g.  $\phi_A$ ):

$$\chi^{AB} = \frac{\ln \left[ \frac{1 - \phi_A}{\phi_A} \right]}{1 - 2\phi_A}$$

- Simulations set up by putting beads of each species in one half of an elongated box: can set this up by supplying DL\_MESO\_DPD with a **CONFIG** file specifying initial positions of beads
- Vary  $A_{ij}^{AB}$ , calculate resulting  $\chi^{AB}$  values and plot to get relationship

# Tutorial 1, Exercise 2

## Python script to automate calculations

- `flory_huggins.py`: Runs series of DL\_MESO\_DPD calculations to find volume fraction profiles and estimates  $\chi^{AB}$  for various  $A_{ij}^{AB}$  values (creates CONTROL, FIELD and CONFIG files automatically)
  - Runs in notebook, displays progress bars for calculations/analyses
  - Suggest recompiling DL\_MESO\_DPD to use OpenMP multithreading to speed up calculations
- Script supplied with notebook to read file produced by `flory_huggins.py` and plot results
  - Concentration profiles:  $\phi_A$  vs.  $x$
  - Plot of  $\chi^{AB}$  vs.  $\Delta A_{ij} = A_{ij}^{AB} - A_{ij}^{AA}$
- Questions:
  - Does value of  $A_{ij}^{AA}$  affect above relationship and the graph?
  - (Optional) What happens if the total bead density  $\rho$  is changed?



# Tutorial 2, Exercise 1

## Applying DPD to molecular systems: adding bonds to form amphiphilic dimers

- Notebook: [Day\\_9Meso/Day9DPDTutorial2Ex1.ipynb](#)
- Including harmonic bonds between two beads that interact differently with a solvent – one hydrophilic, one hydrophobic – to form two-bead molecules
- Solutions of these molecules form mesophases (stable structures at equilibrium)
  - Functions of temperature and concentration of solute (dimers)
- Simulation and analysis workflow:
  - Run DL\_MESO\_DPD with different **FIELD** files (but same **CONTROL** file each time), each representing a different dimer concentration
    - Can run more than one calculation at a time
  - Look at trajectories and create isosurface volume plots of one species in molecule (hydrophobic beads) to visualise phases
  - Calculate order parameters to help detect different mesophases (semi-)automatically (i.e. without needing to visualise them)

# Tutorial 2, Exercise 1

```
DL_MESO amphiphile mesophase example
```

```
volume 10.0 10.0 20.0
temperature 1.0
cutoff 1.0
global bonds

timestep 0.02
steps 50000
equilibration steps 10000
scale temperature every 1000
trajectory 10000 1000
stats every 1000
stack size 100
print every 1000
job time 3600.0
close time 10.0

ensemble nvt mdvv

l_scr

finish
```

**CONTROL**

```
DL_MESO amphiphile mesophase example (30% composition)
```

```
SPECIES 3
A 1.0 0.0 0
B 1.0 0.0 0
S 1.0 0.0 8400

MOLECULES 1
AB
nummols 1800
beads 2
A 0.0107636 -0.0112540 -0.249515
B -0.0107636 0.0112540 0.249515
bonds 1
harm 1 2 100.0 0.50
finish

INTERACTIONS 6
A A dpd 25.0 1.0 5.625
A B dpd 30.0 1.0 5.625
A S dpd 0.0 1.0 5.625
B B dpd 25.0 1.0 5.625
B S dpd 50.0 1.0 5.625
S S dpd 25.0 1.0 5.625

CLOSE
```

NOTE: beads involved in molecules not included in totals here

name of molecule type

number of molecules in system

configuration for molecule insertion

bond stretching interaction

**FIELD**

(30 vol% concentration)

# Tutorial 2, Exercise 1

## Order parameters for automatic mesophase detection

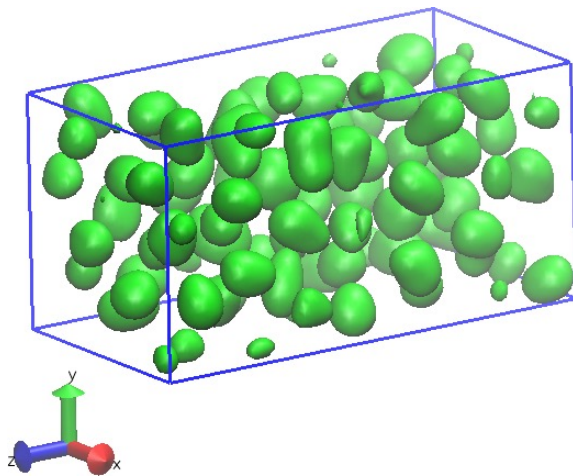
- Assign function (often a Gaussian) for selected particles based on their positions onto a density grid
- Density grid can be used to find isosurfaces and determine normals
- Can calculate second moment of isosurface normal distribution

$$\mathbf{M} = \int \mathbf{n} n p(\mathbf{n}) d\mathbf{n}$$

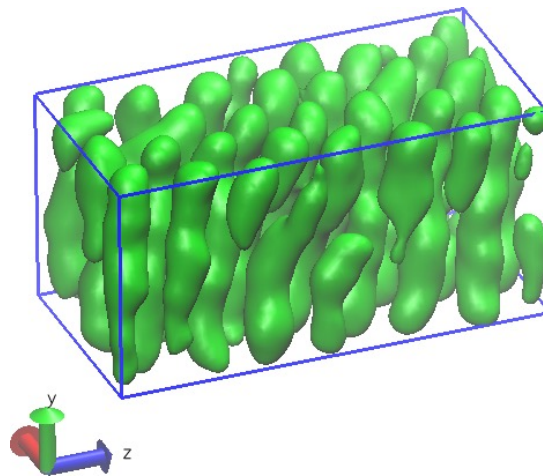
- Eigenvalues of second moment ( $\mu_1, \mu_2, \mu_3$ ) in numerical order normalised to sum to 1 can be used to determine mesophase shape
- All of the above can be determined by `isosurfaces.exe` utility
  - Directly reads `HISTORY` file, uses one bead species selected by user for density plots written to VTK files (readable by ParaView)
  - Eigenvalues are printed on screen and in text file for each trajectory frame
  - Python scripts in notebook can launch utility and plot eigenvalues

# Tutorial 2, Exercise 1

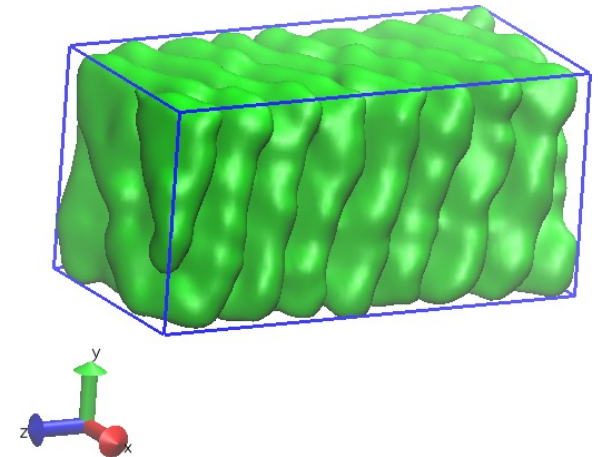
## Probable mesophases



Isotropic ( $L_1$ )  
 $\mu_1 \approx \mu_2 \approx \mu_3 \approx \frac{1}{3}$



Hexagonal ( $H_1$ )  
 $\mu_1 < 0.1, \mu_1 \ll \mu_2, \mu_3$



Lamellar ( $L_a$ )  
 $\mu_1 < 0.1, \mu_2 < 0.15,$   
 $\mu_1, \mu_2 \ll \mu_3$

Eigenvalues shown here are for guidance based on experience  
in carrying out simulations: they are not definitive!

# Running DL\_MESO\_DPD on SCARF

## DL\_MESO\_DPD available on SCARF for Tutorial 2, Exercise 2

- Parallel version of DL\_MESO\_DPD has been compiled on SCARF, available for practical exercises inside a shared folder
- Logging on to SCARF:

```
ssh scarf
```

- Submitting batch job to queue using Slurm:

```
sbatch dpdjob
```

- Job script runs `traject_vtf.exe` utility immediately after DL\_MESO\_DPD finishes to produce VTF file from `HISTORY` file
  - VTF file can be transferred to your own machine for visualisation in VMD
  - Folders available in environment to copy results from each calculation

# Running DL\_MESO\_DPD on SCARF

## Job submission script

- Runs DL\_MESO\_DPD on 32 cores for up to an hour and creates VTF trajectory file for visualisation
- Runs DPD code in directory where job is submitted
- Creates standard output and error files with filenames based on job number
- Uses reservation code for CCP5 Summer School and exclusive running of job on allocated cores
- Uses precompiled executables for DL\_MESO\_DPD and `traject_vtf.exe` utility in shared folder

```
#!/usr/bin/env bash
#SBATCH --nodes=1
#SBATCH --ntasks-per-node=32
#SBATCH -t 01:00:00
#SBATCH -o %j.out
#SBATCH -e %j.err
#SBATCH --exclusive
#SBATCH --reservation=ccp5_pre
#SBATCH -C scarf21
cd $PWD
CWD=/work4/training/ccp5/meso
mpirun -srun $CWD/dpd.exe
$CWD/traject_vtf.exe
```

**dpdjob**

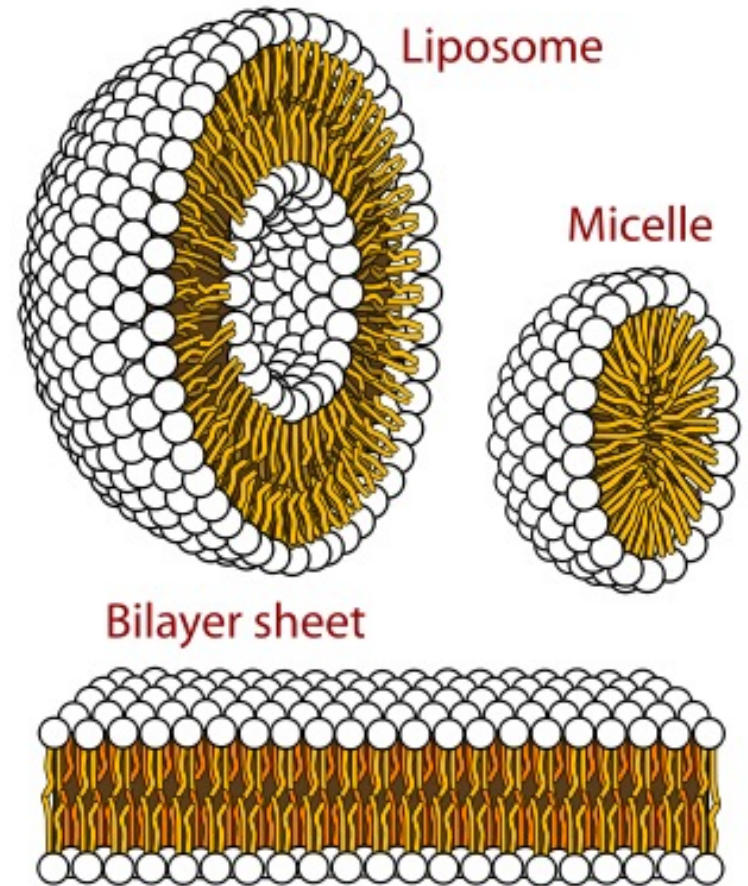
# Tutorial 2, Exercise 2

## Lipid molecules in solution

- Notebook:  
[Day\\_9Meso/Day9DPDTutorial2Ex2.ipynb](#)
- Each molecule consists of one hydrophilic bead and six hydrophobic beads bonded together
- We can optionally include additional potentials to fix angles between pairs of bonds:

$$U_{ijk} = A[1 + \cos(m\theta_{ijk} - \theta_0)]$$

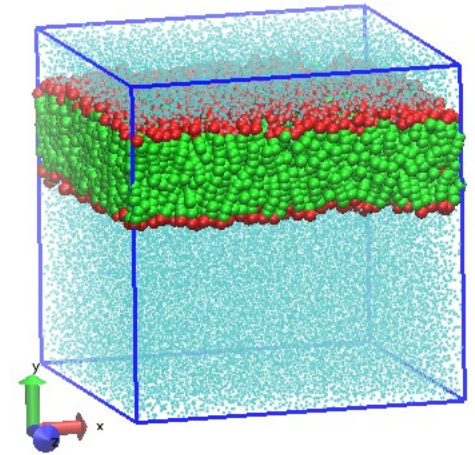
- Depending on molecule concentration and bond straightness, molecules can form into bilayers, micelles or vesicles (liposomes)



# Tutorial 2, Exercise 2

## Lipid molecules in solution

- Starting with DL\_MESO\_DPD simulation that **should** generate a bilayer sheet
  - **FIELD** file sets number of lipid molecules and interactions (including bond angles)
- Start by modifying FIELD file to switch off bond angle potentials, e.g. setting  $A = 0$ 
  - This will still enable angles to be measured and recorded in **CORREL** file
- Try a lower molecule concentration (with and without bond angles)
  - What structures will result?





# Tutorial 2, Exercise 2

```
[...]  
SPECIES 3  
H      1.0 0.0 0  
C      1.0 0.0 0  
W      1.0 0.0 35152  
  
MOLECULES 1  
HC6  
nummols 2000  
beads 7  
H      -0.0172847 0.383451 0.2849  
C      -0.016165 0.381111 -0.215093  
C      0.280007 0.0703806 0.0412832  
C      0.440339 -0.359403 -0.157664  
C      0.00469362 -0.136529 -0.260333  
C      -0.455453 -0.119133 -0.0654933  
C      -0.236136 -0.219878 0.3724  
  
bonds 6  
harm 1 2 128.0 0.50  
harm 2 3 128.0 0.50  
harm 3 4 128.0 0.50  
harm 4 5 128.0 0.50  
harm 5 6 128.0 0.50  
harm 6 7 128.0 0.50  
  
angles 5  
cos 1 2 3 20.0 0.0 1.0  
cos 2 3 4 20.0 0.0 1.0  
cos 3 4 5 20.0 0.0 1.0  
cos 4 5 6 20.0 0.0 1.0  
cos 5 6 7 20.0 0.0 1.0  
finish  
[...]
```

adjust for molecule concentration  
(keep total number of beads constant)

adjust these values for bond angle potentials ( $A$ ,  $\theta_0$ ,  $m$ )

# Before you finish ...

## Registering for DL\_MESO

- If you want to use DL\_MESO for your own research (either DPD or LBE), please register via the website

[www.ccp5.ac.uk/DL\\_MESO](http://www.ccp5.ac.uk/DL_MESO)

## DL\_Software Digital Guide

- If you want more information about DL\_MESO, the mesoscale modelling methods it includes and/or other DL\_Software packages, we have an online guide available at

<https://dl-sdg.github.io/>