Session 2 – Van der Waals Interactions

Important Stuff

Linux Cheat Sheet:

‘cd’ – Change directory	‘ls’ – List files & directories
‘./’ – Current directory	‘../’ – Parent directory

Programs you’ll need:

Mousepad – Text Editor	PyMOL – Molecular Visualiser
Terminator – Command Line	

General Things to Do:

Avoid spaces in filenames (spaces are weird for computers)
Please ask us for help 😊
Please laugh at how funny biological molecules are!

Goal – To build a dynein & microtubule system in which the dynein interacts with the microtubule through steric and Lennard-Jones interactions

Working Directory - /home/ffea/Workshop/2-VdWInteractions

Task 1 – Make the Dynein VdW File

We need to firstly make the dynein microtubule binding domain (MTBD) attractive. How can we do this? We need to select a series of faces and edit their associated types, and create a file defining their interaction. So…

1. Use mousepad and open **“DyneinMonomer.ffea”** and have a quick look at the contents
2. Use PyMOL and open **“DyneinMonomer.ffea”** with the following settings:
    - <Show Solid> = VdW
    - <Show Mesh> = Surface Mesh
    - <Add Atoms> = Onto Faces
3. Select faces at end of stalk that best represent the MTBD (see Fig 1,2 & Box 1)
4. In PyMOL FFEA Loader ‘Editor’ tab:
    - Select <VdW Type 1> = 0
    - Click <Update VdW File>
5. Save the file as “**structure/dynein_MTBD.vdw**”

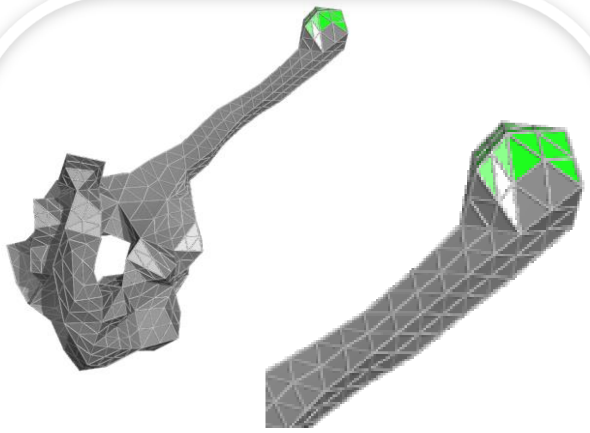


Fig 1:
The Dynein Microtubule Binding Domain

Now we have the file, we should check that it's correct by visualising the result in PyMOL.

6. Use mousepad and open **"DyneinMonomer.ffea"** for editing
7. In the <system> block, replace the dynein .vdw file (first <blob> block) with the one you just made!

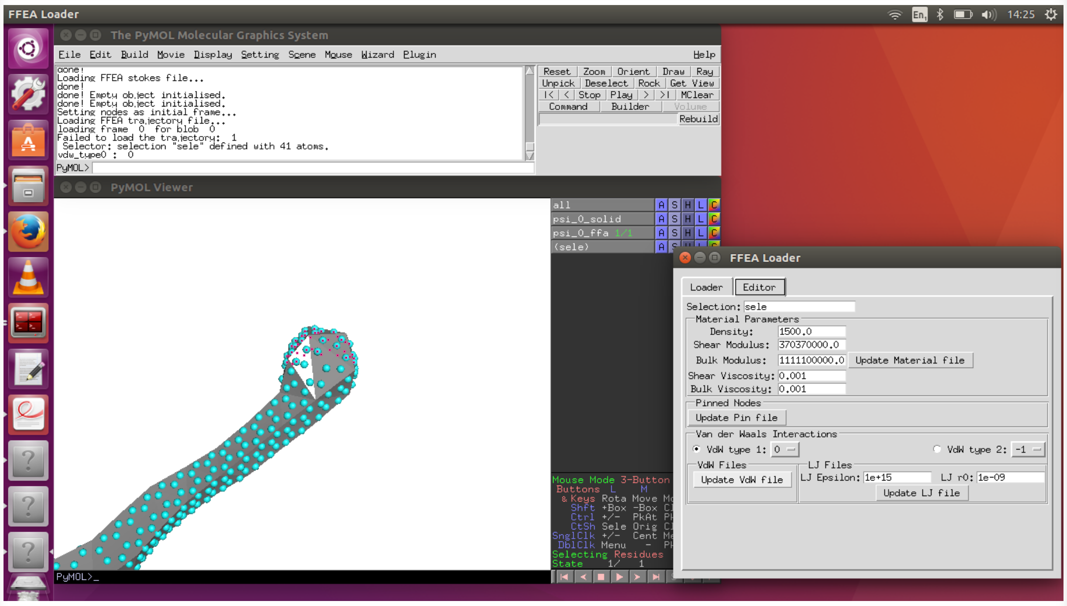
   <vdw = structure/dynein_MTBD.vdw>

8. Save the file as "**DyneinMonomerVdW.ffea**"

9. In PyMOL, remove the currently loaded system (see Box 2)
10. Use PyMOL and open **"DyneinMonomerVdW.ffea"** with the following settings:

    - <Show Solid> = VdW

Does it look correct? See Fig 1 for an example



```
done!
Loading FFEA stokes file...
done!
done! Empty object initialised.
done! Empty object initialised.
Setting nodes as initial frame...
Loading FFEA trajectory file...
loading frame  0  for blob  0
Failed to load the trajectory:  1
 Selector: selection "sele" defined with 41 atoms.
vdw_type0 :  0
PyMOL>
```

```
Selection: sele
Material Parameters
    Density:         1500.0
  Shear Modulus:     370370000.0
  Bulk Modulus:      1111100000.0
Shear Viscosity:     0.001
 Bulk Viscosity:     0.001
Pinned Nodes
Van der Waals Interactions
(•) VdW type 1: 0        ( ) VdW type 2: -1
VdW Files    LJ Files
             LJ Epsilon: 1e+15    LJ r0: 1e-09
```

Fig 2:
Selecting Faces of the MTBD

Box 1: Selecting Faces

To select faces in PyMOL, simply:

- Left-click each face one by one
- Hold shift and left-click to select by area

Note that area select will **select all nodes** within the rectangle, **even ones 'behind' the molecule** that you can't see! Please watch out for this 😉

Box 2: Clearing PyMOL

To clear a system from PyMOL:

- Click File->Reinitialize->Everything

  OR

- In the right-hand bar, that contains all your loaded objects, click all [A]->delete everything and close FFEA Loader

Task 2 – Make the Microtubule VdW File

We also need to make the microtubule track attractive. We can define multiple types of attraction using the .lj matrix, so let's define a steric microtubule track with a binding site on each tubulin molecule! I've already made a default file that makes the microtubule sterically interact with other molecules, so now we need to make those specific binding sites.

1. Use mousepad and open **"DyneinMicrotubule.ffea"** and have a look at the contents (Box 3)
2. Use PyMOL and open **"DyneinMicrotubule.ffea"** with the following settings:

    - <Show Solid> = VdW
    - <Show Mesh> = Surface Mesh
    - <Add Atoms> = Onto Faces

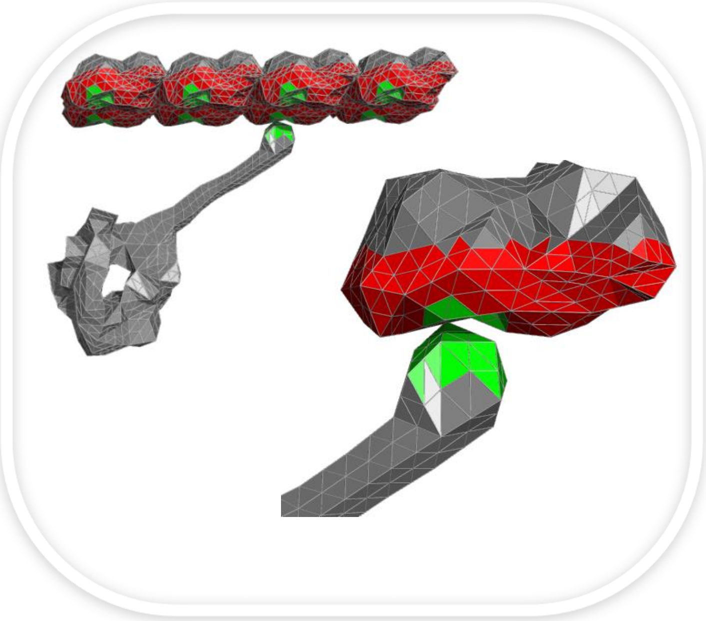


Fig 3:
The Microtubule Binding Site

Now, each tubulin molecule we see uses the *same structure files!* In other words, in the .ffea file, you'll see that each tubulin blob points to exactly the same files, and each one differs only in their position and orientation. Therefore, we only need to edit one tubulin molecule to edit all of them! (see Fig 3 & 4)

3. To define the binding site, select some faces on a single tubulin molecule near to the MTBD of the dynein molecule (see Fig 3, 4 & Box 4 for a hint or two)

4. In PyMOL FFEA Loader 'Editor' tab, select <VdW Type 1> = 0
5. In PyMOL FFEA Loader 'Editor' tab, click <Update VdW File>
6. Save the file as "**structure/tubulin_trackMTBD.vdw**"

Now we have the file, we should check that it's correct.

7. Use mousepad and open **"DyneinMicrotubule.ffea"**
8. In the <system> block, replace all tubulin .vdw files (all <blob> blocks except the first) with the one you just made! ('ctrl+r' in mousepad gives you a find/replace functionality)

    <vdw = structure/tubulin_trackMTBD.vdw>

9. Save the file as "**DyneinMicrotubuleVdW.ffea**"
10. In PyMOL, click File->Reininitialize->Everything to remove the currently loaded system
11. Use PyMOL and open **"DyneinMicrotubuleVdW.ffea"** with the following settings:

    - <Show Solid> = VdW

Box 3: DYNAMIC & STATIC Blobs

Perhaps you noticed the <motion_state> parameter in the <blob> blocks?

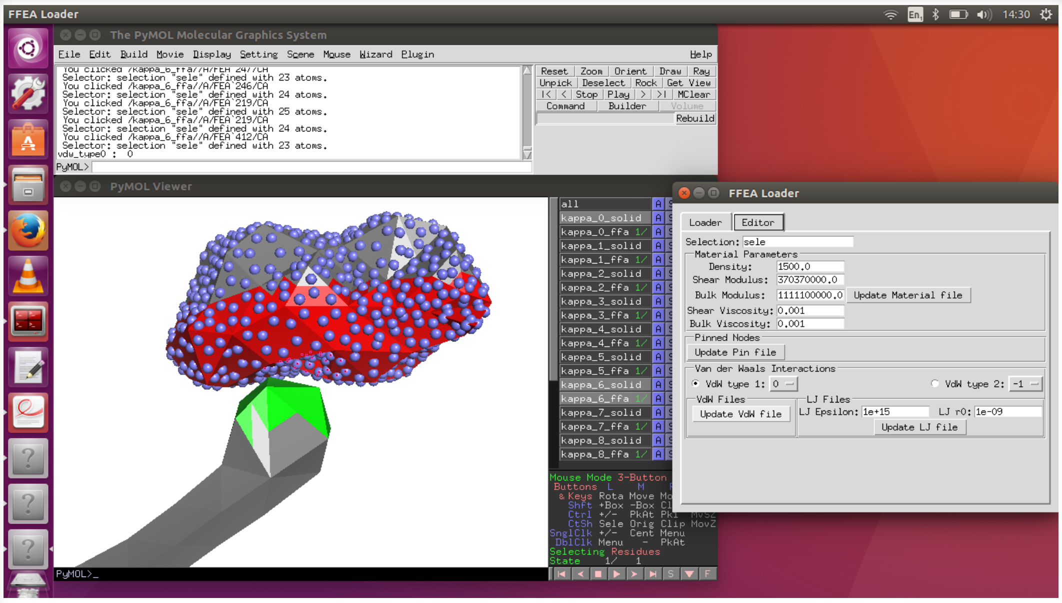
- DYNAMIC – Blob will undergo stochastic dynamics
- STATIC – Blob will not undergo stochastic dynamics, but will still interact through external potentials!

Box 4: Hiding PyMOL Objects

To show/hide loaded objects in PyMOL:

- In the right-hand bar, click any object and it will disappear / reappear
- If you click 'all', then all objects will disappear!

So, if you make all objects disappear and then load only the <systemname>_0_solid, and all <systemname>_6 objects, you may find defining a binding site easier…


FFEA Loader
14:30
The PyMOL Molecular Graphics System
File Edit Build Movie Display Setting Scene Mouse Wizard Plugin Help
Reset Zoom Orient Draw Ray
Unpick Deselect Rock Get View
|< < Stop Play > >| MClear
Command Builder Volume
Rebuild
PyMOL Viewer
FFEA Loader
Loader Editor
Update Material file
Update Pin file
Update VdW file
Update LJ file

```
 You clicked /kappa_6_ffa//A/FEA`247/CA
 Selector: selection "sele" defined with 23 atoms.
 You clicked /kappa_6_ffa//A/FEA`246/CA
 Selector: selection "sele" defined with 24 atoms.
 You clicked /kappa_6_ffa//A/FEA`219/CA
 Selector: selection "sele" defined with 25 atoms.
 You clicked /kappa_6_ffa//A/FEA`219/CA
 Selector: selection "sele" defined with 24 atoms.
 You clicked /kappa_6_ffa//A/FEA`412/CA
 Selector: selection "sele" defined with 23 atoms.
vdw_type0 :  0
PyMOL>
```

```
all
kappa_0_solid
kappa_0_ffa 1/
kappa_1_solid
kappa_1_ffa 1/
kappa_2_solid
kappa_2_ffa 1/
kappa_3_solid
kappa_3_ffa 1/
kappa_4_solid
kappa_4_ffa 1/
kappa_5_solid
kappa_5_ffa 1/
kappa_6_solid
kappa_6_ffa 1/
kappa_7_solid
kappa_7_ffa 1/
kappa_8_solid
kappa_8_ffa 1/
```

```
Mouse Mode 3-Button
 Buttons  L    M    R
 & Keys Rota Move Mo
   Shft +Box -Box Cl
   Ctrl +/-  PkAt Pk1  MvSZ
   CtSh Sele Orig Clip MovZ
SnglClk +/-  Cent Menu
 DblClk Menu  -   PkAt
Selecting Residues
State    1/   1
PyMOL>_
```

```
Selection: sele
Material Parameters
      Density: 1500.0
Shear Modulus: 370370000.0
 Bulk Modulus: 1111100000.0
Shear Viscosity: 0.001
 Bulk Viscosity: 0.001
Pinned Nodes
Van der Waals Interactions
(•) VdW type 1: 0        ( ) VdW type 2: -1
VdW Files   LJ Files
            LJ Epsilon: 1e+15    LJ r0: 1e-09
```

Fig 4:
Selecting Faces of the Tubulin Binding Site

Task 3 – Make the System LJ File

Now we've defined where the interaction sites are on all our molecules, we need to define their strength and equilibrium distance. We'll need to use both <VdW Type> selections in the PyMOL FFEA Loader to do this.

1. Use PyMOL and open **"DyneinMicrotubuleVdW.ffea"** with the following settings:

    - <Show Solid> = VdW
    - < Add Atoms> = Onto Faces

2. In PyMOL FFEA Loader 'Editor' tab, select <VdW Type 1> = 0
3. In PyMOL FFEA Loader 'Editor' tab, select <VdW Type 2> = 0
4. In 'Editor' tab, select appropriate LJ parameters for a '00' interaction (see Box 5)
5. In 'Editor' tab, click <Update LJ File>
6. Save the file as "**structure/DyneinMicrotubule.lj**"

Box 5: LJ Parameters

If you remember from the slides, each *pair* of VdW types has an associated set of LJ parameters. Therefore, we require two VdW types when editing .lj files.
There are two parameters defining each Lennard-Jones interaction, ε and r_0, the interaction strength and equilibrium distance respectively.

r_0 is simple. What is the equilibrium distance of your interaction? Between 5Å and 2nm perhaps?

ε is slightly more complicated. The total interaction energy, E, in FFEA is formed by an integration over both surfaces involved in the interaction. Therefore, the LJ parameter ε has units of J/m^4. We can calculate ϵ by choosing a desired interaction energy and dividing by the area of the MTBD, and the area of the binding site:

$$\varepsilon \approx n\frac{k_B T}{A_1 A_2} \approx n\frac{4.11\times10^{-21}}{(2\times10^{-18})^2}$$

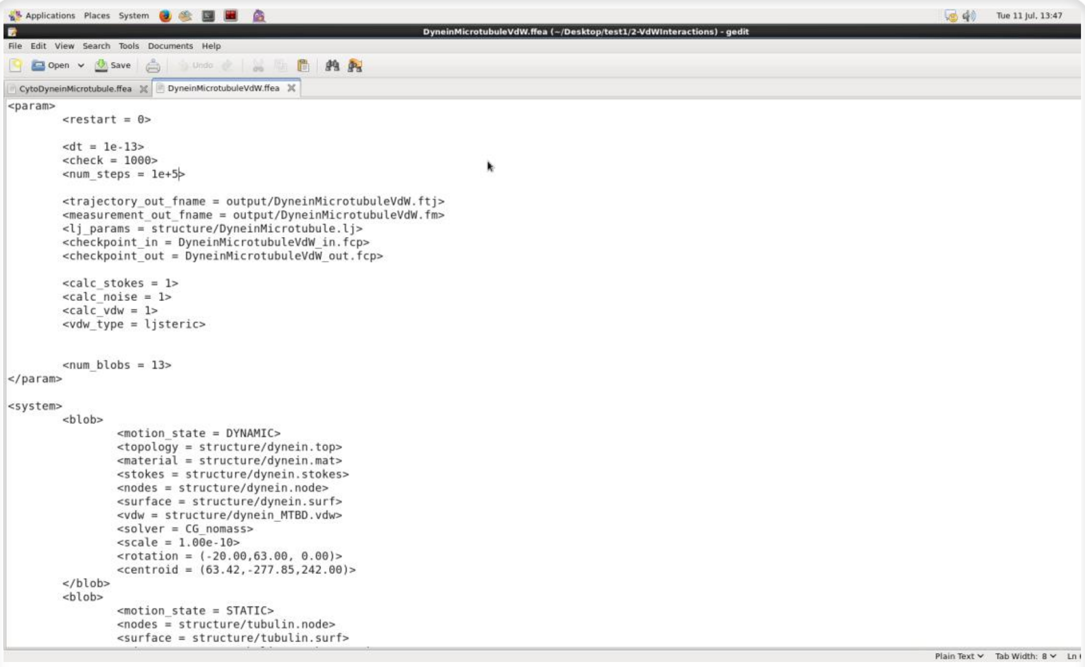
$$\boldsymbol{\varepsilon = 1\times10^{15} n\, J/m^4}$$

where n is some quantity of $k_B T$. Feel free to choose any value of ε you wish, but an interaction energy $\sim k_B T$ makes for a more interesting simulation…

Task 4 – Make the FFEA Script File

Now we have all of the individual files we need, we can bring them together in a .ffea script file to define the simulation we want to run. We'll add some parameters defining what we want to include, how often we want the simulation to output, output filenames etc. Some of the parameter tags below are not included by default, so you'll need to add them yourselves. If you get stuck, check Fig 5.

1. Use mousepad and open **"DyneinMicrotubuleVdW.ffea"**
2. In <params> block, set <trajectory_out_fname> and <measurement_out_fname> to anything you like. "thisworkshopisawesome.ftj" and "thisworkshopisawesome.fm" if you really want 😉
3. In <params> block, set <lj_params> to the .lj file you just made! (don't forget the directory…)
4. In <params> block, set <calc_vdw = 1>
5. In <params> block, set <vdw_type = ljsteric> (see Box 6 for details on this)
6. In <params> block, given <dt = 1e-13>, set <num_steps> and <check> such that you run a 10ns simulation with 100 frames
7. Save the file as "**DyneinMicrotubuleVdW.ffea**"


Applications Places System
DyneinMicrotubuleVdW.ffea (~/Desktop/test1/2-VdWInteractions) - gedit
File Edit View Search Tools Documents Help
Open Save Undo
CytoDyneinMicrotubule.ffea
DyneinMicrotubuleVdW.ffea
Tue 11 Jul, 13:47
Plain Text Tab Width: 8 Ln

```
<param>
        <restart = 0>

        <dt = 1e-13>
        <check = 1000>
        <num_steps = 1e+5>

        <trajectory_out_fname = output/DyneinMicrotubuleVdW.ftj>
        <measurement_out_fname = output/DyneinMicrotubuleVdW.fm>
        <lj_params = structure/DyneinMicrotubule.lj>
        <checkpoint_in = DyneinMicrotubuleVdW_in.fcp>
        <checkpoint_out = DyneinMicrotubuleVdW_out.fcp>

        <calc_stokes = 1>
        <calc_noise = 1>
        <calc_vdw = 1>
        <vdw_type = ljsteric>


        <num_blobs = 13>
</param>

<system>
        <blob>
                <motion_state = DYNAMIC>
                <topology = structure/dynein.top>
                <material = structure/dynein.mat>
                <stokes = structure/dynein.stokes>
                <nodes = structure/dynein.node>
                <surface = structure/dynein.surf>
                <vdw = structure/dynein_MTBD.vdw>
                <solver = CG_nomass>
                <scale = 1.00e-10>
                <rotation = (-20.00,63.00, 0.00)>
                <centroid = (63.42,-277.85,242.00)>
        </blob>
        <blob>
                <motion_state = STATIC>
                <nodes = structure/tubulin.node>
                <surface = structure/tubulin.surf>
```

Fig 5:
Format of the final .ffea script

Box 6: VdW Modes

Within FFEA, the steric interactions are calculated using a much softer potential than the Lennard-Jones function usually allows. It does this as a volume-volume interaction, rather than the face-face interaction. This means that for us to include both steric and Lennard-Jones type interactions, we need some type of interpolation function.

This gives us three different possible VdW modes:

• <vdw_type = steric>	- Pure steric repulsion (volume-volume)
• <vdw_type = lennard-jones>	- Pure Lennard-Jones (face-face) interactions
• < vdw_type = ljsteric>	- The combination of the above two

<u>Task 5 – Run Simulation!</u>

1. Open a command window
2. Change directory to the 2-VdWInteractions directory:
   ➔ cd /home/ffea/Workshop/2-VdWInteractions
3. Run:
   ➔ ffea DyneinMicrotubuleVdW.ffea

4. Wait until it's finished (come chat to us / drink coffee & eat food)
5. Open in PyMOL and watch the simulation!

<u>Task 6 – Plot a Graph</u>

A really quick final thing, we can graph the interaction energy profile.

1. Open a command window
2. Change directory to the 2-VdWInteractions/analysis-scripts directory
   ➔ cd /home/ffea/Workshop/2-VdWInteractions/analysis-scripts
3. Run:
   ➔ python plotVdWEnergy.py ../DyneinMicrotubuleVdW.ffea

So what was your interaction energy at the end? Did it match up with the simulation you saw? If your energy was $|E_{VdW}| \gg k_B T$ then your dynein should be stuck very strongly to the microtubule. $|E_{VdW}| \ll k_B T$ and it should have escaped the attractive potential. $|E_{VdW}| \approx k_B T$…well we'd have to run a much longer simulation to see what that would do.

Extras 1 – Pins & Springs?

FFEA has functionality for additional constraints that we can use to build more effective simulations for our purposes. For example:

1. Use PyMOL and open **"CytoDyneinMicrotubule.ffea"** with the following settings:

    - Pinned Nodes [✓]
    - Springs [✓]

We can see that one of the dynein molecules, the MTBD is covered in small red spheres. These are *pinned* nodes, and will not move at all throughout a simulation. We can use this to artificially bind a molecule to a surface to better control its motion. Check out **"structure/dynein_MTBD.pin"** to see the list of pinned nodes I've defined. By loading <Add Atoms> = Onto Nodes, you can also define pinned regions in the same way you defined VdW regions.

You may also notice that between the dynein monomers is a spring-like object. That is exactly what it is, a harmonic spring that exists between any two nodes in your FFEA universe! A less restrictive constraint than pinned nodes, we can artificially control relative motion between specific nodes. This can be useful for a variety of reasons, and physicists love it because to us the entire universe can be modelled as harmonic springs to a decent approximation 😉
You can check out the spring file in **"structure/cytodynein.springs"**. It's a bit of an odd format so feel free to ask us about it.

I modelled the dynein dimer system in this way to represent the artificial GST dimerization that was done in experiments on the same molecule! The next extra section will show you what these extra additions can accomplish.

Extras 2 – Recovering Atomic Structures

Got some spare time? Consider the following…

At the most basic level, an FFEA model is simply a series of points connected through elements. Perhaps you noticed, the topology of these elements never changes throughout the simulation! Elements cannot invert or change their connectivity, and because of this, we have the ability to recover atomic structures (see Box 7 for some details). Let's do it.

1. Open a command window and change directory to 2-VdWInteractions/
    ➔ cd /home/ffea/Workshop/2-VdWInteractions
2. Run:
    ➔ ffea CytoDyneinMicrotubule.ffea (it's already set up for you)
3. Now, run:
    ➔ ffeatools maptraj output/CytoDyneinMicrotubule.ftj output/CytoDyneinMicrotubule.pdb structure/dyneinFFEAtoPDB.map structure/dynein.pdb

4. Open CytoDyneinMicrotubule.ffea AND output/CytoDyneinMicrotubule.pdb in pymol. The .pdb file can just be opened without the FFEALoader

These 'pseudo-trajectories' are suitable for minimisation and subsequent simulation in something like GROMACS.

Box 7: Atomic Mapping

If we have a series of FFEA nodes, $\vec{x}$, and a set of atoms, $\vec{y}$, we can make a linear mapping as follows:

$$\vec{y} = \boldsymbol{M}\vec{x}$$

where $\boldsymbol{M}$ is a non-square matrix. The above matrix can have many structures that all correspond to the equation. However, by aligning the atomic and continuum structures, and then associating each atom with its containing / nearest element, we can use the element edge vectors to uniquely determine every atom position. The file structure/dyneinFFEAtoPDB.map is a mapping matrix I made, connecting the dynein.node and dynein.top structure to dynein.pdb and written in the Yale Sparse format. This extra task has you apply the matrix to the simulation you ran of the full Cytoplasmic Dynein molecule. You can also apply it to the dynein monomer simulation you ran in Task 5…