DynamO Workshop

Tutorial: Thermostats, transport properties, and compression

Dr Marcus N. Bannerman & Dr Leo Lue

m.campbellbannerman@abdn.ac.uk leo.lue@strath.ac.uk

- This tutorial session concerns content from three online tutorials:
  - Tutorial 3: Configuration file format.
  - Tutorial 4: Square-wells, thermostats, and transport properties.
  - Tutorial 5: Multi-component simulations.
- However, as understanding the file format is essential, some key points are highlighted here.

Section Outline

- To exploit the generality of DynamO you must become familiar with the file format.
- The file format is written in XML, as this is easy for computers and humans to read, it is self descriptive, and is insenstive to whitespace (you may format it however you like).
- There is also support for XML in EVERY programming/scripting language (to help you write your own set-up scripts).
- The format is fully documented online: `http://dynamomd.org/index.php/reference`
- However, the best method to learn is by example, and **dynamod** can generate 25 types of system using its mode "-m" tag.
- There's typically more than one way to do something and I'm also happy to send out what I think are clean/optimal examples (and will be adding an example gallery to the site).
- But the concepts of **Ranges** and **Properties** are critical, so I will introduce them here.

- The configuration file is split into three main sections/tags:

```
<?xml version="1.0"?>
<DynamOconfig version="1.5.0">
  <Simulation>...</Simulation>
  <Properties>...</Properties>
  <ParticleData>...</ParticleData>
</DynamOconfig>
```

- **Simulation:** holds almost all the information on the actual set up of the simulation (interactions, boundaries, dynamics, species).
- **Properties:** is the (optional) way to define per-particle properties, such as diameters, mass, etc (more on this shortly).
- **ParticleData:** contains the position and velocity of every particle.

```
<DynamOconfig>
  ...
  <ParticleData>
    <Pt ID="0">
      <P x="-6.5" y="-6.5" z="-6.5"/>
      <V x="-1.89" y="-0.761" z="-0.469"/>
    </Pt>
    <Pt ID="1">
      <P x="-5.5" y="-5.5" z="-6.5"/>
      <V x="0.771" y="0.546" z="-1.72"/>
    </Pt>
    ...
  </ParticleData>
</DynamOconfig>
```

- Above is the most basic form of the **ParticleData** tag.
- Each **Pt** tag within **ParticleData** represents a single particle.
- The **P** and **V** tags hold the position and velocity.
- The **ID** attribute is just for information on the last computed ID.
- DynamO always numbers particles sequentially in the order they appear, and counts them (no need to specify how many particles there are).

- In particle simulation, we often want to implement statements like:
  - *All particles are point masses and have a mass of one.*
  - *The first 50 particles rotate and have unique/random masses.*
  - *Particle 12 has an infinite mass.*
  - *Particles 32–96 are a single molecule.*
- Implementing these statements through lists or look-up tables can be both difficult to read and inefficient.
- DynamO uses **Ranges** to specify what particles you're talking about, and **Properties** for the actual data.

- For example, a **Species** tag in the configuration file is used to specify the inertia tensors and masses of particles.
- Consider: *All particles are point (non-rotating) masses and have a mass of one*:

```
<Species Mass="1" Name="Bulk" Type="Point">
  <IDRange Type="All"/>
</Species>
```

`<IDRange Type="All"/>` = *All particles*

`<Species ... Type="Point" ...>` = *are point masses*

`<Species ... Mass="1" ...>` = *and have a mass of one.*

- The `Name` attribute is used to identify this **Species** in output (e.g., when $g(r)$ is collected between each Species) and the default value used in **dynamod** is "Bulk". You may use whatever you like.
- The `Mass` attribute is what we term a **Property specifier**, in that it can be a numeric value OR a string identifier for a **Property**.

- *The first 50 particles rotate and have unique/random masses.*

```
<Species InertiaConstant="0.5" Mass="PMass" Name="Spins"
 Type="SphericalTop">
  <IDRange Type="Ranged" Start="0" End="50"/>
</Species>
```

`<IDRange Type="Ranged" Start="0" End="50"/>` = *The first 50 particles*
`<Species ... Type="SphericalTop" ...>` = *rotate (as spherical tops)*
`<Species ... Mass="PMass" ...>` = *and have unique/random masses.*

- `Mass="PMass"` states that there is a **Property** called "PMass", which we should use to look up/calculate the mass of the particle.

- **Properties** must also be defined separately in the property tag:

```
<Properties>
  <Property Type="PerParticle" Name="M" Units="PMass"/>
</Properties>
```

- The "PerParticle" type, means that each particle (**Pt**) tag may have an additional PMass attribute:

```
<ParticleData>
  <Pt PMass="0.0690" ID="0">
    <P x="-6.5" y="-6.5" z="-6.5"/>
    <V x="-3.86" y="-2.95" z="-2.68"/>
  </Pt>
  ...
</ParticleData>
```

- Almost every attribute is a **Property specifier**, and the property mechanism is used for complex potentials (e.g., the PRIME protein potential).
- There are some restrictions (i.e., each particle must be assigned to exactly one Species), but DynamO tests while loading.

- The concept of **Ranges** is also used to specify how particle pairs interact.
- For example: *All particles are hard spheres with a diameter of 1*

```
<Interactions>
  <Interaction Type="HardSphere" Diameter="1" Name="Bulk">
    <IDPairRange Type="All"/>
  </Interaction>
</Interactions>
```

- If there are multiple **Interaction** tags, then the highest one with a correct **IDPairRange** will be used.
- All particle pairings must have at least one matching **Interaction**
- The `Diameter` attribute is also a **Property specifier** so polydisperse/random diameters may be introduced.
- There are many types of **IDRange**, **IDPairRange**, **Species**, and **Interaction** available, please see the online Tutorial 3 and the reference documentation for more information.