# Coarse-graining of Anisotropic Molecules for Energy Materials Simulations

BOISE STATE UNIVERSITY
COLLEGE OF ENGINEERING

Noah Montrose[1], Eridani Rojas[1], Dr. Rainier Barrett[1], and Dr. Eric Jankowski[1]

Boise State University[1]

## ABSTRACT

Coarse-graining---simplifying models of molecules by representing a collection of atoms with a simulation element like a sphere or ellipsoid---can significantly increase the timescales accessible to simulations without loss of structural accuracy. Spherical simulation elements are inaccurate representations of flat molecular structures, though, which are better represented with anisotropic shapes like ellipsoids. In this work we debug and extend open-source software (GRiTS) for calculating the shapes and orientations of an ellipsoid representing a collection of atoms. These functionalities are useful for both validating the correctness of coarse-grained models and for training advanced anisotropic potentials that can be used in accelerated molecular simulations.

## OBJECTIVE

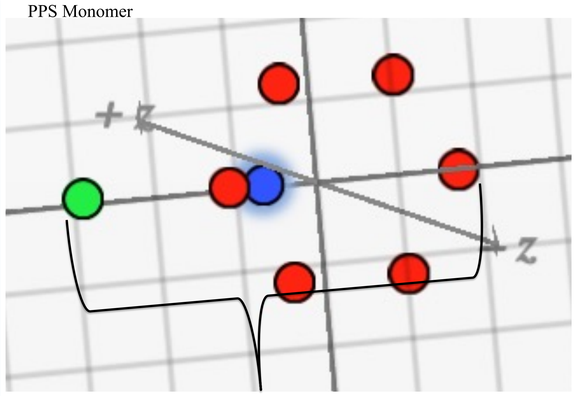
The objective of this project is to create an anisotropic bead option to the existing open-source software (GRiTS) in order to visualize orientational phenomena.

## COARSE-GRAIN PARAMETERS

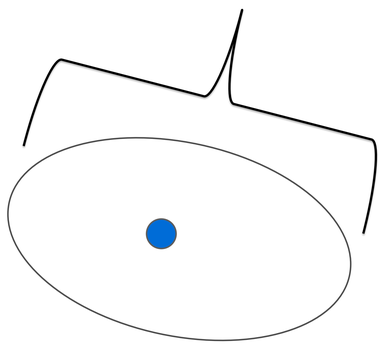
The program needs the following:
- The atom positions
- The atom weights
- The atom element types

To return:
- The major axis
- The center of mass
- The orientation

PPS Monomer




Major Axis



PPS Coarse-grained Bead

Figure 1. The image above shows the heavy atomic positions (excluding hydrogen) for a polyphenylene sulfide (PPS) monomer. The image below shows an anisotropically coarse-grained bead of PPS.

Important things to note:
- The center of mass (CoM) was chosen over the center of geometry (CoG) because in practice, the center of rotation is more realistically represented by the CoM rather than the CoG.
- The sulfur atom has an atomic weight of 32.065u, while the carbon atoms have an atomic weight of 12.011u thus the CoM isn't "off-center"

## COARSE-GRAIN PARAMETERS (CONT.)

In order to the calculate the orientation, a series of steps must first be completed:
1) A plane must be created using the dimensions of the monomer. In order to do this a second vector which is unique and non-parallel to the direction vector is used, called A_CoM. This will be referred to as the reference plane.
2) A second monomer is created, which is meant to represent any other orientation than the reference plane.
3) The cross product of the normal vector of the reference plane and the normal vector of the particle's reference plane is calculated to produce an axis of rotation.
4) The reference plane is rotated by some angle theta along this axis in order to transform the reference plane into the other plane.
5) The axis angle then can be transformed into a quaternion using a python package called rowan, so that they are compatible with our simulation engine, HOOMD-blue.

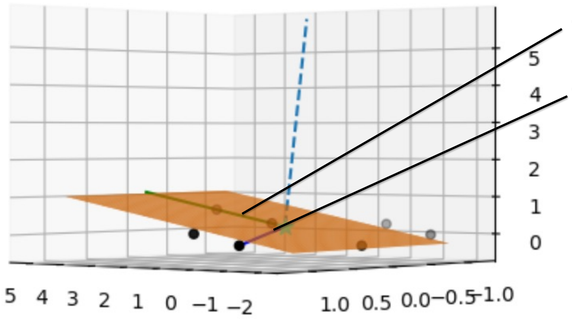



Figure 2. The plane (highlighted in orange) created by the major axis vector ($\overrightarrow{\ell_{||_0}}$) and the A_CoM vector

$$\overrightarrow{v_{ax}} = \underbrace{\left(\overrightarrow{\ell_{||_0}} \times \overrightarrow{A_{CoM_0}}\right)}_{\overrightarrow{n_0}} \times \underbrace{\left(\overrightarrow{\ell_{||_1}} \times \overrightarrow{A_{CoM_1}}\right)}_{\overrightarrow{n_1}}$$

$$\theta_{rot} = arccos\left(\frac{\overrightarrow{n_0} \cdot \overrightarrow{n_1}}{\left|\overrightarrow{n_0}\right| * \left|\overrightarrow{n_1}\right|}\right)$$

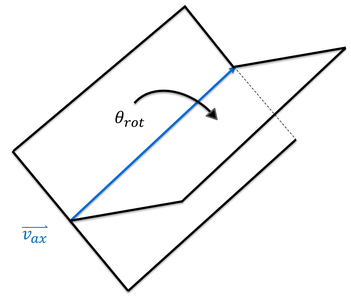



Figure 3. The formula on the left shows the math required in order to perform the rotation between the reference plane ($n_0$ ) and the new plane ($n_1$ ). $\overrightarrow{v_{ax}}$ represents the axis about which the reference plane is being rotated in order to create the new plane, and $\theta_{rot}$ represents the amount that the reference plane needs to be rotated in order to create the new plane

```python
def get_quaternion(n1, n0=np.array([0, 0, 1])):
    """Calculates axis and angle of rotation given two planes normal vectors"""
    V_axis = np.cross(n0, n1)
    theta_numerator = np.dot(n0, n1)
    theta_denominator = np.linalg.norm(n0) * np.linalg.norm(n1)
    theta_rotation = np.arccos(theta_numerator / theta_denominator)
    quaternion = rowan.from_axis_angle(V_axis, theta_rotation)
    return quaternion
```

Figure 4. The function above converts the two planes into a rotational quaternion.

Why use quaternions over Euler angles? Quaternions are more compact, efficient, and stable than Euler angles, but are harder to understand. They extend to imaginary numbers, and rather than returning a typical (X, Y, Z) they return (C, XS, YS, ZS) where C represents $\cos(\theta/2)$, and S represents $\sin(\theta/2)$. Because of this added fourth element, they are resistant to gimbal lock. Gimbal lock is the alignment of rotational planes, and once two separate rotational planes align, they can no longer be represented uniquely using normal vectors.

## APPLICATION VISUALIZED

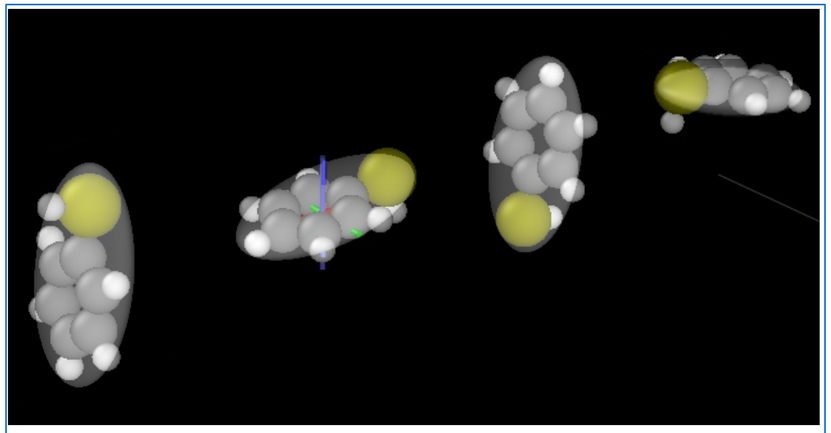


Figure 5. The image above is a rendering of four PPS monomers with different orientations using a visualization software called OVITO. The transparent ellipsoids enveloping each of the four PPS monomers represent the anisotropic coarse-grained beads for each respective monomer, reflective of their orientation.

## CONCLUSIONS AND FUTURE WORK

Conclusions
- Anisotropic coarse-grained particles can be mapped from arbitrary sets of atoms.
- The mapping can be done while remaining agnostic of the bead dimensions/forcefield parameters at the time.
- This mapping can be performed on all atom trajectory files in one workflow step.

Future work
- These coarse-grain mappings can be used to train those forcefield parameters with various techniques (e.g., machine learning, iterative Boltzmann inversion, force-matching, etc.).
- Revise and integrate code with existing workflows (GRiTS).
- Write tests to ensure that code will apply to any input.
- Compare performance and accuracy of anisotropic-mapped and traditional sphere-mapped coarse-grain systems.
- Implement additional mapping schemes (e.g., 3-axis anisotropic beads).
- Create solutions for edge cases (e.g., linear monomers, n<3 particle systems, etc.).

## ACKNOWLEDGEMENTS

A large thanks is given towards Dr. Amy Moll and Dr. Eric Jankowski at Boise State University for pushing me towards the Research Experience for Undergraduate program. This poster wouldn't have been possible without the guidance of Dr. Rainier Barrett, a postdoctoral student here at BSU. This research was supported by the NSF via the REU Site: Materials for Society (Award No. 1950305) and was partially supported by the National Aeronautics and Space Administration (NASA) under the University Leadership Initiative program; grant number 80NSSC20M0165.

BOISE STATE UNIVERSITY