UGM 2024

HACKATHON SHOWCASE

Instructions:

- *Add a slide summarising the work you/your group did during this hackathon!*
- *Copy the template or do your own thing (but avoid large inserts e.g. movies)*
- *Be ready to give a ~2 min summary when your slide comes up!*
- *We may share the slides later - let us know if you'd rather not have your slide shared*

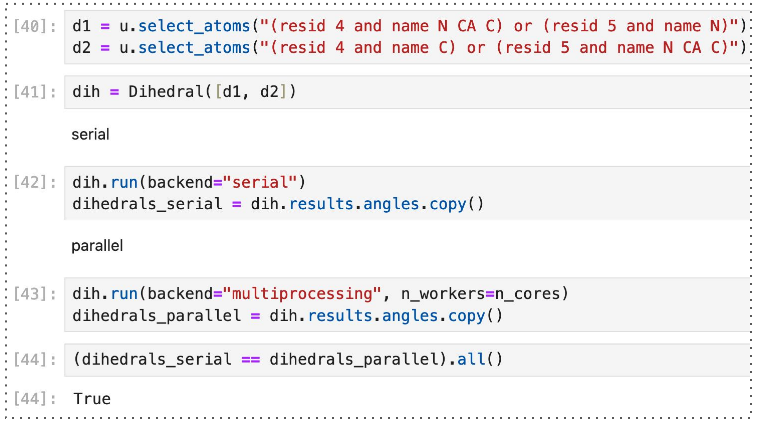
Everyone can edit this presentation - don't delete other people's stuff!

MDANALYSIS

Parallel analysis

Adding parallel supports (or un-parallelizable flag) to Dihedral, RDF, PCA

- Parallelized: Dihedrals, Ramachandran, Janin (PR #4682)
- PCA marked not parallelizable (PR #4684)
- RDF still work-in-progress



```
[40]: d1 = u.select_atoms("(resid 4 and name N CA C) or (resid 5 and name N)")
      d2 = u.select_atoms("(resid 4 and name C) or (resid 5 and name N CA C)")

[41]: dih = Dihedral([d1, d2])
```

serial

```
[42]: dih.run(backend="serial")
      dihedrals_serial = dih.results.angles.copy()
```

parallel

```
[43]: dih.run(backend="multiprocessing", n_workers=n_cores)
      dihedrals_parallel = dih.results.angles.copy()

[44]: (dihedrals_serial == dihedrals_parallel).all()

[44]: True
```

Bugs discovered

- Error when n_frames < n_workers (issue #4685)
- `require_all_aggregators` cannot be set by users

Valerij, Oliver, Yuxuan

ProteinDataKit

Converting protein trajectory data into suitable format for ML (tensors, graphs, embeddings)

Problem:

create datasets (training + validation + validation) for ML

represent trj data as tensors, graphs or embeddings

Solution:

MDAKit based on two classes (ProteinDataSet and MLDataSet)

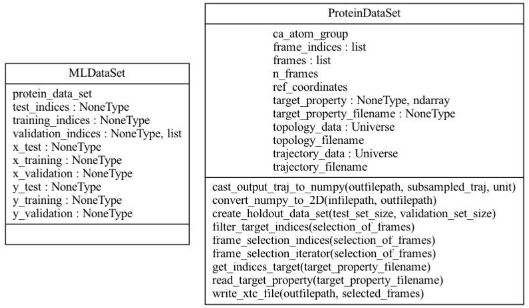
ability to ‘attach’ target variables based on time-dependent properties

Status:

(see figure…)

Where to find it: https://github.com/alepandini/ProteinDataKit

Ferdoos, Namir, Asal and Ale



```
MLDataSet
---------
protein_data_set
test_indices : NoneType
training_indices : NoneType
validation_indices : NoneType, list
x_test : NoneType
x_training : NoneType
x_validation : NoneType
y_test : NoneType
y_training : NoneType
y_validation : NoneType
```

```
ProteinDataSet
--------------
ca_atom_group
frame_indices : list
frames : list
n_frames
ref_coordinates
target_property : NoneType, ndarray
target_property_filename : NoneType
topology_data : Universe
topology_filename
trajectory_data : Universe
trajectory_filename
--------------
cast_output_traj_to_numpy(outfilepath, subsampled_traj, unit)
convert_numpy_to_2D(infilepath, outfilepath)
create_holdout_data_set(test_set_size, validation_set_size)
filter_target_indices(selection_of_frames)
frame_selection_indices(selection_of_frames)
frame_selection_iterator(selection_of_frames)
get_indices_target(target_property_filename)
read_target_property(target_property_filename)
write_xtc_file(outfilepath, selected_frames)
```

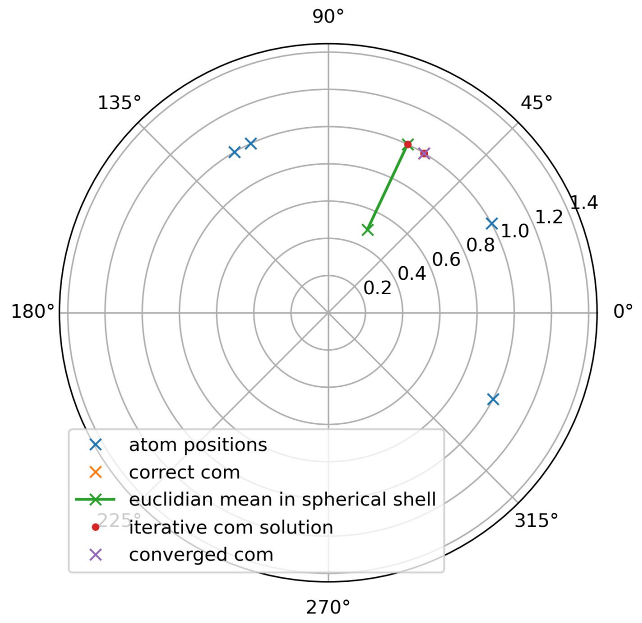
Fast and correct center of mass calculation

Outline

- COM calculation needs unwrap, which is slow
- We can map on a circle and do weighted averages on the circle
- 1st approx for small angles: Projected euclidian average
- Find the COM iteratively and use correct distance metric

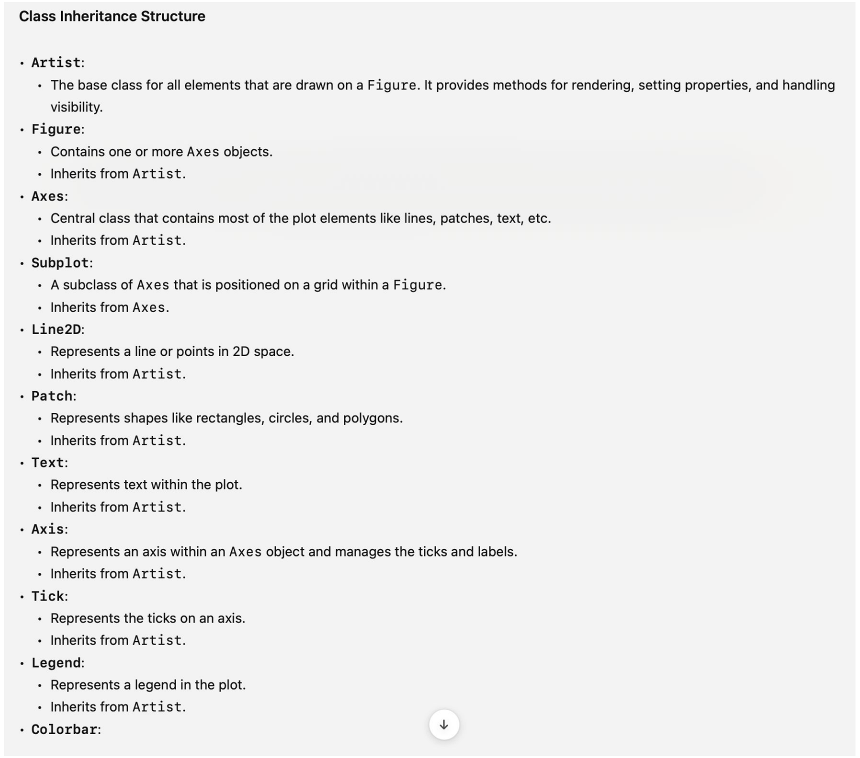
Open Questions

- Is it actually faster? (for small molecules 1 to 2 iterations is enough)
- Can it be used in triclinic boxes?



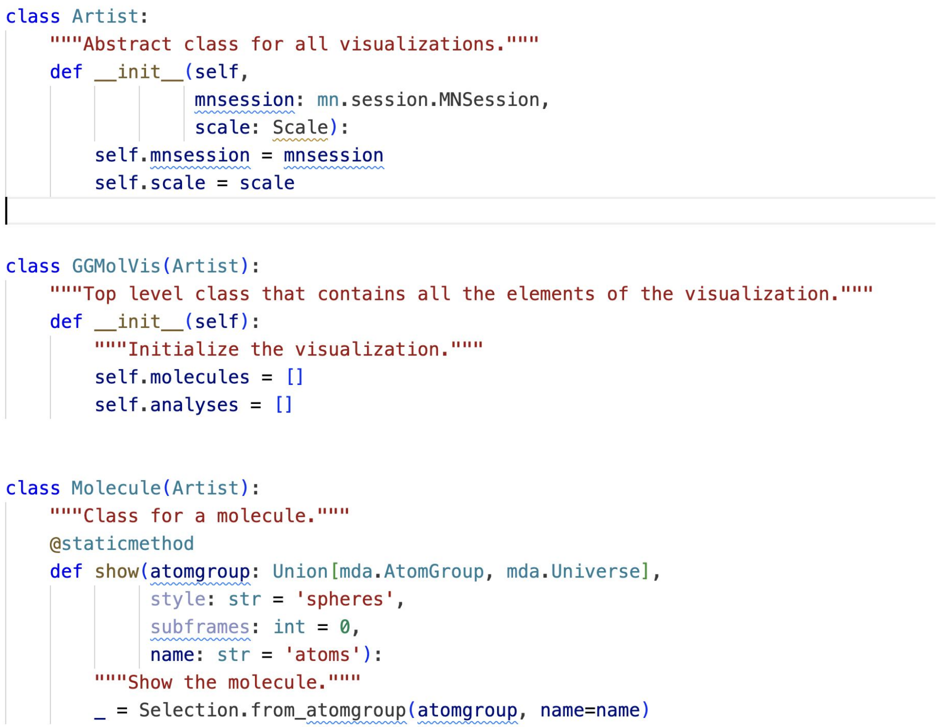
Henrik, Kira, Josh

Learning from Matplotlib for Molecular visualization



Class Inheritance Structure

- `Artist`:
  - The base class for all elements that are drawn on a `Figure`. It provides methods for rendering, setting properties, and handling visibility.
- `Figure`:
  - Contains one or more `Axes` objects.
  - Inherits from `Artist`.
- `Axes`:
  - Central class that contains most of the plot elements like lines, patches, text, etc.
  - Inherits from `Artist`.
- `Subplot`:
  - A subclass of `Axes` that is positioned on a grid within a `Figure`.
  - Inherits from `Axes`.
- `Line2D`:
  - Represents a line or points in 2D space.
  - Inherits from `Artist`.
- `Patch`:
  - Represents shapes like rectangles, circles, and polygons.
  - Inherits from `Artist`.
- `Text`:
  - Represents text within the plot.
  - Inherits from `Artist`.
- `Axis`:
  - Represents an axis within an `Axes` object and manages the ticks and labels.
  - Inherits from `Artist`.
- `Tick`:
  - Represents the ticks on an axis.
  - Inherits from `Artist`.
- `Legend`:
  - Represents a legend in the plot.
  - Inherits from `Artist`.
- `Colorbar`:



```python
class Artist:
    """Abstract class for all visualizations."""
    def __init__(self,
                 mnsession: mn.session.MNSession,
                 scale: Scale):
        self.mnsession = mnsession
        self.scale = scale


class GGMolVis(Artist):
    """Top level class that contains all the elements of the visualization."""
    def __init__(self):
        """Initialize the visualization."""
        self.molecules = []
        self.analyses = []


class Molecule(Artist):
    """Class for a molecule."""
    @staticmethod
    def show(atomgroup: Union[mda.AtomGroup, mda.Universe],
             style: str = 'spheres',
             subframes: int = 0,
             name: str = 'atoms'):
        """Show the molecule."""
        _ = Selection.from_atomgroup(atomgroup, name=name)
```

Yuxuan

CIF Reader

It’s hard!

- Use gemmi
  - PDB’s reference implementation seems not supported
- Coordinates work
- Topologies are hard

- Note: Protein Databank deprecated PDB format
- No more PDB files, you’ll get mmcif/pdbx
  - Text mmcif/pdbx
  - binary

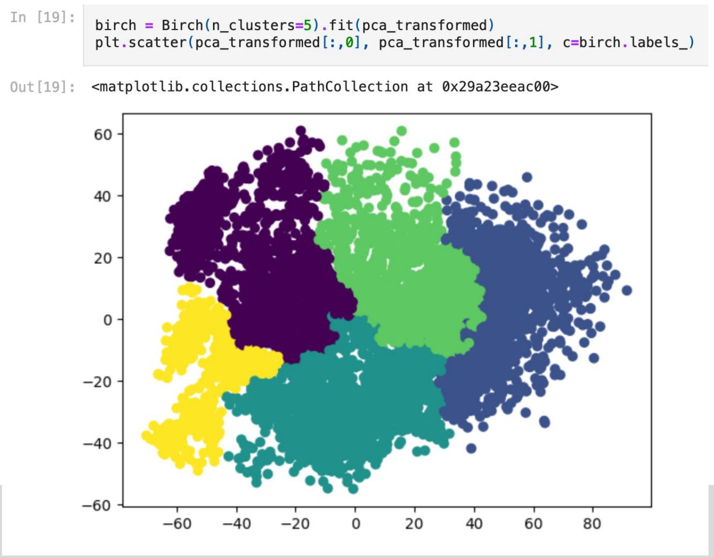
Teaching notebook for clustering

How to use MDAnalysis for manual clustering

https://github.com/lexin-chen/cluster

“There is no free lunch in clustering”.

Demonstrate how to do simple dimensionality reduction with PCA and clustering with common clustering algorithms in sklearn.



```
In [19]: birch = Birch(n_clusters=5).fit(pca_transformed)
         plt.scatter(pca_transformed[:,0], pca_transformed[:,1], c=birch.labels_)

Out[19]: <matplotlib.collections.PathCollection at 0x29a23eeac00>
```

Lexi