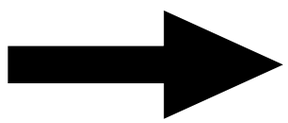
CECAM Macromolecular Simulation Workshop, Jülich, 2015-10-14

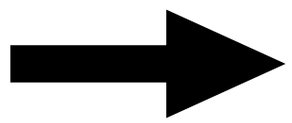
Oliver Beckstein, David Dotson
Arizona State University

Naveen Michaud-Agrawal, Elizabeth J. Denning, Joshua Adelman, Jonathan Barnoud, Christian Beckstein (logo), Alejandro Bernardin, Sébastien Buchoux, David Caplan, Matthieu Chavent, Xavier Deupi, Jan Domański, Lennard van der Feltz, Philip Fowler, Joseph Goose, Richard J. Gowers, Lukas Grossar, Benjamin Hall, Joe Jordan, Max Linke, Jinju Lu, Robert McGibbon, Alex Nesterenko, Manuel Nuno Melo, Caio S. Souza, Danny Parton, Joshua L. Phillips, Tyler Reddy, Paul Rigor, Sean L. Seyler, Andy Somogyi, Lukas Stelzl, Gorman Stock, Isaac Virshup, Zhuyi Xue, Carlos Yáñez S

simulation trajectory



"accessible" structured data



analysis algorithm

processed data

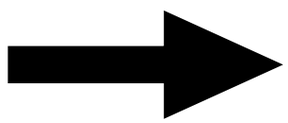
tables

images

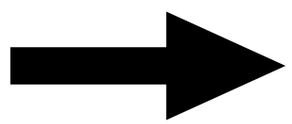
graphs

RESULTS!

simulation trajectory



"accessible" structured data



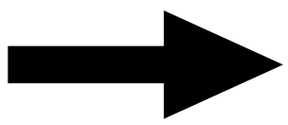
analysis algorithm

processed data

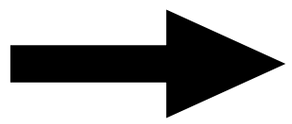
- tables
- images
- graphs

RESULTS!

simulation trajectory



"accessible" structured data



analysis algorithm

processed data

tables

images

graphs

RESULTS!

simulation trajectory

dcd, xtc, trr, ncdf, trj, pdb, pqr, gro, crd, dms, trz, mol2, xyz, config, history, gms, ...

psf, tpr, prmtop, dms, mol2, hoomd xml, ...

"accessible" structured data

analysis algorithm

processed data

tables

images

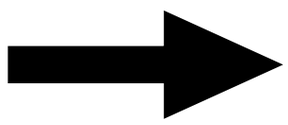
graphs

RESULTS!

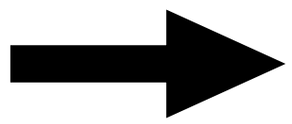
simulation trajectory

dcd, xtc, trr, ncdf, trj, pdb, pqr, gro, crd, dms, trz, mol2, xyz, config, history, gms, ...

psf, tpr, prmtop, dms, mol2, hoomd xml, ...

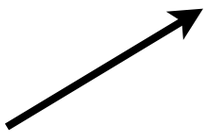


"accessible" structured data



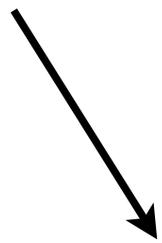
analysis algorithm

processed data



tables

images



graphs

RESULTS!

simulation trajectory

dcd, xtc, trr, ncdf, trj, pdb, pqr, gro, crd, dms, trz, mol2, xyz, config, history, gms, ...

psf, tpr, prmtop, dms, mol2, hoomd xml, ...

"accessible" structured data

analysis algorithm

$$\rho_i = \sqrt{\left\langle (\mathbf{x}_i(t) - \langle \mathbf{x}_i \rangle)^2 \right\rangle}$$

C_α RMSF

processed data

tables

images

graphs

RESULTS!

simulation trajectory

dcd, xtc, trr, ncdf, trj, pdb, pqr, gro, crd, dms, trz, mol2, xyz, config, history, gms, ...

psf, tpr, prmtop, dms, mol2, hoomd xml, ...

"accessible" structured data

analysis algorithm

processed data

tables

images

graphs

RESULTS!

```python
import numpy as np
import MDAnalysis as mda

u = mda.Universe("topol.tpr", "trj.xtc")
ca = u.select_atoms("name CA")
means = np.zeros((len(ca), 3))
sumsq = np.zeros_like(means)
for k, ts in enumerate(u.trajectory):
    sumsq += (k/(k+1.0)) * (ca.positions - means)**2
    means[:] = (k*means + ca.positions)/(k+1.0)
rmsf = np.sqrt(sumsq.sum(axis=1)/(k+1.0))

matplotlib.pyplot.plot(ca.residues.resids, rmsf)
```

$$\rho_i = \sqrt{\left\langle (\mathbf{x}_i(t) - \langle \mathbf{x}_i \rangle)^2 \right\rangle}$$

Cα RMSF

simulation trajectory

dcd, xtc, trr, ncdf, trj, pdb, pqr, gro, crd, dms, trz, mol2, xyz, config, history, gms, ...

psf, tpr, prmtop, dms, mol2, hoomd xml, ...

"accessible" structured data

analysis algorithm

processed data

tables

images

graphs

RESULTS!

```python
import numpy as np
import MDAnalysis as mda

u = mda.Universe("topol.tpr", "trj.xtc")
ca = u.select_atoms("name CA")
means = np.zeros((len(ca), 3))
sumsq = np.zeros_like(means)
for k, ts in enumerate(u.trajectory):
    sumsq += (k/(k+1.0)) * (ca.positions - means)**2
    means[:] = (k*means + ca.positions)/(k+1.0)
rmsf = np.sqrt(sumsq.sum(axis=1)/(k+1.0))

matplotlib.pyplot.plot(ca.residues.resids, rmsf)
```

$$\rho_i = \sqrt{\left\langle (\mathbf{x}_i(t) - \langle \mathbf{x}_i \rangle)^2 \right\rangle}$$

$C\alpha$ RMSF

simulation trajectory

dcd, xtc, trr, ncdf, trj, pdb, pqr, gro, crd, dms, trz, mol2, xyz, config, history, gms, ...

psf, tpr, prmtop, dms, mol2, hoomd xml, ...

"accessible" structured data

analysis algorithm

processed data

tables

images

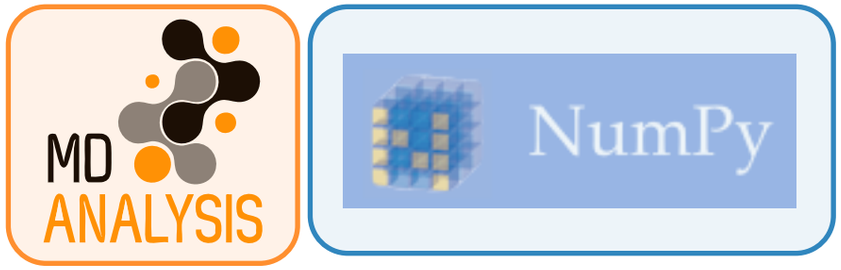
graphs

RESULTS!

```python
import numpy as np
import MDAnalysis as mda

u = mda.Universe("topol.tpr", "trj.xtc")
ca = u.select_atoms("name CA")
means = np.zeros((len(ca), 3))
sumsq = np.zeros_like(means)
for k, ts in enumerate(u.trajectory):
    sumsq += (k/(k+1.0)) * (ca.positions - means)**2
    means[:] = (k*means + ca.positions)/(k+1.0)
rmsf = np.sqrt(sumsq.sum(axis=1)/(k+1.0))

matplotlib.pyplot.plot(ca.residues.resids, rmsf)
```

$$\rho_i = \sqrt{\left\langle (\mathbf{x}_i(t) - \langle \mathbf{x}_i \rangle)^2 \right\rangle}$$

$C\alpha$ RMSF



simulation trajectory

dcd, xtc, trr, ncdf, trj, pdb, pqr, gro, crd, dms, trz, mol2, xyz, config, history, gms, ...

psf, tpr, prmtop, dms, mol2, hoomd xml, ...

"accessible" structured data

analysis algorithm

processed data

tables

images

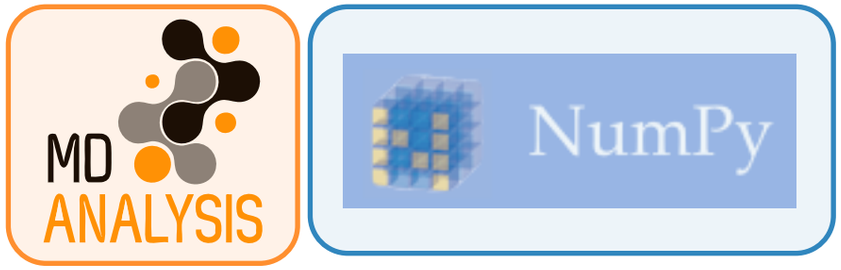
graphs

RESULTS!

```python
import numpy as np
import MDAnalysis as mda

u = mda.Universe("topol.tpr", "trj.xtc")
ca = u.select_atoms("name CA")
means = np.zeros((len(ca), 3))
sumsq = np.zeros_like(means)
for k, ts in enumerate(u.trajectory):
    sumsq += (k/(k+1.0)) * (ca.positions - means)**2
    means[:] = (k*means + ca.positions)/(k+1.0)
rmsf = np.sqrt(sumsq.sum(axis=1)/(k+1.0))

matplotlib.pyplot.plot(ca.residues.resids, rmsf)
```

$$\rho_i = \sqrt{\left\langle (\mathbf{x}_i(t) - \langle \mathbf{x}_i \rangle)^2 \right\rangle}$$

$C\alpha$ RMSF



simulation trajectory

dcd, xtc, trr, ncdf, trj, pdb, pqr, gro, crd, dms, trz, mol2, xyz, config, history, gms, ...

psf, tpr, prmtop, dms, mol2, hoomd xml, ...

"accessible" structured data

analysis algorithm

processed data

tables

images

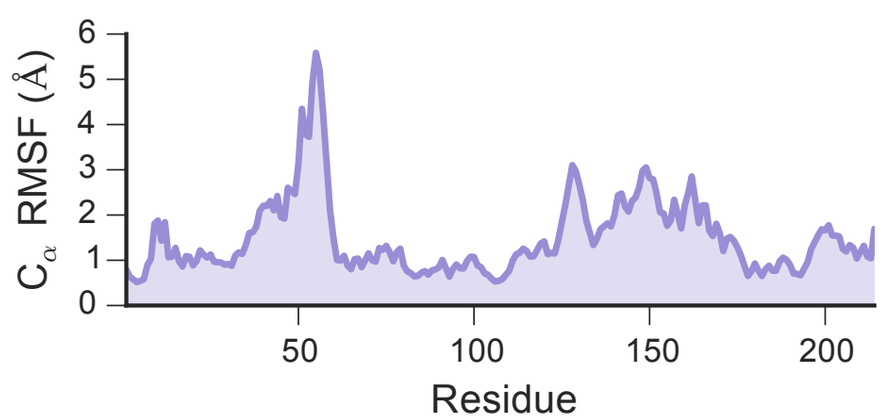
graphs

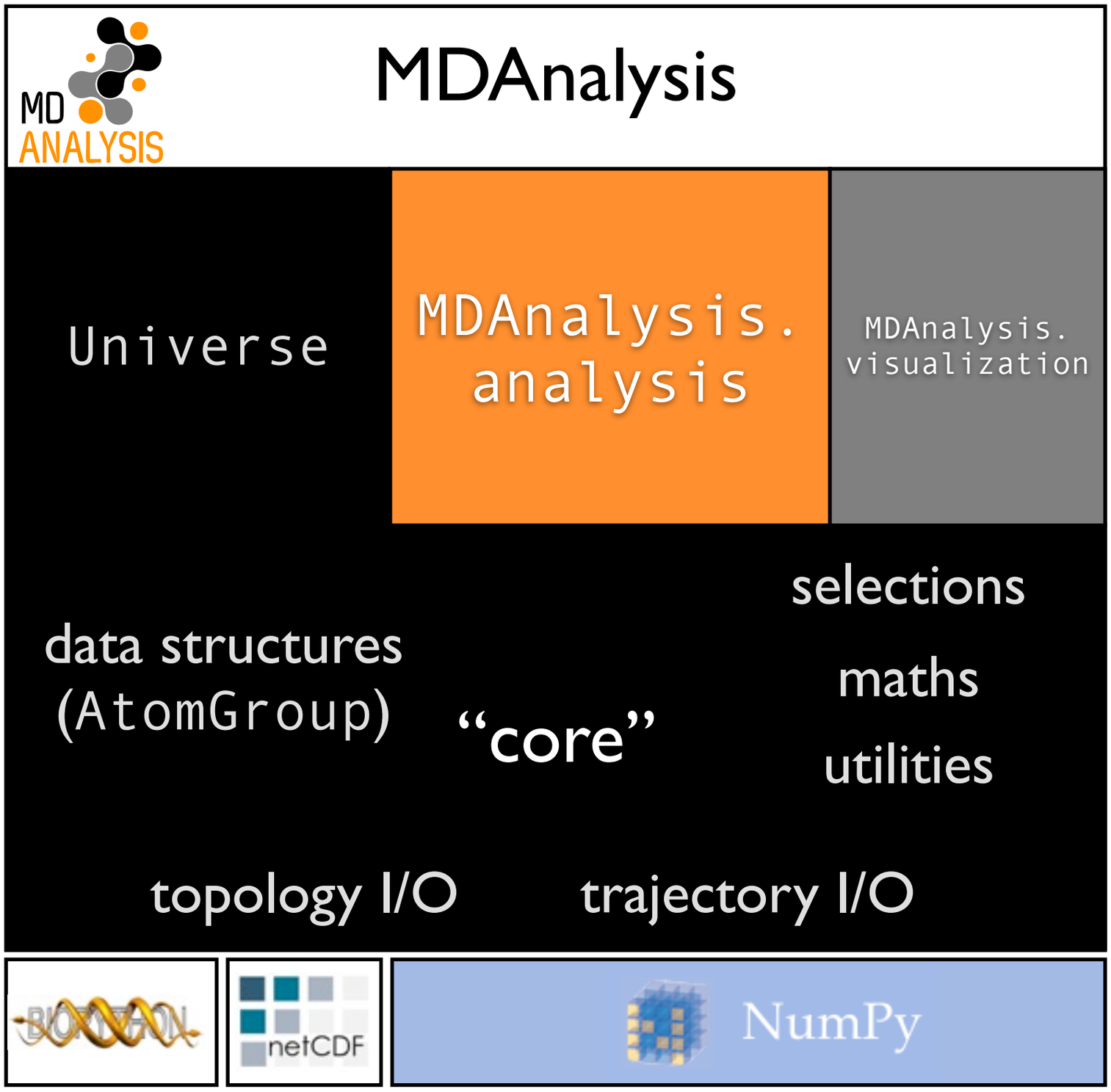
$$\rho_i = \sqrt{\left\langle (\mathbf{x}_i(t) - \langle \mathbf{x}_i \rangle)^2 \right\rangle}$$

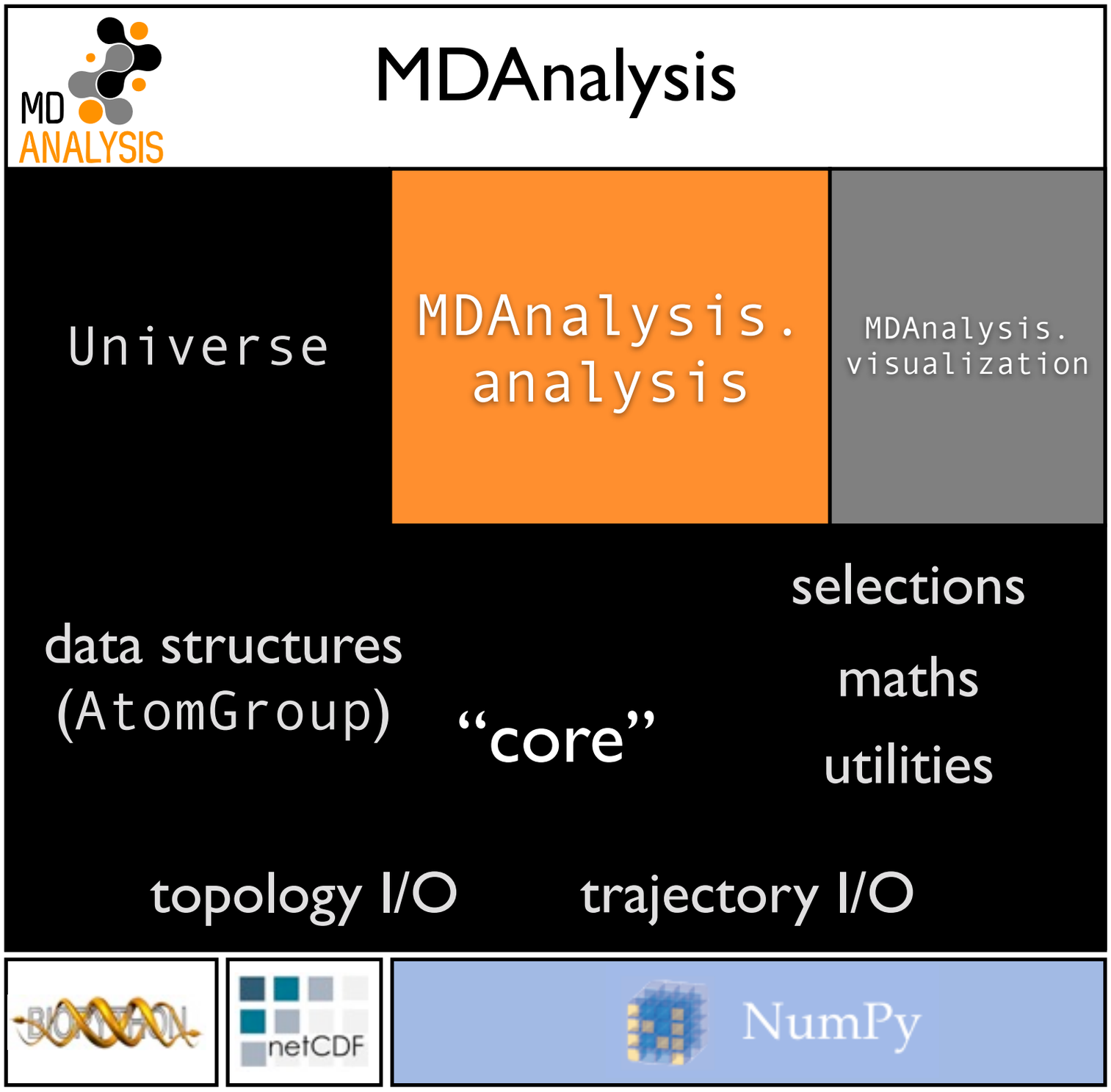
Cα RMSF

```
import numpy as np
import MDAnalysis as mda

u = mda.Universe("topol.tpr", "trj.xtc")
ca = u.select_atoms("name CA")
means = np.zeros((len(ca), 3))
sumsq = np.zeros_like(means)
for k, ts in enumerate(u.trajectory):
    sumsq += (k/(k+1.0)) * (ca.positions - means)**2
    means[:] = (k*means + ca.positions)/(k+1.0)
rmsf = np.sqrt(sumsq.sum(axis=1)/(k+1.0))

matplotlib.pyplot.plot(ca.residues.resids, rmsf)
```

RESULTS!



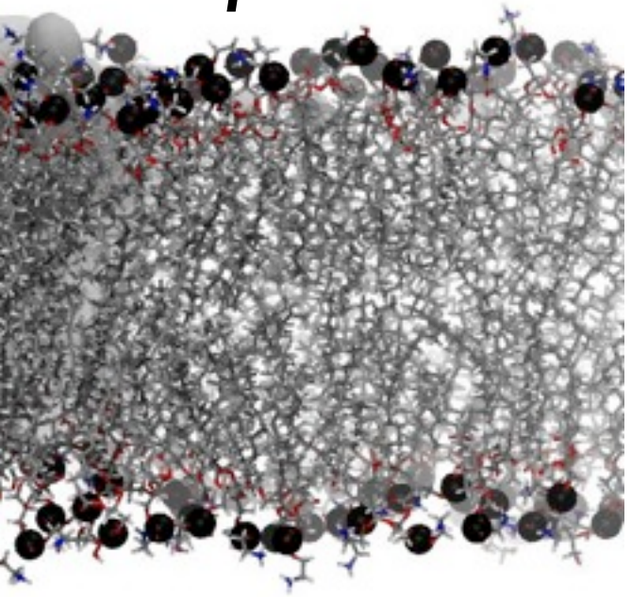

MD ANALYSIS
MDAnalysis
Universe
MDAnalysis.
analysis
MDAnalysis.
visualization
selections
data structures
(AtomGroup)
maths
“core”
utilities
topology I/O
trajectory I/O
netCDF
NumPy

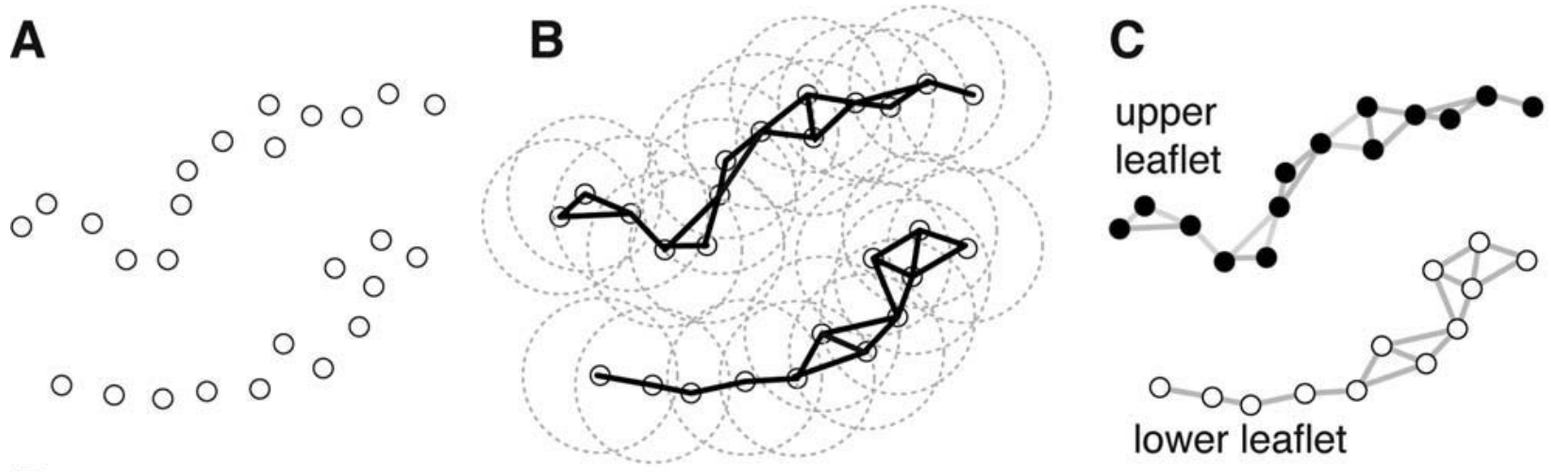


Code base:

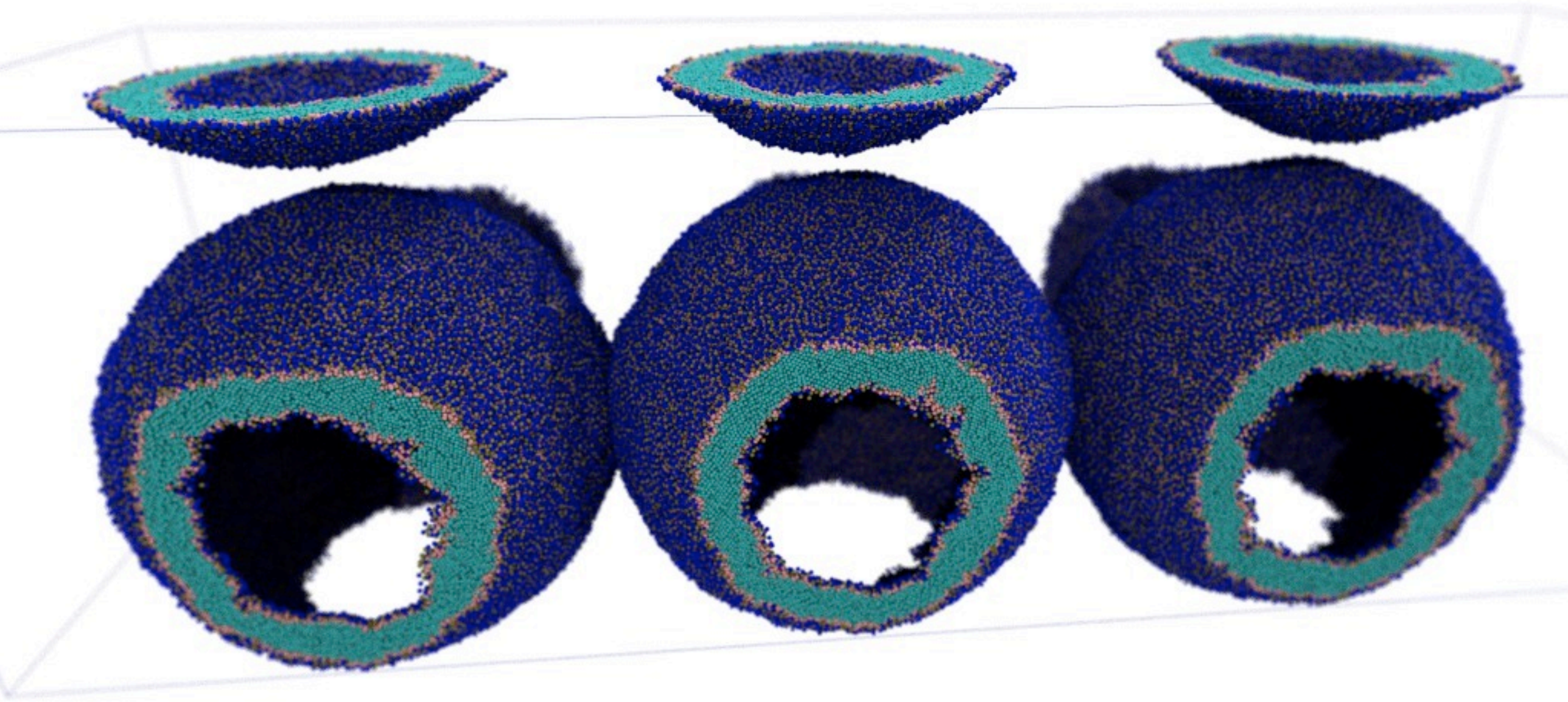
- python 2.7
- cython
- C

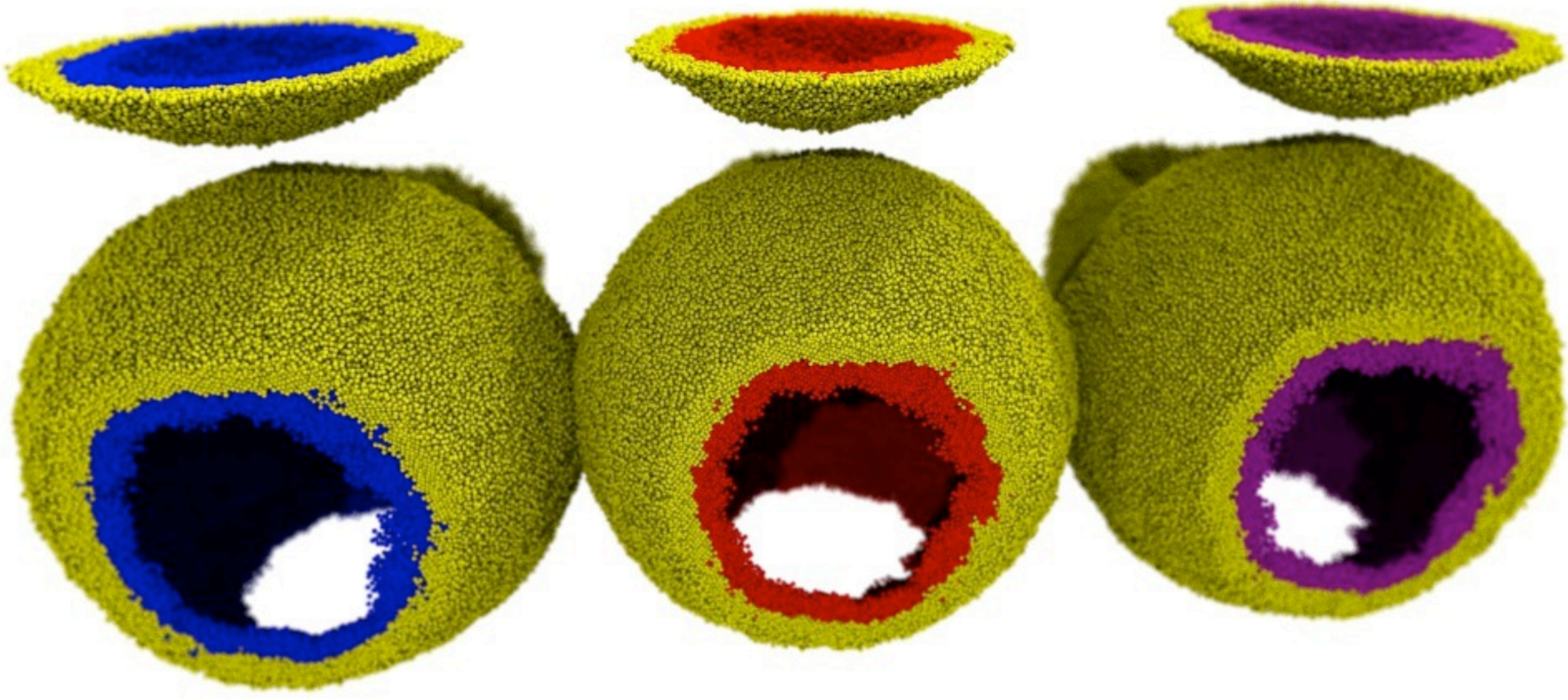
LeafletFinder

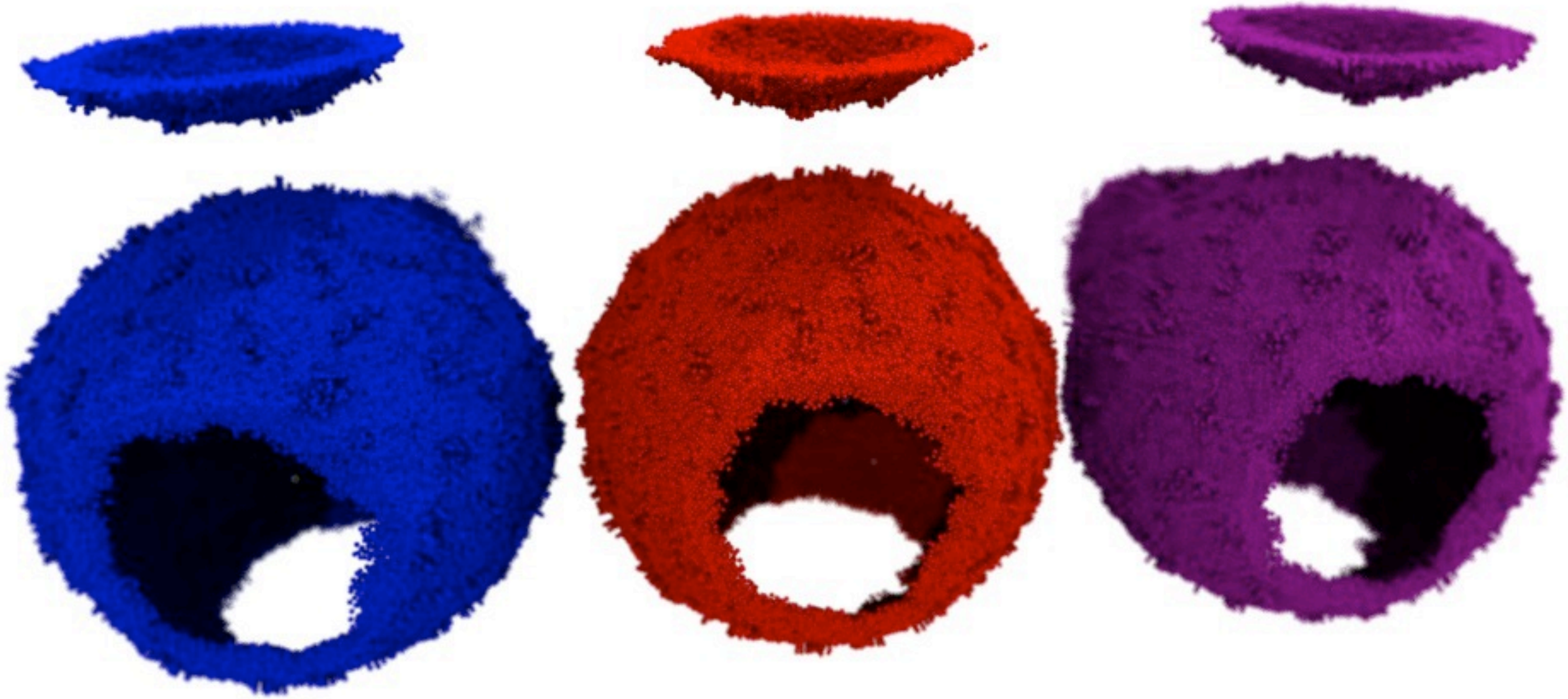


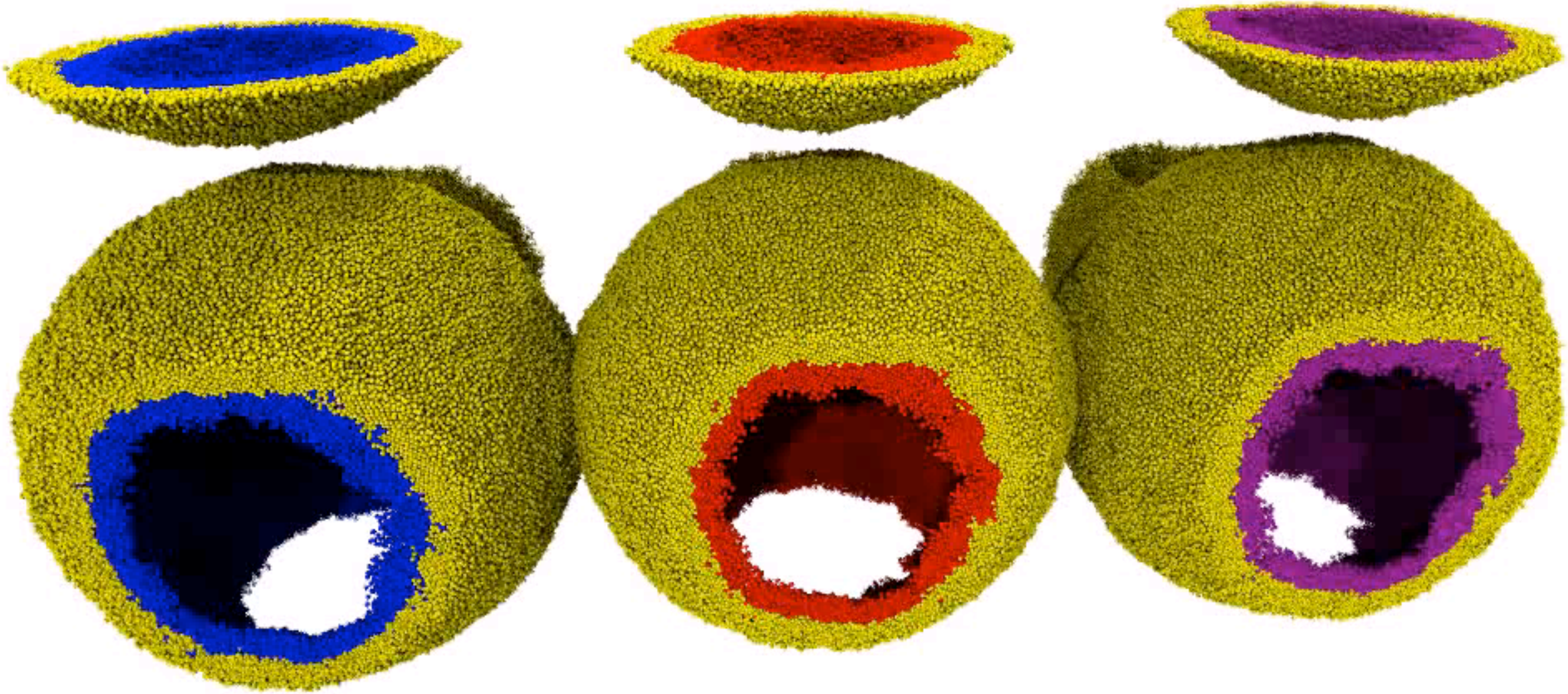



```
import MDAnalysis as mda
import networkx as nx
from MDAnalysis.lib.distances import distance_array

u = mda.Universe(pdb, xtc)
headgroup_atoms = u.select_atoms("name P*")
A x = headgroup_atoms.positions

B adj = (distance_array(x, x) < 12)
C leaflets = sorted(nx.connected_components(nx.Graph(adj)), key=len, reverse=True)

A_lipids = headgroup_atoms[leaflets[0]].residues
B_lipids = headgroup_atoms[leaflets[1]].residues
```







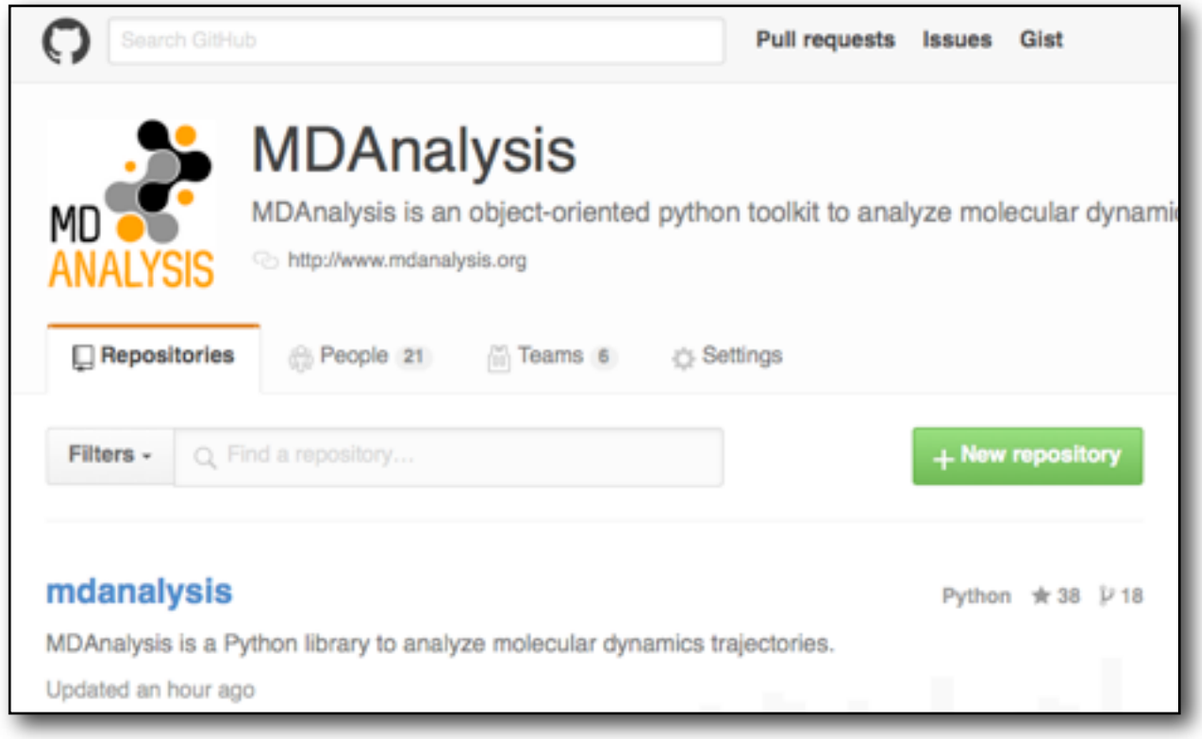


Runs on:

- Linux
- Mac OS X

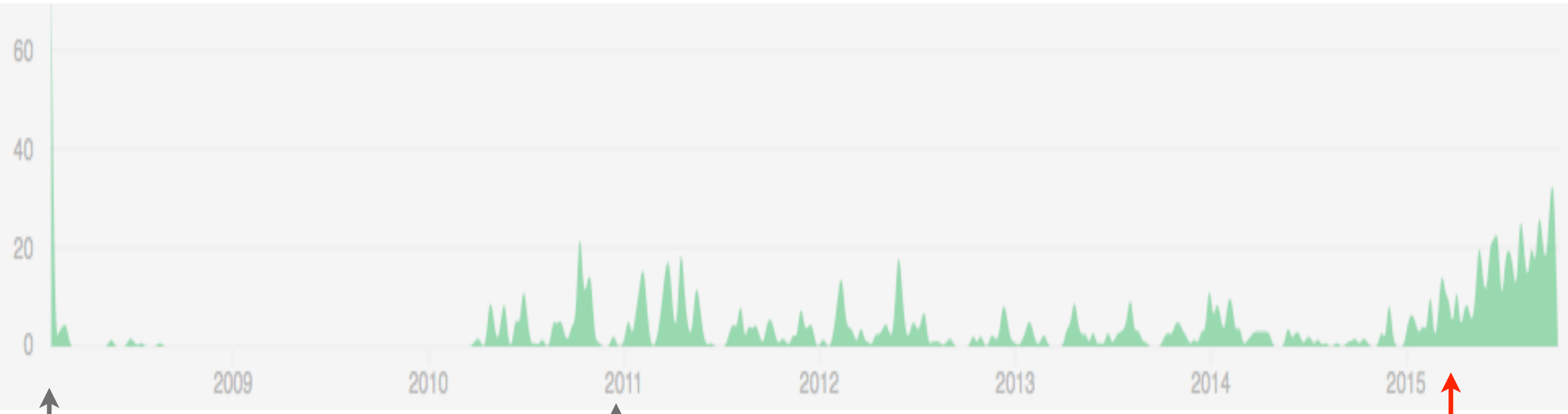
Open source

- GPL v2
- github.com/MDAnalysis



Open and inclusive community:

- questions are answered (mailing list)
- *pull requests* welcome!
- community code review
- continuous integration with > 2,500 unit tests
- 36 contributing authors (Oct 2015)




```
Author: Naveen Michaud-Agrawal
Date:   Thu Jan 31 11:42:33 2008 +0000

initial import
```

GitHub

Software News and Updates

MDAnalysis: A Toolkit for the Analysis of Molecular Dynamics Simulations

NAVEEN MICHAUD-AGRAWAL,[1] ELIZABETH J. DENNING,[1,2] THOMAS B. WOOLF,[1,3] OLIVER BECKSTEIN[3,4]

Received 23 October 2010; Revised 6 February 2011; Accepted 12 February 2011
DOI 10.1002/jcc.21787
Published online 15 April 2011 in Wiley Online Library (wileyonlinelibrary.com).

J Comput Chem 32: 2319–2327, 2011

Naveen Michaud-Agrawal, Elizabeth J. Denning, Joshua Adelman, Jonathan Barnoud, Christian Beckstein (logo), Alejandro Bernardin, Sébastien Buchoux, David Caplan, Matthieu Chavent, David L. Dotson, Xavier Deupi, Jan Domański, Lennard van der Feltz, Philip Fowler, Joseph Goose, Richard J. Gowers, Lukas Grossar, Benjamin Hall, Joe Jordan, Max Linke, Jinju Lu, Robert McGibbon, Alex Nesterenko, Manuel Nuno Melo, Caio S. Souza, Danny Parton, Joshua L. Phillips, Tyler Reddy, Paul Rigor, Sean L. Seyler, Andy Somogyi, Lukas Stelzl, Gorman Stock, Isaac Virshup, Zhuyi Xue, Carlos Yáñez S, and Oliver Beckstein

Join us at www.mdanalysis.org

github.com/MDAnalysis

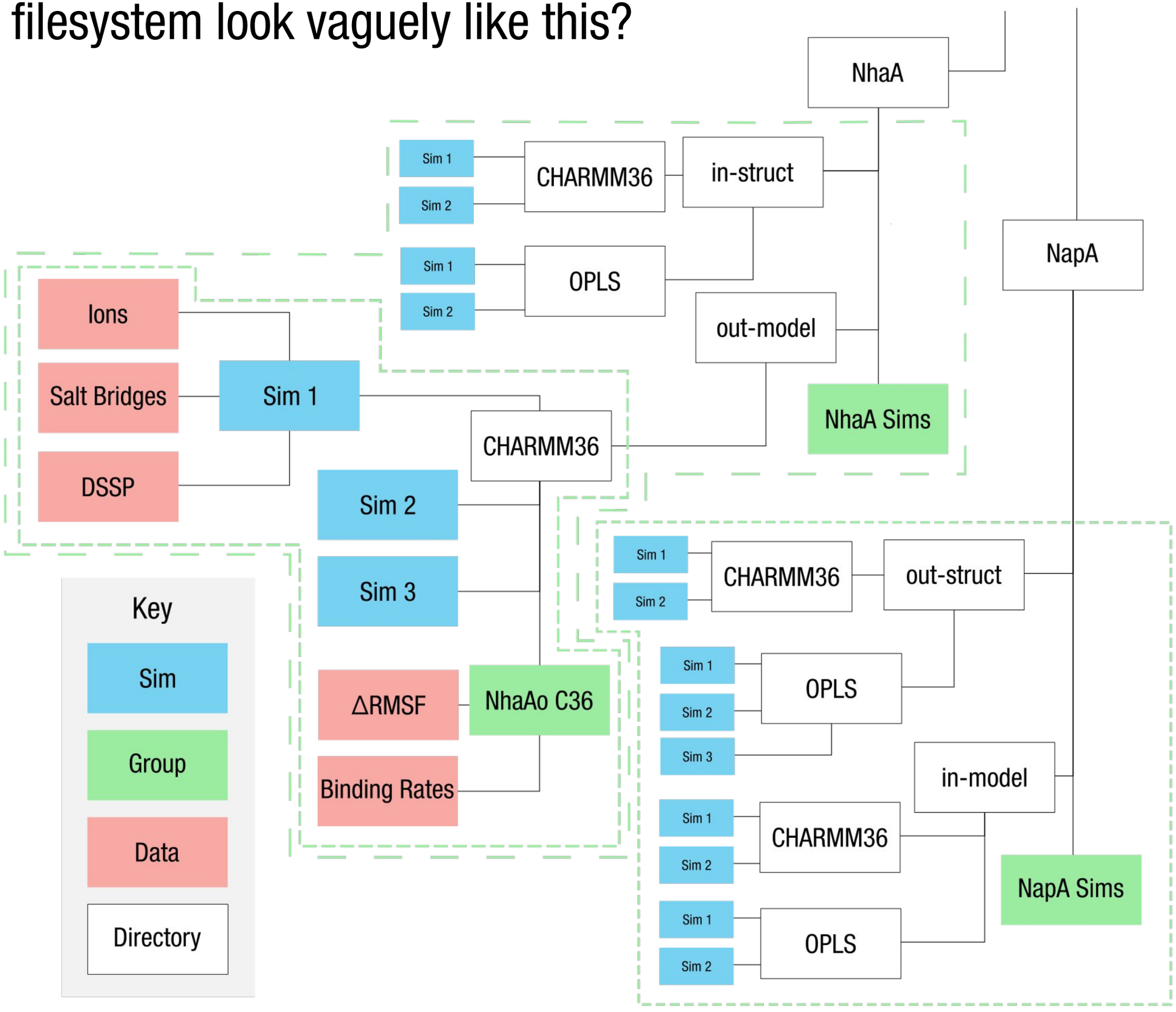
MD SYNTHESIS

persistent python objects for molecular dynamics data science

David Dotson, Oliver Beckstein

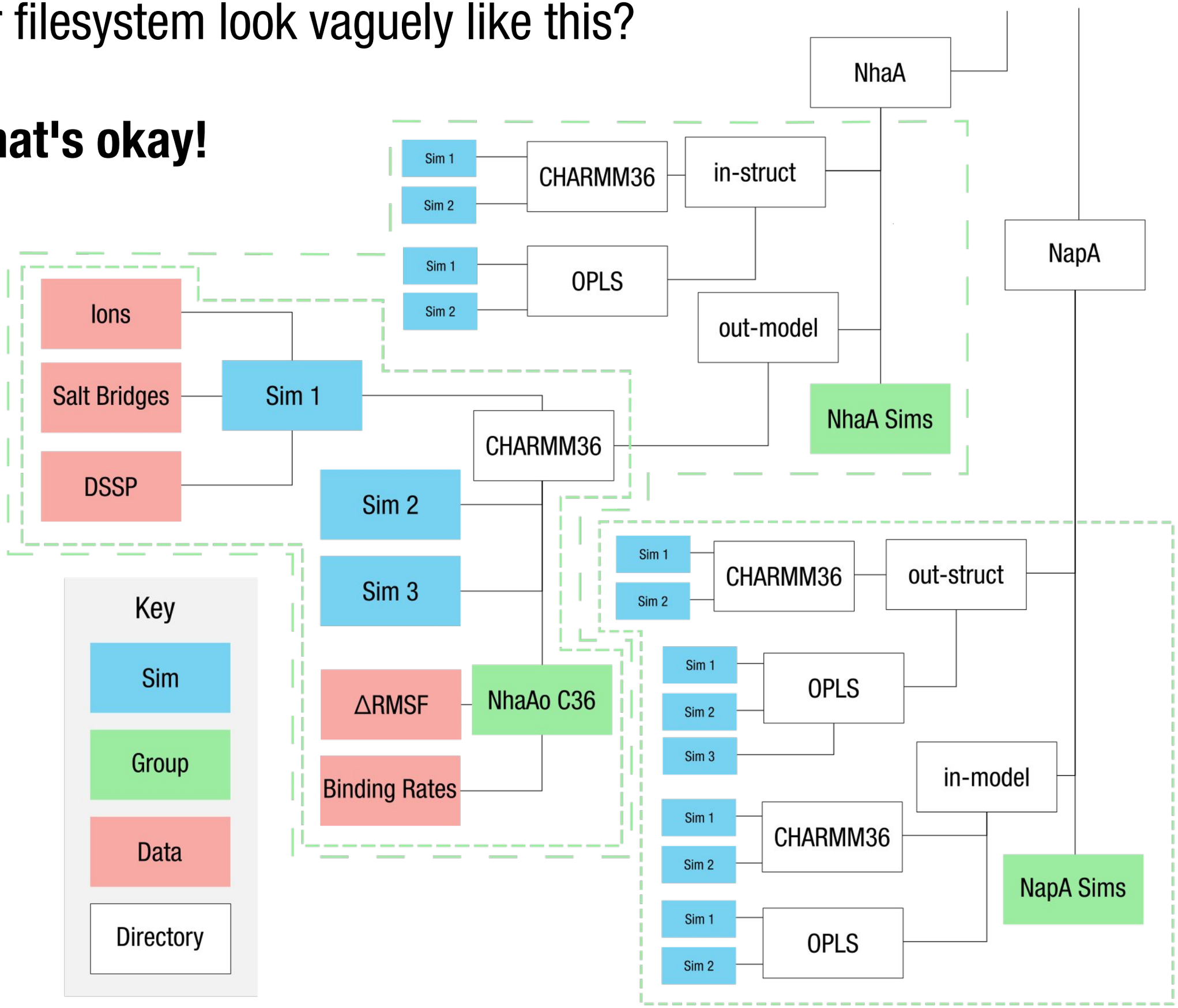
https://github.com/Becksteinlab/MDSynthesis

Does your filesystem look vaguely like this?

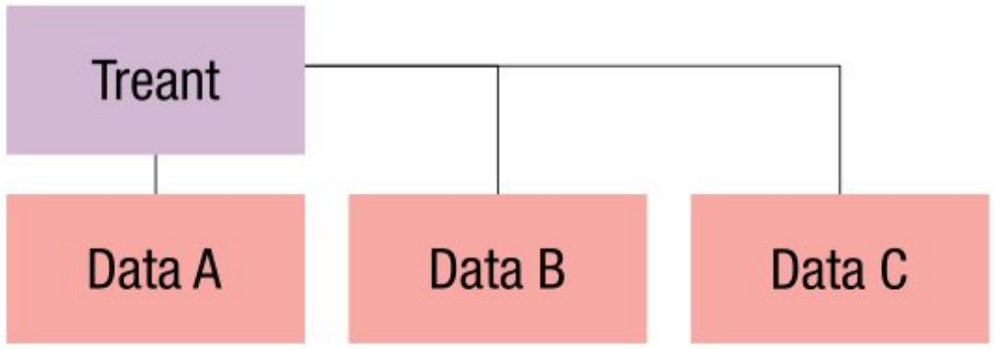


Does your filesystem look vaguely like this?

That's okay!



MDSynthesis is built to make data efficient storage and recall pythonic.




```
Name: 'sprout'
Tags(['DIMS', 'gbsw', 'dimer'])
Categories({'forcefield': 'CHARMM36',
            'transition': 'in to out',
            'protein': 'NhaA'})
```

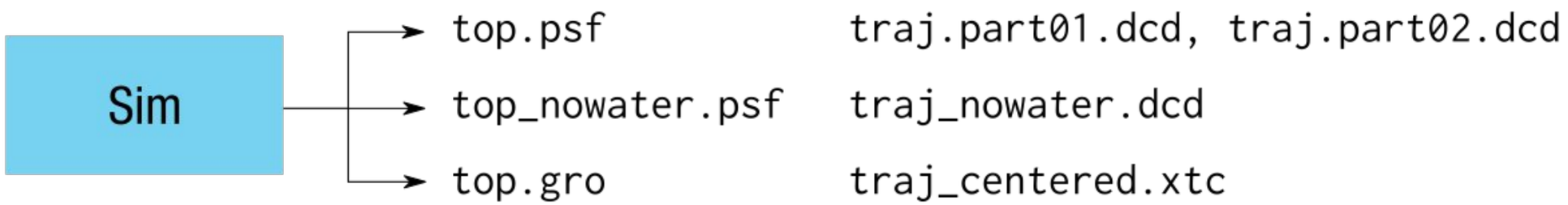
Treants can store identifying metadata that can be used to differentiate them from other **Treants**, but their main purpose is to store datasets, in particular pandas objects and numpy arrays using the efficient HDF5 data format.

Treants live in the filesystem as directory trees, with a state file storing their state persistently on disk as they are used.

```
sprout/
├── Treant.bc5c5c78-83d5-4164-85b9-e069367ae00a.h5
└── something_wicked
    └── pdData.h5
```

The same **Treant** can be used in multiple independent python sessions, since it stores no state in memory.

MDSynthesis is built to make data efficient storage and recall pythonic.



Sims are Treants that store universe definitions and atom selections using MDAnalysis as an interface layer, allowing

- programmatic access to trajectory details with less work
- enough abstraction to write analysis code that works across very different simulations

Groups can store the locations of other Treants in the filesystem, and find them when they move; they offer convenience methods for

- querying for subselections of their members (in development)
- mapping functions across members, in parallel
- aggregation of member datasets

Time permitting, we will demo **MDSynthesis** at the end of the tutorial.

David Dotson, Oliver Beckstein

https://github.com/Becksteinlab/MDSynthesis