lipidize

May 2, 2016

```
In [202]: import MDAnalysis
          import numpy as np
          from random import shuffle

          def norm(x):
              return x / np.sqrt(x.dot(x))

          def flatten(x):
              return [item for sublist in x for item in sublist]
```

```
In [211]: def replaceDum(dummypdb,
                         lipidpdb=[],
                         composition=[],
                         labels=[]):

              # Structure with dummy atoms
              uprot = MDAnalysis.Universe(dummypdb)

              ulipid = []
              for lip in lipidpdb:
                  ulipid.append(MDAnalysis.Universe(lip))

              # Selection for specified lipids
              sel_lipid = []; init_lipid_pos = [];
              nlipid = len(ulipid)
              for u in ulipid:
                  sel = u.select_atoms('all')
                  sel_lipid.append(sel)
                  init_lipid_pos.append(sel.positions)

              # Create selections for dummy atoms, select dummys
              # within a distance of the protein if necessary
              dum_n = uprot.select_atoms('resname DUM and name N')
              dum_o = uprot.select_atoms('resname DUM and name O')
              dum_no = dum_n + dum_o

              # Number of dummy atoms
              ndum =  dum_no.n_atoms
              ndumn = dum_n.n_atoms
              ndumo = dum_o.n_atoms

              # We need to specify some form of composition / ratio
              # of the lipids. By Default, assume a balanced amount
              # of the specified lipids
# TODO: Make robust when ndum * species_frac != integer
if composition == []:
    composition = (np.array([1.]*nlipid)) / nlipid
composition *= ndum

# Create a list of lipids to be merged
tomerge = []
for l,c in zip(sel_lipid,composition):
    tomerge.append([l]*np.int(c))

# Flatten the list and shuffle so placement on the
# surface is random
tomerge = flatten(tomerge)
shuffle(tomerge)

# Select all the atoms for lipid and create a new
# universe that has ndum lipid molecules
mergemol = MDAnalysis.Merge(*tomerge)
sel = mergemol.select_atoms('all')
nlipidatoms = sel.n_atoms

#TODO: Cleanup the lipid numbering, segments etc..
resid = 0; segnames = []; resids = []
for lipid in tomerge:
    resids.append([resid]*lipid.n_atoms)
    resid += 1
sel.set_resids(flatten(resids))

# Array to store coordinates of lipid molecules
newcoords = np.empty([nlipidatoms,3])
start = 0; end = tomerge[0].n_atoms

# Generate the new popc coordinates for each 'O' dummy atom
np.random.shuffle(dum_o.positions)
for pos, lipid in zip(dum_o.positions,tomerge[0:ndumo]):

    # Store the lipid position
    init_lipid_pos = lipid.positions

    # Align the principal axis with the dummy atom vector
    lipid.align_principal_axis(0,norm(-1*pos))

    # Rotate about the principal axis some random amount to
    # give the illusion (MAGIC!) of disorder
    lipid.rotateby(np.random.uniform(low=-180, high=180,size=1),
                   norm(pos),[0, 0, 0])

    # Translate to the dummy atom position, but add some fuzzyness
    end = start + lipid.n_atoms
    newcoords[start:end] = pos + lipid.positions +\
        np.random.uniform(low=-3.0,high=3.0,size=[1,3])

    # Coordinate array offsets
    start += lipid.n_atoms
```

```
                # Return the lipid to its initial position before moving
                lipid.positions = init_lipid_pos

            # Generate the new popc coordinates for each 'N' dummy atom
            np.random.shuffle(dum_n.positions)
            for pos, lipid in zip(dum_n.positions,tomerge[ndumo:ndum]):
                init_lipid_pos = lipid.positions
                lipid.align_principal_axis(0,norm(pos))
                lipid.rotateby(np.random.uniform(low=-180, high=180, size=1),
                               norm(pos),[0, 0, 0])
                end = start + lipid.n_atoms
                newcoords[start:end] = pos + lipid.positions +\
                    np.random.uniform(low=-3.0,high=3.0,size=[1,3])
                start += lipid.n_atoms
                lipid.positions = init_lipid_pos

            # Apply the coordinates to the selection
            sel.set_positions(newcoords)

            # Write out the pdb
            mergemol.atoms.write('lipidized.pdb')
```

```
In [212]: replaceDum('./pdbs/result.pdb',
                    ['./pdbs/POPC_1.pdb','./pdbs/CLOL_1.pdb','./pdbs/NSM_1.pdb'])
```

```
In [ ]:
```