PSU PDB Geometry

A python library to explore the geometric features and experimental evidence of pdb structures.

This library has four simple classes that bring together the protein structure atomic coordinates, the structure's electron density, and the secondary structures.

A querying facility is implemented to analyse the pdb on geometric measures, with the ability to produce reports easily in html format.

There are a set of predefined reports for ease of use, or a simple interface to create any reports required. An example is given at the end of the hand-crafted report to explore a reported non-planar omega in structure 2bw4 (references).

The bringing together of secondary structure, coordinates and electron density requires 3 bioinformatics libraries, along with some standard libraries.

Required Libraries

Library	Description	Link
BioPython	For protein structure coordinates and information	https://biopython.org/
DSSP	For secondary structure	Installation: http://biskit.pasteur.fr/install/applications/dssp Information: https://biopython.org/DIST/docs/api/Bio.PDB.DSSP%27-module.html
pdb_eda	For access to the ccp4 electron density files	https://pdb-eda.readthedocs.io/en/latest/index.html
Pandas	For dataframes	Installation https://pandas.pydata.org/docs/getting_started/index.html User guide https://pandas.pydata.org/docs/user_guide/10min.html
Numpy	For matrices	https://numpy.org/
Matplotlib	Data visualisation	Installation https://matplotlib.org/3.1.0/index.html Colour Maps https://matplotlib.org/3.1.0/tutorials/colors/colormaps.html
Seaborn	Data visualisation	https://seaborn.pydata.org/index.html#

Classes and Methods

GeoPdb

GeoPdb('pdbcode','dataPath')

Initialises the object.
PdbCode: Initialising this object with the pdb 4 letter code automatically loads the atomic coordinates, electron density and dssp and creates a list of atom objects as well as a dataframe that can be queried.
dataPath: The dataPath is the directory in which the pdb files are stored, in the BioPython format of 'pdbABCD.ent'. If the pdb file does not exist it will attempt to download it.

GeoPdb.getStructureCsv()

Returns the dataframe that was created on initialisation. The headers are:
Structure level: pdbCode,resolution
Residue level: chain,rid,dssp,aa
Atom level: atom,atomNo,electrons,element,x,y,z,bfactor,occupant,occupancy,
Density information: 2FoFc, FoFc,Fo,Fc
e.g one of the rows of 2bw4
Structure level: 2bw4, 0.9
Residue level: A, 7, VAL
Atom level: C, 3, 6, C, 26.64, 36.89, 39.89, 12.34, A, 0.58
Density information: 78.05, -7.47, 85.52, 92.99

GeoPdb.getGeometryCsv(geoList,hueList)

Returns a dataframe created from the geometric measures and hues that you request.
geoList: is a list of geometric measures definitions. Each geometric measure is defined as either a distance, angle or dihedral angle with 2,3 or 4 atoms respectively. The atoms are defined using the standard pdb atom naming system, with + used to go in the C-terminus direction and – in the N-terminus. For example, the distance between a residues Cα and the following Cα would be defined as 'CA:CA+1'. The backbone dihedral PHI would be defined as 'C-1:N:CA:C' and the angle TAU would be defined as 'N:CA:C'. e.g.
geoList = ['CA:C:N+1:CA+1','N:CA:C','C:N+1','CA:CA+1']
Note, there is no restriction on bonded geometric measures or distance, so you could choose to look at 'C-5:N' and 'N-6:CA:C+2' if you wish. This gives the flexibility to design reports that look at specific features of interest such as planarity around certain atoms.
Note, important, the resulting dataframe has no nulls, so a row is returned for a given residue only if there is a geometric measures for all of the specified measures. If you specify 'CA-100:CA' you will not get very many results. This is a design decision to ensure you can correlate sets of data, if you do not want all the data correlated against each other you can specify separate dataframes for those you do want to be correlated.
hueList: is a list of attributes that the plots can be coloured on, and is in the format of a string list, e.g.
hueList = ['dssp','aa','bfactor','2FoFc','rid']
The possible hues and information available are:
Structure level: pdbCode, resolution
Residue level: chain, rid, dssp, aa
Atom Level: atom, atomNo, x, y, z, bfactor, occupant, occupancy, electrons, element
Density information: Fo, Fc, FoFc, 2FoFc
Note that where the hue is numeric and per atom, the average will be given for the residue.

GeoDensity

GeoDensity('pdbcode','normalisation')

Initialises the object. This object is created automatically as a member object of GeoPdb above.
pdbCode: Initialising this object with the pdb 4 letter code automatically loads the atomic coordinates, electron density and dssp and creates a list of atom objects as well as a dataframe that can be queried.
normalisation: In order to compare density matrices across pdbs the density matrices need to be normalised. Currently this is not used as the only method implemented is 'Fifty'

GeoPdb.getStructureCsv()normalisation

Returns the dataframe that was created on initialisation. The headers are:
Structure level: pdbCode,resolution
Residue level: chain,rid,dssp,aa
Atom level: atom,atomNo,electrons,element,x,y,z,bfactor,occupant,occupancy,
Density information: 2FoFc, FoFc,Fo,Fc
e.g one of the rows of 2bw4
Structure level: 2bw4, 0.9
Residue level: A, 7, -, VAL
Atom level: CA, 2, 6, C, 26.97, 36.99, 38.41, 13.95, A, 0.58
Density information: 83.55, 4.55, 79.00, 74.45

Useful colour link:
https://matplotlib.org/3.1.0/tutorials/colors/colormaps.html

Note the numerical hues are the averages of the atoms

Classes and Methods

GeoReport
GeoDensity
GeoAtom

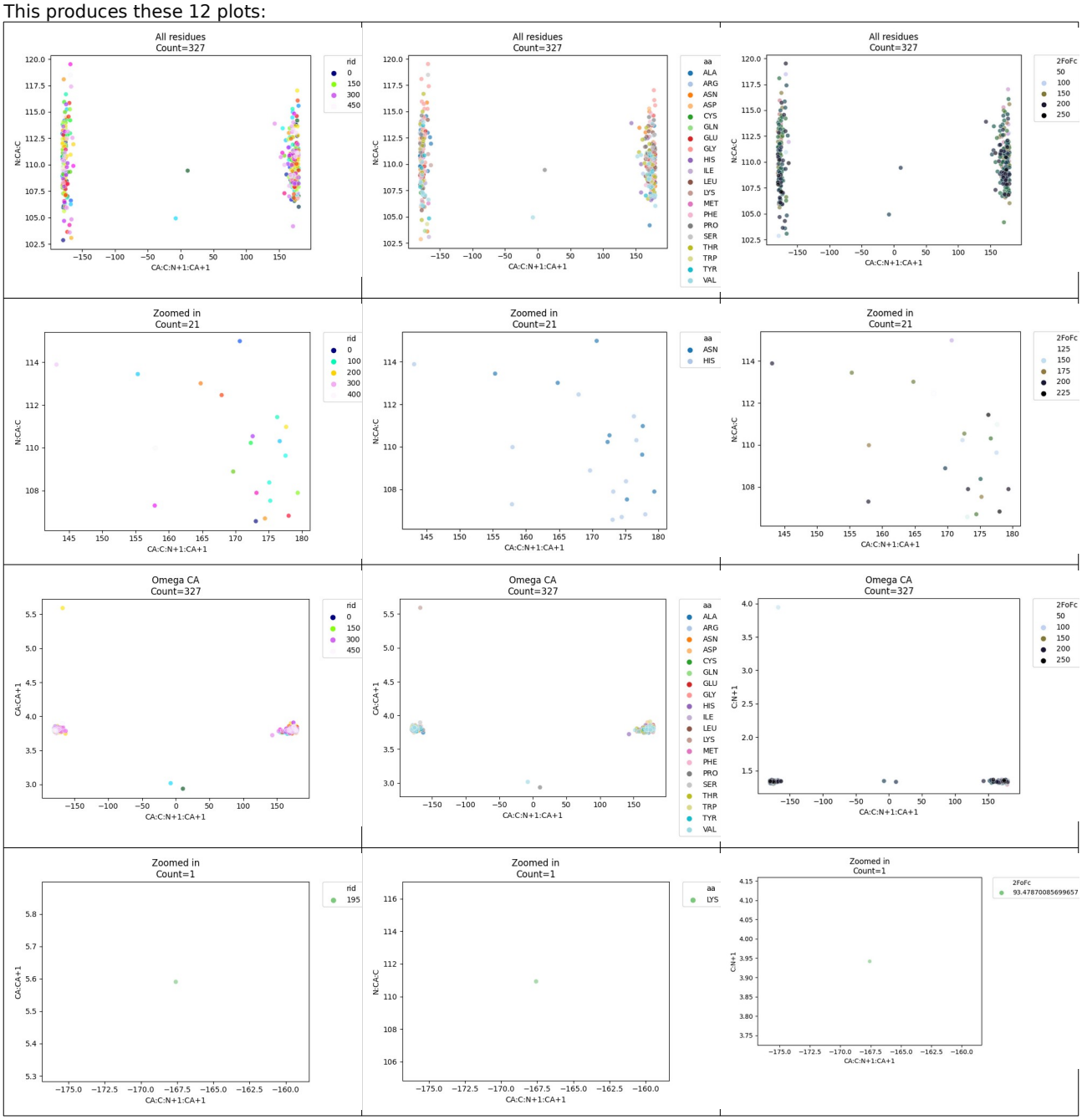
Pre Created Reports

An example of a hand-crafted report

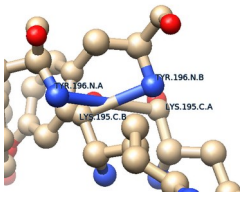
This report is for 2bw4 and taken from the literature in which the active site is reported to have a non-planar omega (Antonyuk et al, 2005; Berkholz et al, 2009)

```
import GeoReport as geor
import GeoPdb as geop

printPath = '???' #'wherever on your computer you wish to save the file'

# Create the GeoPdb object and the report object with just that single pdb
geoPdb = geop.GeoPdb('2bw4', pdbDataPath)
georep = geor.GeoReport([geoPdb])

# Choose the geometric calculations desired and the hues we might want to look at
geoList = ['CA:C:N+1:CA+1','N:CA:C','C:N+1','CA:CA+1']
hueList = ['dssp','aa','bfactor','2FoFc','rid'] # note the hues are the sum od the atoms
data = georep.getGeoemtryCsv(geoList, hueList)
printList = []
printList.append(['All residues',data,'CA:C:N+1:CA+1','N:CA:C','','rid','gist_ncar',False,0,0])
printList.append(['All residues', data, 'CA:C:N+1:CA+1', 'N:CA:C', '', 'aa', 'tab20', False, 0, 0])
printList.append(['All residues', data, 'CA:C:N+1:CA+1', 'N:CA:C', '', '2FoFc', 'cubehelix_r', False, 0, 0])

# Narrow down the data to a zoomed-in area around the residue of interest
riddata = data[data['CA:C:N+1:CA+1'] > 100]
riddata = riddata.query('aa =="HIS" or aa=="ASN"')
riddata = riddata.sort_values(by='rid', ascending=True)
printList.append(['Zoomed in',riddata,'CA:C:N+1:CA+1','N:CA:C','','rid','gist_ncar',False,0,0])
printList.append(['Zoomed in', riddata, 'CA:C:N+1:CA+1', 'N:CA:C', '', 'aa', 'tab20', False, 0, 0])
printList.append(['Zoomed in', riddata, 'CA:C:N+1:CA+1', 'N:CA:C', '', '2FoFc', 'cubehelix_r', False, 0, 0])

# Another validation check of omega against distances
printList.append(['Omega CA', data, 'CA:C:N+1:CA+1', 'CA:CA+1', '', 'rid', 'gist_ncar', False, 0, 0])
printList.append(['Omega CA', data, 'CA:C:N+1:CA+1', 'N:CA:C', '', 'aa', 'tab20', False, 0, 0])
printList.append(['Omega CA', data, 'CA:C:N+1:CA+1', 'C:N+1', '', '2FoFc', 'cubehelix_r', False, 0, 0])


# Narrow down the data to the residue of interest
serdata = data[data['CA:CA+1'] > 5]
printList.append(['Zoomed in', serdata, 'CA:C:N+1:CA+1', 'CA:CA+1', '', 'rid', 'Accent', False, 0, 0])
printList.append(['Zoomed in', serdata, 'CA:C:N+1:CA+1', 'CA:CA+1', '', 'aa', 'Accent', False, 0, 0])
printList.append(['Zoomed in', serdata, 'CA:C:N+1:CA+1', 'C:N+1', '', '2FoFc', 'Accent', False, 0, 0])

# And finally create the report with a file name of choice
georep.printCsvToHtml(printList,georep.pdbs,'Non Planar Omega in 2bw4',3,printPath,'2bw4_omega')
```

This produces these 12 plots:



Note that the non-planar omega HIS306 is highlighted as an active site in the deposition paper (Antonyuk et al, 2005) but the apparently ageometric LYS195 is not mentioned. Further analysis of structure can be performed quickly by clicking on the PDB or PDBe links in the generated report. Chimera analysis (reference) readily shows the bond between LYS.195.C and TYR.196.N is too long - a muddle of occupants suggests an error.



Managing memory problems

References

Antonyuk, S. V., Strange, R. W., Sawers, G., Eady, R. R., & Hasnain, S. S. (2005). Atomic resolution structures of resting-state, substrate- and product-complexed Cu-nitrite reductase provide insight into catalytic mechanism. Proceedings of the National Academy of Sciences of the United States of America, 102(34), 12041–12046. https://doi.org/10.1073/pnas.0504207102

Berkholz, D. S., Shapovalov, M. V., Dunbrack, R. L., & Karplus, P. A. (2009). Conformation Dependence of Backbone Geometry in Proteins. Structure, 17(10), 1316–1325. https://doi.org/10.1016/j.str.2009.08.012