

The Biopython Structural Bioinformatics FAQ

Bioinformatics center
Institute of Molecular Biology
University of Copenhagen
Universitetsparken 15, Bygning 10
DK-2100 København Ø
Denmark
thamelry@binf.ku.dk
<http://www.binf.ku.dk/users/thamelry/>

1 Introduction

The Biopython Project is an international association of developers of freely available Python (<http://www.python.org>) tools for computational molecular biology. Python is an obinterced,12356(inp6(fr)-ced)1(,1-34103exiilabter)-languagabterisygningrfreelyris2is2syntax(is2)-

- mmLib: <http://pymmlib.sourceforge.net/>
- VMD: <http://www.ks.uiuc.edu/Research/vmd/>
- MMTK: <http://starship.python.net/crew/hinsen/MMTK/>

```
resolution=structure.header['resolution']  
keywords=structure.header['keywords']
```

The available keys are name, head, deposition_date, release_date, structure_method, resolution, structure_reference (**maps to a list of references**), journal_reference, author **and** compound (**maps to a dictionary with various information about the crystallized compound**).


```
class GlySelect(Select):
    def accept_residue(self, residue):
        if residue.get_name() == 'GLY':
            return 1
        else:
            return 0
io=PDBIO()
io.set_structure(s)
io.save('gly_only.pdb', GlySelect())
```

If this is all too complicated for you, the Dice

residues and atoms. The philosophy of Bio.PDB is to provide a reasonably fast, clean,

How do I measure torsion angles?

Again, this can easily be done via the vector representation of the atomic coordinates, this time using the `calc_dihedral` function from the `Vector` module:

```
vector1=atom1.get_vector()
vector2=atom2.get_vector()
vector3=atom3.get_vector()
vector4=atom4.get_vector()
angle=calc_dihedral(vector1, vector2, vector3, vector4)
```

How do I determine atom-atom contacts?

Use `NeighborSearch`. This uses a KD tree data structure coded in C++ behind the screens, so it's pretty darn fast (see `Bio.KDTree`).

How do I extract polypeptides from a Structure object?

Use `PolypeptideBuilder`. You can use the resulting `Polypeptide` object to get the sequence as a `Seq` object or to get a list of `C` atoms as well. Polypeptides can be built using a C-N or a C-C distance criterion.

Example:

```
# Using C-N
ppb=PPBuilder()
for pp in ppb.build_peptides(structure):
    print pp.get_sequence()
# Using CA-CA
ppb=CaPPBuilder()
for pp in ppb.build_peptides(structure):
    print pp.get_sequence()
```

Note that in the above case only model 0 of the structure is considered by `PolypeptideBuilder`.


```
print sup.rms
# Apply rotation/translation to the moving atoms
sup.apply(moving)
```