# Class 10 Structural Bioinformatics

Emily Chen (PID:A16925878)

## Table of contents



### The PDB Database

The main repository of biomolecular structure data is called the Protein Data Bank (PDB). It is the second oldest database (after GenBank)

(https://www.rcsb.org/)

What is currently in the PDB? We can access current composition stats here

```
stats <- read.csv("Data Export Summary.csv", row.names=1)
head(stats)
```

```
                        X.ray     EM    NMR Multiple.methods Neutron Other
Protein (only)        171,959 18,083 12,622              210      84    32
Protein/Oligosaccharide  10,018  2,968     34               10       2     0
Protein/NA                8,847  5,376    286                7       0     0
Nucleic acid (only)       2,947    185  1,535               14       3     1
Other                       170     10     33                0       0     0
Oligosaccharide (only)       11      0      6                1       0     4
                        Total
Protein (only)        202,990
Protein/Oligosaccharide  13,032
Protein/NA               14,516
Nucleic acid (only)       4,685
Other                       213
Oligosaccharide (only)       22
```

```
202990/252188522 *100
```

```
[1] 0.08049137
```

Q1: What percentage of structures in the PDB are solved by X-Ray and Electron Microscopy.

```
stats$X.ray
```

```
[1] "171,959" "10,018"  "8,847"   "2,947"   "170"     "11"
```

```
as.numeric(stats$X.ray)
```

```
Warning: NAs introduced by coercion
```

```
[1]  NA  NA  NA  NA 170  11
```

we can see here that when we run the code `as.numeric(stats$X.ray)` we get NA for the first four values. The reason for this is because in the first four values there are commas and in the last two 170 and 11 there are no commas. So we need to get rid of these commas and the way we do that is by using 'gsub(). The first argument is what you want ot get rid of, the second argument is what you want to repalce it with If nothing, then just have quotation marks, then the third argument is the function with which you want to work.

```
x<-stats$X.ray

  #Substitute commas for nothing
  y<-gsub(",", "", x)

  # convert to numeric
  sum(as.numeric(y))
```

```
[1] 193952
```

Turn this snippet into a functions so I can use it any time I have a comma problem (i.e the other columns of this `stats` table)

```
comma.sum <- function(x){
  y<- gsub(",", "", x)
  return( sum(as.numeric(y)))
}
```

```
xray.sum <- comma.sum(stats$X.ray)
em.sum <- comma.sum(stats$EM)
total.sum <- comma.sum (stats$Total)
```

```
xray.sum/total.sum*100
```

```
[1] 82.37223
```

Q2: What proportion of structures in the PDB are protein?

```
protein.sum <- comma.sum(stats["Protein (only)", "Total"])
protein.sum/total.sum *100
```

```
[1] 86.2107
```

```
sum(stats$Neutron)
```

```
[1] 89
```

```
em.sum/total.sum*100
```

```
[1] 11.30648
```

Q3: Type HIV in the PDB website search box on the home page and determine how many HIV-1 protease structures are in the current PDB?

Skipped

##2 Visualizing with Mol-Star

Explore the HIV-1 protease strcuture with PDB code: `1HSG` Mol-star homepage at https://molestar.org/viewer/

The code that is used to insert an image is `![]()` inside the bracket. You will put your caption in here, and then for the other parentheses, we will add the name of the file of the image. The code for the image below is `![Figure 1. A first view of HIV-Pr] (1HSG.png)`

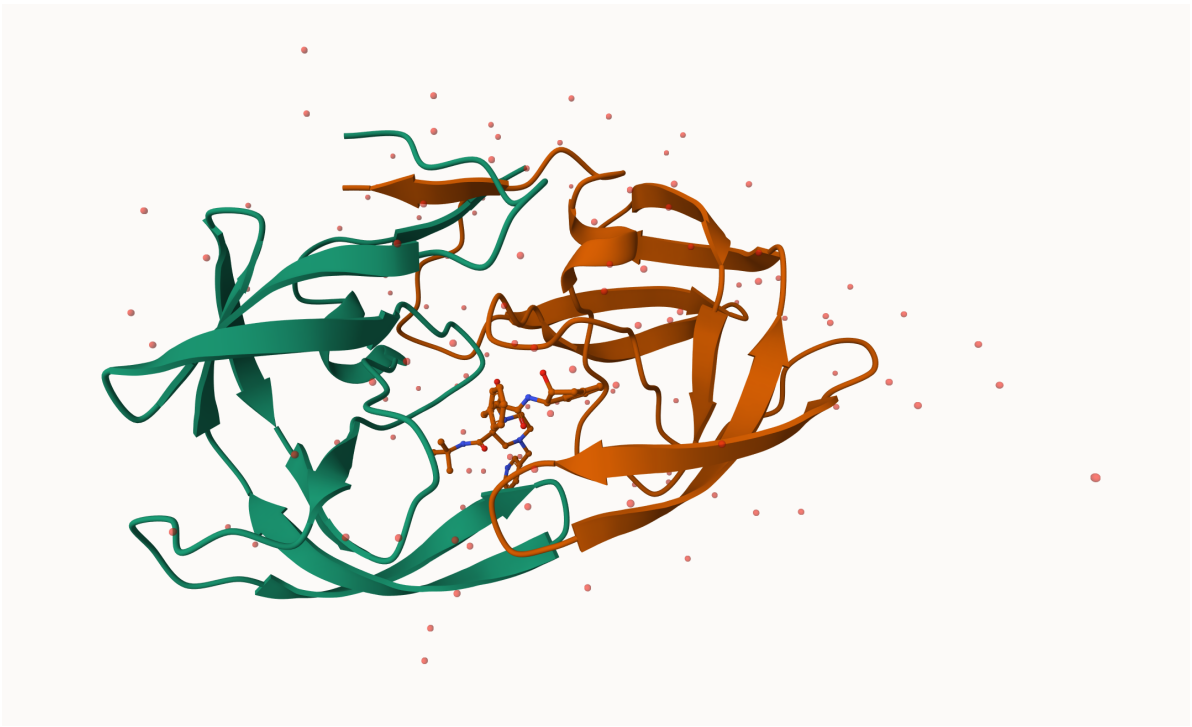


Figure 1: Figure 1. A first view of HIV-Pr

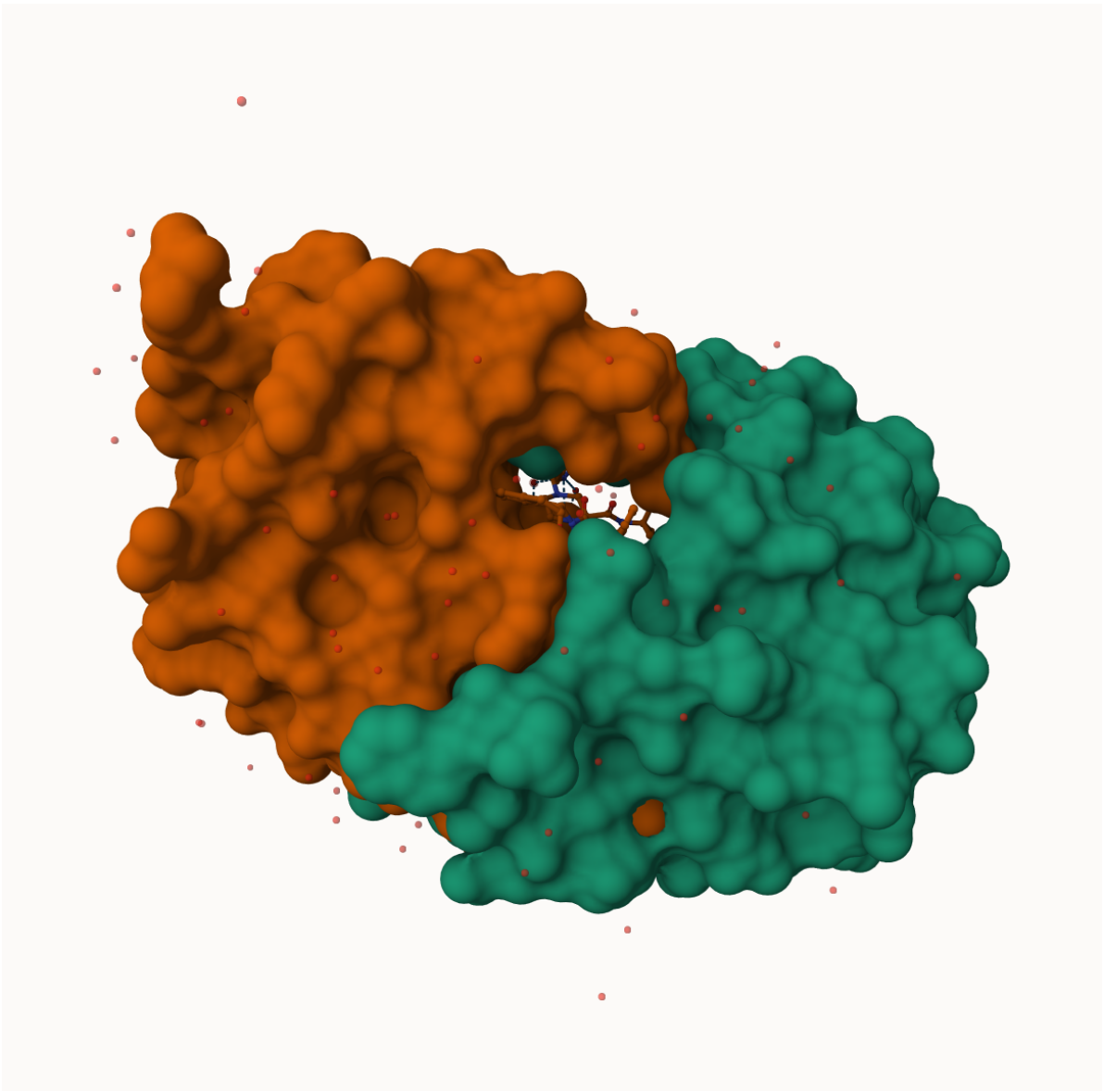


Figure 2: Figure 2. Molecular surface showing binding cavity

Figure 3: Figure 3. The Catatiliically important ASP 25 amino acids and drug interacting HOH 308 water molecule

## 3. Using the bio3d package in R

The Bio3D package is focused on structural bioinformatics analysis and allows us to read and analyse PDB and related data

```
library(bio3d)
```

```
pdb<- read.pdb("1hsg")
```

```
  Note: Accessing on-line PDB file
```

```
pdb
```

```
 Call:  read.pdb(file = "1hsg")

   Total Models#: 1
     Total Atoms#: 1686,  XYZs#: 5058  Chains#: 2  (values: A B)

     Protein Atoms#: 1514  (residues/Calpha atoms#: 198)
     Nucleic acid Atoms#: 0  (residues/phosphate atoms#: 0)

     Non-protein/nucleic Atoms#: 172  (residues: 128)
     Non-protein/nucleic resid values: [ HOH (127), MK1 (1) ]

   Protein sequence:
      PQITLWQRPLVTIKIGGQLKEALLDTGADDTVLEEMSLPGRWKPKMIGGIGGFIKVRQYD
      QILIEICGHKAIGTVLVGPTPVNIIGRNLLTQIGCTLNFPQITLWQRPLVTIKIGGQLKE
      ALLDTGADDTVLEEMSLPGRWKPKMIGGIGGFIKVRQYDQILIEICGHKAIGTVLVGPTP
      VNIIGRNLLTQIGCTLNF

+ attr: atom, xyz, seqres, helix, sheet,
        calpha, remark, call
```

```
attributes(pdb)
```

```
$names
[1] "atom"   "xyz"    "seqres" "helix"  "sheet"  "calpha" "remark" "call"

$class
[1] "pdb" "sse"
```

We can see atom data wtih `pdb$atom`

```
head(pdb$atom)
```

```
  type eleno elety  alt resid chain resno insert      x      y     z o     b
1 ATOM     1     N <NA>   PRO     A     1   <NA> 29.361 39.686 5.862 1 38.10
2 ATOM     2    CA <NA>   PRO     A     1   <NA> 30.307 38.663 5.319 1 40.62
3 ATOM     3     C <NA>   PRO     A     1   <NA> 29.760 38.071 4.022 1 42.64
4 ATOM     4     O <NA>   PRO     A     1   <NA> 28.600 38.302 3.676 1 43.40
5 ATOM     5    CB <NA>   PRO     A     1   <NA> 30.508 37.541 6.342 1 37.87
6 ATOM     6    CG <NA>   PRO     A     1   <NA> 29.296 37.591 7.162 1 38.40
  segid elesy charge
```

```
1  <NA>     N   <NA>
2  <NA>     C   <NA>
3  <NA>     C   <NA>
4  <NA>     O   <NA>
5  <NA>     C   <NA>
6  <NA>     C   <NA>
```

```
head(pdbseq(pdb))
```

```
  1   2   3   4   5   6
"P" "Q" "I" "T" "L" "W"
```

We can make quick 3D viz with the `view.pdb`

```
library(bio3dview)
library(NGLVieweR)

#view.pdb(pdb, backgroundColor= "skyblue", colorScheme= "sse") |>
#setSpin()
```

```
library(bio3d)
#sel <- atom.select(pdb, resno=25)
#view.pdb(pdb, highlight=sel,
        # highlight.style = "spacefill") |>
  #setRock()
```

### Prediciting functional motion os a single structure

We can finish off today with a bioinformatic prediction of the functional motions of a protein.

We will run a Normal Mode Analysis (NMA)

```
adk <- read.pdb("6s36")
```

```
  Note: Accessing on-line PDB file
   PDB has ALT records, taking A only, rm.alt=TRUE
```

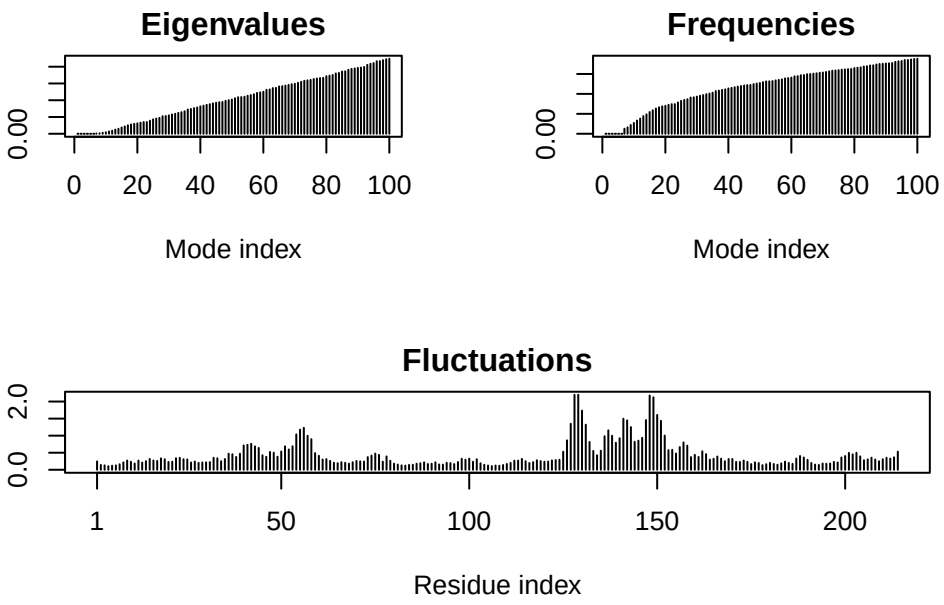
```
adk
```

```
 Call:  read.pdb(file = "6s36")

   Total Models#: 1
     Total Atoms#: 1898,  XYZs#: 5694  Chains#: 1  (values: A)

     Protein Atoms#: 1654  (residues/Calpha atoms#: 214)
     Nucleic acid Atoms#: 0  (residues/phosphate atoms#: 0)

     Non-protein/nucleic Atoms#: 244  (residues: 244)
     Non-protein/nucleic resid values: [ CL (3), HOH (238), MG (2), NA (1) ]

   Protein sequence:
      MRIILLGAPGAGKGTQAQFIMEKYGIPQISTGDMLRAAVKSGSELGKQAKDIMDAGKLVT
      DELVIALVKERIAQEDCRNGFLLDGFPRTIPQADAMKEAGINVDYVLEFDVPDELIVDKI
      VGRRVHAPSGRVYHVKFNPPKVEGKDDVTGEELTTRKDDQEETVRKRLVEYHQMTAPLIG
      YYSKEAEAGNTKYAKVDGTKPVAEVRADLEKILG

+ attr: atom, xyz, seqres, helix, sheet,
        calpha, remark, call
```

```
m<- nma(adk)
```

```
 Building Hessian...        Done in 0.018 seconds.
 Diagonalizing Hessian...   Done in 0.382 seconds.
```

```
plot(m)
```




```
#view.nma(m)
```

We can write out a trajectory of the predicetd dynamic and iew this in Mol-star

```
mktrj(m, file="nma.pdb")
```

A file called nma.pdb will be saved into class 10 project and now we can go to molstar and look at the sequence