Class 09 Structural Bioinformatics part 1

Carolina Merino (PID 14484883)

1: Introduction to the RCSB Protein Data Bank (PDB)

The main database for structural biology is called PDB. Let's have a look at what it contains:

Download a CSV file from the PDB site

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

Around **81.5%** of structures are solved by X-ray and **12.2%** are solved by Electron Microscopy.

```
library(readr)


# Read CSV
pdb_data <- read_csv("Data Export Summary.csv")
head(pdb_data)
```

```
# A tibble: 6 x 9
  `Molecular Type`    `X-ray`    EM   NMR Integrative `Multiple methods` Neutron
  <chr>                 <dbl> <dbl> <dbl>       <dbl>               <dbl>   <dbl>
1 Protein (only)       176204 20299 12708         342                 218      83
2 Protein/Oligosacch~   10279  3385    34           8                  11       1
3 Protein/NA             9007  5897   287          24                   7       0
4 Nucleic acid (only)    3066   200  1553           2                  15       3
5 Other                   173    13    33           3                   0       0
6 Oligosaccharide (o~      11     0     6           0                   1       0
# i 2 more variables: Other <dbl>, Total <dbl>
```

```
# Step 1: Clean numeric columns (remove commas if they exist)
pdb_data$`X-ray` <- as.numeric(gsub(",", "", pdb_data$`X-ray`))
pdb_data$EM <- as.numeric(gsub(",", "", pdb_data$EM))
pdb_data$Total <- as.numeric(gsub(",", "", pdb_data$Total))


# Step 2: Calculate total of all structures
total_all <- sum(pdb_data$Total, na.rm = TRUE)


# Step 3: Calculate percentage for X-ray and EM
xray_total <- sum(pdb_data$`X-ray`, na.rm = TRUE)
xray_percent <- (xray_total / total_all) * 100


em_total <- sum(pdb_data$EM, na.rm = TRUE)
em_percent <- (em_total / total_all) * 100


# Step 4: Show results
xray_percent
```

```
[1] 81.48087
```

```
em_percent
```

```
[1] 12.21516
```

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

Around **86%** of the structures in the PDB are proteins.

```
# Protein percentage round((stats$Total[1] / total_all) * 100)




protein_percent <- 86
protein_percent
```

```
[1] 86
```

> 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?

According to the data 1,091 HIV-1 protease structures in the PDB.

2. Visualizing the HIV-1 protease structure

> Q4: Water molecules normally have 3 atoms. Why do we see just one atom per water molecule in this structure?

With the structure we only see one atom per water molecule in this structure because hydrogen atoms are too small to be detected by x-ray crystallography, so only the one oxygen atom is visible.

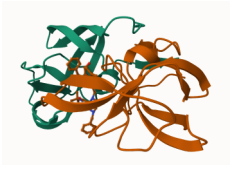
> Q5: There is a critical "conserved" water molecule in the binding site. Can you identify this water molecule? What residue number does this water molecule have?

We can see that the conserved water molecule in the binding site is HOH 307, located in chain D, which interacts with the active site residues.

> Q6: Generate and save a figure clearly showing the two distinct chains of HIV-protease along with the ligand. You might also consider showing the catalytic residues ASP 25 in each chain and the critical water (we recommend "Ball & Stick" for these side-chains). Add this figure to your Quarto document.

Follow the figure below.

```
knitr::include_graphics("1HSG.PNG")
```

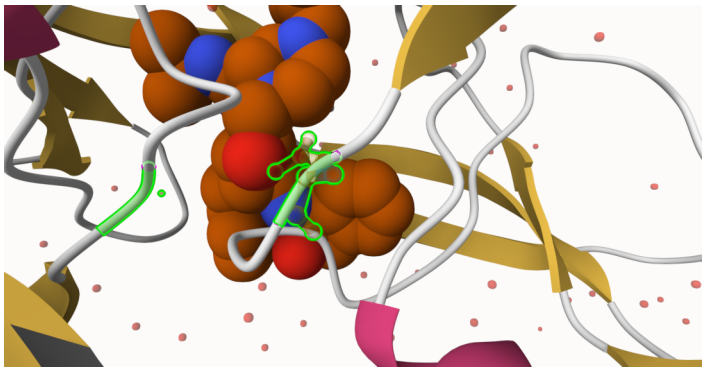


Deeper Look..

```
knitr::include_graphics("1HSG6.PNG")
```

Closer View..

```
knitr::include_graphics("1HSGQ5.PNG")
```



3. Introduction to Bio3D in R

```
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
```

Q7: How many amino acid residues are there in this pdb object?

We can see that there are **198** AA residues in this pdb project.

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

```
  Note: Accessing on-line PDB file

Warning in get.pdb(file, path = tempdir(), verbose = FALSE):
C:\Users\LINDAK~1\AppData\Local\Temp\RtmpWwF493/1hsg.pdb exists. Skipping
download
```

```
hiv
```

```
 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
```

```
length(unique(hiv$atom$resno))
```

```
[1] 227
```

```
length(pdbseq(hiv))
```

```
[1] 198
```

Q8: Name one of the two non-protein residues?

One of the two non-protein residues is named as: "MK1"

```
unique(hiv$atom$resid[hiv$atom$type=="HETATM"])
```

```
[1] "MK1" "HOH"
```

Q9: How many protein chains are in this structure?

There are **2 protein chains** in this structure.

```
# Number of protein chains in the PDB
length(unique(hiv$atom$chain))
```

```
[1] 2
```

4. Comparative structure analysis of Adenylate Kinase

We can see the source species for each PDB, which can be very useful for clustering or grouping in PCA.

```
hits <- NULL
hits$pdb.id <- c('1AKE_A','6S36_A','6RZE_A','3HPR_A','1E4V_A','5EJE_A',
                 '1E4Y_A','3X2S_A','6HAP_A','6HAM_A','4K46_A','3GMT_A','4PZL_A')
```

Q10. Which of the packages above is found only on BioConductor and not CRAN?

We see that "msa: is found on BioConductor, not CRAN.

Q11. Which of the above packages is not found on BioConductor or CRAN?:

"bio3dview" cannot be found on either CRAN or BioConductor.

Q12. True or False? Functions from the pak package can be used to install packages from GitHub and BitBucket?

True

##Search and retrieve ADK structures

We perform a BLAST search of the PDB to find structures related to Adenylate Kinase (ADK). Here, `get.seq()` fetches the chain A sequence of PDB ID 1AKE, which is used as input for `blast.pdb()`

```
library(bio3d)
aa <- get.seq("1ake_A")
```

```
Warning in get.seq("1ake_A"): Removing existing file: seqs.fasta
```

```
Fetching... Please wait. Done.
```

```
aa
```

```
              1        .         .         .         .         .         60
pdb|1AKE|A    MRIILLGAPGAGKGTQAQFIMEKYGIPQISTGDMLRAAVKSGSELGKQAKDIMDAGKLVT
              1        .         .         .         .         .         60

             61        .         .         .         .         .         120
pdb|1AKE|A    DELVIALVKERIAQEDCRNGFLLDGFPRTIPQADAMKEAGINVDYVLEFDVPDELIVDRI
             61        .         .         .         .         .         120

            121        .         .         .         .         .         180
pdb|1AKE|A    VGRRVHAPSGRVYHVKFNPPKVEGKDDVTGEELTTRKDDQEETVRKRLVEYHQMTAPLIG
            121        .         .         .         .         .         180

            181        .         .         .   214
pdb|1AKE|A    YYSKEAEAGNTKYAKVDGTKPVAEVRADLEKILG
            181        .         .         .   214
```

```
Call:
  read.fasta(file = outfile)

Class:
  fasta

Alignment dimensions:
  1 sequence rows; 214 position columns (214 non-gap, 0 gap)

+ attr: id, ali, call
```

Q13. How many amino acids are in this sequence, i.e. how long is this sequence?

In this sequence, there are **214 amino acids**.

```
# b <- blast.pdb(aa)  # aa is your ADK sequence from earlier
```

Now we can visualize the Blast

```
# Visualize the BLAST hits
#plot.blast(b)
```

Filter the hits, this makes sure we filter out the hits

```
# Check the top hits
# head(b)       # shows first few hits
# length(b)     # shows total number of hits
```

```
hits <- NULL
hits$pdb.id <- c('1AKE_A','6S36_A','6RZE_A','3HPR_A','1E4V_A','5EJE_A','1E4Y_A','3X2S_A','6H
```

```
files <- get.pdb(hits$pdb.id, path="pdbs", split=TRUE, gzip=TRUE)
```

```
Warning in get.pdb(hits$pdb.id, path = "pdbs", split = TRUE, gzip = TRUE):
pdbs/1AKE.pdb exists. Skipping download

Warning in get.pdb(hits$pdb.id, path = "pdbs", split = TRUE, gzip = TRUE):
pdbs/6S36.pdb exists. Skipping download

Warning in get.pdb(hits$pdb.id, path = "pdbs", split = TRUE, gzip = TRUE):
pdbs/6RZE.pdb exists. Skipping download
```

```
Warning in get.pdb(hits$pdb.id, path = "pdbs", split = TRUE, gzip = TRUE):
pdbs/3HPR.pdb exists. Skipping download

Warning in get.pdb(hits$pdb.id, path = "pdbs", split = TRUE, gzip = TRUE):
pdbs/1E4V.pdb exists. Skipping download

Warning in get.pdb(hits$pdb.id, path = "pdbs", split = TRUE, gzip = TRUE):
pdbs/5EJE.pdb exists. Skipping download

Warning in get.pdb(hits$pdb.id, path = "pdbs", split = TRUE, gzip = TRUE):
pdbs/1E4Y.pdb exists. Skipping download

Warning in get.pdb(hits$pdb.id, path = "pdbs", split = TRUE, gzip = TRUE):
pdbs/3X2S.pdb exists. Skipping download

Warning in get.pdb(hits$pdb.id, path = "pdbs", split = TRUE, gzip = TRUE):
pdbs/6HAP.pdb exists. Skipping download

Warning in get.pdb(hits$pdb.id, path = "pdbs", split = TRUE, gzip = TRUE):
pdbs/6HAM.pdb exists. Skipping download

Warning in get.pdb(hits$pdb.id, path = "pdbs", split = TRUE, gzip = TRUE):
pdbs/4K46.pdb exists. Skipping download

Warning in get.pdb(hits$pdb.id, path = "pdbs", split = TRUE, gzip = TRUE):
pdbs/3GMT.pdb exists. Skipping download

Warning in get.pdb(hits$pdb.id, path = "pdbs", split = TRUE, gzip = TRUE):
pdbs/4PZL.pdb exists. Skipping download


  |
  |                                                                      |   0%
  |
  |=====                                                                 |   8%
  |
  |===========                                                           |  15%
  |
  |================                                                      |  23%
```

```
  |
  |======================                                                |  31%
  |
  |=========================                                             |  38%
  |
  |===============================                                       |  46%
  |
  |====================================                                  |  54%
  |
  |==========================================                            |  62%
  |
  |================================================                      |  69%
  |
  |======================================================                |  77%
  |
  |============================================================          |  85%
  |
  |==================================================================    |  92%
  |
  |======================================================================| 100%
```

```
pdbs <- pdbaln(files, fit = TRUE, exefile="msa")  # aligns structures
```

```
Reading PDB files:
pdbs/split_chain/1AKE_A.pdb
pdbs/split_chain/6S36_A.pdb
pdbs/split_chain/6RZE_A.pdb
pdbs/split_chain/3HPR_A.pdb
pdbs/split_chain/1E4V_A.pdb
pdbs/split_chain/5EJE_A.pdb
pdbs/split_chain/1E4Y_A.pdb
pdbs/split_chain/3X2S_A.pdb
pdbs/split_chain/6HAP_A.pdb
pdbs/split_chain/6HAM_A.pdb
pdbs/split_chain/4K46_A.pdb
pdbs/split_chain/3GMT_A.pdb
pdbs/split_chain/4PZL_A.pdb
   PDB has ALT records, taking A only, rm.alt=TRUE
.   PDB has ALT records, taking A only, rm.alt=TRUE
.   PDB has ALT records, taking A only, rm.alt=TRUE
.   PDB has ALT records, taking A only, rm.alt=TRUE
..   PDB has ALT records, taking A only, rm.alt=TRUE
```

```
....   PDB has ALT records, taking A only, rm.alt=TRUE
.   PDB has ALT records, taking A only, rm.alt=TRUE
...

Extracting sequences

pdb/seq: 1   name: pdbs/split_chain/1AKE_A.pdb
   PDB has ALT records, taking A only, rm.alt=TRUE
pdb/seq: 2   name: pdbs/split_chain/6S36_A.pdb
   PDB has ALT records, taking A only, rm.alt=TRUE
pdb/seq: 3   name: pdbs/split_chain/6RZE_A.pdb
   PDB has ALT records, taking A only, rm.alt=TRUE
pdb/seq: 4   name: pdbs/split_chain/3HPR_A.pdb
   PDB has ALT records, taking A only, rm.alt=TRUE
pdb/seq: 5   name: pdbs/split_chain/1E4V_A.pdb
pdb/seq: 6   name: pdbs/split_chain/5EJE_A.pdb
   PDB has ALT records, taking A only, rm.alt=TRUE
pdb/seq: 7   name: pdbs/split_chain/1E4Y_A.pdb
pdb/seq: 8   name: pdbs/split_chain/3X2S_A.pdb
pdb/seq: 9   name: pdbs/split_chain/6HAP_A.pdb
pdb/seq: 10   name: pdbs/split_chain/6HAM_A.pdb
   PDB has ALT records, taking A only, rm.alt=TRUE
pdb/seq: 11   name: pdbs/split_chain/4K46_A.pdb
   PDB has ALT records, taking A only, rm.alt=TRUE
pdb/seq: 12   name: pdbs/split_chain/3GMT_A.pdb
pdb/seq: 13   name: pdbs/split_chain/4PZL_A.pdb
```

```
# ids <- basename.pdb(pdbs$id)                          # get PDB IDs
# png("pdb_alignment.png", width=1200, height=900)  # adjust resolution/size if needed
# par(mar=c(4,4,2,2))                               # reduce margins to avoid plotting errors
# plot(pdbs, labels=ids)
# dev.off()                          #
```

Now we annotate structures

```
# anno <- pdb.annotate(ids)
# unique(anno$source)   # see which organisms they came from
```

PCA structural variations

The `pca()` function performs principal component analysis (PCA) on structural data. PCA is a statistical method that reduces a large dataset to a few key components that capture

the main directions of variation. For protein structures, PCA highlights the major structural differences across a group of related structures. You can run PCA on a set of structures stored in the pdbs object using either `pca.xyz()` or the simpler `pca()` function.

```
# pc.xray <- pca(pdbs)
# plot(pc.xray)
```

Each dot = one PDB structure. Shows conformational differences.

Let's calculate RMSD

```
# Calculate RMSD
rd <- rmsd(pdbs)
```

```
Warning in rmsd(pdbs): No indices provided, using the 204 non NA positions
```

```
# Structure-based clustering
hc.rd <- hclust(dist(rd))
grps.rd <- cutree(hc.rd, k=3)


#plot(pc.xray, 1:2, col="grey50", bg=grps.rd, pch=21, cex=1)
```

Now for clustering...

```
# rd <- rmsd(pdbs)                    # pairwise RMSD matrix
# hc.rd <- hclust(dist(rd))           # hierarchical clustering
# grps.rd <- cutree(hc.rd, k=3)       # divide into 3 groups


# plot PCA colored by clusters
# library(ggplot2)
# library(ggrepel)
# df <- data.frame(PC1=pc.xray$z[,1],
                #  PC2=pc.xray$z[,2],
               #   col=as.factor(grps.rd),
                  # ids=ids)
#ggplot(df) +
 # aes(PC1, PC2, col=col, label=ids) +
 # geom_point(size=2) +
 # geom_text_repel(max.overlaps = 20) +
 # theme(legend.position = "none")
```

6. Normal mode analysis *optional section*

```
# NMA of all structures
modes <- nma(pdbs)
```

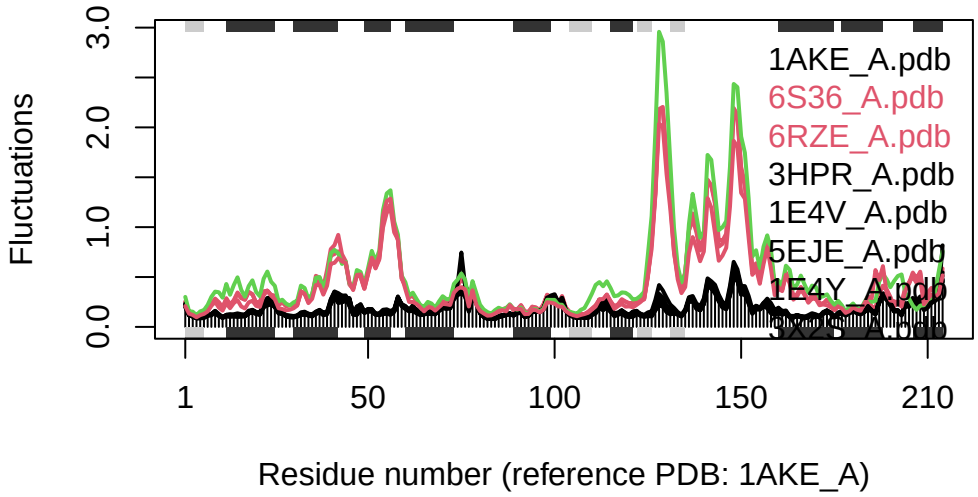
```
Details of Scheduled Calculation:
  ... 13 input structures
  ... storing 606 eigenvectors for each structure
  ... dimension of x$U.subspace: ( 612x606x13 )
  ... coordinate superposition prior to NM calculation
  ... aligned eigenvectors (gap containing positions removed)
  ... estimated memory usage of final 'eNMA' object: 36.9 Mb
```

```
  |
  |                                                                      |   0%
  |
  |=====                                                                 |   8%
  |
  |===========                                                           |  15%
  |
  |================                                                      |  23%
  |
  |======================                                                |  31%
  |
  |============================                                          |  38%
  |
  |==================================                                    |  46%
  |
  |========================================                              |  54%
  |
  |=============================================                         |  62%
  |
  |=================================================                     |  69%
  |
  |=========================================================             |  77%
  |
  |=============================================================         |  85%
  |
  |=================================================================     |  92%
```

```
  |
  |======================================================================| 100%
```

```
plot(modes, pdbs, col=grps.rd)
```

```
Extracting SSE from pdbs$sse attribute
```



Q14. What do you note about this plot? Are the black and colored lines similar or different? Where do you think they differ most and why?

Fluctuations of the reference protein are shown by the black line, while the colored lines represent other aligned structures. The overall patterns are similar, but differences appear in the peak regions, especially around residues *35–55 and 130–150*, likely reflecting flexible regions like loops or terminal ends that vary between structures.