lab 10: structural bioinformatics (part 1)

Torrey Rhyne (A14397504)

Introduction to PDB

PDB Statistics.

Download csv file from PDB (accessible from "Analyze" > "PDB Statistics" > "by Experimental Method and Molecular Type"):

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

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

```
(sum(pdb_stats$X.ray) + sum(pdb_stats$EM)) / sum(pdb_stats$Total)
```

```
[1] 0.9315962
```

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

```
# protein only
pdb_stats[1,7] / sum(pdb_stats[,7])
```

```
[1] 0.8667026
```

```
# protein / oligosaccharide or NA
sum(pdb_stats[1:3,7]) / sum(pdb_stats[,7])
```

```
[1] 0.9784556
```

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? (skip)

PDB format. Download the "PDB File" for the HIV-1 protease structure with the PDB identifier 1HSG and view in terminal.

Visualizing the HIV-1 protease structure with Mol*

Image of HIV-Pr that is not very useful yet.

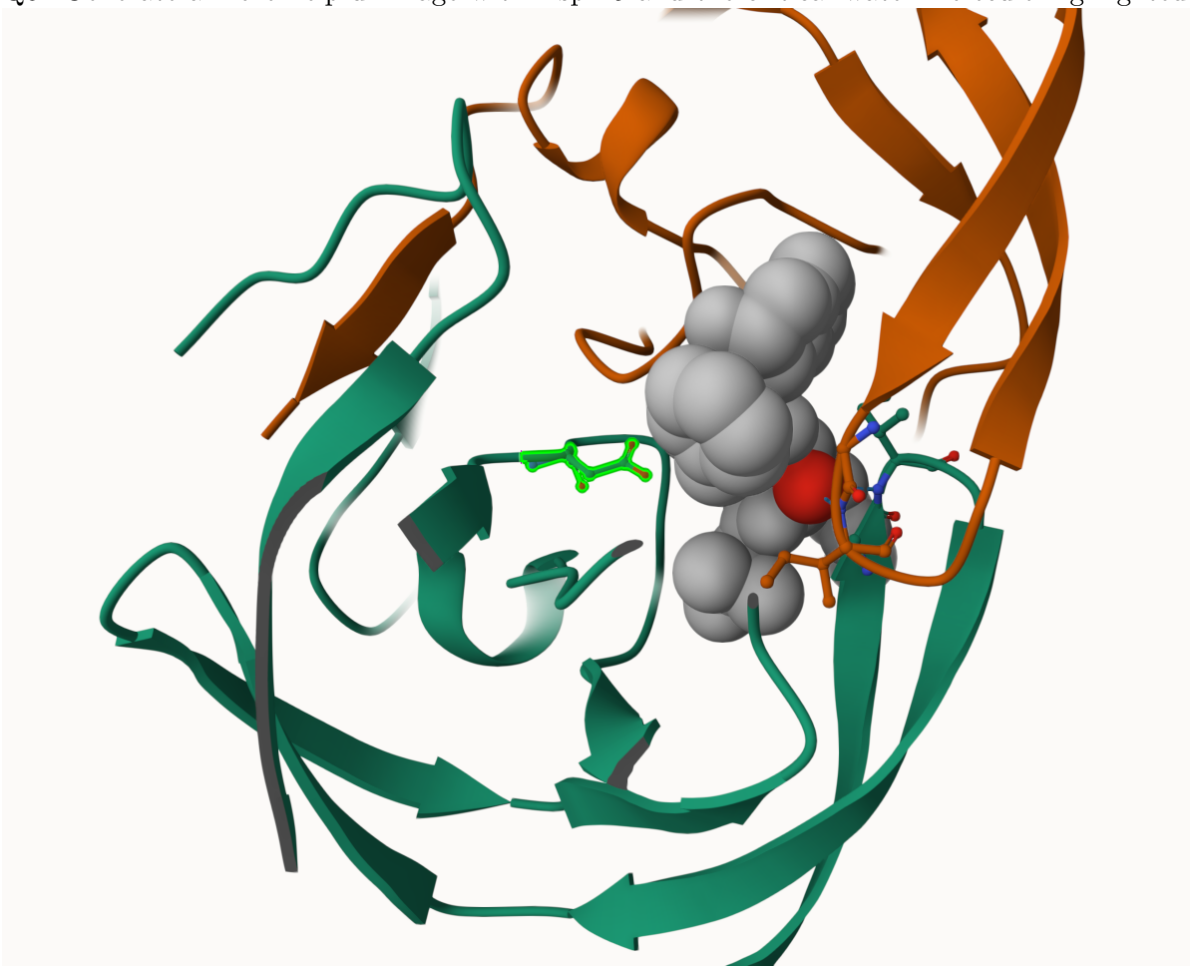
Highlight Asp 25 (D25) residue.

Introduction to Bio3D in R

Q4: Water molecules normally have 3 atoms. Why do we see just one atom per water molecule in this structure? Hydrogen atoms are too small for the resolution of this structure (2 A).

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? 308

Q6: Generate a more helpful image with Asp 25 and the critical water molecule highlighted.



Asp25 = bright green Ligand = grey critical water = red ball

Introduction to Bio3D in R

Load package:

```
library(bio3d)
```

Load pdb file:

```
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? 198

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

Q9: How many protein chains are in this structure? 2

Attributes:

```
attributes(pdb)
```

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

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

```
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>
```

Predicting functional motions of a single molecule. Run a normal mode analysis (NMA) - a bioinformatics method to predict functional motions.

Load new pdb file:

```
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
```

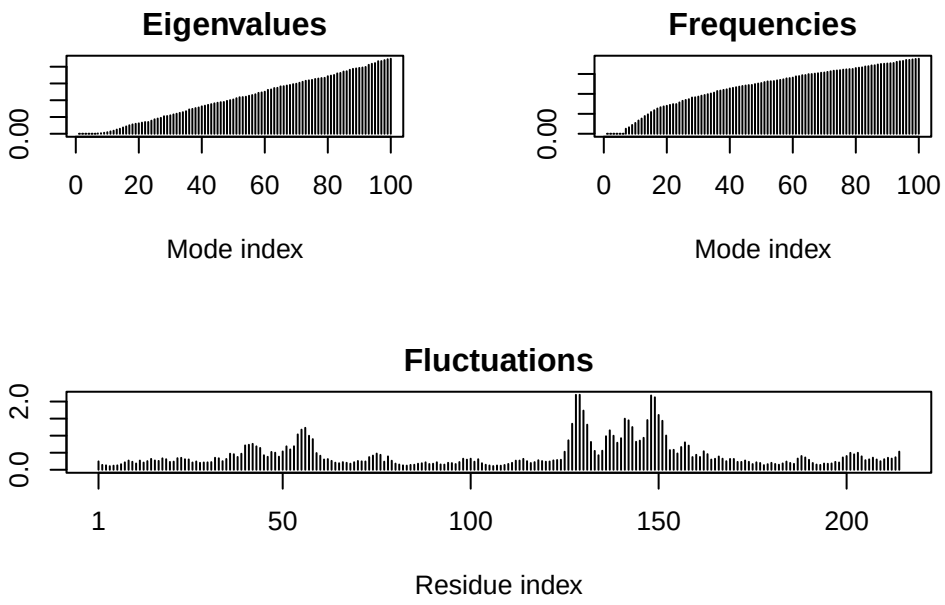
```
     Total Atoms#: 1898,  XYZs#: 5694  Chains#: 1  (values: A)

     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
```

Perform flexibility prediction:

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

```
 Building Hessian...        Done in 0.013 seconds.
 Diagonalizing Hessian...   Done in 0.255 seconds.
```

```
plot(m)
```



Make a movie:

```
mktrj(m, file="adk_m7.pdb")
# load this file into Mol*
```

Comparative structure analysis of Adenylate Kinase

Starting from only one Adk PDB identifier (PDB ID: 1AKE) we will search the entire PDB for related structures using BLAST, fetch, align and superpose the identified structures, perform PCA and finally calculate the normal modes of each individual structure in order to probe for potential differences in structural flexibility.

Setup:

```
# Install packages in the R console NOT your Rmd/Quarto file
# install.packages("bio3d")
# install.packages("devtools")
# install.packages("BiocManager")
# BiocManager::install("msa")
# devtools::install_bitbucket("Grantlab/bio3d-view")
```

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

Q11. Which of the above packages is not found on BioConductor or CRAN? bio3d-view

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

Search and retrieve ADK structures. Below we perform a blast search of the PDB database to identify related structures to our query Adenylate kinase (ADK) sequence. In this particular example we use function get.seq() to fetch the query sequence for chain A of the PDB ID 1AKE and use this as input to blast.pdb(). Note that get.seq() would also allow the corresponding UniProt identifier.

```
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? 214

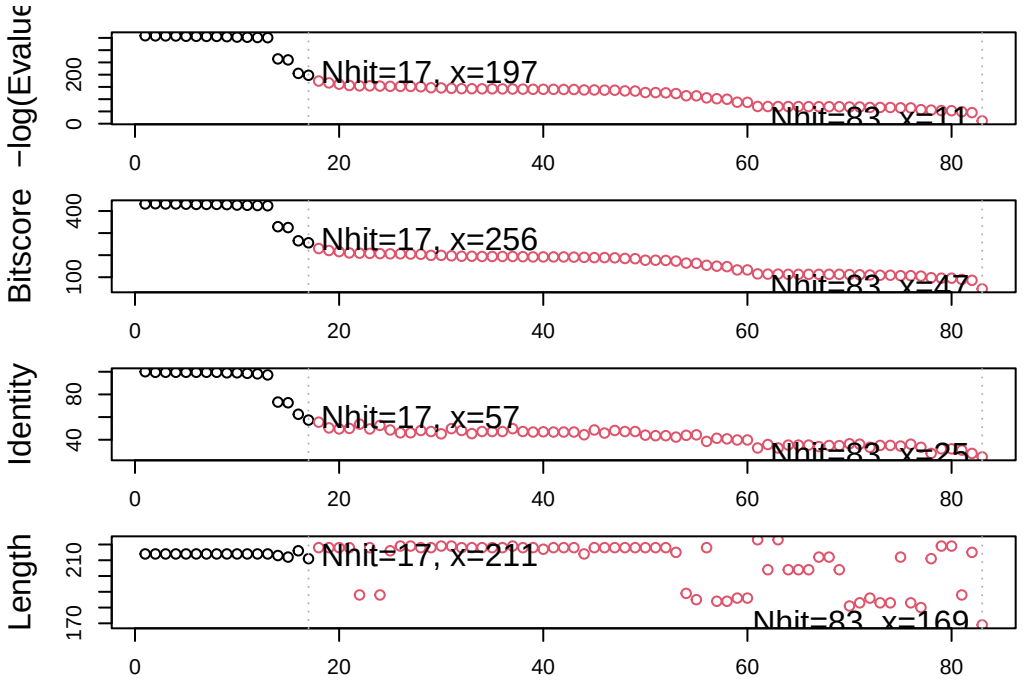
Blast search:

```
b <- blast.pdb(aa)
```

```
Searching ... please wait (updates every 5 seconds) RID = MAXSX38U016
.................................
Reporting 83 hits
```

```
hits <- plot(b)
```

```
 * Possible cutoff values:    197 11
           Yielding Nhits:    17 83

 * Chosen cutoff value of:    197
           Yielding Nhits:    17
```



List some of the top hits:

```
head(hits$pdb.id)
```

```
[1] "1AKE_A" "8BQF_A" "4X8M_A" "6S36_A" "6RZE_A" "4X8H_A"
```

Download related PDB files:

```
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.gz exists. Skipping download
```

```
Warning in get.pdb(hits$pdb.id, path = "pdbs", split = TRUE, gzip = TRUE):
pdbs/8BQF.pdb.gz exists. Skipping download
```

```
Warning in get.pdb(hits$pdb.id, path = "pdbs", split = TRUE, gzip = TRUE):
pdbs/4X8M.pdb.gz exists. Skipping download
```

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

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

Warning in get.pdb(hits$pdb.id, path = "pdbs", split = TRUE, gzip = TRUE):
pdbs/4X8H.pdb.gz exists. Skipping download

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

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

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

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

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

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

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

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

Warning in get.pdb(hits$pdb.id, path = "pdbs", split = TRUE, gzip = TRUE):
pdbs/4NP6.pdb.gz exists. Skipping download

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

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


  |
  |                                                                      |   0%
  |
  |====                                                                  |   6%
  |
  |========                                                              |  12%
  |
  |============                                                          |  18%
  |
  |================                                                      |  24%
  |
  |====================                                                  |  29%
  |
  |========================                                              |  35%
  |
  |============================                                          |  41%
  |
  |================================                                      |  47%
  |
  |====================================                                  |  53%
  |
  |========================================                              |  59%
  |
  |============================================                          |  65%
  |
  |================================================                      |  71%
  |
  |====================================================                  |  76%
  |
  |============================================================          |  82%
  |
  |================================================================      |  88%
  |
  |====================================================================  |  94%
  |
  |======================================================================| 100%
```

Align and superpose structures. Next we will use the pdbaln() function to align and also optionally fit (i.e. superpose) the identified PDB structures.