class11 redo

Kai Movellan

11/7/2021

Download a CSV file from the PDB site (accessible from "Analyze" > "PDB Statistics" > "by Experimental Method and Molecular Type". Move this CSV file into your RStudio project and use it to answer the following questions:

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

```
dataexport<-"DataExport.csv"
db<-read.csv(dataexport,row.names = 1)
head(db)
```

```
##                           X.ray   NMR   EM Multiple.methods Neutron Other  Total
## Protein (only)           142303 11804 5999               177      70    32 160385
## Protein/Oligosaccharide    8414    31  979                 5       0     0   9429
## Protein/NA                 7491   274 1986                 3       0     0   9754
## Nucleic acid (only)        2368  1372   60                 8       2     1   3811
## Other                       149    31    3                 0       0     0    183
## Oligosaccharide (only)       11     6    0                 1       0     4     22
```

```
xray<-db$X.ray
em<-db$EM
total<-db$Total

((sum(xray)+sum(em))/sum(total))*100
```

```
## [1] 92.47157
```

```
method.sums<-colSums(db)
round((method.sums/method.sums["Total"])*100,2)
```

```
##            X.ray              NMR               EM Multiple.methods
##            87.55             7.36             4.92             0.11
##          Neutron            Other            Total
##             0.04             0.02           100.00
```

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

```
#type.sums <- rowSums(db)
#round((type.sums[1]/method.sums["Total"]),2)
round((db$Total/method.sums["Total"])*100,2)
```

```
## [1] 87.36  5.14  5.31  2.08  0.10  0.01
```

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?

183581 HIV-1 protease structres in the current PDB

#The PDB format

```
#Alternatively, you can examine the contents of your downloaded file in a suitable text editor or use t

#less ~/Downloads/1hsg.pdb         ## (use 'q' to quit)
```

#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?

- We selected and displayed all water molecules as red spheres

Q5: There is a conserved water molecule in the binding site. Can you identify this water molecule? What residue number does this water molecule have (see note below)?

- The water molecule is MK1902

```
#install.packages("bio3d")
library(bio3d)
##   Note: Accessing on-line PDB file
pdb <- read.pdb("1hsg")
```

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

```
print(pdb)
```

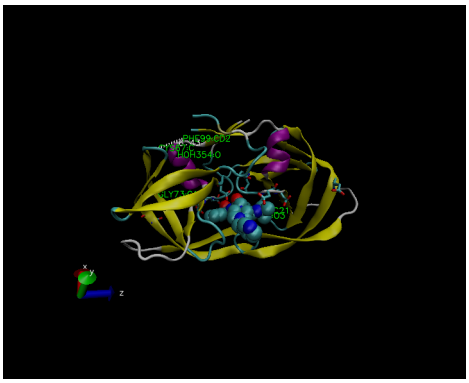
```
##
##  Call:  read.pdb(file = "1hsg")
##
##    Total Models#: 1
##      Total Atoms#: 1686,  XYZs#: 5058  Chains#: 2  (values: A B)
##
##      Protein Atoms#: 1514  (residues/Calpha atoms#: 198)
##      Nucleic acid Atoms#: 0  (residues/phosphate atoms#: 0)
##
##      Non-protein/nucleic Atoms#: 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"
```

```
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>
```



Q6: As you have hopefully observed HIV protease is a homodimer (i.e. it is composed of two identical chains). With the aid of the graphic display and the sequence viewer extension can you identify secondary structure elements that are likely to only form in the dimer rather than the monomer?

- The secondary structures in the purple region are likely to form a dimer rather than the monomer

#3. Introduction to Bio3D in R

Using the bio3d package

```
library(bio3d)

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

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

```
pdb
```

```
##
##  Call:  read.pdb(file = "1hel")
##
##    Total Models#: 1
##      Total Atoms#: 1186,  XYZs#: 3558  Chains#: 1  (values: A)
##
##      Protein Atoms#: 1001  (residues/Calpha atoms#: 129)
##      Nucleic acid Atoms#: 0  (residues/phosphate atoms#: 0)
##
##      Non-protein/nucleic Atoms#: 185  (residues: 185)
##      Non-protein/nucleic resid values: [ HOH (185) ]
##
##    Protein sequence:
##       KVFGRCELAAAMKRHGLDNYRGYSLGNWVCAAKFESNFNTQATNRNTDGSTDYGILQINS
##       RWWCNDGRTPGSRNLCNIPCSALLSSDITASVNCAKKIVSDGNGMNAWVAWRNRCKGTDV
##       QAWIRGCRL
##
## + 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(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>   LYS     A     1   <NA>  3.294 10.164 10.266 1 11.18
## 2 ATOM     2    CA <NA>   LYS     A     1   <NA>  2.388 10.533  9.168 1  9.68
## 3 ATOM     3     C <NA>   LYS     A     1   <NA>  2.438 12.049  8.889 1 14.00
```

```
## 4 ATOM     4     O <NA>   LYS     A     1   <NA>  2.406 12.898  9.815 1 14.00
## 5 ATOM     5    CB <NA>   LYS     A     1   <NA>  0.949 10.101  9.559 1 13.29
## 6 ATOM     6    CG <NA>   LYS     A     1   <NA> -0.050 10.621  8.573 1 13.52
##   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>
```

#4.Comparative analysis of protein structure

```
# Install packages in the R console not your Rmd

#install.packages("bio3d")
#install.packages("ggplot2")
#install.packages("ggrepel")
#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?:

-Grantlab/bio3d-view

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

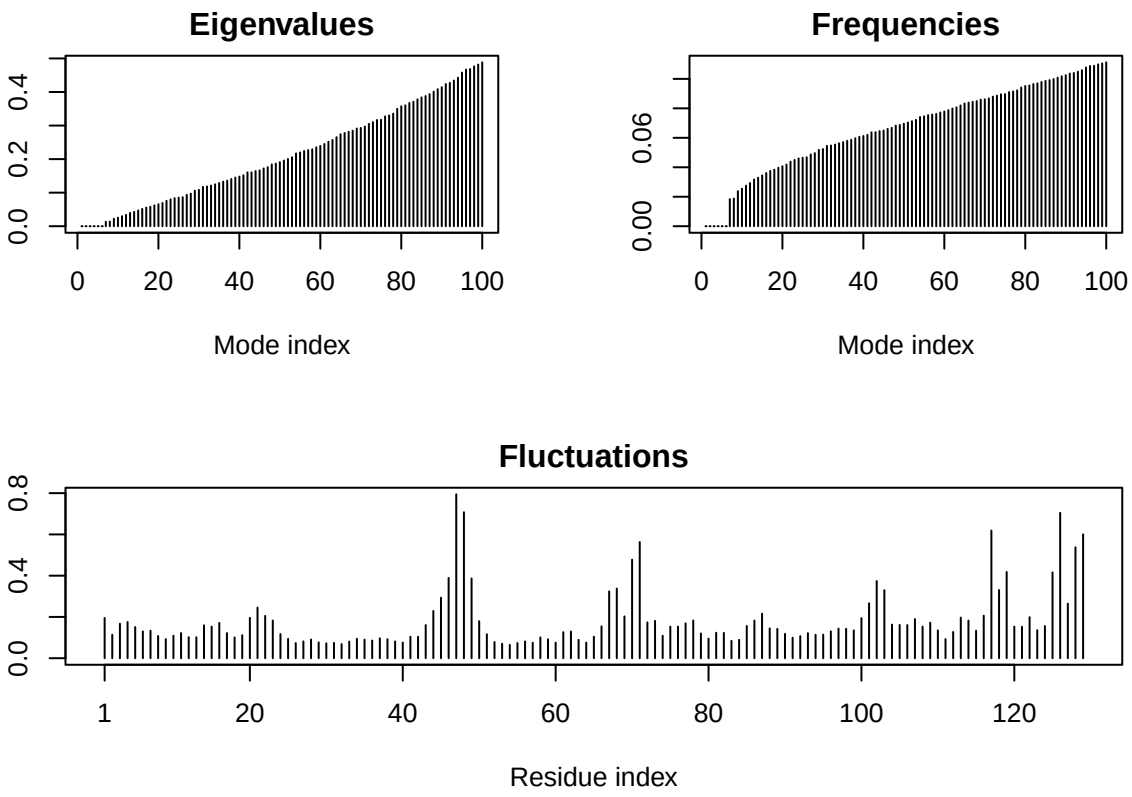
-TRUE

Let's use a mathematics method called NMA (Normal Made Analysis) to predict the dynamics (flexibility) of this enzyme.

```
modes<-nma(pdb)
```

```
##  Building Hessian...     Done in 0.025 seconds.
##  Diagonalizing Hessian...    Done in 0.099 seconds.
```

```
plot(modes)
```



Make a "movie" of its predicted motion.We often call this a "trajectory"

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

##Search and retrieve ADK structures

```
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 or hmmer search
#blast<-blast.pdb(aa)
```

```
hits <- NULL
hits$pdb.id <- c('1AKE_A','4X8M_A','6S36_A','6RZE_A','4X8H_A','3HPR_A','1E4V_A','5EJE_A','1E4Y_A','3X2S_
```

```
# Plot a summary of search results
#hits <- plot(b)
```

```
# List out some 'top hits'
head(hits$pdb.id)
```

```
## [1] "1AKE_A" "4X8M_A" "6S36_A" "6RZE_A" "4X8H_A" "3HPR_A"
```

```
# Download releated 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/
## 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

##   |                                                                              |
```

##Align and superpose structures

```
# Align releated PDBs
pdbs <- pdbaln(files, fit = TRUE)#, exefile="msa")
```

```
## Reading PDB files:
## pdbs/split_chain/1AKE_A.pdb
## pdbs/split_chain/4X8M_A.pdb
## pdbs/split_chain/6S36_A.pdb
## pdbs/split_chain/6RZE_A.pdb
## pdbs/split_chain/4X8H_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/4NP6_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/4X8M_A.pdb
## pdb/seq: 3   name: pdbs/split_chain/6S36_A.pdb
##    PDB has ALT records, taking A only, rm.alt=TRUE
## pdb/seq: 4   name: pdbs/split_chain/6RZE_A.pdb
##    PDB has ALT records, taking A only, rm.alt=TRUE
## pdb/seq: 5   name: pdbs/split_chain/4X8H_A.pdb
## pdb/seq: 6   name: pdbs/split_chain/3HPR_A.pdb
##    PDB has ALT records, taking A only, rm.alt=TRUE
## pdb/seq: 7   name: pdbs/split_chain/1E4V_A.pdb
## pdb/seq: 8   name: pdbs/split_chain/5EJE_A.pdb
##    PDB has ALT records, taking A only, rm.alt=TRUE
## pdb/seq: 9   name: pdbs/split_chain/1E4Y_A.pdb
## pdb/seq: 10   name: pdbs/split_chain/3X2S_A.pdb
## pdb/seq: 11   name: pdbs/split_chain/6HAP_A.pdb
## pdb/seq: 12   name: pdbs/split_chain/6HAM_A.pdb
##    PDB has ALT records, taking A only, rm.alt=TRUE
## pdb/seq: 13   name: pdbs/split_chain/4K46_A.pdb
##    PDB has ALT records, taking A only, rm.alt=TRUE
## pdb/seq: 14   name: pdbs/split_chain/4NP6_A.pdb
## pdb/seq: 15   name: pdbs/split_chain/3GMT_A.pdb
## pdb/seq: 16   name: pdbs/split_chain/4PZL_A.pdb
```

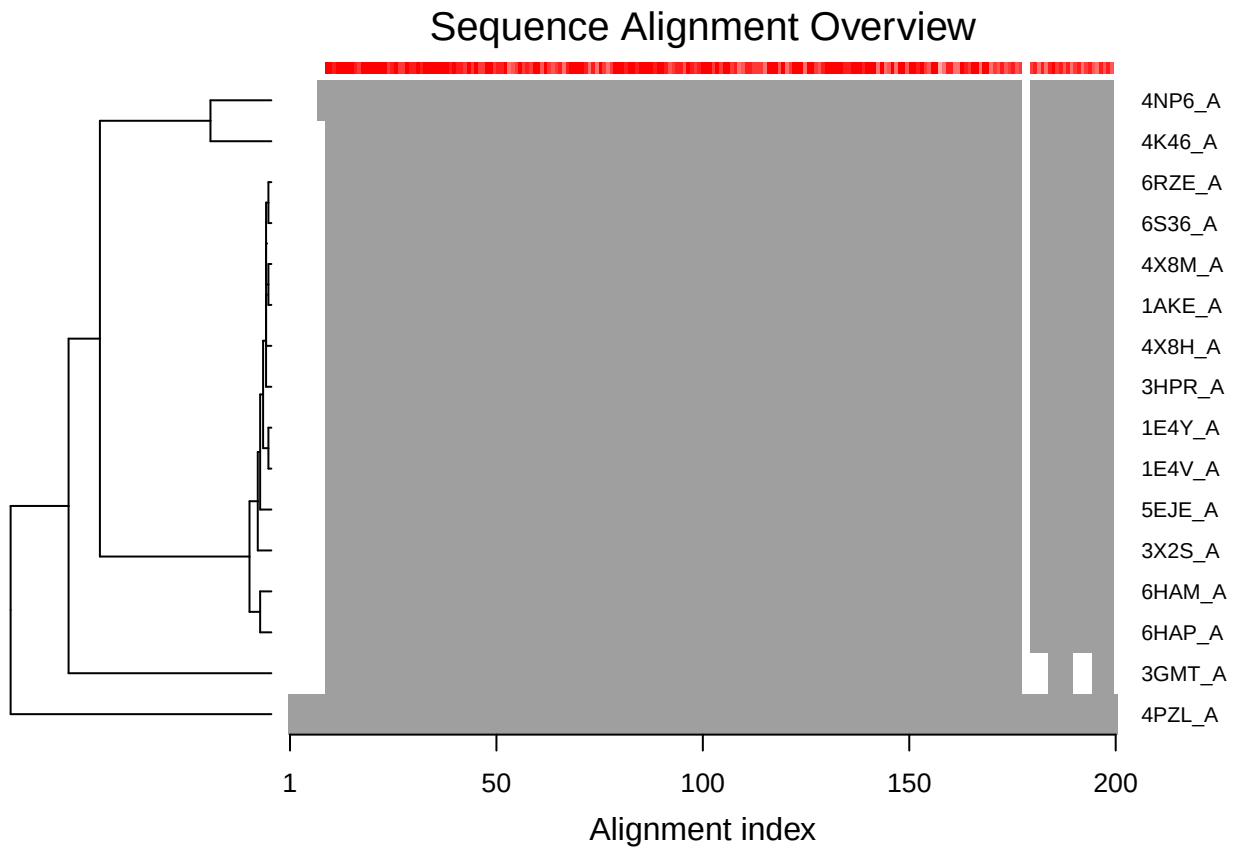
```
#pdbs
```

#PCA

We will use the bio3d pca() function which is designed for protein structure data

```
# Vector containing PDB codes for figure axis
ids <- basename.pdb(pdbs$id)
```

```
# Draw schematic alignment
plot(pdbs, labels=ids)
```



#Viewing our superposed structures

```
library(bio3d.view)
library(rgl)

view.pdbs(pdbs)
```

#Annotate collected PDB structures

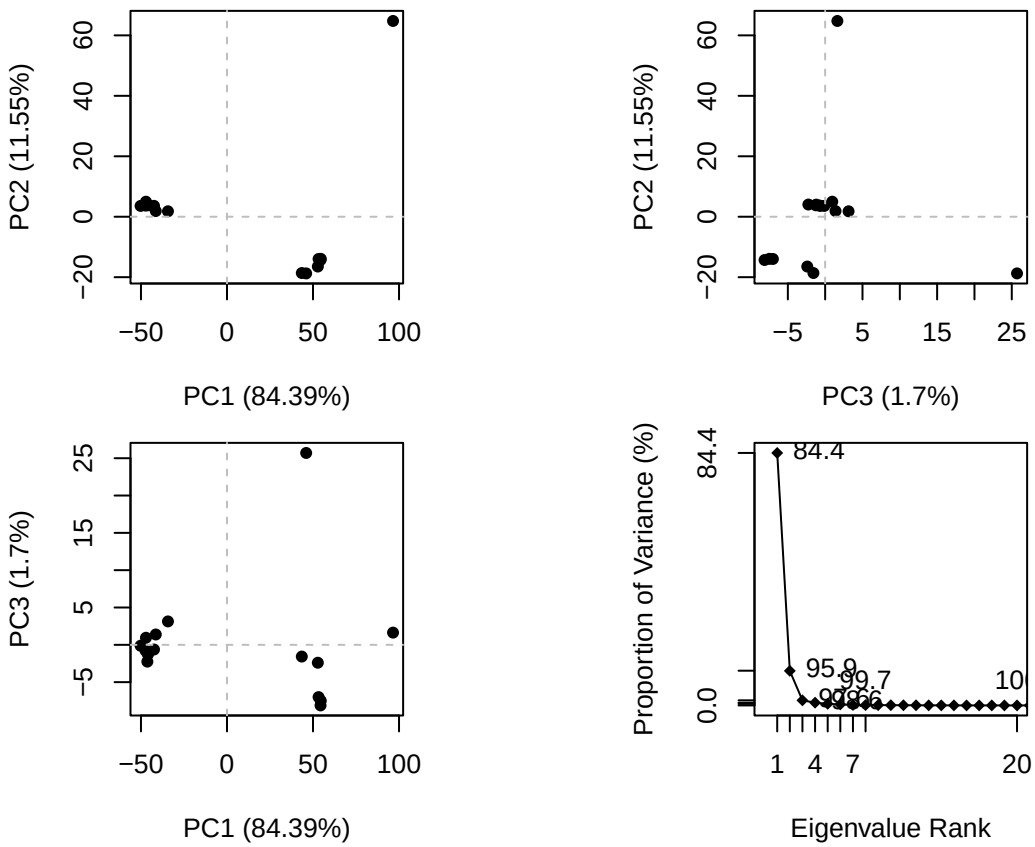
Optional

```
#anno <- pdb.annotate(ids)
#unique(anno$source)
```

```
#anno
```

#Principal component analysis

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

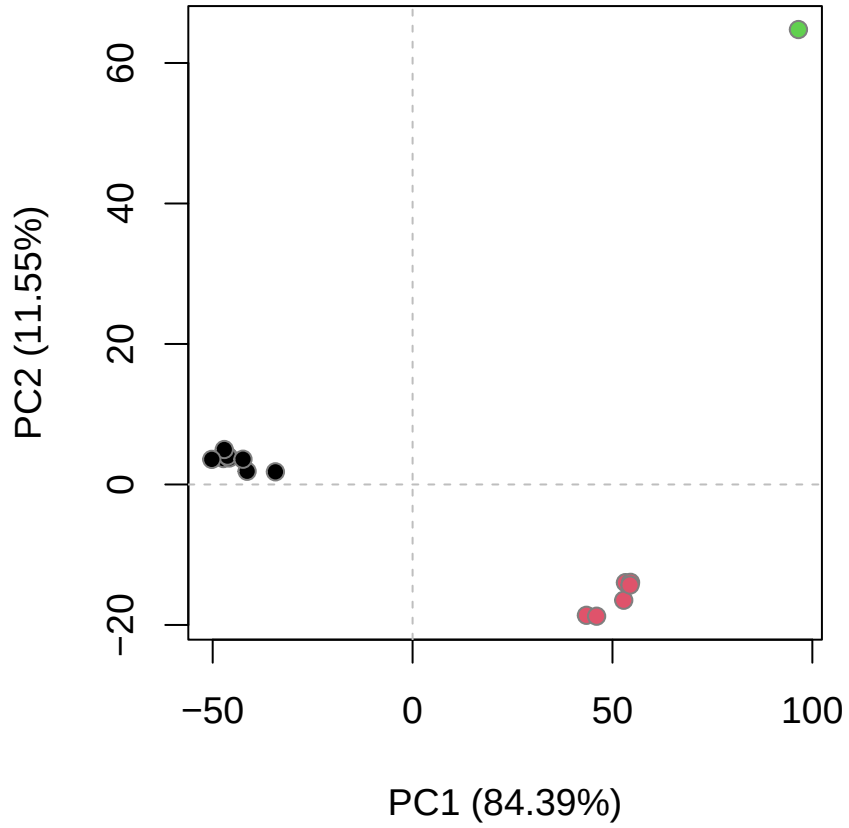



```
# Calculate RMSD
rd <- rmsd(pdbs)
```

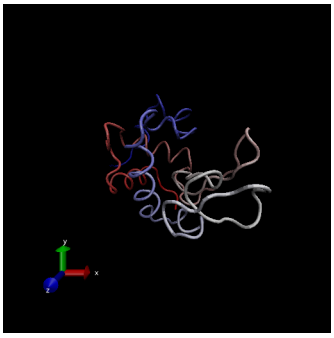
```
## Warning in rmsd(pdbs): No indices provided, using the 204 non NA positions
```

```
# 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)
```



#Optional further visualization



```
#Visualize first principal component
pc1<-mktrj(pc.xray, pc=1, file="pc_1.pdb")
```

```
view.xyz(pc1)
```

```
## Potential all C-alpha atom structure(s) detected: Using calpha.connectivity()
```

```
## Potential all C-alpha atom structure(s) detected: Using calpha.connectivity()
```

```
view.xyz(pc1, col=vec2color( rmsf(pc1) ))
```

```
## Potential all C-alpha atom structure(s) detected: Using calpha.connectivity()
```

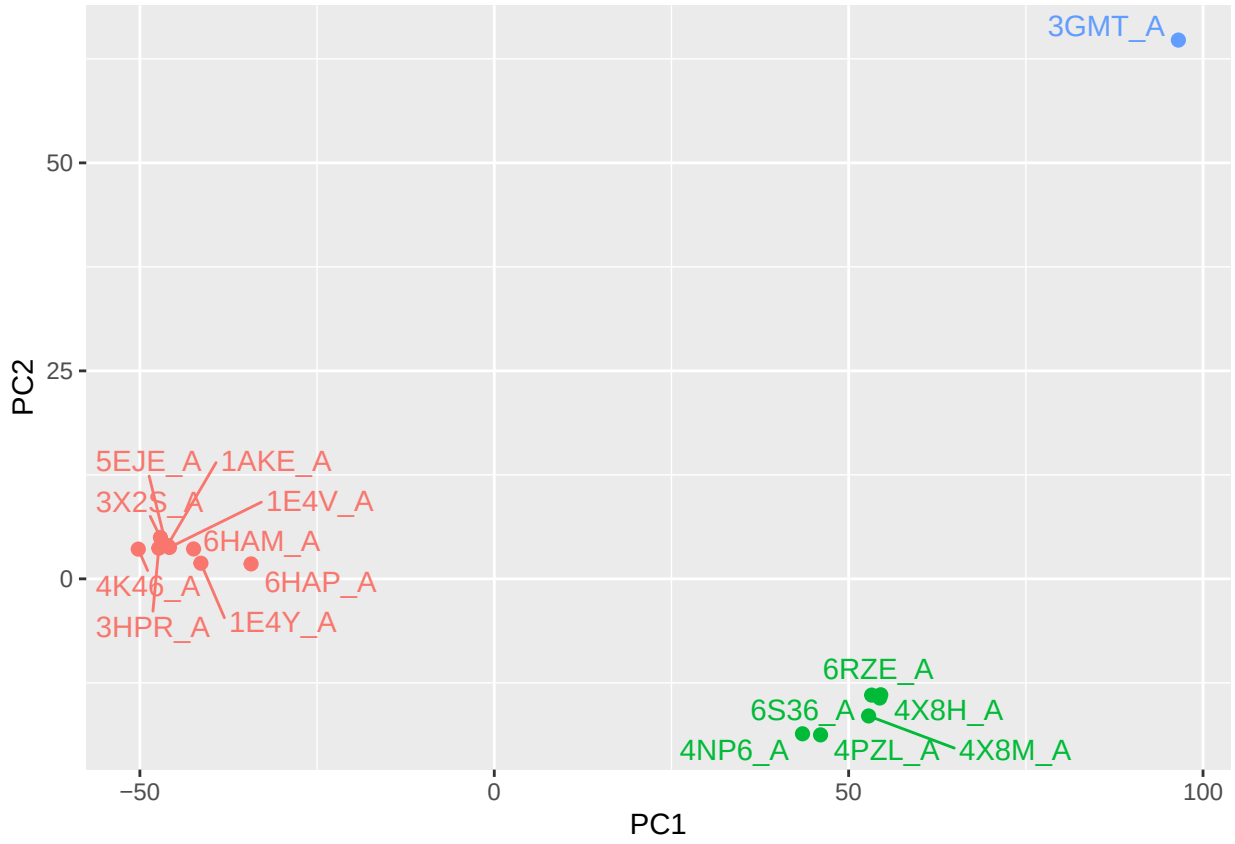
```
## Potential all C-alpha atom structure(s) detected: Using calpha.connectivity()
```

We can also plot our main PCA results with ggplot:

```
#Plotting results with ggplot2
library(ggplot2)
library(ggrepel)

df <- data.frame(PC1=pc.xray$z[,1],
                 PC2=pc.xray$z[,2],
                 col=as.factor(grps.rd),
                 ids=ids)

p <- ggplot(df) +
  aes(PC1, PC2, col=col, label=ids) +
  geom_point(size=2) +
  geom_text_repel(max.overlaps = 20) +
  theme(legend.position = "none")
p
```



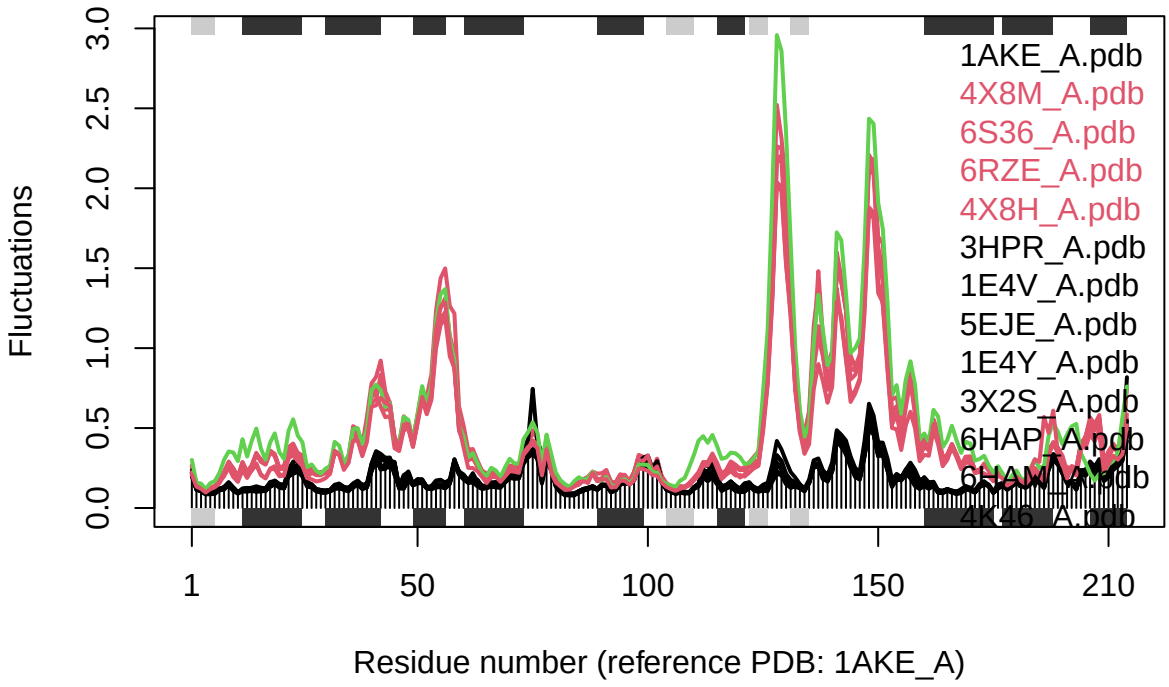
#6. Normal mode analysis

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

```
##
## Details of Scheduled Calculation:
##    ... 16 input structures
##    ... storing 606 eigenvectors for each structure
##    ... dimension of x$U.subspace: ( 612x606x16 )
##    ... coordinate superposition prior to NM calculation
##    ... aligned eigenvectors (gap containing positions removed)
##    ... estimated memory usage of final 'eNMA' object: 45.4 Mb
##
##    |                                                                              |
```

```
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?

- the graph depicts two conformation states
- the black and colored lines are different
- they differ from the low frequency displacement of two nucleotide binding regions showing the distinct nucleotides