Class 9: Structual Bioinformatics

Bianca Barriga

```
PDB.df <- read.csv("/Users/biancabarriga/Desktop/bggn213/Class09_files/Data Export Summary
head(PDB.df)
```

	X.ray	EM	NMR	Multiple.methods	Neutron	Other	Total
Protein (only)	152914	9495	12121	191	72	32	174825
Protein/Oligosaccharide	9008	1663	32	7	1	0	10711
Protein/NA	8069	2949	282	6	0	0	11306
Nucleic acid (only)	2602	78	1434	12	2	1	4129
Other	163	9	31	0	0	0	203
Oligosaccharide (only)	11	0	6	1	0	4	22

```
knitr::include_graphics("/Users/biancabarriga/Desktop/bggn213/1HSG.png")
```

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

```
percent <- (sum(PDB.df$X.ray) + sum(PDB.df$EM)) / sum(PDB.df$Total)
percent
```

```
[1] 0.9292481
```

The percent of structures in the PDB solved by X-ray and EM is ~93%.

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

```
percentProtein <- PDB.df$Total[1]/ sum(PDB.df$Total)
percentProtein
```

```
[1] 0.8689288
```

The proportion of structures in the PDB that are protein are ~87%.

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?

```
knitr::include_graphics("/Users/biancabarriga/Desktop/bggn213/1HSG3.png")
```

There are 4,791 structures for HIV-1.

The PDB format

Now download the "PDB File" for the HIV-1 protease structure with the PDB identifier 1HSG. On the website you can "Display" the contents of this "PDB format" file.

Alternatively, you can examine the contents of your downloaded file in a suitable text editor or use the Terminal tab from within RStudio (or your favorite Terminal/Shell) and try the following command:

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

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

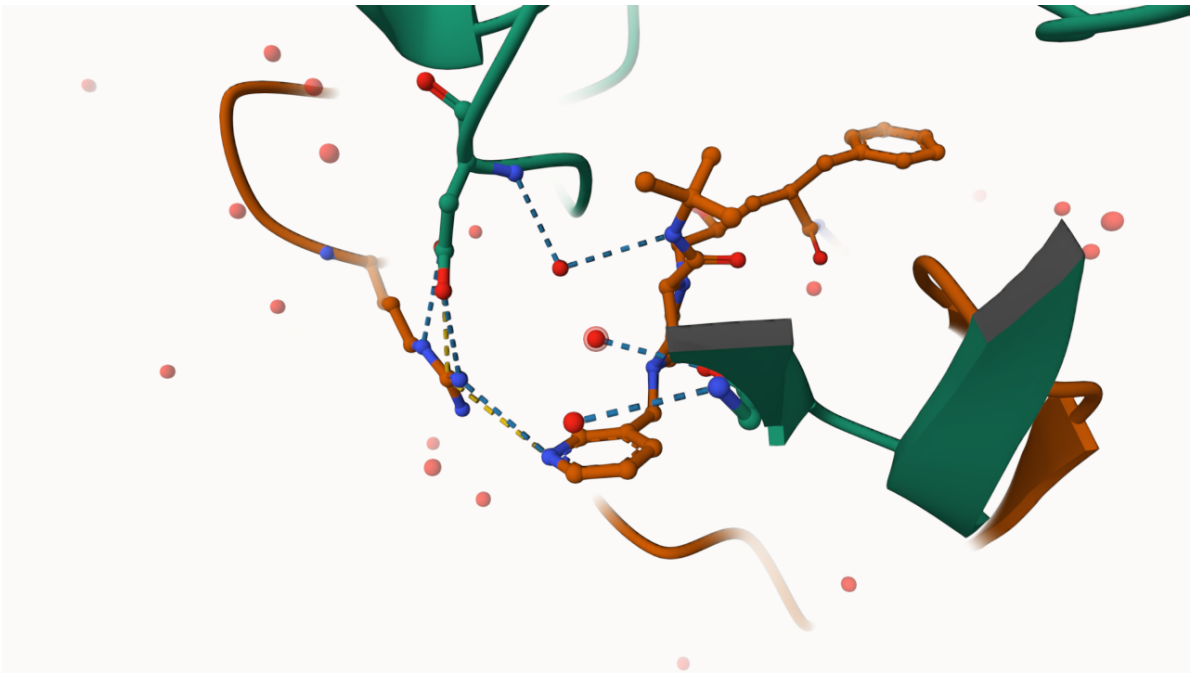
Ball and stick representation - not showing H atoms only oxygen.

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

HOH332

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

```
knitr::include_graphics("/Users/biancabarriga/Desktop/bggn213/1HSG5.png")
```



.

Introduction to Bio3D in R

Install the Bio3D package

```
#install.packages("bio3d")
```

Load the Bio3D package

```
library(bio3d)
```

Read PDB file into R

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

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

Get a summary of the contents

```
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?

There are 198 amino acid residues.

```
aa <- length(pdb$seqres)
aa
```

```
[1] 198
```

There are 198 amino acids.

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

HOH[127]

Q9. How many protein chains are in this structure?

2 protein chains

Note that the attributes (+ attr:) of this object are listed on the last couple of lines. To find the attributes of any such object you can use:

```
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 structure

Let's read a new PDB structure of Adenylate Kinase and perform Normal mode analysis.

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

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

```
adk
```

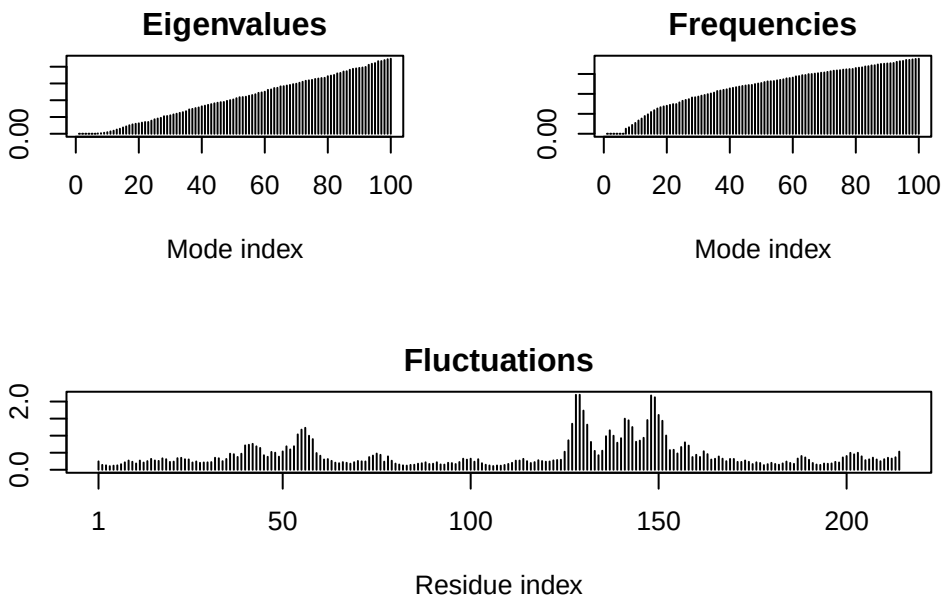
```
 Call:  read.pdb(file = "6s36")

   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 flexbility predicion

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

```
 Building Hessian...        Done in 0.026 seconds.
 Diagonalizing Hessian...   Done in 0.538 seconds.
```

```
plot(m)
```



To view a "movie" of these predicted motions we can generate a molecular "trajectory" with the mktrj() function.

```
mktrj(m, file="adk_m7.pdb")
```

Now we can load the resulting "adk_m7.pdb" PDB into Mol* with the "Open Files" option on the right side control panel. Once loaded click the "play" button to see a movie (see image below). We will discuss how this method works at the end of this lab when we apply it across a large set of homologous structures.

Comparative structure analysis of Adenylate Kinase

The goal of this section is to perform principal component analysis (PCA) on the complete collection of Adenylate kinase structures in the protein data-bank (PDB).

```
# 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 package

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

Error need to instal httr package

```
#install.packages("httr")
```

```
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 amino acids

Now we can use this sequence as a query to BLAST search the PDB to find similar sequences and structures. #load the sequence and then blast

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

```
Searching ... please wait (updates every 5 seconds) RID = YBKH9VUU013
.
Reporting 96 hits
```

The function plot.blast() facilitates the visualization and filtering of the Blast results. It will attempt to set a seed position to the point of largest drop-off in normalized scores (i.e. the biggest jump in E-values). In this particular case we specify a cutoff (after initial plotting) of to include only the relevant E.coli structures:

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

```
  * Possible cutoff values:    197 -3
            Yielding Nhits:    16 96
```

```
* Chosen cutoff value of:    197
          Yielding Nhits:    16
```

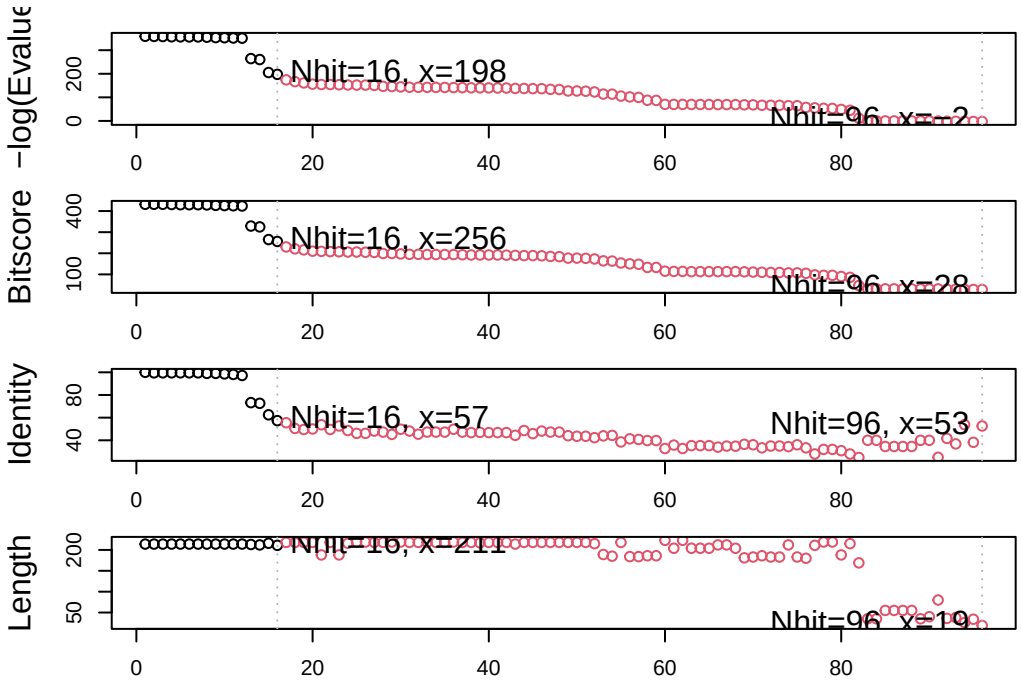


Figure 6: Blast results. Visualize and filter blast results through function plot.blast(). Here we proceed with only the top scoring hits (black).

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

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

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

The Blast search and subsequent filtering identified a total of 13 related PDB structures to our query sequence. The PDB identifiers of this collection are accessible through the `$pdb.id` attribute to the hits object (i.e. `hits$pdb.id`). Note that adjusting the cutoff argument (to `plot.blast()`) will result in a decrease or increase of hits.

We can now use function `get.pdb()` and `pdbslit()` to fetch and parse the identified structures.

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

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

```
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
```

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

# Draw schematic alignment
op <-par(mar = c(0, 0, 0, 0))
#plot(pdbs, labels=ids)
par(op)
```

Annotate collected PDB structures

The function `pdb.annotate()` provides a convenient way of annotating the PDB files we have collected. Below we use the function to annotate each structure to its source species. This will come in handy when annotating plots later on:

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

```
[1] "Escherichia coli"
[2] "Escherichia coli K-12"
[3] "Escherichia coli O139:H28 str. E24377A"
[4] "Escherichia coli str. K-12 substr. MDS42"
[5] "Photobacterium profundum"
[6] "Burkholderia pseudomallei 1710b"
[7] "Francisella tularensis subsp. tularensis SCHU S4"
```

```
anno
```

```
       structureId chainId macromoleculeType chainLength experimentalTechnique
1AKE_A        1AKE       A           Protein         214                X-ray
6S36_A        6S36       A           Protein         214                X-ray
6RZE_A        6RZE       A           Protein         214                X-ray
3HPR_A        3HPR       A           Protein         214                X-ray
```

1E4V_A	1E4V	A	Protein	214		X-ray
5EJE_A	5EJE	A	Protein	214		X-ray
1E4Y_A	1E4Y	A	Protein	214		X-ray
3X2S_A	3X2S	A	Protein	214		X-ray
6HAP_A	6HAP	A	Protein	214		X-ray
6HAM_A	6HAM	A	Protein	214		X-ray
4K46_A	4K46	A	Protein	214		X-ray
3GMT_A	3GMT	A	Protein	230		X-ray
4PZL_A	4PZL	A	Protein	242		X-ray

	resolution	scopDomain	pfam	ligandId
1AKE_A	2.00	Adenylate kinase	Adenylate kinase (ADK)	AP5
6S36_A	1.60	<NA>	Adenylate kinase (ADK)	CL (3),NA,MG (2)
6RZE_A	1.69	<NA>	Adenylate kinase (ADK)	NA (3),CL (2)
3HPR_A	2.00	<NA>	Adenylate kinase (ADK)	AP5
1E4V_A	1.85	Adenylate kinase	Adenylate kinase (ADK)	AP5
5EJE_A	1.90	<NA>	Adenylate kinase (ADK)	AP5,CO
1E4Y_A	1.85	Adenylate kinase	Adenylate kinase (ADK)	AP5
3X2S_A	2.80	<NA>	Adenylate kinase (ADK)	JPY (2),AP5,MG
6HAP_A	2.70	<NA>	Adenylate kinase (ADK)	AP5
6HAM_A	2.55	<NA>	Adenylate kinase (ADK)	AP5
4K46_A	2.01	<NA>	Adenylate kinase (ADK)	ADP,AMP,PO4
3GMT_A	2.10	<NA>	Adenylate kinase (ADK)	SO4 (2)
4PZL_A	2.10	<NA>	Adenylate kinase (ADK)	CA,FMT,GOL

	ligandName
1AKE_A	BIS(ADENOSINE)-5'-PENTAPHOSPHATE
6S36_A	CHLORIDE ION (3),SODIUM ION,MAGNESIUM ION (2)
6RZE_A	SODIUM ION (3),CHLORIDE ION (2)
3HPR_A	BIS(ADENOSINE)-5'-PENTAPHOSPHATE
1E4V_A	BIS(ADENOSINE)-5'-PENTAPHOSPHATE
5EJE_A	BIS(ADENOSINE)-5'-PENTAPHOSPHATE,COBALT (II) ION
1E4Y_A	BIS(ADENOSINE)-5'-PENTAPHOSPHATE
3X2S_A	N-(pyren-1-ylmethyl)acetamide (2),BIS(ADENOSINE)-5'-PENTAPHOSPHATE,MAGNESIUM ION
6HAP_A	BIS(ADENOSINE)-5'-PENTAPHOSPHATE
6HAM_A	BIS(ADENOSINE)-5'-PENTAPHOSPHATE
4K46_A	ADENOSINE-5'-DIPHOSPHATE,ADENOSINE MONOPHOSPHATE,PHOSPHATE ION
3GMT_A	SULFATE ION (2)
4PZL_A	CALCIUM ION,FORMIC ACID,GLYCEROL

	source
1AKE_A	Escherichia coli
6S36_A	Escherichia coli
6RZE_A	Escherichia coli
3HPR_A	Escherichia coli K-12
1E4V_A	Escherichia coli

```
5EJE_A           Escherichia coli O139:H28 str. E24377A
1E4Y_A                                Escherichia coli
3X2S_A         Escherichia coli str. K-12 substr. MDS42
6HAP_A           Escherichia coli O139:H28 str. E24377A
6HAM_A                             Escherichia coli K-12
4K46_A                          Photobacterium profundum
3GMT_A                   Burkholderia pseudomallei 1710b
4PZL_A Francisella tularensis subsp. tularensis SCHU S4

1AKE_A STRUCTURE OF THE COMPLEX BETWEEN ADENYLATE KINASE FROM ESCHERICHIA COLI AND THE INHIBI
6S36_A
6RZE_A
3HPR_A
1E4V_A
5EJE_A                                                                                 Cryst
1E4Y_A
3X2S_A
6HAP_A
6HAM_A
4K46_A
3GMT_A
4PZL_A                                                                              The cryst
                                                    citation rObserved   rFree
1AKE_A                 Muller, C.W., et al. J Mol Biol (1992)   0.19600      NA
6S36_A                  Rogne, P., et al. Biochemistry (2019)   0.16320 0.23560
6RZE_A                  Rogne, P., et al. Biochemistry (2019)   0.18650 0.23500
3HPR_A  Schrank, T.P., et al. Proc Natl Acad Sci U S A (2009)   0.21000 0.24320
1E4V_A                   Muller, C.W., et al. Proteins (1993)   0.19600      NA
5EJE_A  Kovermann, M., et al. Proc Natl Acad Sci U S A (2017)   0.18890 0.23580
1E4Y_A                   Muller, C.W., et al. Proteins (1993)   0.17800      NA
3X2S_A                Fujii, A., et al. Bioconjug Chem (2015)   0.20700 0.25600
6HAP_A               Kantaev, R., et al. J Phys Chem B (2018)   0.22630 0.27760
6HAM_A               Kantaev, R., et al. J Phys Chem B (2018)   0.20511 0.24325
4K46_A                    Cho, Y.-J., et al. To be published   0.17000 0.22290
3GMT_A Buchko, G.W., et al. Biochem Biophys Res Commun (2010)   0.23800 0.29500
4PZL_A                       Tan, K., et al. To be published   0.19360 0.23680
         rWork spaceGroup
1AKE_A 0.19600  P 21 2 21
6S36_A 0.15940    C 1 2 1
6RZE_A 0.18190    C 1 2 1
3HPR_A 0.20620  P 21 21 2
1E4V_A 0.19600  P 21 2 21
5EJE_A 0.18630  P 21 2 21
```

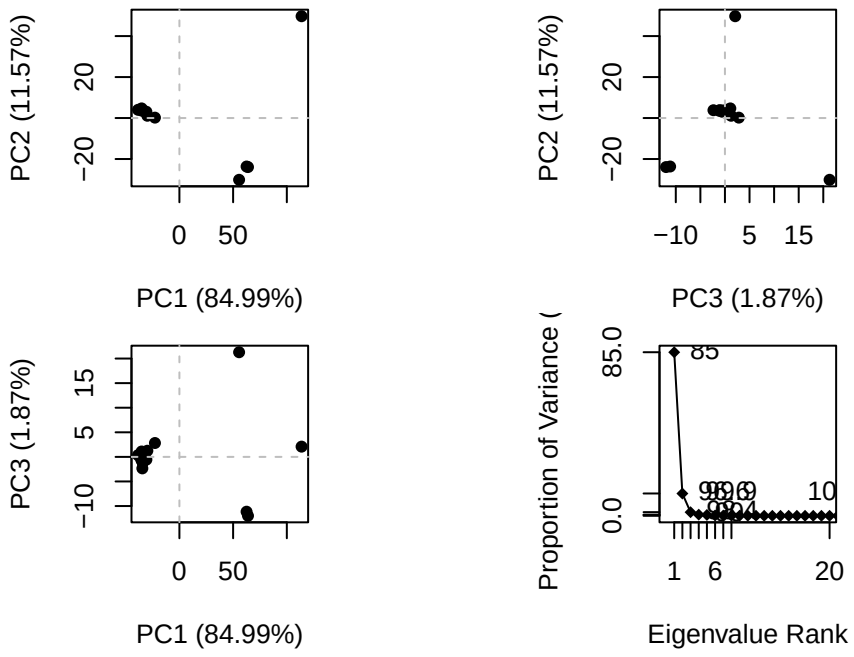
```
1E4Y_A 0.17800   P 1 21 1
3X2S_A 0.20700 P 21 21 21
6HAP_A 0.22370    I 2 2 2
6HAM_A 0.20311       P 43
4K46_A 0.16730 P 21 21 21
3GMT_A 0.23500   P 1 21 1
4PZL_A 0.19130       P 32
```

#Principal componen analysis

Function `pca()` provides principal component analysis (PCA) of the structure data. PCA is a statistical approach used to transform a data set down to a few important components that describe the directions where there is most variance. In terms of protein structures PCA is used to capture major structural variations within an ensemble of structures.

PCA can be performed on the structural ensemble (stored in the pdbs object) with the function `pca.xyz()`, or more simply `pca()`.

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



Function `rmsd()` will calculate all pairwise RMSD values of the structural ensemble. This facilitates clustering analysis based on the pairwise structural deviation:

```
# 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)
```