class9r

Rachel Kraft

2/16/2022

Download CSV file from the PDB site

```
pdb.data <- read.csv("Data Export Summary.csv")
pdb.data
```

```
##            Molecular.Type  X.ray   NMR   EM Multiple.methods Neutron Other
## 1          Protein (only) 144433 11881 6732              182      70    32
## 2 Protein/Oligosaccharide   8543    31 1125                5       0     0
## 3              Protein/NA   7621   274 2165                3       0     0
## 4     Nucleic acid (only)   2396  1399   61                8       2     1
## 5                   Other    150    31    3                0       0     0
## 6  Oligosaccharide (only)     11     6    0                1       0     4
##    Total
## 1 163330
## 2   9704
## 3  10063
## 4   3867
## 5    184
## 6     22
```

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

sum of the xray column+ sum of the EM column divided by the total *100 for a percentage

```
str.xray.elec <- (sum(pdb.data[,2])+sum(pdb.data[,4])) /  sum(pdb.data[,8]) *100
str.xray.elec
```

```
## [1] 92.55757
```

92.55757% of structures are solved by X-Ray and EM

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

```
pr.protein <- pdb.data[1,8] / sum(pdb.data[,8])
pr.protein
```

```
## [1] 0.8726292
```

Proportion of 0.8726292 are protein(only)

> 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?

There are many results when we just simply search HIV, and it is hard to tell how many are HIV-1 protease structures just from the name. A more reliable search method would be to find an HIV protease sequence from refseq on NCBI and use this to find how many structures are there

> Download PDB for 1HSG

1hsg.pdb downloaded in project

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

The water molecules are just represented as 1 sphere since we are selecting for HOH as a molecule in the graphics window

> 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)?

residue number: HOH308:0

> Load Bio3D package

```
library(bio3d)
```

> Read our 1HSG.pdb file

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

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

find attributes

```
attributes(pdb)
```

```
## $names
## [1] "atom"   "xyz"    "seqres" "helix"  "sheet"  "calpha" "remark" "call"
##
## $class
## [1] "pdb" "sse"
```

access atom attribute

```
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>
```

Section 4: PCA on adenylate kinase structures in the PDB

Install packages needed

The install.packages() function is used to install packages from the main CRAN repository for R packages. BioConductor is a separate repository of R packages focused on high-throughput biomolecular assays and biomolecular data. Packages from BioConductor can be installed using the BiocManager::install() function. Finally, R packages found on GitHub or BitBucket can be installed using devtools::install_github() and devtools::install_bitbucket() functions.

```r
#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?:

bio3d-view

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

true

We will perform a blast search of PDB database to identify related structures to our ADK sequence. get.seq() gets query sequence for chain A of PDB ID 1AKE and use this as input to blast.pdb()

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

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

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

```r
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

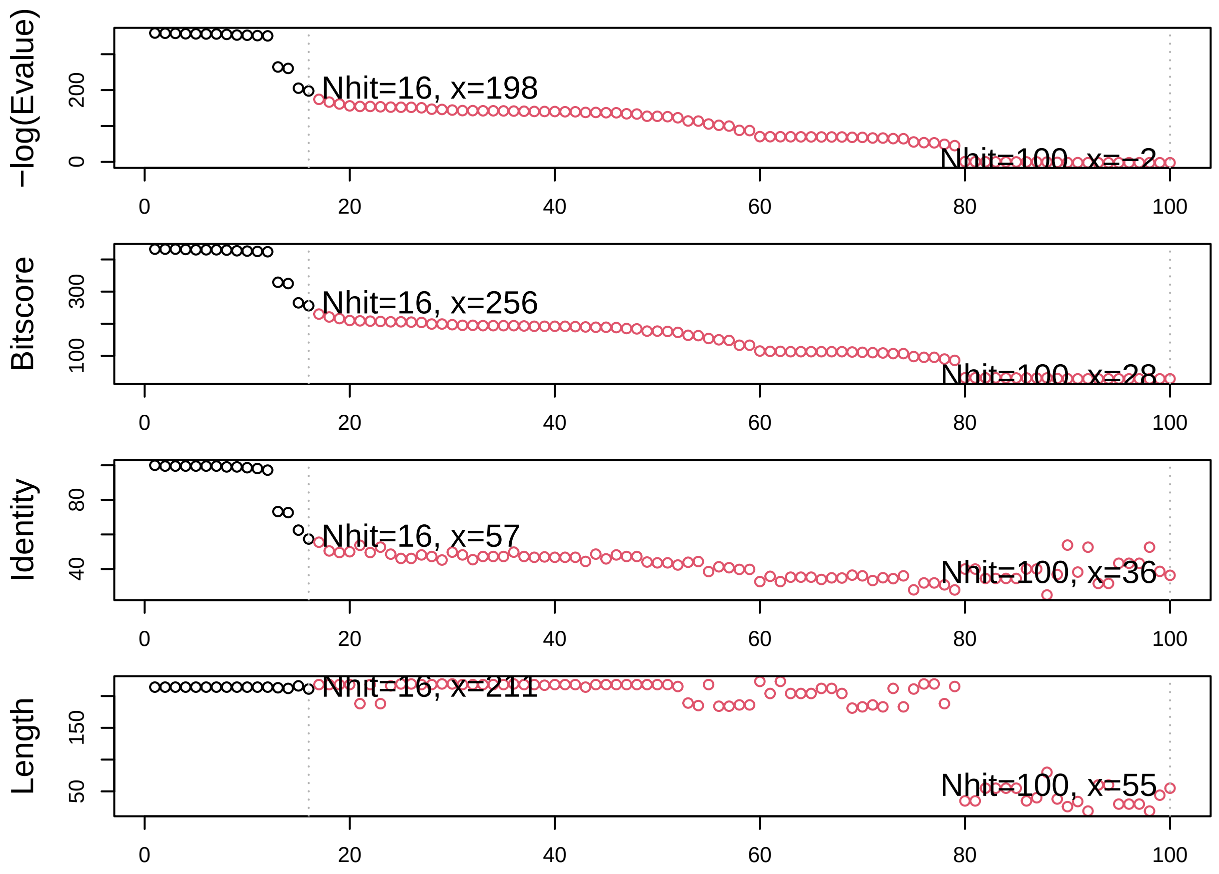
Now we can use this sequence as a query to blast to look for similar sequences and structures

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

```
##  Searching ... please wait (updates every 5 seconds) RID = 15Z9HENE013
##  .......................................
##  Reporting 100 hits
```

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

```
##   * Possible cutoff values:    197 -3
##             Yielding Nhits:    16 100
##
##   * Chosen cutoff value of:    197
##             Yielding Nhits:    16
```



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

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

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

```r
# 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 exists. Skipping download

## Warning in get.pdb(hits$pdb.id, path = "pdbs", split = TRUE, gzip = TRUE): pdbs/
## 4X8M.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$pdb.id, path = "pdbs", split = TRUE, gzip = TRUE): pdbs/
## 4X8H.pdb exists. Skipping download
## Warning in get.pdb(hits$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/
## 4NP6.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

##   |                                                                              |
```

Use pdbaln() to align and fit identified PDB structures

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

- I attempted to download and install muscle but I am having issues getting it to run on my computer

  Principal component analysis

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)
```

- This outputs plots for 3 principal components and eigenvalue rank
- I can't download muscle to make the alignment object pdbs, so I can't do the principal component analysis

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)

# 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)
```

- This outputs a PC1 vs PC2 conformer plot for projection of adenylate kinase x-ray structures

To visualize the major structural variations in the ensemble the function mktrj() can be used to generate a trajectory PDB file by interpolating along a give PC (eigenvector)

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

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
```

Normal mode analysis

Function nma() provides normal mode analysis (NMA) on both single structures (if given a singe PDB input object) or the complete structure ensemble (if provided with a PDBS input object). This facilitates characterizing and comparing flexibility profiles of related protein structures.

```
# NMA of all structures
#modes <- nma(pdbs)
#plot(modes, pdbs, col=grps.rd)
```

> 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 black and colored lines look similar in overall shape and trajectory, but differ the most in heights (fluctuations). They differ because they represent two different conformational states for Adk