Class 06 R Functions HW

Cindy Tran

1/22/2022

Running the Original Code

First, I am installing the bio3d package and then loading it.

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

I am now going to run the original code.

```
s1 <- read.pdb("4AKE") # kinase with drug
```

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

```
s2 <- read.pdb("1AKE") # kinase no drug
```

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

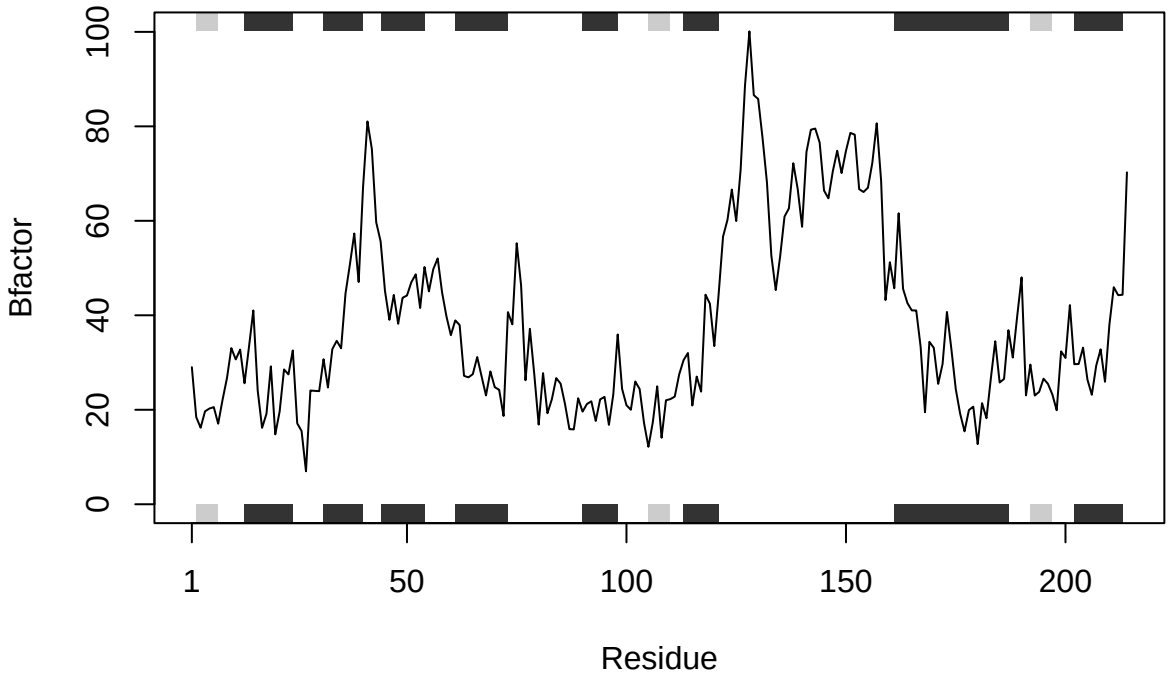
```
s3 <- read.pdb("1E4Y") # kinase with drug
```

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

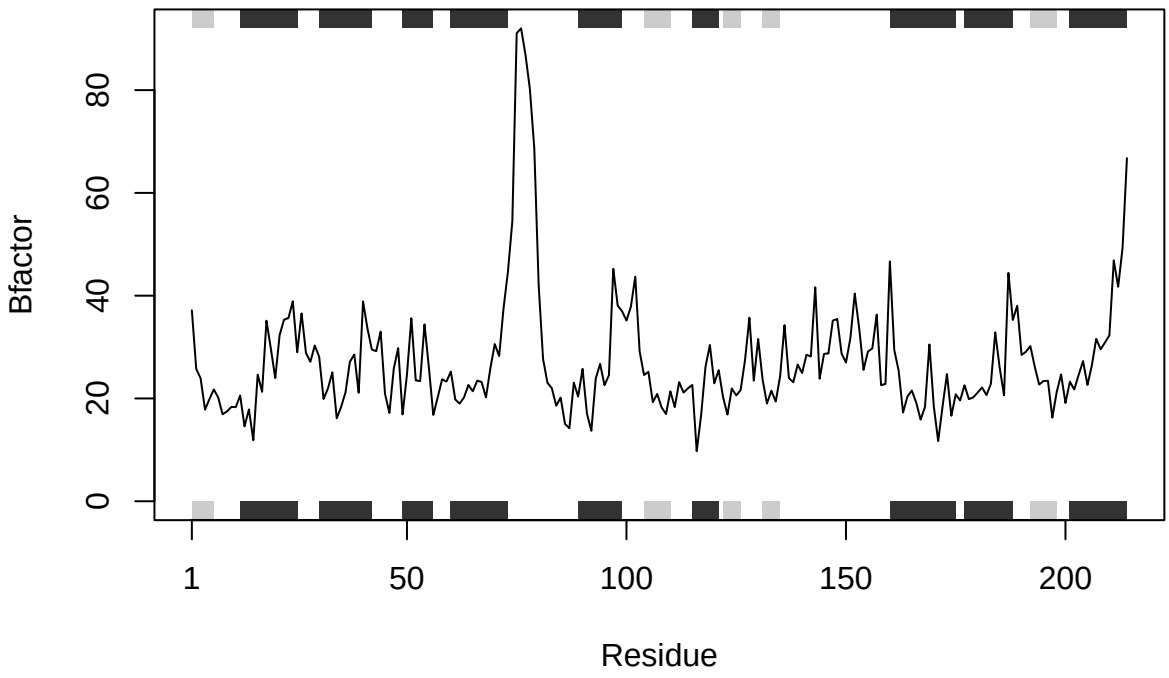
```
s1.chainA <- trim.pdb(s1, chain="A", elety="CA")
s2.chainA <- trim.pdb(s2, chain="A", elety="CA")
s3.chainA <- trim.pdb(s3, chain="A", elety="CA")
```

```
s1.b <- s1.chainA$atom$b
s2.b <- s2.chainA$atom$b
s3.b <- s3.chainA$atom$b
```

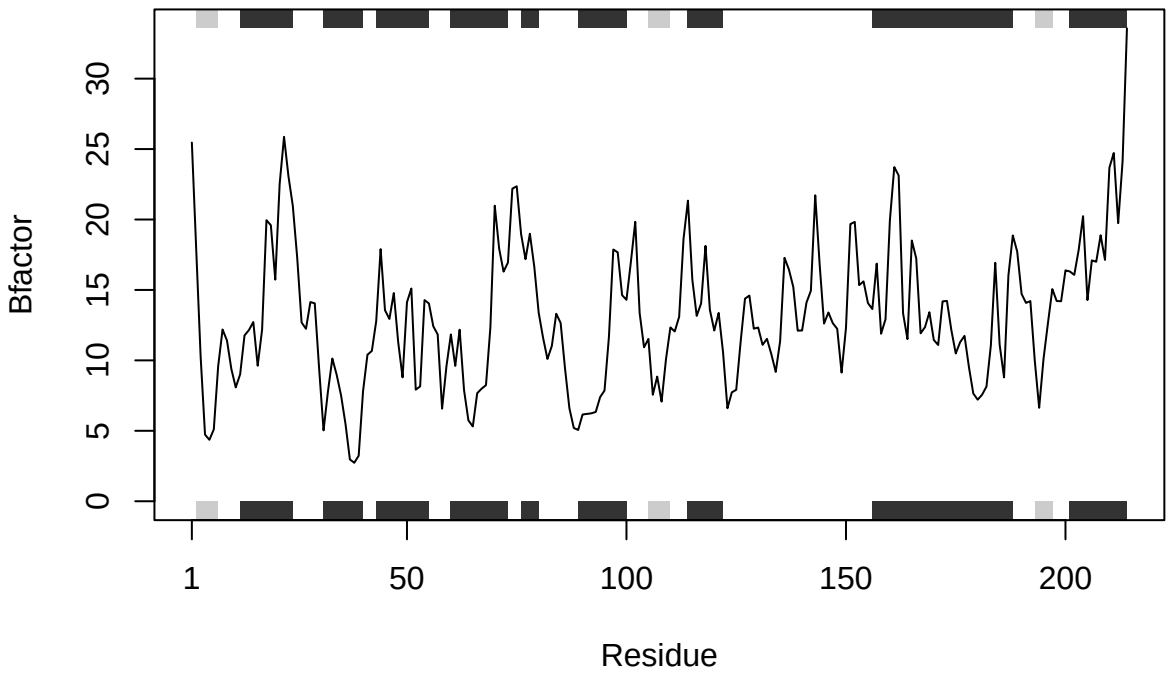
```
plotb3(s1.b, sse=s1.chainA, typ="l", ylab="Bfactor")
```




```
plotb3(s2.b, sse=s2.chainA, typ="l", ylab="Bfactor")
```




```
plotb3(s3.b, sse=s3.chainA, typ="l", ylab="Bfactor")
```



Q1. What type of object is returned from the read.pdb() function?

The read.pdb() function returns a list containing 8 elements.

Q2. What does the trim.pdb() function do?

The trim.pdb() function filters the pdb file. In our case, it filters the pdb file to only show data that has an A for the chain column.

Q3. What input parameter would turn off the marginal black and grey rectangles in the plots and what do they represent in this case?

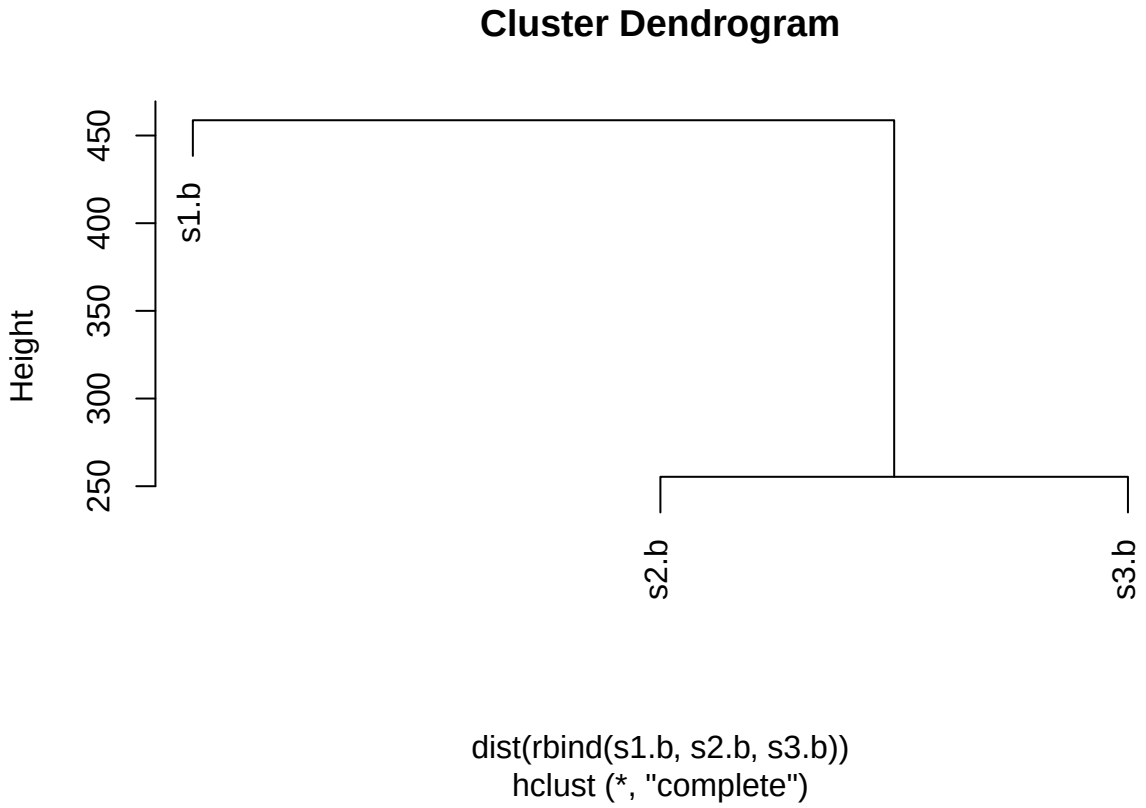
To turn off the marginal black and grey rectangles, add "top = FALSE" and "bot = FALSE" to the argument of plotb3. The black and grey rectangles represent the secondary structure of the protein. The black rectangles represent alpha helices, and the grey represent beta strands.

Q4. What would be a better plot to compare across the different proteins?

A better plot might be one that plots the B-factors of all the different proteins into a single graph.

Q5. Which proteins are more similar to each other in their B-factor trends. How could you quantify this?

```
hc <- hclust(dist(rbind(s1.b, s2.b, s3.b)))
plot(hc)
```



s2 and s3 are more similar to each other.

Improving the Analysis Code

Q6. How would you generalize the original code above to work with any set of input protein structures?

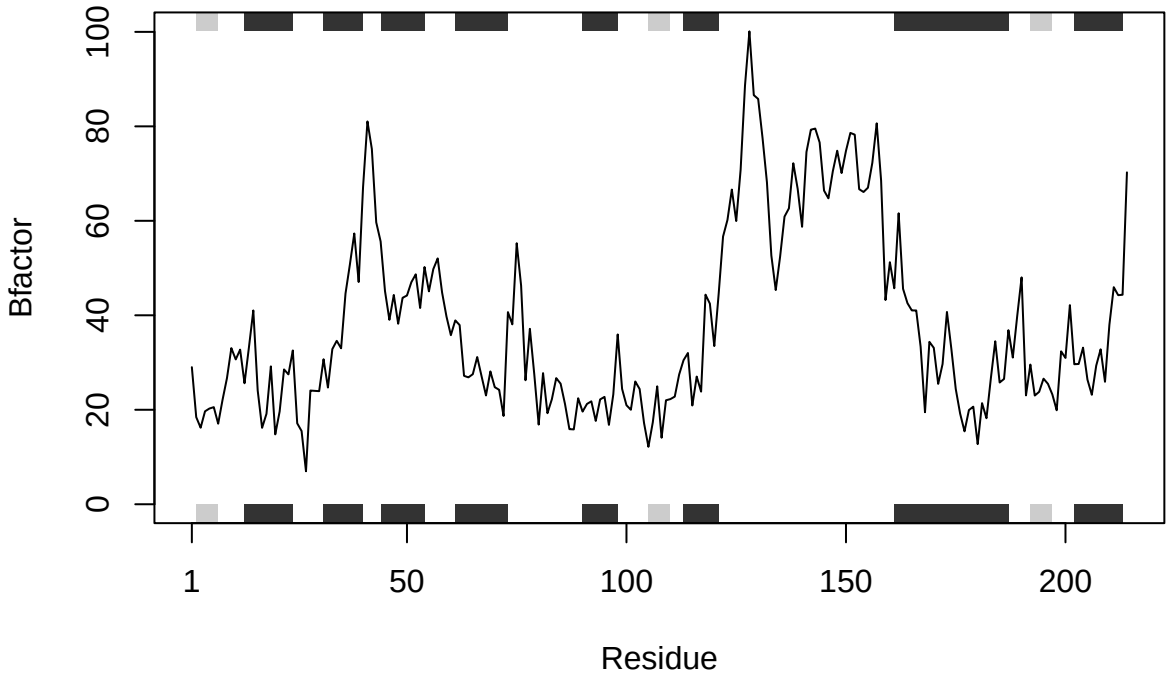
```
#' Generate a plot of 'B-factor vs residue' for residues from
#' a specific chain of a protein by providing its PDB accession code and specified chain.
#' and specified chain.
#'
#' @param x A PDB accession code (put the accession code in quotes)
#' @param y Chain of interest (in quotes)
#'
#' @return Plot of 'B-factor vs residue' for residues in a specified chain of the specified protein.
#'          in a specified chain of the specified protein.
#' @export
```

```
#'
#' @examples
#' plotpdb("2SPL", "A")

plotpdb <- function(x, y) {
  #This loads the bio3d package
  library(bio3d)

  #This reads the PDB file
  prot <- read.pdb(x)

  #This filters for residues belonging to a specific chain
  prot.chain <- trim.pdb(prot, chain = y, elety = "CA")

  #This obtains the B-factor values for the chain "y" residues
  prot.b <- prot.chain$atom$b

  #This plots 'B-factor vs residue' for the chain "y" residues
  plotb3(prot.b, sse = prot.chain, typ="l", ylab = "Bfactor")
}
```

Testing the code on s1 protein (4AKE) from the original code.

```
plotpdb("4AKE", "A")
```

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

## Warning in get.pdb(file, path = tempdir(), verbose = FALSE): C:
## \Users\cindy\AppData\Local\Temp\RtmpoptyRo/4AKE.pdb exists. Skipping download
```



The plot matches what was obtained from the original code. The plotpdb() function works!