# HW Class 6: R Functions

Xiaoxuan Teng (PID: A69028742)

2024-01-29

## Section 1: Improving analysis code by writing functions

### A

```r
# (A. Can you improve this analysis code?
# Also consider copy and paste error in df$b and df$d
df <- data.frame(a=1:10, b=seq(200,400,length=10),c=11:20,d=NA)
df$a <- (df$a - min(df$a)) / (max(df$a) - min(df$a))
df$b <- (df$b - min(df$b)) / (max(df$b) - min(df$b))
df$c <- (df$c - min(df$c)) / (max(df$c) - min(df$c))
df$d <- (df$d - min(df$d)) / (max(df$d) - min(df$d))
```

```r
# Write function to perform normalization
Min_Max_Normalization <- function(x){
  (x - min(x))/(max(x) - min(x)) # min_max_normalization=(value-min)/(max-min)
}
```

```r
# Apply normalization function to each column
apply(df, 2, Min_Max_Normalization)
```

```
             a         b         c  d
[1,] 0.0000000 0.0000000 0.0000000 NA
[2,] 0.1111111 0.1111111 0.1111111 NA
[3,] 0.2222222 0.2222222 0.2222222 NA
[4,] 0.3333333 0.3333333 0.3333333 NA
[5,] 0.4444444 0.4444444 0.4444444 NA
[6,] 0.5555556 0.5555556 0.5555556 NA
[7,] 0.6666667 0.6666667 0.6666667 NA
```

```
 [8,] 0.7777778 0.7777778 0.7777778 NA
 [9,] 0.8888889 0.8888889 0.8888889 NA
[10,] 1.0000000 1.0000000 1.0000000 NA
```

**B**

```
# install.packages("bio3d")
# Can you improve this analysis code?
library(bio3d)
s1 <- read.pdb("4AKE") # kinase with drug
```

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

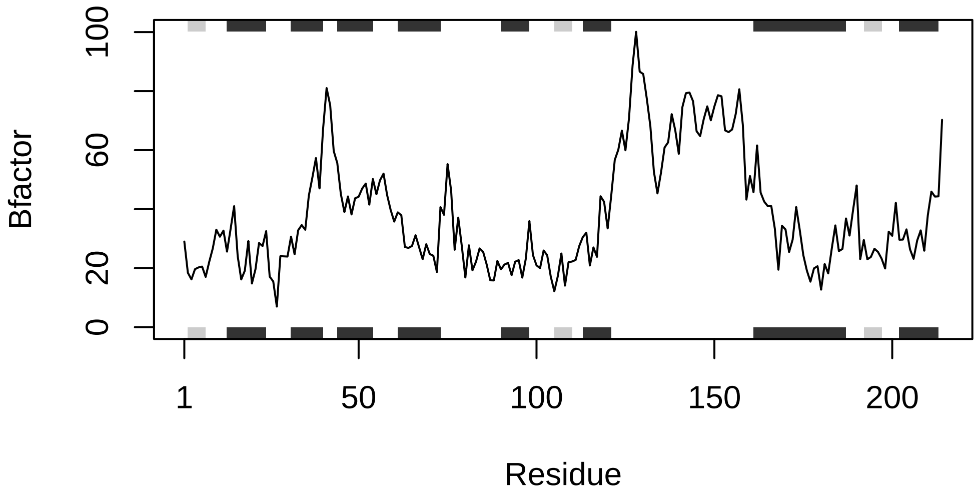
```
s2 <- read.pdb("1AKE") # kinase no drug
```

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

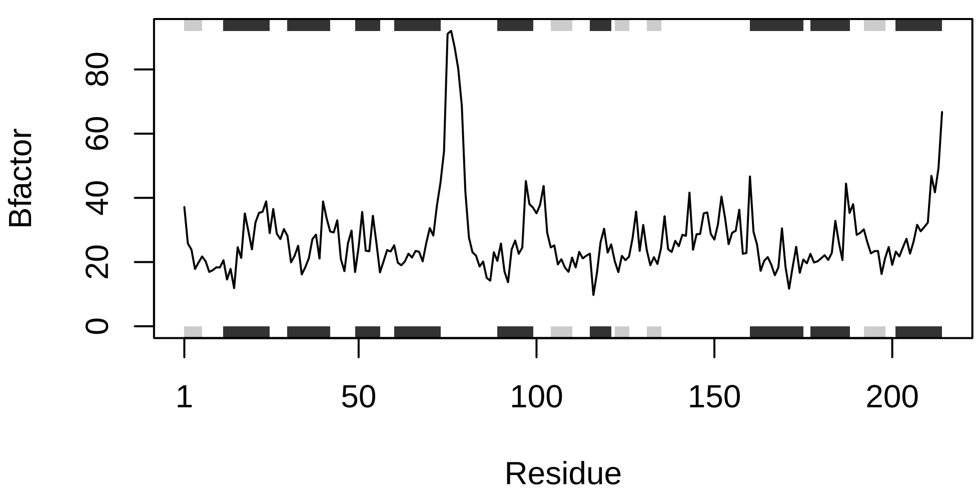
```
s3 <- read.pdb("1E4Y") # kinase with drug
```

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

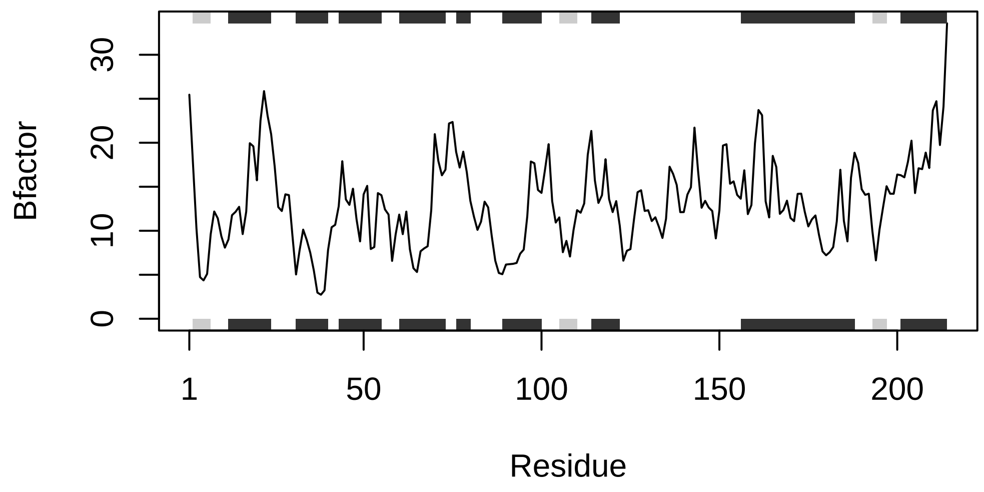
```
s1.chainA <- trim.pdb(s1, chain="A", elety="CA")
s2.chainA <- trim.pdb(s2, chain="A", elety="CA")
s3.chainA <- trim.pdb(s3, chain="A", elety="CA")
s1.b <- s1.chainA$atom$b
s2.b <- s2.chainA$atom$b
s3.b <- s3.chainA$atom$b
plotb3(s1.b, sse=s1.chainA, typ="l", ylab="Bfactor")
```



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



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

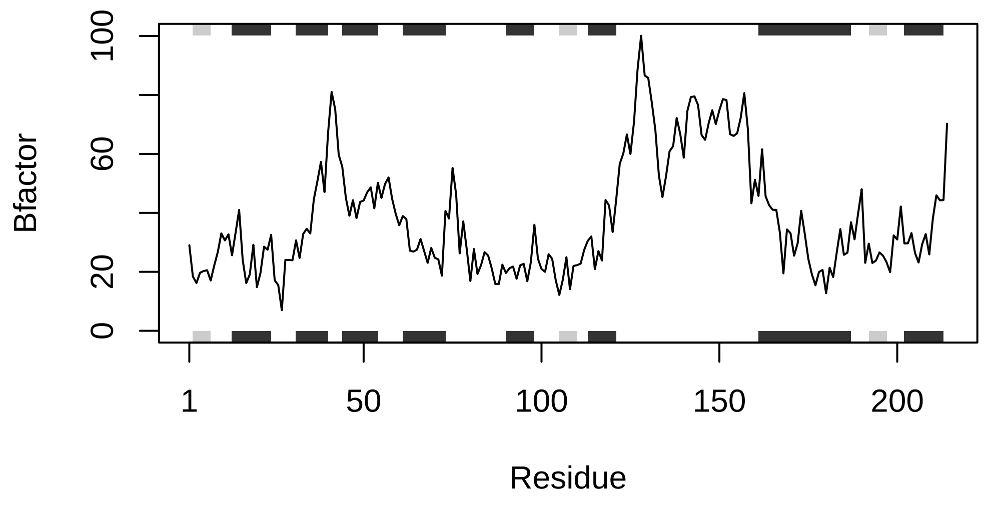


```
# Write function to read, trim PDB file, and plot B factor
# Arguments: PDB ID, chian identifier, and atom type
plot_pdb <- function(pdbcode, chain, elety){
  # there are more than 1 pdb ID, so will run this function in loop
  for (x in pdbcode){
    # read a PDB file by read.pdb()
    s <- read.pdb(x)
    # create a new PDB object based on our selection of backbone atoms by trim.pdb()
    s.chain <- trim.pdb(s, chain = chain, elety = elety)
    # access the b-factor column
    s.b <- s.chain$atom$b
    # plot b-factors with sse annotation added to linear plot
    plotb3(s.b, sse = s.chain, type = "l", ylab = "Bfactor")
  }
}
```

```
# Inputs: pdb code, chian identifier, and atom type
pdbcode <- c("4AKE", "1AKE", "1E4Y")
chain <- "A"
```

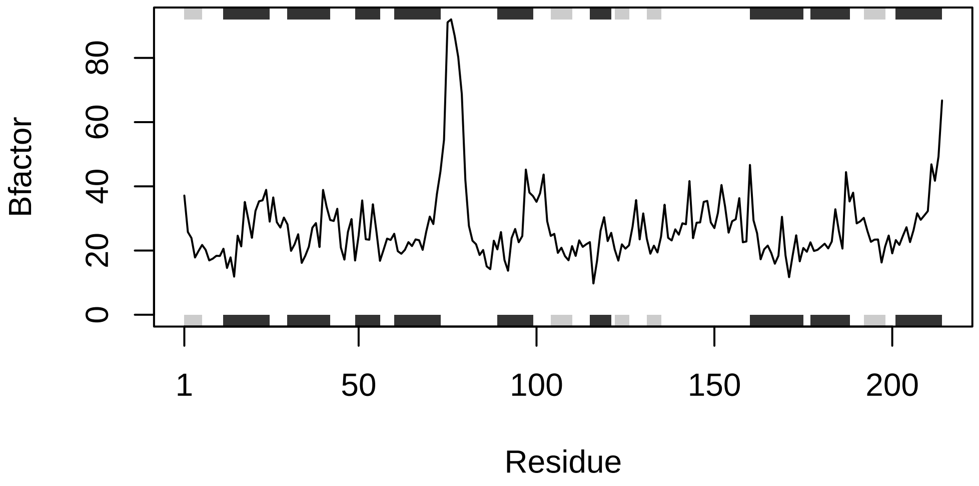
```
elety <- "CA"

# The output of the function are plots of B-factor vs chain A residue from our list of pro
plot_pdb(pdbcode, chain, elety)
```

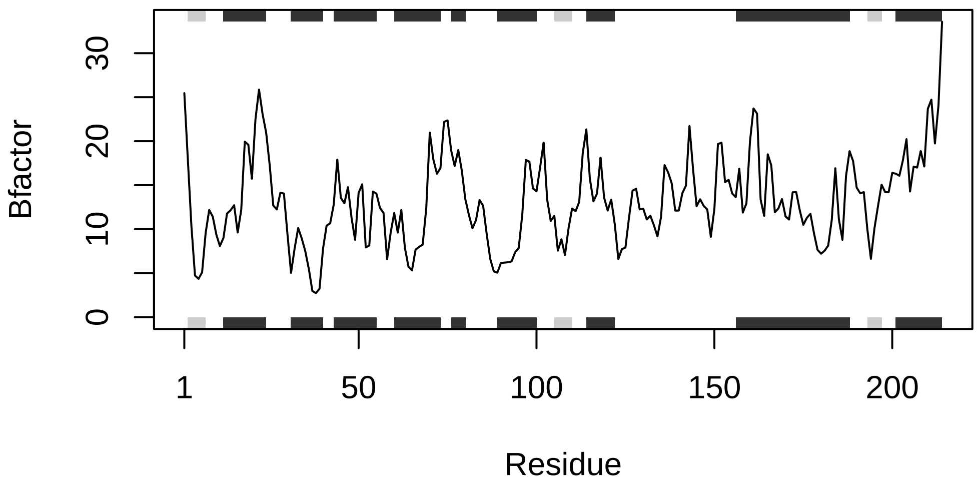
```
  Note: Accessing on-line PDB file

Warning in get.pdb(file, path = tempdir(), verbose = FALSE):
/var/folders/8f/2dcl0ls90c9fwxxs0_dz20bh0000gp/T//RtmpLP7ZBk/4AKE.pdb exists.
Skipping download

  Note: Accessing on-line PDB file

Warning in get.pdb(file, path = tempdir(), verbose = FALSE):
/var/folders/8f/2dcl0ls90c9fwxxs0_dz20bh0000gp/T//RtmpLP7ZBk/1AKE.pdb exists.
Skipping download
```



```
   PDB has ALT records, taking A only, rm.alt=TRUE

  Note: Accessing on-line PDB file
```

```
Warning in get.pdb(file, path = tempdir(), verbose = FALSE):
/var/folders/8f/2dcl0ls90c9fwxxs0_dz20bh0000gp/T//RtmpLP7ZBk/1E4Y.pdb exists.
Skipping download
```





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

**Answer: The 'read.pdb()' function returns a list of class "pdb" with several components. This includes atom, xyz, seqres, helix, sheet, calpha, remark, and call.**

```
str(s1)
```

```
List of 8
 $ atom  :'data.frame': 3459 obs. of  16 variables:
  ..$ type  : chr [1:3459] "ATOM" "ATOM" "ATOM" "ATOM" ...
  ..$ eleno : int [1:3459] 1 2 3 4 5 6 7 8 9 10 ...
  ..$ elety : chr [1:3459] "N" "CA" "C" "O" ...
  ..$ alt   : chr [1:3459] NA NA NA NA ...
  ..$ resid : chr [1:3459] "MET" "MET" "MET" "MET" ...
  ..$ chain : chr [1:3459] "A" "A" "A" "A" ...
  ..$ resno : int [1:3459] 1 1 1 1 1 1 1 1 2 2 ...
  ..$ insert: chr [1:3459] NA NA NA NA ...
  ..$ x     : num [1:3459] -10.93 -9.9 -9.17 -9.8 -10.59 ...
  ..$ y     : num [1:3459] -24.9 -24.4 -23.3 -22.3 -24 ...
  ..$ z     : num [1:3459] -9.52 -10.48 -9.81 -9.35 -11.77 ...
  ..$ o     : num [1:3459] 1 1 1 1 1 1 1 1 1 1 ...
  ..$ b     : num [1:3459] 41.5 29 27.9 26.4 34.2 ...
  ..$ segid : chr [1:3459] NA NA NA NA ...
  ..$ elesy : chr [1:3459] "N" "C" "C" "O" ...
  ..$ charge: chr [1:3459] NA NA NA NA ...
 $ xyz   : 'xyz' num [1, 1:10377] -10.93 -24.89 -9.52 -9.9 -24.42 ...
 $ seqres: Named chr [1:428] "MET" "ARG" "ILE" "ILE" ...
  ..- attr(*, "names")= chr [1:428] "A" "A" "A" "A" ...
 $ helix :List of 4
  ..$ start: Named num [1:19] 13 31 44 61 75 90 113 161 202 13 ...
  .. ..- attr(*, "names")= chr [1:19] "" "" "" "" ...
  ..$ end  : Named num [1:19] 24 40 54 73 77 98 121 187 213 24 ...
  .. ..- attr(*, "names")= chr [1:19] "" "" "" "" ...
  ..$ chain: chr [1:19] "A" "A" "A" "A" ...
  ..$ type : chr [1:19] "5" "1" "1" "1" ...
 $ sheet :List of 4
  ..$ start: Named num [1:14] 192 105 2 81 27 123 131 192 105 2 ...
  .. ..- attr(*, "names")= chr [1:14] "" "" "" "" ...
  ..$ end  : Named num [1:14] 197 110 7 84 29 126 134 197 110 7 ...
  .. ..- attr(*, "names")= chr [1:14] "" "" "" "" ...
  ..$ chain: chr [1:14] "A" "A" "A" "A" ...
  ..$ sense: chr [1:14] "0" "1" "1" "1" ...
 $ calpha: logi [1:3459] FALSE TRUE FALSE FALSE FALSE FALSE ...
 $ remark:List of 1
  ..$ biomat:List of 4
  .. ..$ num   : int 1
  .. ..$ chain :List of 1
  .. .. ..$ : chr [1:2] "A" "B"
  .. ..$ mat   :List of 1
  .. .. ..$ :List of 1
  .. .. .. ..$ A B: num [1:3, 1:4] 1 0 0 0 1 0 0 0 1 0 ...
  .. ..$ method: chr "AUTHOR"
 $ call  : language read.pdb(file = "4AKE")
 - attr(*, "class")= chr [1:2] "pdb" "sse"
```

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

**Answer: 'trim.pdb()' function trim a pdb object to a subset of atoms and produce a new smaller PDB object based on our selection of atoms. In our preview code, we used 'trim.pdb()' function to extract only the alpha carbon atoms "CA" of chain A from our protein of interest.**

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

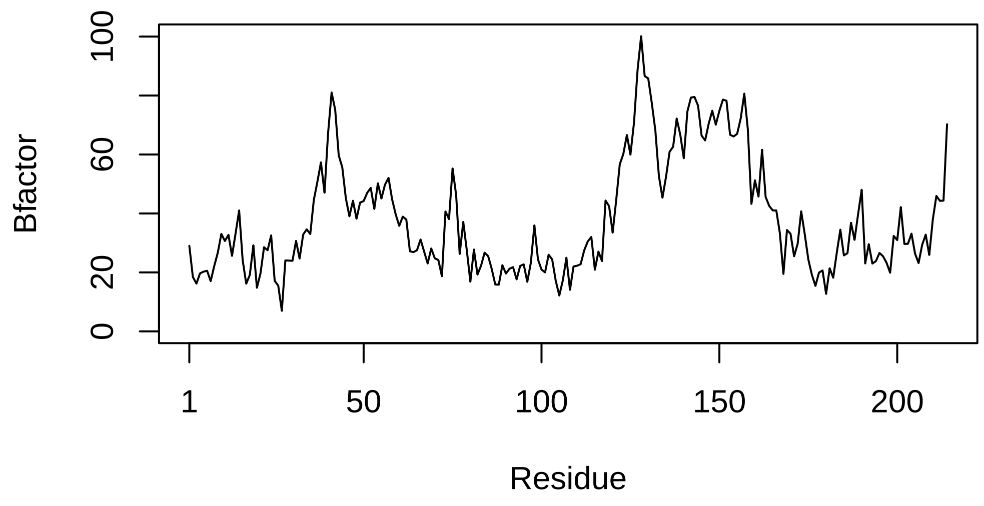
**Answer: set "sse = NULL" parameter would turn off the marginal rectangles. The "sse" parameter provides the information about the protein secondary structure, with black marginal rectangles represent alpha helix and gery indicates beta strand regions.**

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

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

```
Warning in get.pdb(file, path = tempdir(), verbose = FALSE):
/var/folders/8f/2dcl0ls90c9fwxxs0_dz20bh0000gp/T//RtmpLP7ZBk/4AKE.pdb exists.
Skipping download
```

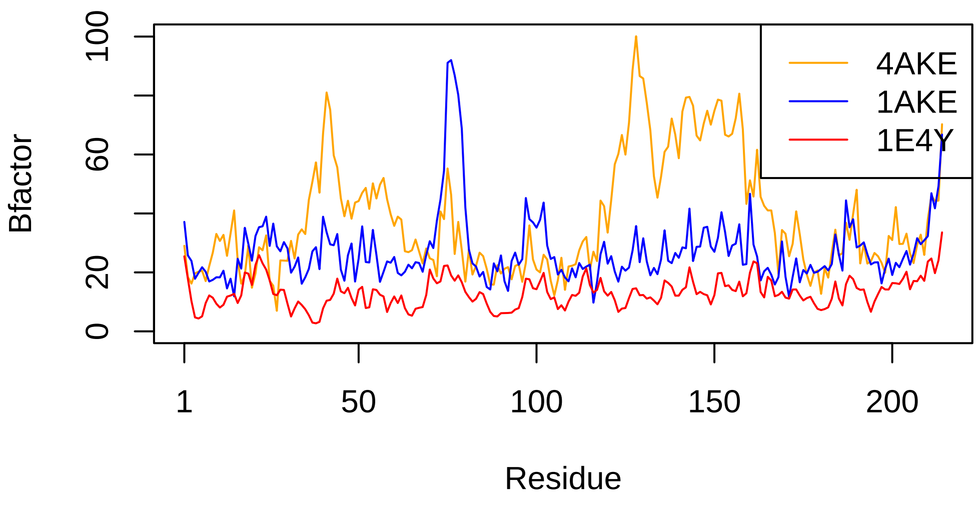
```
s1.chainA <- trim.pdb(s1, chain="A", elety="CA")
s1.b <- s1.chainA$atom$b
plotb3(s1.b, sse=NULL, typ="l", ylab="Bfactor")
```



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

**Answer: A better plot would be a overlay line plot with all three proteins plotted in the same graph.**

```
plotb3(s1.b, typ="l", ylab="Bfactor", col = "orange", ylim = range(c(s1.b, s2.b, s3.b)))
lines(s2.b, typ = "l", col = "blue")
lines(s3.b, typ = "l", col = "red")
legend("topright", legend = c("4AKE", "1AKE", "1E4Y"), col = c("orange", "blue", "red"), l
```



**Q5.** Which proteins are more similar to each other in their B-factor trends. How could you quantify this? HINT: try the rbind(), dist() and hclust() functions together with a resulting dendrogram plot. Look up the documentation to see what each of these functions does.

**Answer: 'rbind()' function combines the B-factor data from different proteins by rows. 'dist()' function computes and returns the distance matrix based on their B-factors. 'hclust()' function performs hierarchical cluster analysis on the dissimilarity structures produced by dist().**

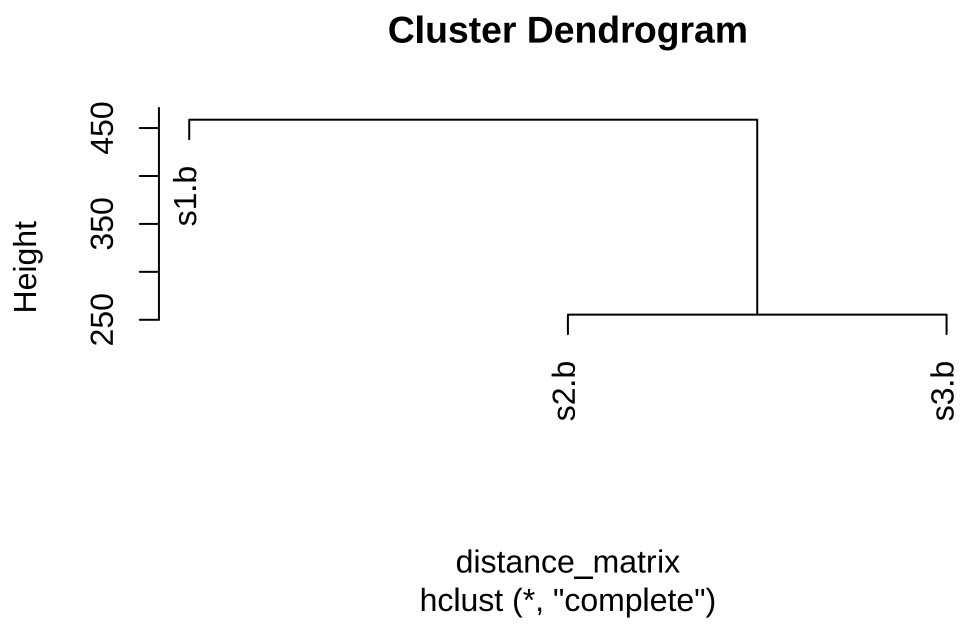
**From the cluster dendrogram, we can tell that protein s2 (1AKE) and s3 (1E4Y) are more similar to each other in terms of their B-factor trends as they have been grouped closer in the dendrogram.**

**The heights of which two objects are joined together represent the distance between two clusters. Lower heights mean more similarity. This could be quantified by the 'dist()' function, that generates the distance between any two samples.**

```
distance_matrix <- dist(rbind(s1.b, s2.b, s3.b))
distance_matrix
```

```
         s1.b     s2.b
s2.b 340.4926         
s3.b 458.7081 255.3615
```

```
hc <- hclust(distance_matrix)
plot(hc)
```

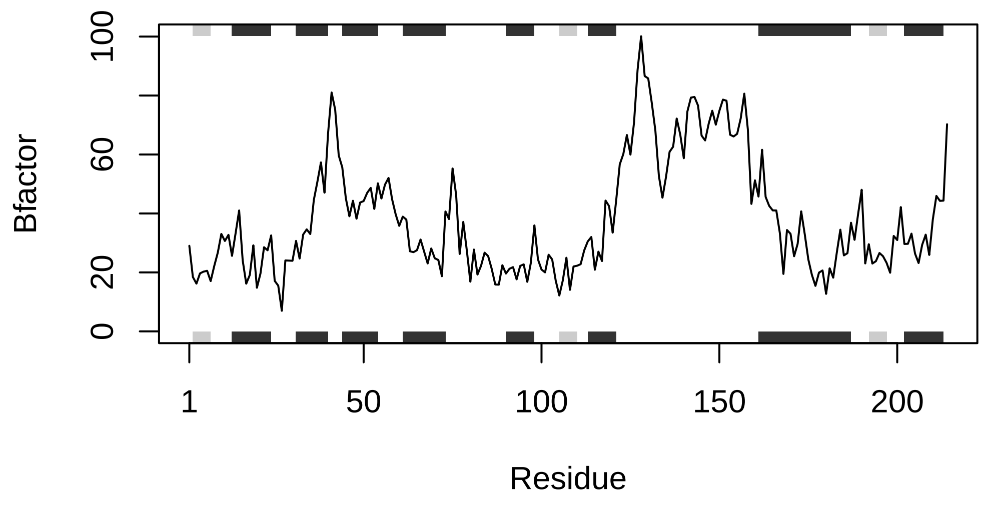


**Q6.** How would you generalize the original code above to work with any set of input protein structures?

**Answer: I need to define a function with input arguments of a list of pdb codes, chian identifier, atom type, and/or factor. Then process each one in loop.**

```
# Write function to read, trim PDB file, and plot B factor
# Arguments: PDB ID, chian identifier, and atom type
plot_pdb <- function(pdbcode, chain, elety, factor){
  # there are more than 1 pdb ID, so will run this function in loop
  for (x in pdbcode){
    # read a PDB file by read.pdb()
    s <- read.pdb(x)
    # create a new PDB object based on our selection of backbone atoms by trim.pdb()
    s.chain <- trim.pdb(s, chain = chain, elety = elety)
    # access the b-factor column
    s.atom <- s.chain$atom
    s.b <- s.atom[,factor]
    # plot b-factors with sse annotation added to linear plot
    plotb3(s.b, sse = s.chain, type = "l", ylab = "Bfactor")
  }
}

# Inputs: pdb code, chian identifier, and atom type
pdbcode <- c("4AKE", "1AKE", "1E4Y")
chain <- "A"
elety <- "CA"
factor <- "b"

# The output of the function are plots of B-factor vs chain A residue from our list of pro
plot_pdb(pdbcode, chain, elety, factor)
```

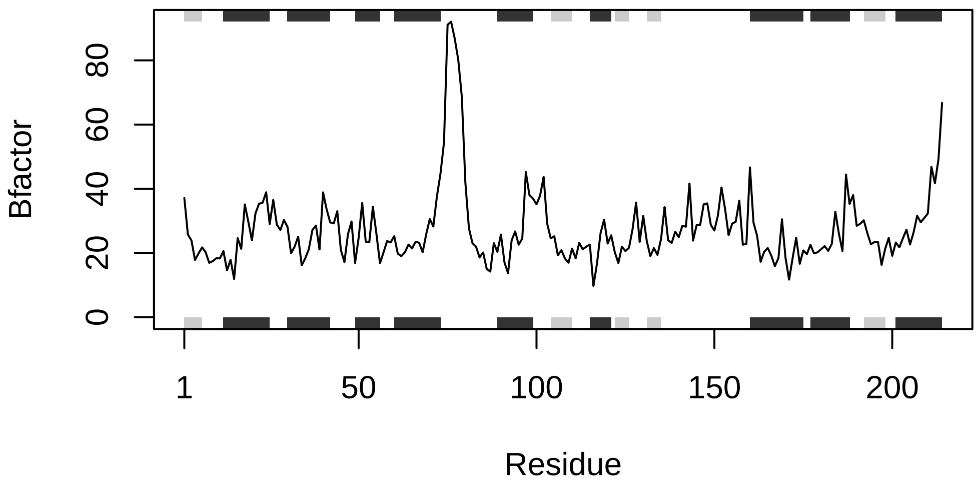
```
  Note: Accessing on-line PDB file

Warning in get.pdb(file, path = tempdir(), verbose = FALSE):
/var/folders/8f/2dcl0ls90c9fwxxs0_dz20bh0000gp/T//RtmpLP7ZBk/4AKE.pdb exists.
Skipping download

  Note: Accessing on-line PDB file

Warning in get.pdb(file, path = tempdir(), verbose = FALSE):
/var/folders/8f/2dcl0ls90c9fwxxs0_dz20bh0000gp/T//RtmpLP7ZBk/1AKE.pdb exists.
Skipping download
```

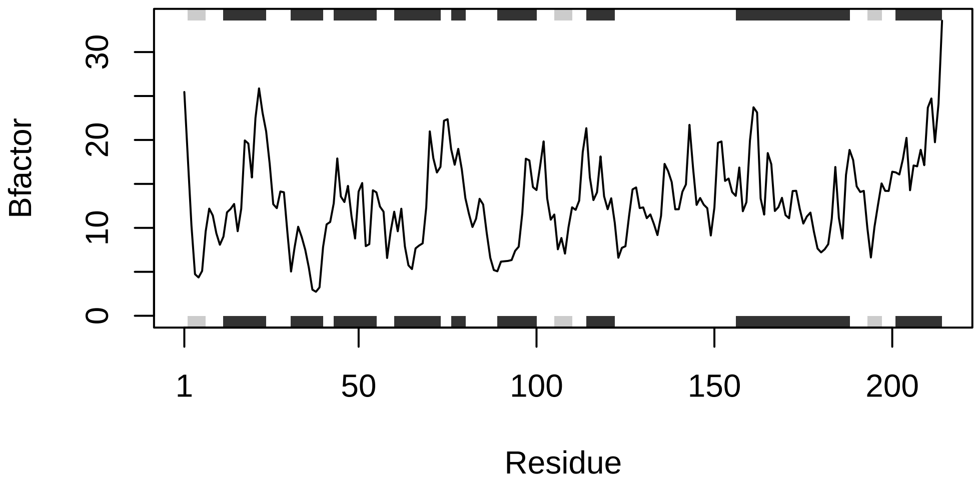


```
   PDB has ALT records, taking A only, rm.alt=TRUE
```

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

```
Warning in get.pdb(file, path = tempdir(), verbose = FALSE):
/var/folders/8f/2dcl0ls90c9fwxxs0_dz20bh0000gp/T//RtmpLP7ZBk/1E4Y.pdb exists.
Skipping download
```


Bfactor
0
20
40
60
80
1
50
100
150
200
Residue


Bfactor
0
10
20
30
1
50
100
150
200
Residue