Homework 6

Brittney Hayes

2024-02-01

Section 1: Improving Analysis Code by Writing Functions

A) Improving Analysis Code

Original Code:

```r
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$b - min(df$a)) / (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$a) - min(df$d))
```

Improved Code:

```r
#Create a data frame
df <- data.frame(a = 1:10, b = seq(200, 400, length = 10), c = 11:20, d = NA)

#Define a function utilizing min and max
min_max_normalize <- function(x) {
  (x - min(x, na.rm = TRUE)) / (max(x, na.rm = TRUE) - min(x, na.rm = TRUE))
}

df
```

```
##     a        b  c  d
## 1   1 200.0000 11 NA
## 2   2 222.2222 12 NA
## 3   3 244.4444 13 NA
## 4   4 266.6667 14 NA
## 5   5 288.8889 15 NA
## 6   6 311.1111 16 NA
## 7   7 333.3333 17 NA
## 8   8 355.5556 18 NA
## 9   9 377.7778 19 NA
## 10 10 400.0000 20 NA
```

B) Improving Analysis Code Using Bio3d

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

Original Code:

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

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

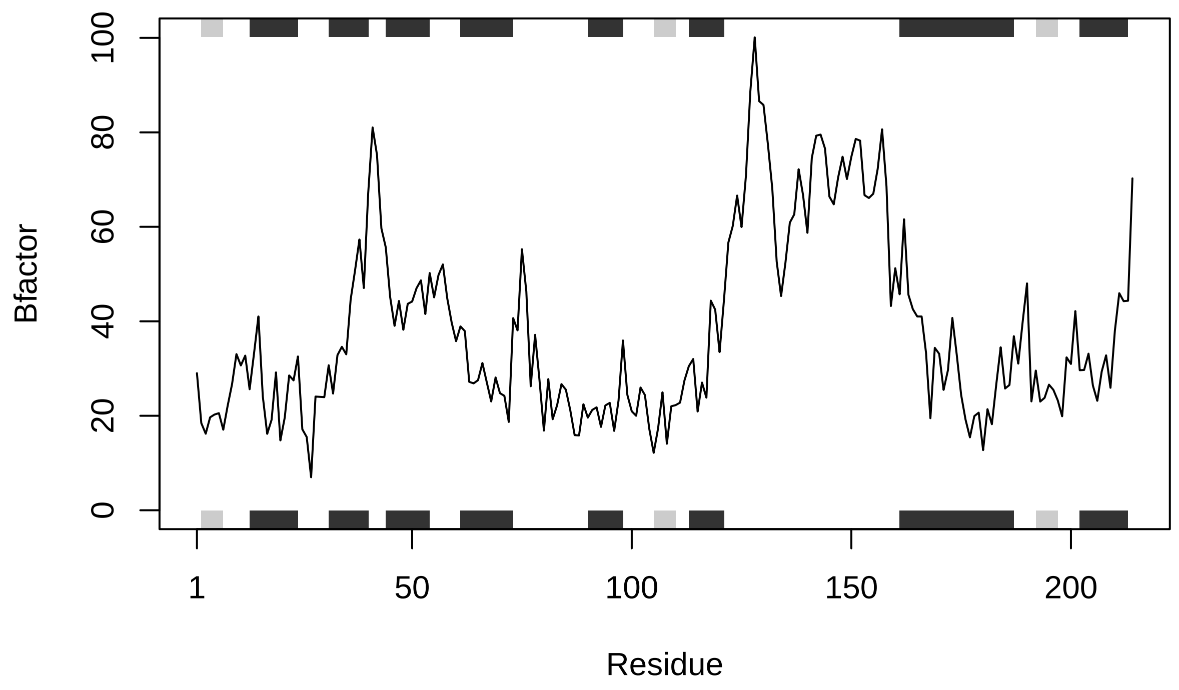
```r
s2 <- read.pdb("1AKE") # kinase no drug
```

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

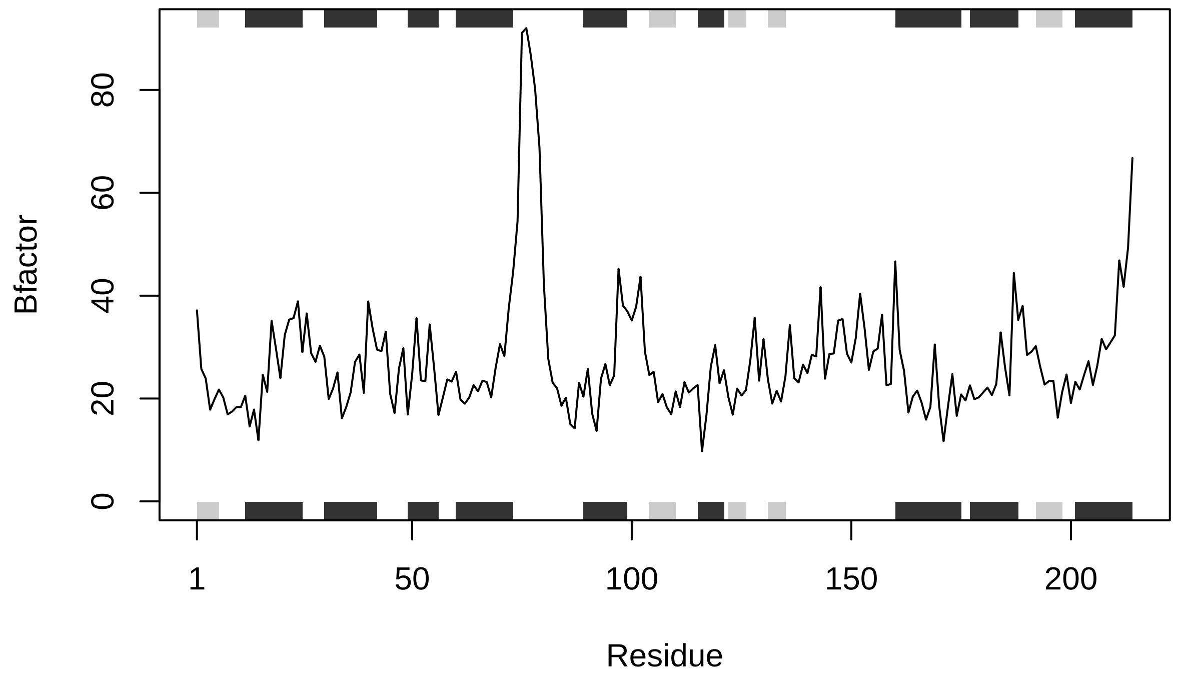
```r
s3 <- read.pdb("1E4Y") # kinase with drug
```

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

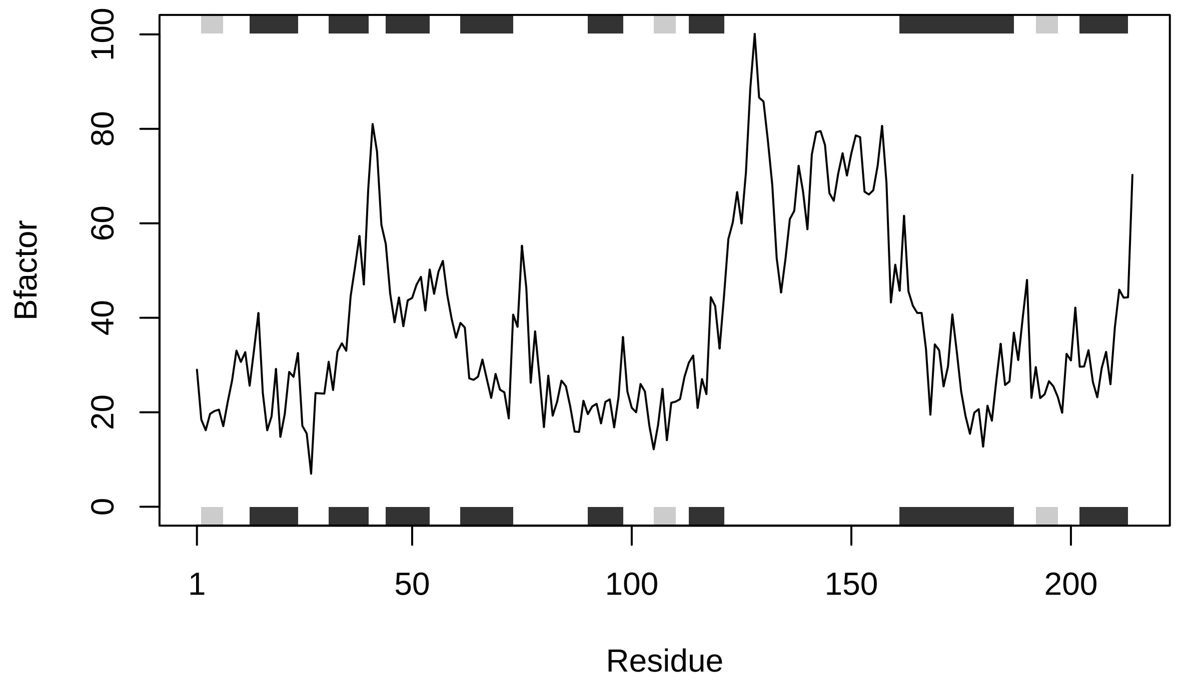
```r
s1.chainA <- trim.pdb(s1, chain="A", elety="CA")
s2.chainA <- trim.pdb(s2, chain="A", elety="CA")
s3.chainA <- trim.pdb(s1, chain="A", elety="CA")
s1.b <- s1.chainA$atom$b
s2.b <- s2.chainA$atom$b
s3.b <- s3.chainA$atom$b
plotb3(s1.b, sse=s1.chainA, typ="l", ylab="Bfactor")
```



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



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



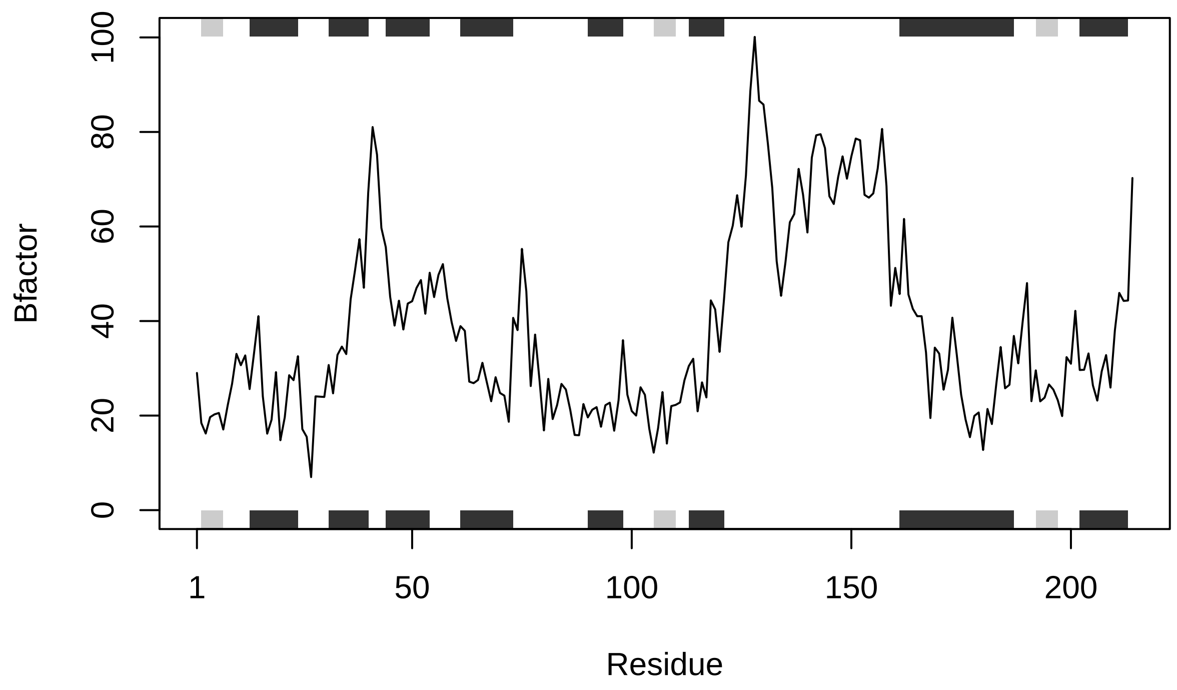
Improved Code:

```
library(bio3d)

#Define a function for analyzing protein drug interactions
analyze_protein_drug_interaction <- function(pdb_file, chain_id = "A", elety_type = "CA") {

#Read PDB file
pdb_data <- read.pdb(pdb_file)

#Trim PDB data
trimmed_data <- trim.pdb(pdb_data, chain = chain_id, elety = elety_type)

#Extract B-factor values
b_factors <- trimmed_data$atom$b

#Plot B-factor values
plotb3(b_factors, sse = trimmed_data, typ = "l", ylab = "Bfactor")
}

#Checking code to see if it works
analyze_protein_drug_interaction("4AKE")
```

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

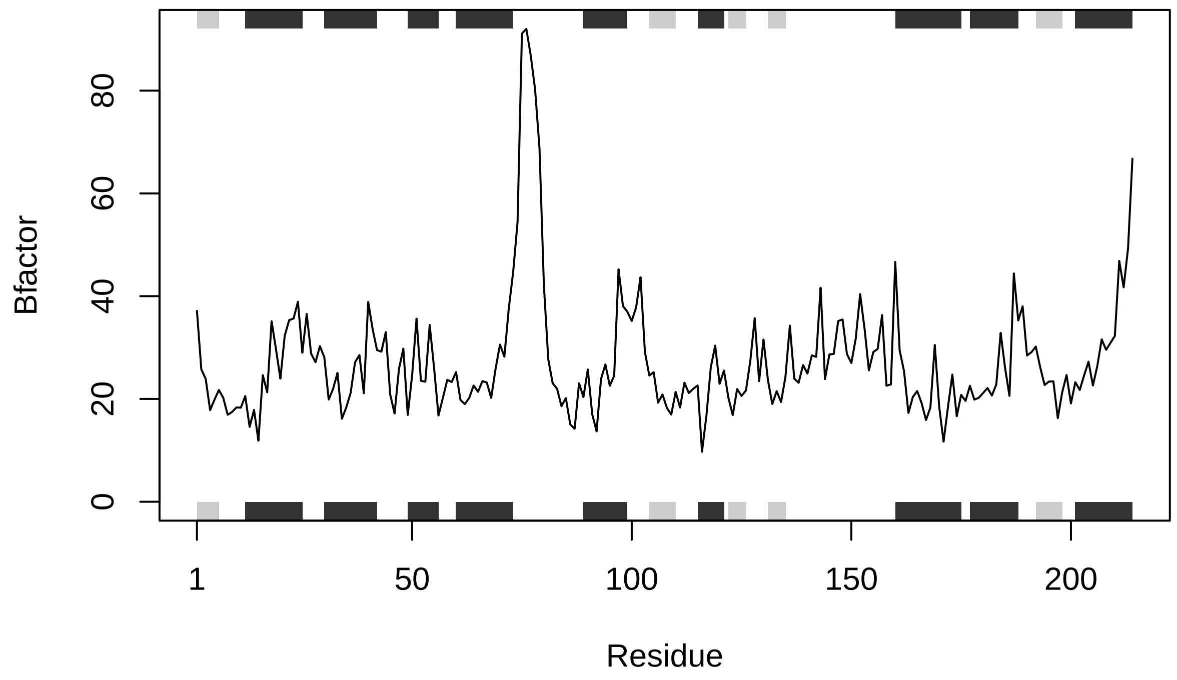
```
## Warning in get.pdb(file, path = tempdir(), verbose = FALSE):
```

```
## /var/folders/4x/1n4r7l7s1ds5_8n783n2v3f00000gn/T//RtmpGK6t0y/4AKE.pdb exists.
## Skipping download
```



```r
analyze_protein_drug_interaction("1AKE")
```

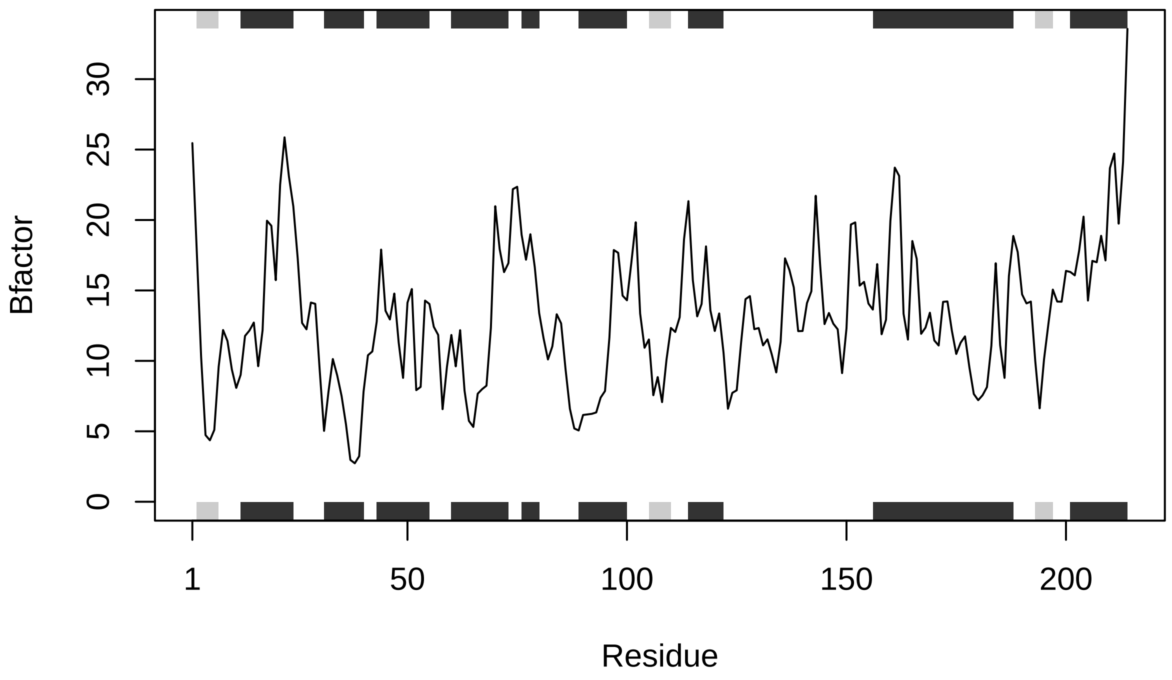
```
##   Note: Accessing on-line PDB file

## Warning in get.pdb(file, path = tempdir(), verbose = FALSE):
## /var/folders/4x/1n4r7l7s1ds5_8n783n2v3f00000gn/T//RtmpGK6t0y/1AKE.pdb exists.
## Skipping download

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



```r
analyze_protein_drug_interaction("1E4Y")
```

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

```
## Warning in get.pdb(file, path = tempdir(), verbose = FALSE):
## /var/folders/4x/1n4r7l7s1ds5_8n783n2v3f00000gn/T//RtmpGK6t0y/1E4Y.pdb exists.
## Skipping download
```



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

The read.pdb() function extracts a protein structure from the Protein Data Bank (PDB) file, which provides information about the structure of a protein.

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

The trim.pdb() function allows the creation of a subset of a protein structure represented by a "pdb" object so that a particular chain, segment, or type of atom may be isolated.

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

The parameter that controls the display of the marginal black and grey rectangles is rect.col. If rect.col is set to NA, the marginal black and grey rectangles in the plots will turn off. In this case, these rectangles represent different types of secondary structure elements.

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

A dendrogram plot or a box plot using the "ggplot2" package would be better for comparing across the various proteins.

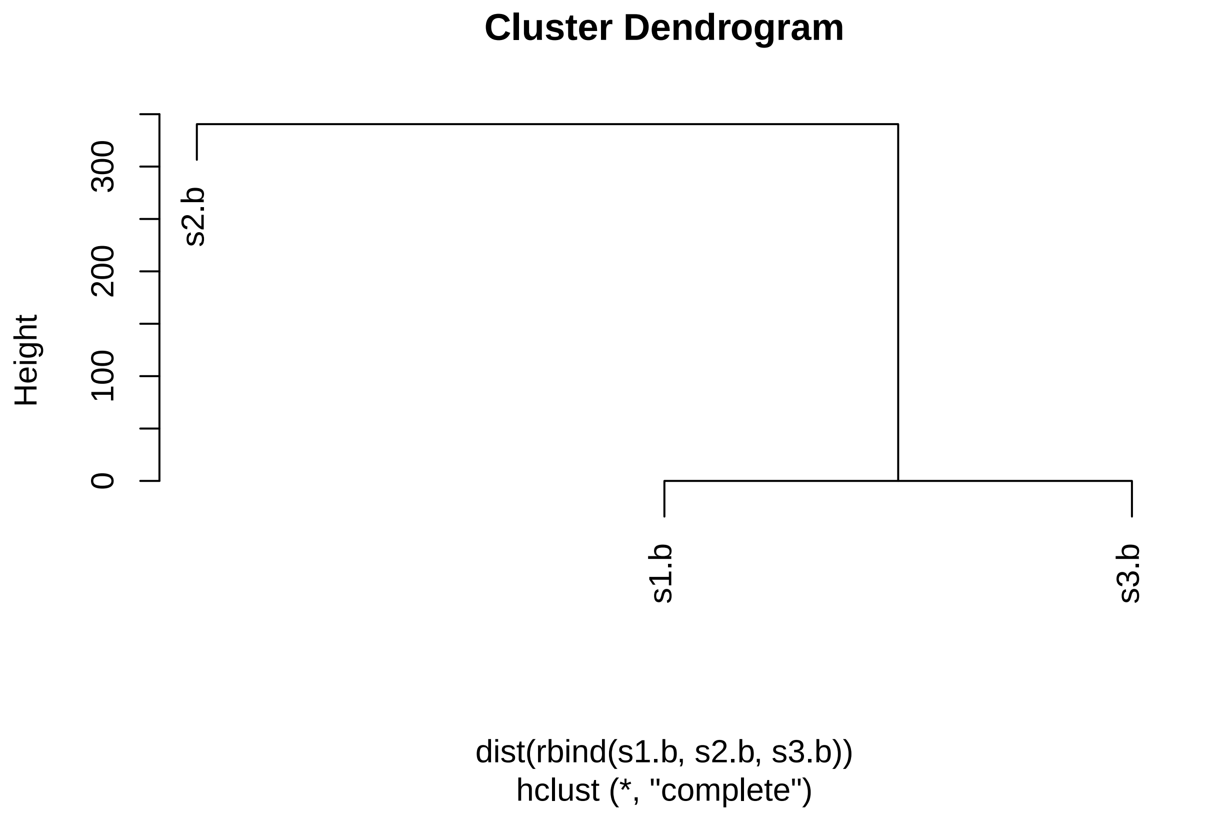
Q5. Which proteins are more similar to each other in their B-factor trends. How could you quantify this?

To quantify the similarity between B-factor trends of different proteins, the correlation coefficients between the protein B-factor vectors may be calculated. Alternatively, you can visualize which B-factor trends are similar by creating a dendrogram plot. S1 and S3 are most similar to each other as they have a correlation coefficient of 1 and are the same height on the dendrogram plot.

```r
#Create a data frame with B-factor values
df <- data.frame(s1 = s1.b, s2 = s2.b, s3 = s3.b)

#Calculate correlation coefficients
cor_matrix <- cor(df)

#Print the correlation matrix
print(cor_matrix)
```

```
##           s1        s2        s3
## s1 1.0000000 0.1843483 1.0000000
## s2 0.1843483 1.0000000 0.1843483
## s3 1.0000000 0.1843483 1.0000000
```

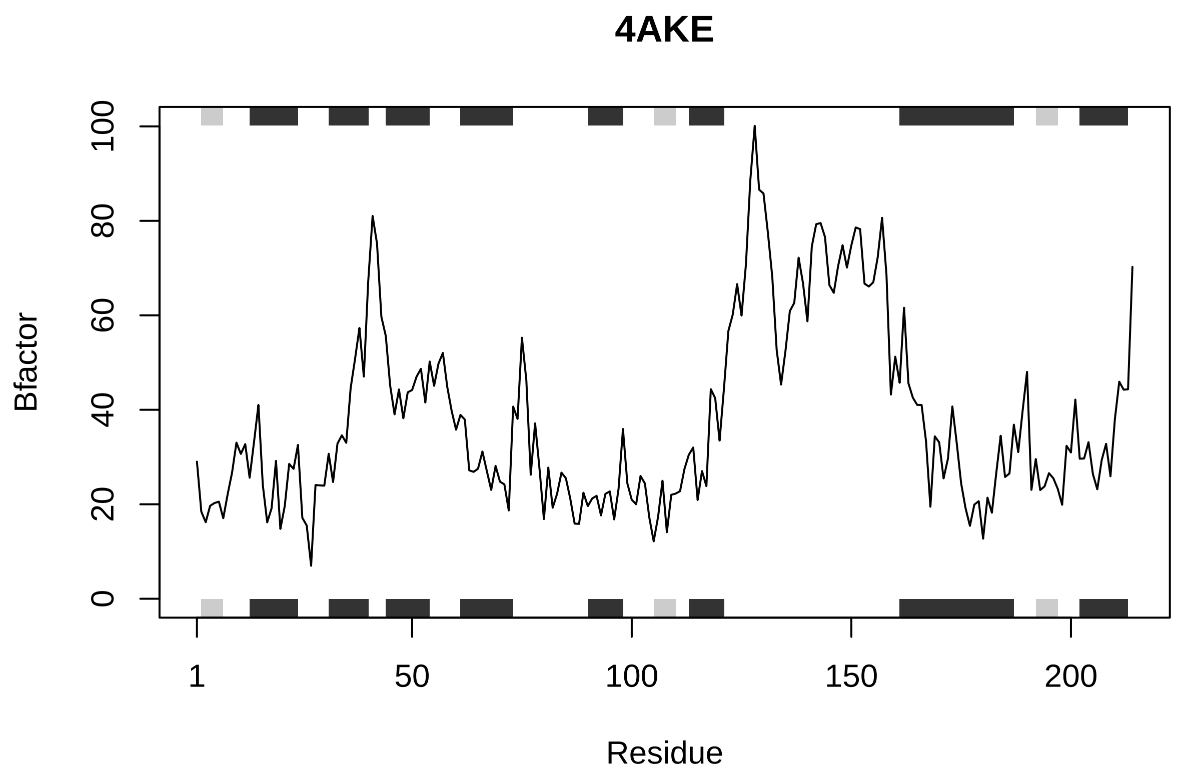
```r
hc <- hclust( dist( rbind(s1.b, s2.b, s3.b) ) )
plot(hc)
```



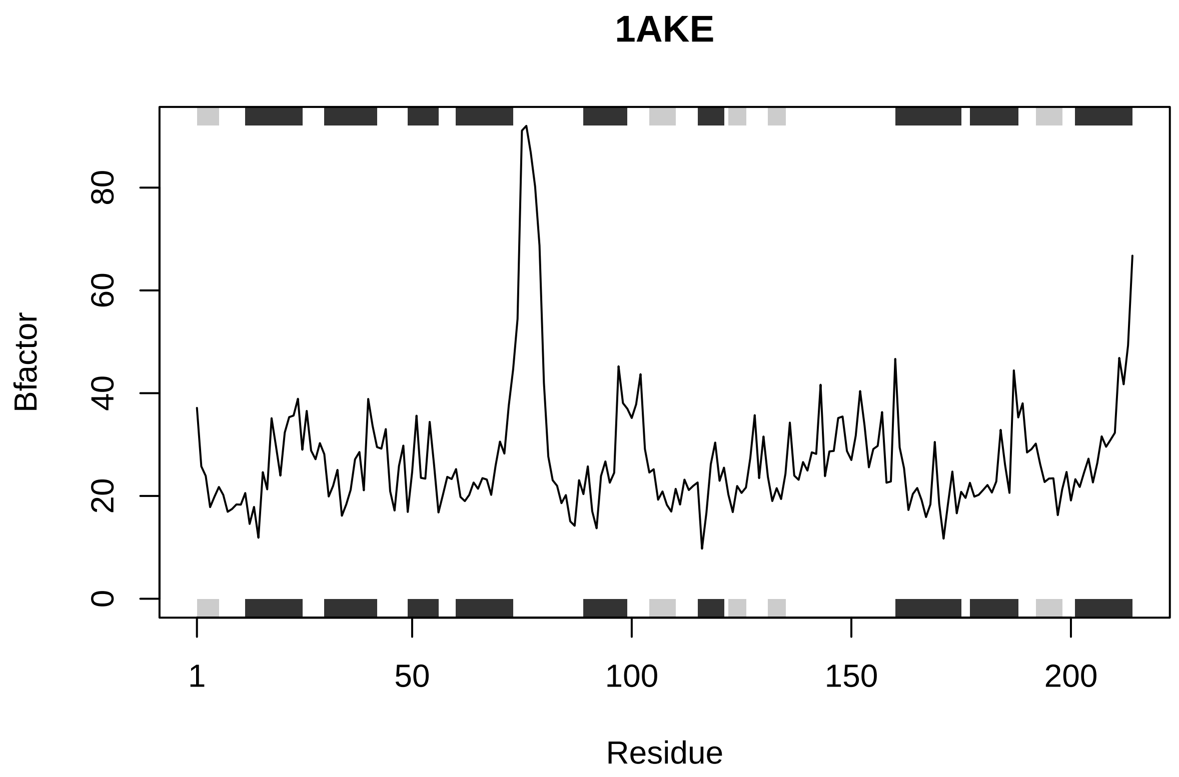
Q6. How would you generalize the original code above to work with any set of input protein structures?

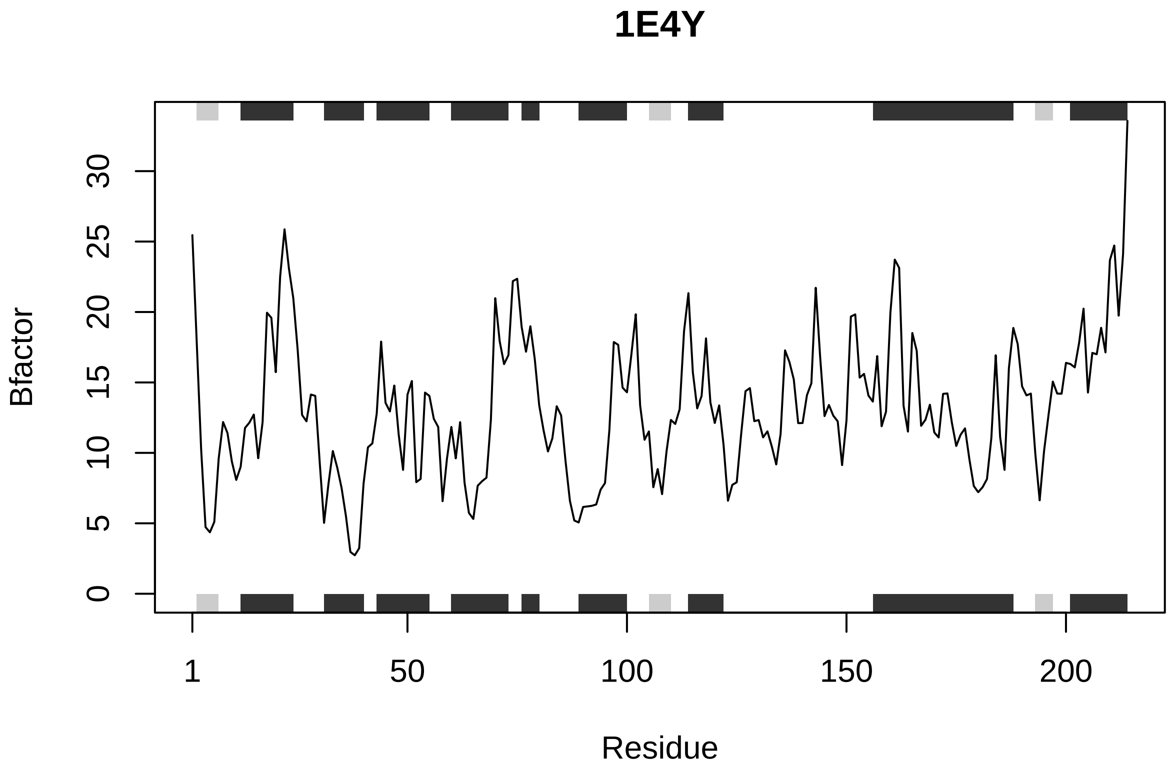
```r
library(bio3d)

#Define a function for analyzing and visualizing B-factor trends
analyze_and_plot_b_factors <- function(pdb_filenames) {

#Create an empty list to store B-factor vectors
b_factors_list <- list()

#Loop through each PDB file
for (pdb_file in pdb_filenames) {

#Read PDB file
pdb_data <- read.pdb(pdb_file)

#Trim PDB data
trimmed_data <- trim.pdb(pdb_data, chain = "A", elety = "CA")

#Extract B-factor values
b_factors <- trimmed_data$atom$b

#Plot B-factor values
plotb3(b_factors, sse = trimmed_data, typ = "l", ylab = "Bfactor", main = pdb_file)

#Store B-factor vector in the list
b_factors_list[[pdb_file]] <- b_factors
  }

#Return the list of B-factor vectors
return(b_factors_list)
}

#Test three protein structures
protein_filenames <- c("4AKE", "1AKE", "1E4Y")
b_factors_list <- analyze_and_plot_b_factors(protein_filenames)
```

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

## Warning in get.pdb(file, path = tempdir(), verbose = FALSE):
## /var/folders/4x/1n4r7l7s1ds5_8n783n2v3f00000gn/T//RtmpGK6t0y/4AKE.pdb exists.
## Skipping download

##   Note: Accessing on-line PDB file

## Warning in get.pdb(file, path = tempdir(), verbose = FALSE):
## /var/folders/4x/1n4r7l7s1ds5_8n783n2v3f00000gn/T//RtmpGK6t0y/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/4x/1n4r7l7s1ds5_8n783n2v3f00000gn/T//RtmpGK6t0y/1E4Y.pdb exists.
## Skipping download
```


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


1E4Y
Bfactor
0
5
10
15
20
25
30
1
50
100
150
200
Residue