Class 06 Homework

Izabelle Querubin

Section 1: Improving analysis code by writing functions

Part A

```
# Can you improve this analysis code?
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))
```

```
# Improving the code

# Recalling df
df <- data.frame(a=1:10, b=seq(200,400,length=10),c=11:20,d=NA, row.names = NULL, check.ro
  FALSE, check.names = TRUE, fix.empty.names = TRUE,stringsAsFactors = FALSE)

# Defining a helper function to scale a vector to the range of 0 to 1
min_max_scale <- function(x) {
  if(all(is.na(x))) {
    return(x)
  } else {
    return((x-min(x, na.rm = TRUE)) / max(x, na.rm = TRUE) - min(x, na.rm = TRUE))
  }
}

# Scaling each column of the df using min-max scaling
df[] <- lapply(df, min_max_scale)
```

Part B

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

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

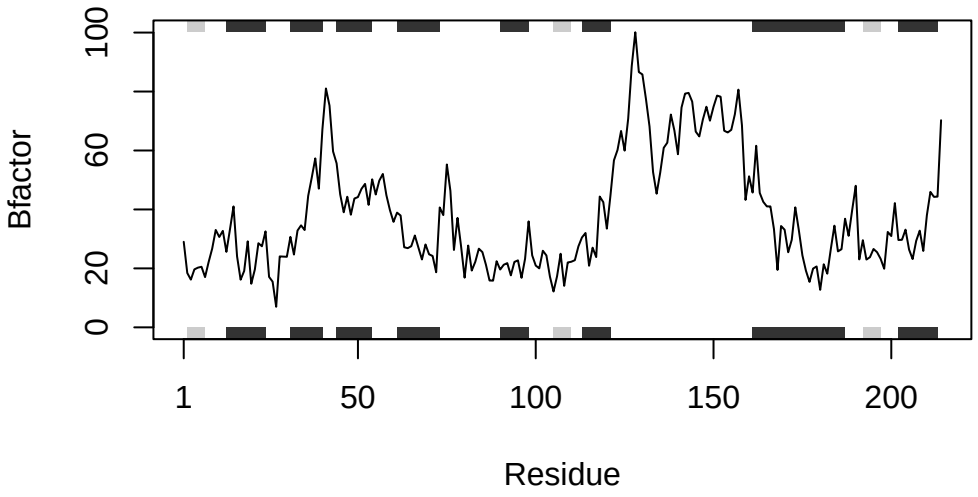
```
s2 <- read.pdb("1AKE") # kinase no drug
```

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

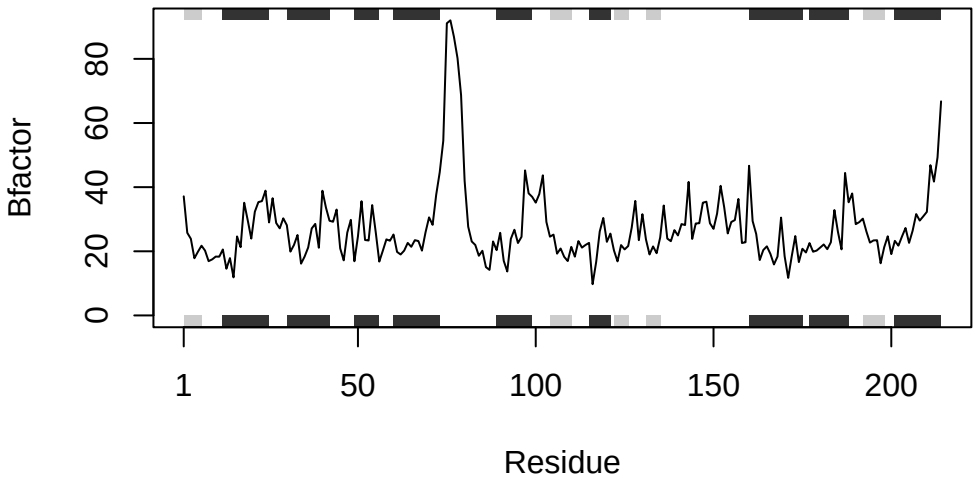
```
s3 <- read.pdb("1E4Y") # kinase with drug
```

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

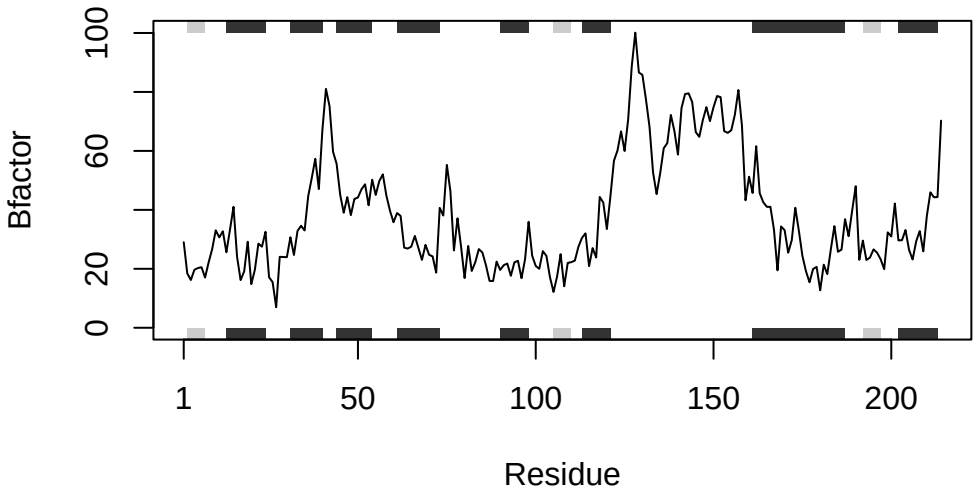
```
s1.chainA <- trim.pdb(s1, chain="A", elety="CA")
s2.chainA <- trim.pdb(s2, chain="A", elety="CA")
s3.chainA <- trim.pdb(s1, chain="A", elety="CA")
s1.b <- s1.chainA$atom$b
s2.b <- s2.chainA$atom$b
s3.b <- s3.chainA$atom$b
plotb3(s1.b, sse=s1.chainA, typ="l", ylab="Bfactor")
```




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



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




```
# Improving the code

# Reading the PDB files and checking for errors
s1 <- read.pdb("4AKE") # kinase with drug
```

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

```
Warning in get.pdb(file, path = tempdir(), verbose = FALSE):
/var/folders/l5/11lkmzlj79s7r8yjkp__gp1c0000gn/T//RtmpuC5plN/4AKE.pdb exists.
Skipping download
```

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

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

```
Warning in get.pdb(file, path = tempdir(), verbose = FALSE):
/var/folders/l5/11lkmzlj79s7r8yjkp__gp1c0000gn/T//RtmpuC5plN/1AKE.pdb exists.
Skipping download
```

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

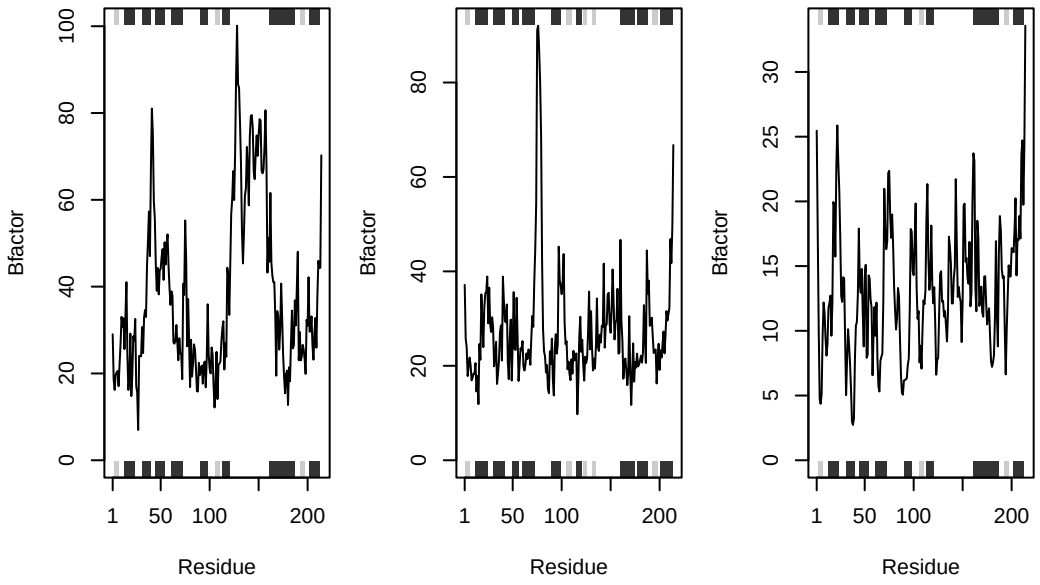
```
s3 <- read.pdb("1E4Y") # kinase with drug
```

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

```
Warning in get.pdb(file, path = tempdir(), verbose = FALSE):
/var/folders/l5/11lkmzlj79s7r8yjkp__gp1c0000gn/T//RtmpuC5plN/1E4Y.pdb exists.
Skipping download
```

```
# Trimming PDB files to chain A and CA atoms only
trim_pdb <- function(s) {
  trim.pdb(s, chain = "A", elety = "CA")$atom$b
}
s1.b <- trim_pdb(s1)
s2.b <- trim_pdb(s2)
s3.b <- trim_pdb(s3)

# Plotting the B-factor values for each structure
par(mfrow = c(1,3), mar = c(4,4,2,1), oma = c(0, 0, 2, 0))
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")
mtext("B factors for Kinases With and Without Drug", outer = TRUE, cex = 1.5)
```



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

The `read.pdb()` function returns an object of S3 class `pdb` (Protein Data Bank), which is used to represent the atomic features in a protein structure (e.g. atomic coordinates, B-factors, residue names, etc.).

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

The `trim.pdb()` function takes the PDB file and extracts a subset of atoms from a protein structure. This gives us the ability to specify a range of residue or a chain to extract from the stored PDB file, and returns a new PDB object containing the snippet of selected data.

Q3. What input parameter would turn off the marginal black and grey rectangles in the

plots and what do they represent in this case?

Setting `show.margins = FALSE` would turn off the marginal rectangles. The marginal rectangles represent the secondary structure elements (SSEs) of the protein–black rectangles represent alpha helices and the grey rectangles represent beta strands.

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

The `plotb3()` function plots the B-factors of the three proteins, so a better plot would be a side-by-side box plot or a violin plot.

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

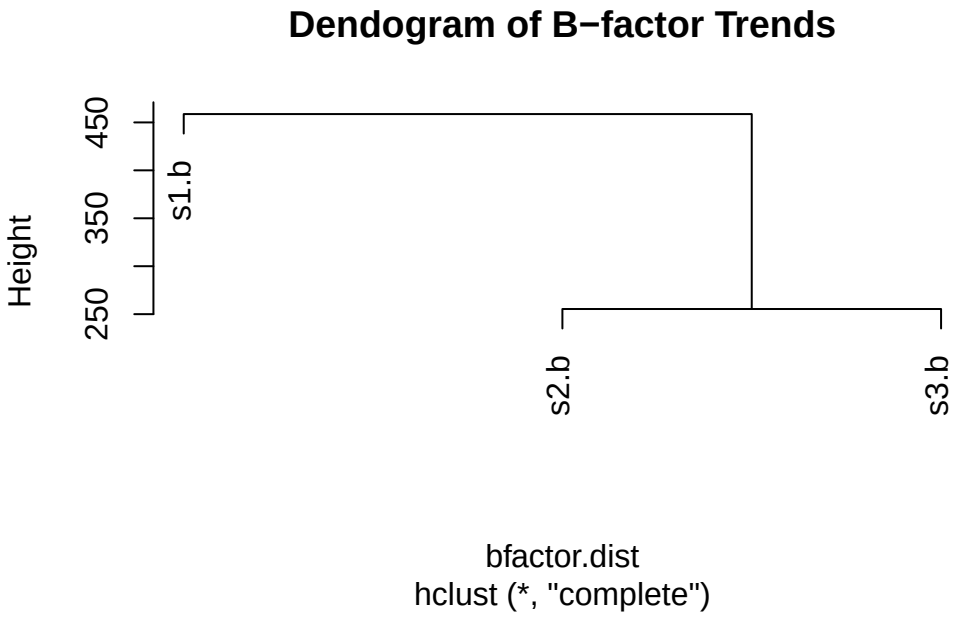
you quantify this? HINT: try the rbind(), dist() and hclust() functions together with a resulting dendrogram plot. Look up the documentation to see what each of these functions does.

```
# Creating a matrix of the B-factors
bfactors <- rbind(s1.b, s2.b, s3.b)

# Calculating the distance matrix
bfactor.dist <- dist(bfactors)

# Performing hierharchal clustering
bfactor.hclust <- hclust(bfactor.dist)

# Plotting the dendogram
plot(bfactor.hclust, main = "Dendogram of B-factor Trends")
```



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

```
library(bio3d)

analyze_protein <- function(pdb_file, chain = "A", ele_type = "CA") {
  # Read PDB file
  s <- read.pdb(pdb_file)

  # Trim protein
  s.chain <- trim.pdb(s, chain = chain, elety = ele_type)

  # Get B-factor values
  s.b <- s.chain$atom$b

  # Plot B-factor values
  plotb3(s.b, sse = s.chain, type = "1", ylab = "Bfactor", main = paste("B-factor plot for
    pdb_file))
}
```