HomeWork6

Mahsa Naeimi

2023-05-24

Section 1: Improving analysis code by writing functions

```
# Can you improve this analysis code?
library(bio3d)
# s1 <- read.pdb("4AKE")  # kinase with drug
# s2 <- read.pdb("1AKE")  # kinase no drug
# s3 <- read.pdb("1E4Y")  # kinase with drug
# 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 copy paste issues:
s1 <- read.pdb("4AKE")  # kinase with drug
```

B.

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

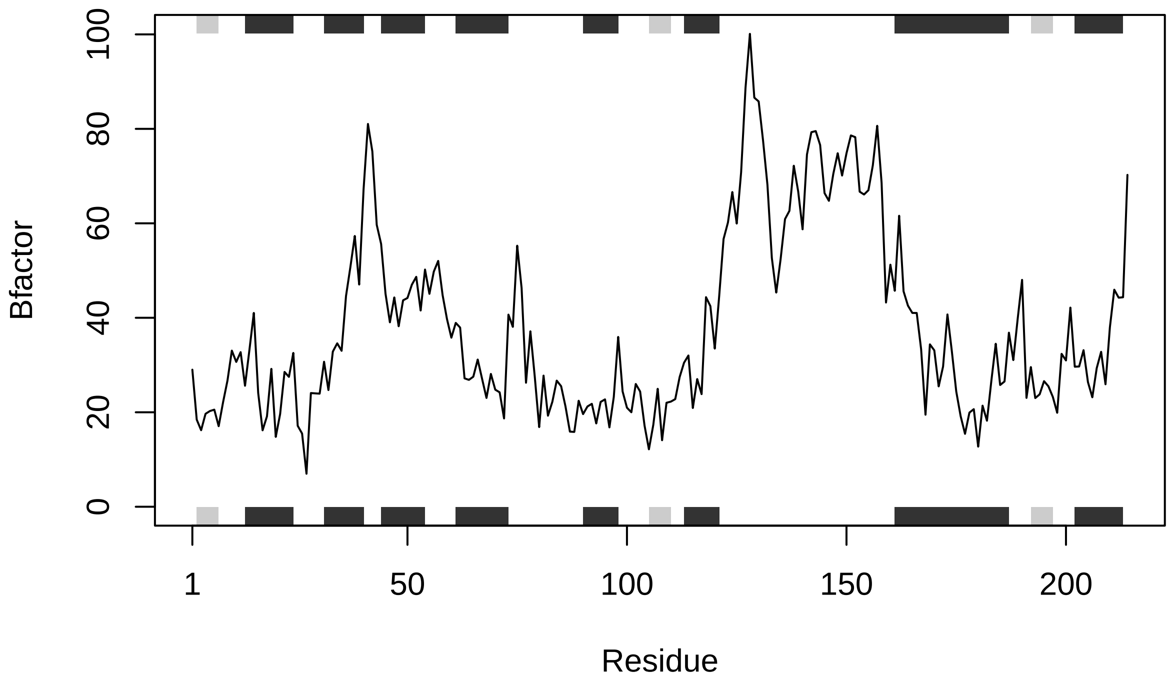
```
s2 <- read.pdb("1AKE")  # kinase no drug
```

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

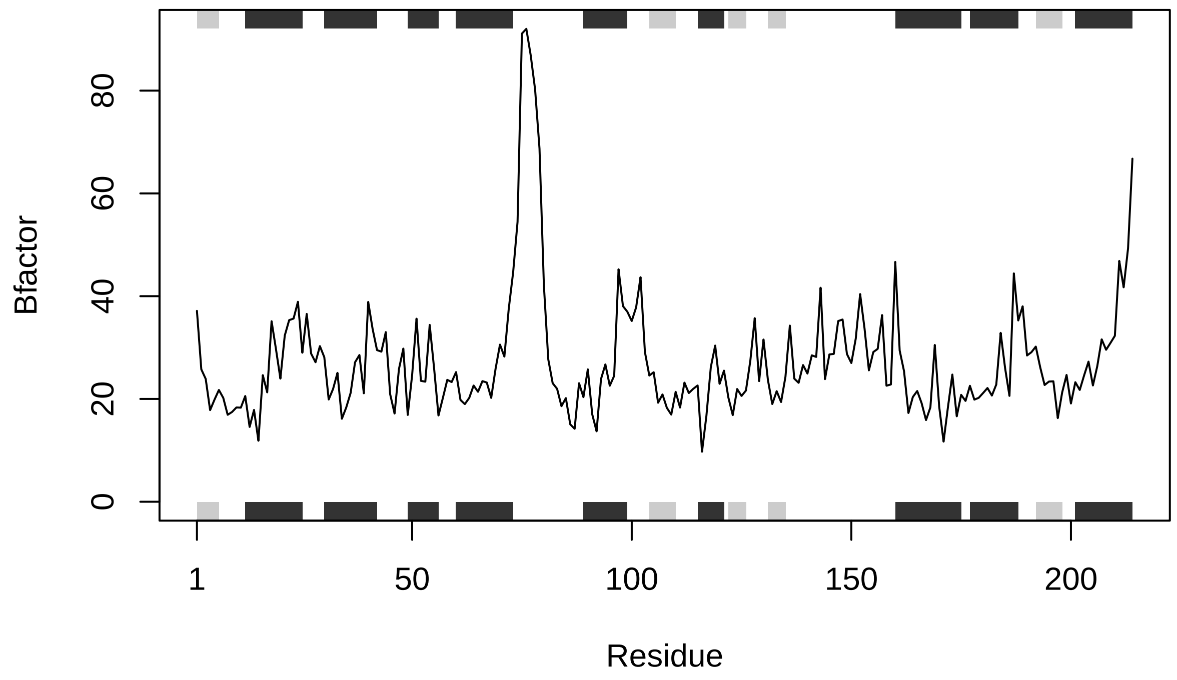
```
s3 <- read.pdb("1E4Y")  # kinase with drug
```

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

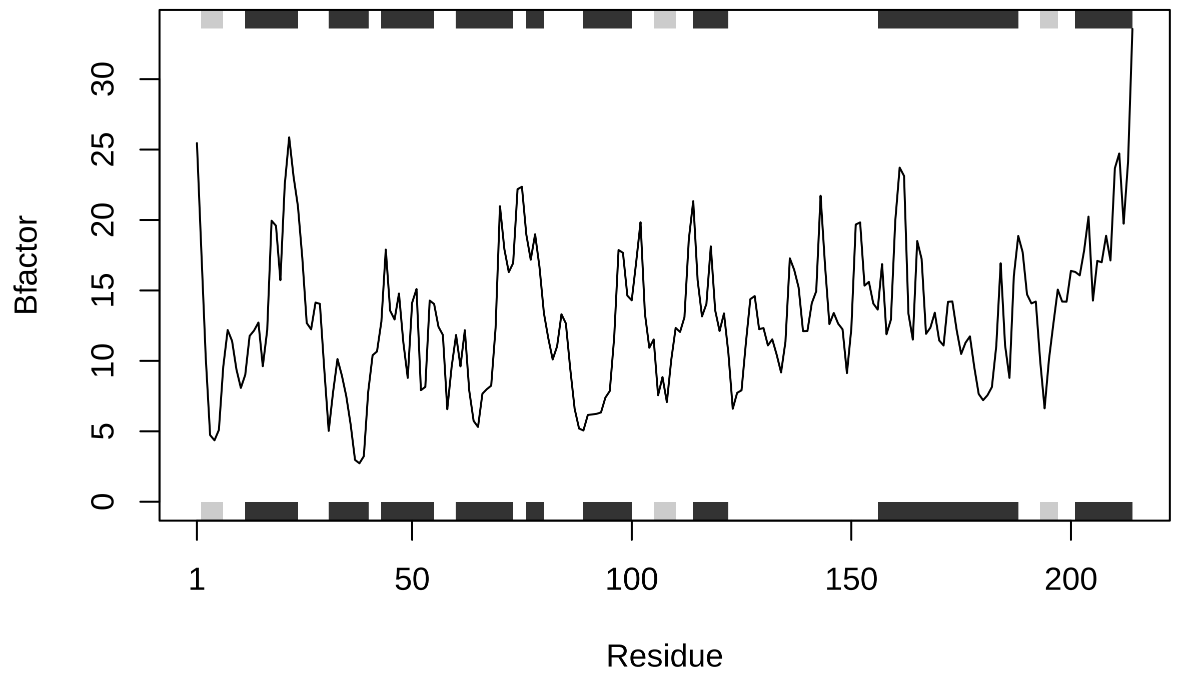
```
s1.chainA <- trim.pdb(s1, chain="A", elety="CA")
s2.chainA <- trim.pdb(s2, chain="A", elety="CA")
s3.chainA <- trim.pdb(s3, chain="A", elety="CA")

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")
```

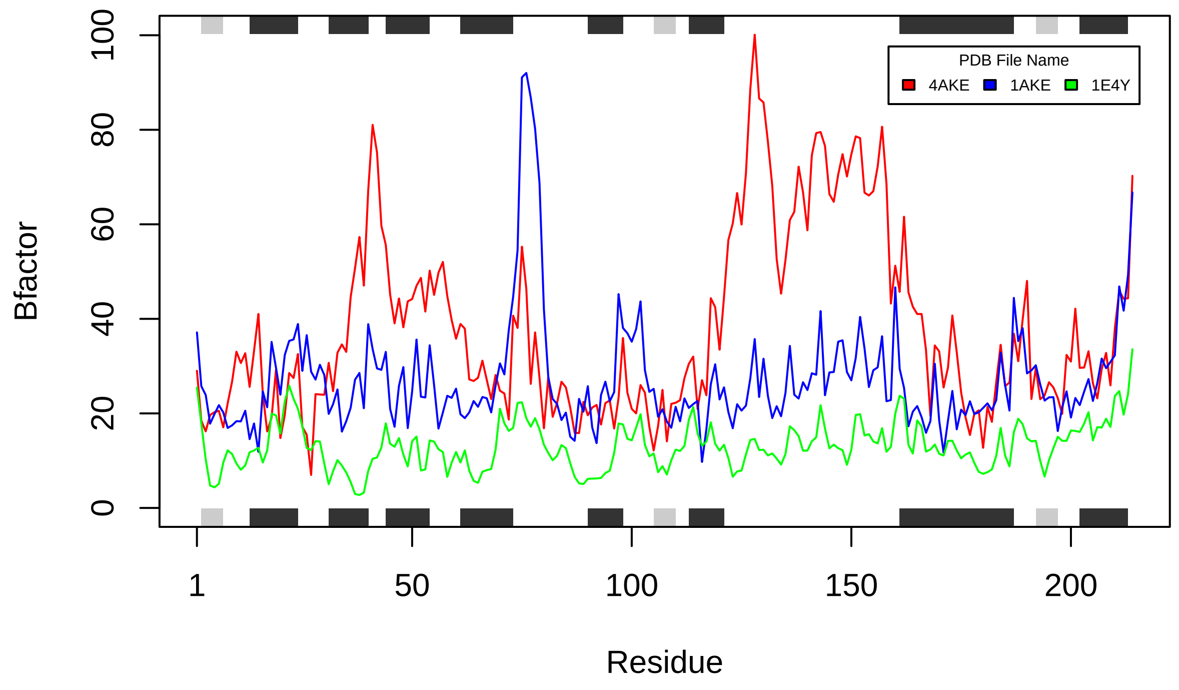


We want to create a function to be able to visualize protein-drug interactions from PDB data:

```
pr_dr <- function(file, chain, elmnt, fctr) {
   plot_colors <- c("red", "blue", "green")
     for (i in 1:length(file)) {
  s1 <- read.pdb(file[i])

  s1.chain <- trim.pdb(s1, chain = chain, elety = elmnt)

  atom_df <- s1.chain$atom
   s1.fctr <- atom_df[, fctr]
     if (i == 1) {
    plotb3(s1.fctr, sse = s1.chain, typ = "l", ylab = paste(toupper(fctr), "factor", sep = ""), col = pl
     } else {
    lines(s1.fctr, col = plot_colors[i])
  }
     }
     legend("topright", title = "PDB File Name", file, fill = plot_colors, horiz=TRUE, cex = 0.5, inset
}
```

```
# Testing the function for theses 3 protein:
files <- c("4AKE", "1AKE", "1E4Y")
chains <- "A"
elements <- "CA"
factors <- "b"

pr_dr(files, chains, elements, factors)
```

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

## Warning in get.pdb(file, path = tempdir(), verbose = FALSE):
## /var/folders/zx/ygntgn2j6lz4sh8bv9d1xsw00000gn/T//Rtmp1qwFKw/4AKE.pdb exists.
## Skipping download

##   Note: Accessing on-line PDB file

## Warning in get.pdb(file, path = tempdir(), verbose = FALSE):
## /var/folders/zx/ygntgn2j6lz4sh8bv9d1xsw00000gn/T//Rtmp1qwFKw/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/zx/ygntgn2j6lz4sh8bv9d1xsw00000gn/T//Rtmp1qwFKw/1E4Y.pdb exists.
## Skipping download
```



Q1. What type of object is returned from the read.pdb() function? This function returns information about the structure of the protein, its building block, and sequence of the protein from PDB

```
read.pdb("4AKE")
```

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

## Warning in get.pdb(file, path = tempdir(), verbose = FALSE):
## /var/folders/zx/ygntgn2j6lz4sh8bv9d1xsw00000gn/T//Rtmp1qwFKw/4AKE.pdb exists.
## Skipping download

##
```

```
##  Call:  read.pdb(file = "4AKE")
##
##    Total Models#: 1
##      Total Atoms#: 3459,  XYZs#: 10377  Chains#: 2  (values: A B)
##
##      Protein Atoms#: 3312  (residues/Calpha atoms#: 428)
##      Nucleic acid Atoms#: 0  (residues/phosphate atoms#: 0)
##
##      Non-protein/nucleic Atoms#: 147  (residues: 147)
##      Non-protein/nucleic resid values: [ HOH (147) ]
##
##    Protein sequence:
##       MRIILLGAPGAGKGTQAQFIMEKYGIPQISTGDMLRAAVKSGSELGKQAKDIMDAGKLVT
##       DELVIALVKERIAQEDCRNGFLLDGFPRTIPQADAMKEAGINVDYVLEFDVPDELIVDRI
##       VGRRVHAPSGRVYHVKFNPPKVEGKDDVTGEELTTRKDDQEETVRKRLVEYHQMTAPLIG
##       YYSKEAEAGNTKYAKVDGTKPVAEVRADLEKILGMRIILLGAPGA...<cut>...KILG
##
## + attr: atom, xyz, seqres, helix, sheet,
##         calpha, remark, call
```

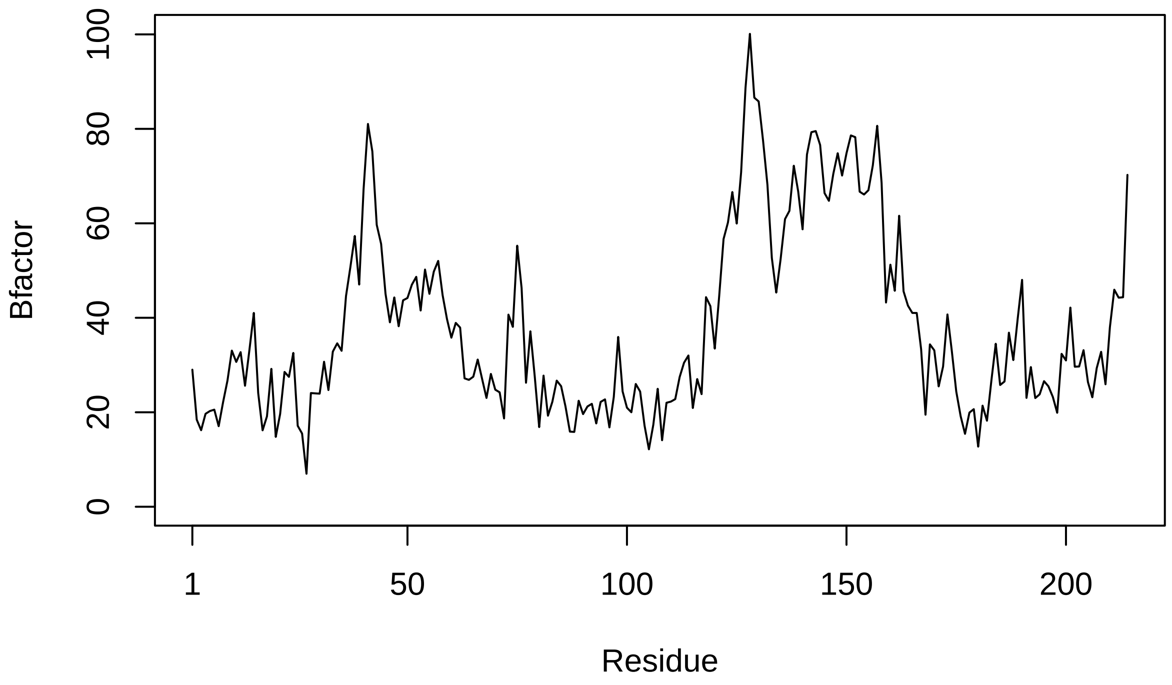
Q2. What does the trim.pdb() function do? This function trims the pdb object by selecting specific chains, residues, or atoms based on the provided parameters. In this case, `trim.pdb()` is used to extract a specific chain ("A") and element ("CA") from the pdb object.

```
trim.pdb(s1, chain="A", elety="CA")
```

```
##
##  Call:  trim.pdb(pdb = s1, chain = "A", elety = "CA")
##
##    Total Models#: 1
##      Total Atoms#: 214,  XYZs#: 642  Chains#: 1  (values: A)
##
##      Protein Atoms#: 214  (residues/Calpha atoms#: 214)
##      Nucleic acid Atoms#: 0  (residues/phosphate atoms#: 0)
##
##      Non-protein/nucleic Atoms#: 0  (residues: 0)
##      Non-protein/nucleic resid values: [ none ]
##
##    Protein sequence:
##       MRIILLGAPGAGKGTQAQFIMEKYGIPQISTGDMLRAAVKSGSELGKQAKDIMDAGKLVT
##       DELVIALVKERIAQEDCRNGFLLDGFPRTIPQADAMKEAGINVDYVLEFDVPDELIVDRI
##       VGRRVHAPSGRVYHVKFNPPKVEGKDDVTGEELTTRKDDQEETVRKRLVEYHQMTAPLIG
##       YYSKEAEAGNTKYAKVDGTKPVAEVRADLEKILG
##
## + attr: atom, helix, sheet, seqres, xyz,
##         calpha, call
```

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 remove the black and grey rectangles, we need delete SSE(secondary structral element) from the chunk.

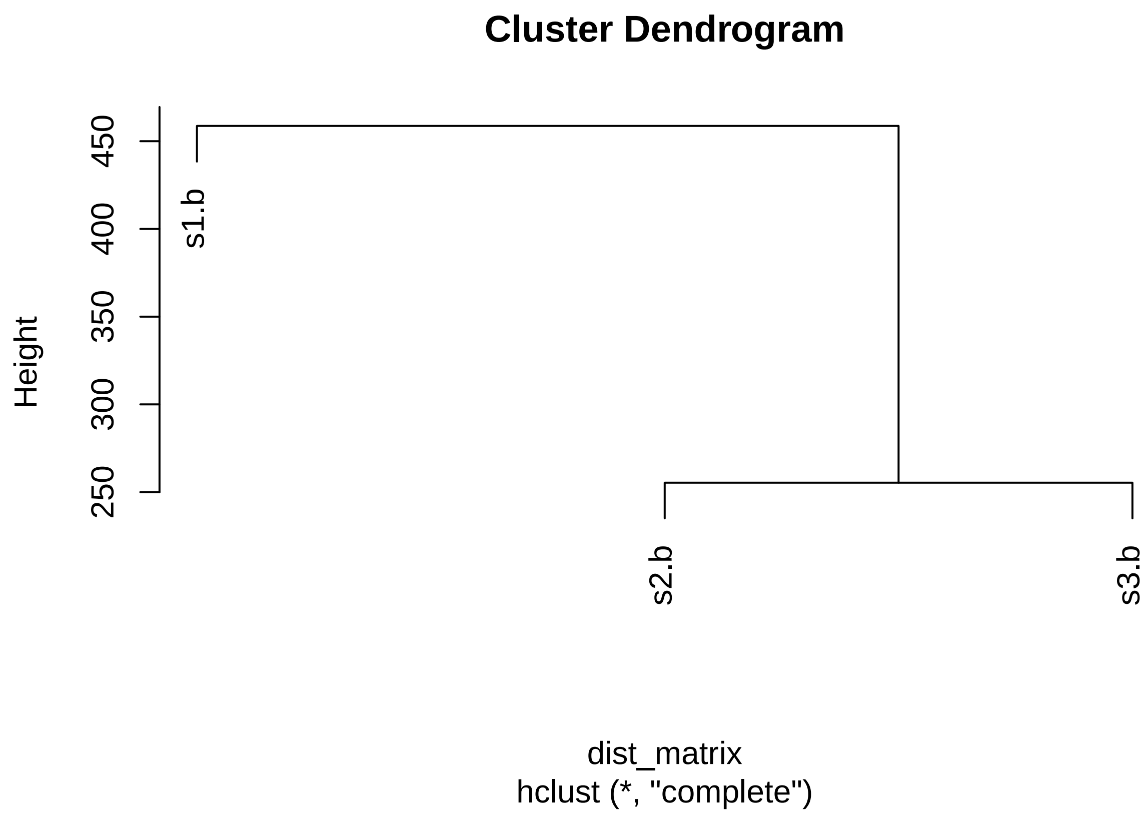
```
plotb3(s1.b, typ = "l", ylab = "Bfactor")
```



Q4. What would be a better plot to compare across the different proteins? a single plot with different colored lines representing each protein's B-factor trends which allows us to compare multiple proteins at the same time.

Q5. Which proteins are more similar to each other in their B-factor trends. How could you quantify this? From the clustering we can see that 1E4Y and 1AKE are the most similar ones.

```
bfactors <- rbind(s1.b, s2.b, s3.b)
dist_matrix <- dist(bfactors)
hclust_result <- hclust(dist_matrix)
plot(hclust_result)
```



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

```
pr_dr <- function(file, chain, elmnt, fctr) {
   plot_colors <- c("red", "blue", "green")
     for (i in 1:length(file)) {
  s1 <- read.pdb(file[i])

  s1.chain <- trim.pdb(s1, chain = chain, elety = elmnt)

  atom_df <- s1.chain$atom
   s1.fctr <- atom_df[, fctr]
     if (i == 1) {
    plotb3(s1.fctr, sse = s1.chain, typ = "l", ylab = paste(toupper(fctr), "factor", sep = ""), col = pl
     } else {
    lines(s1.fctr, col = plot_colors[i])
  }
     }
     legend("topright", title = "PDB File Name", file, fill = plot_colors, horiz=TRUE, cex = 0.5, inset
}
```