Lab 6 Homework: Question 6

Christopher Leone | A16731724

2025-04-20

Contents

Preceding Code Chunk

Here is the code chunk preceding the main code to be edited in Q6.

```
# This calls upon bio3d
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
```

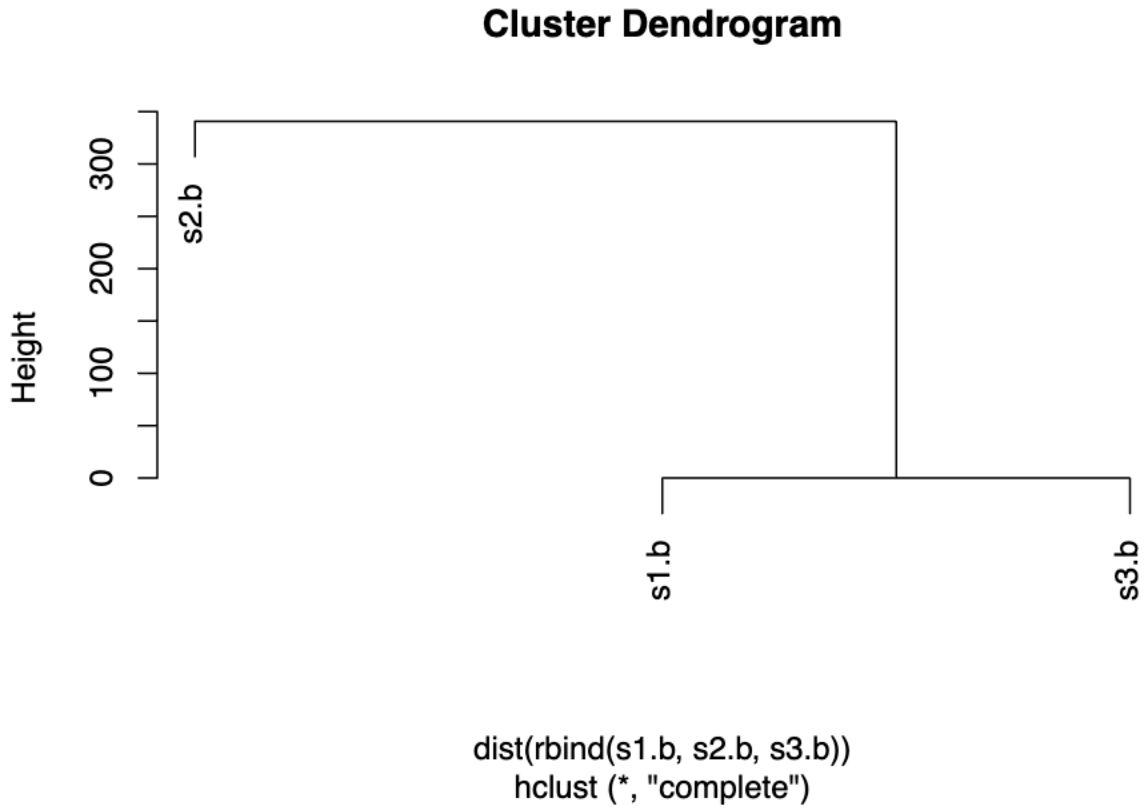
```
##   Note: Accessing on-line PDB file
```

```
# This trims a predefined protein into various chains into a smaller PDB object.
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")

# This gathers the various B-factor values to be plotted
s1.b <- s1.chainA$atom$b
s2.b <- s2.chainA$atom$b
```

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

# The B-factor plots will not be printed here for sake of space.
```

Main Assignment: Question 6

```
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?

1) If we want to generalize the original code above, we have to rely on creating a `function()`. But first, I want to define similar variables to `s1.b`, `s2.b`, `s3.b` to add to the object pool.

```
# This calls upon bio3d
library(bio3d)

s4 <- read.pdb("1O6K") # PKB Kinase
```

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

```
s5 <- read.pdb("2BDJ") # Src Kinase (inhibited)
```

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

```r
s6 <- read.pdb("6F3F") # Src Kinase (not inhibited)
```

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

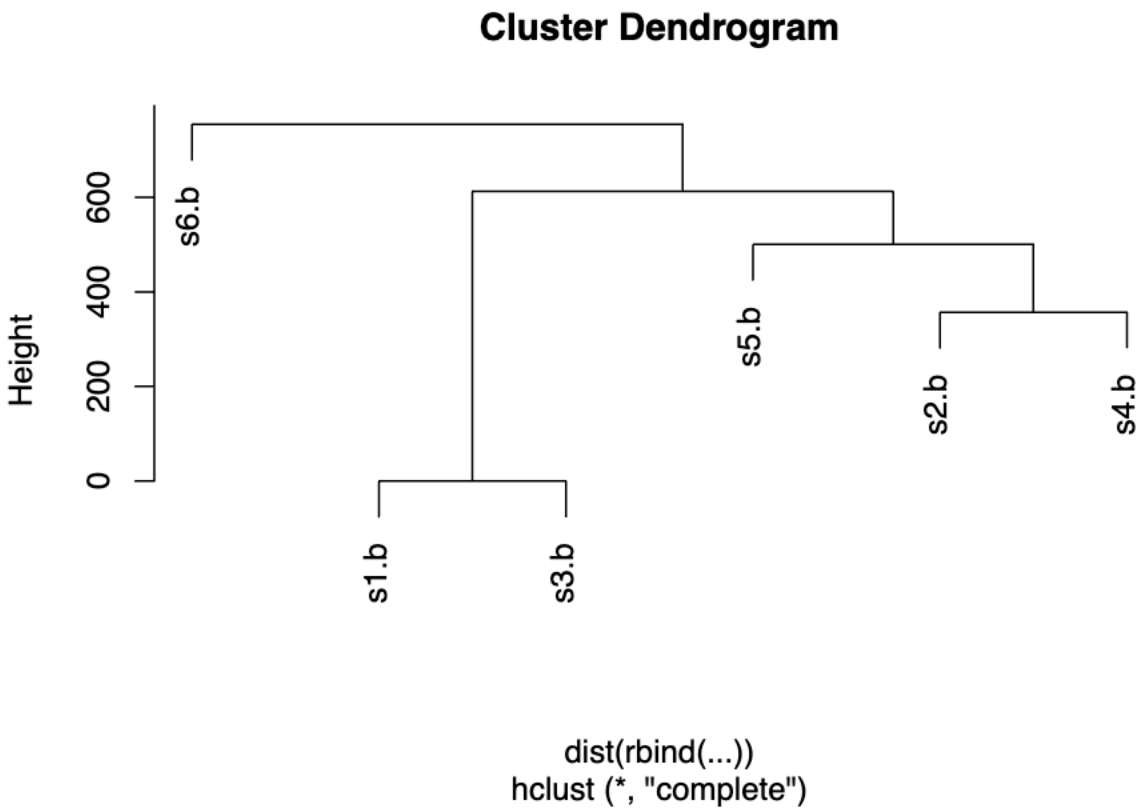
```r
# This trims a predefined protein into various chains into a smaller PDB object.
s4.chainA <- trim.pdb(s4, chain="A", elety="CA")
s5.chainA <- trim.pdb(s5, chain="A", elety="CA")
s6.chainA <- trim.pdb(s6, chain="A", elety="CA")

# This gathers the various B-factor values to be plotted.
s4.b <- s4.chainA$atom$b
s5.b <- s5.chainA$atom$b
s6.b <- s6.chainA$atom$b
```

2) Now that I have additional objects to use, let's create the function. We want it to allow any number or set of inputs to be used with our dendrogram.

```r
# Let's create a function to generalize the process.

# Here, `...` is used to denote an arbitrary number of inputs.
hcFunction <- function(...){

  # Let's paste the original code, but with the `...` to allow for object variation.
  hcAdvanced <- hclust(dist(rbind(...)))

  # And now, to plot.
  plot(hcAdvanced)
}
```

3) We have now defined `hcFunction()`. Let's attempt to create a dendrogram with all 6 protein chains.

```r
# Considering that `hcFunction()` was defined with `...` representing the arguments...
hcFunction(s1.b, s2.b, s3.b, s4.b, s5.b, s6.b)
```




```
# ...we can add whatever we would like to be in our dendrogram.
```

This code would also work with any combination of proteins, so long as there are at least 3 object inputs in `hcFunction()`.