Creating R functions

Yoonjin Lim (PID: A16850635)

Table of contents

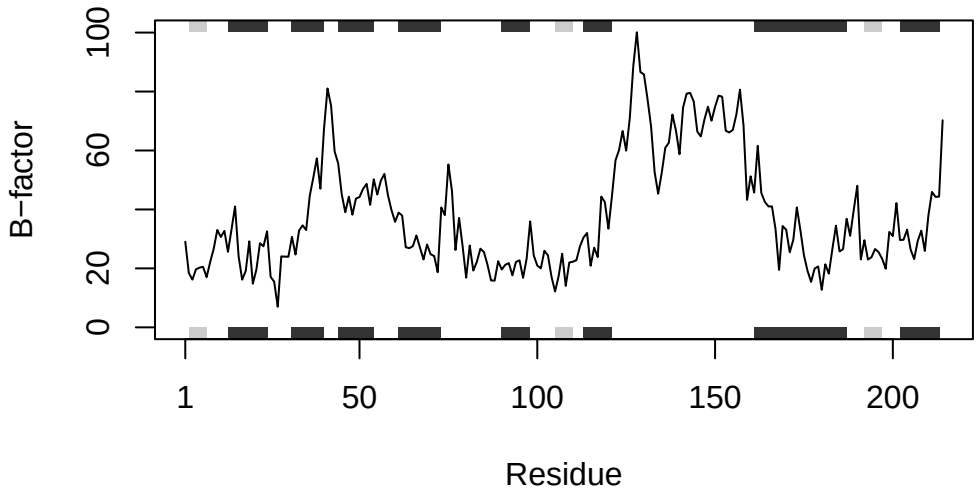
Generalizing the original code to work with any input protein structure

Code:

```
library(bio3d)
generate_protein <- function(pdb_data, sse=TRUE){
   ## Loading protein PDB data
  pdb <- read.pdb(pdb_data)
  ## Trimming a PDB object to subset of CA
  trim <- trim.pdb (pdb, chain="A", elety="CA")
  ## Abstracting B-factor data
  B_factor <- trim$atom$b
  ## Visualizing the plots for the chosen proteins
  sse_data <- sse
  if (sse) {
   sse_data <- trim
  }
  plotb3(B_factor, typ="l", ylab="B-factor", sse=sse_data)
}
```

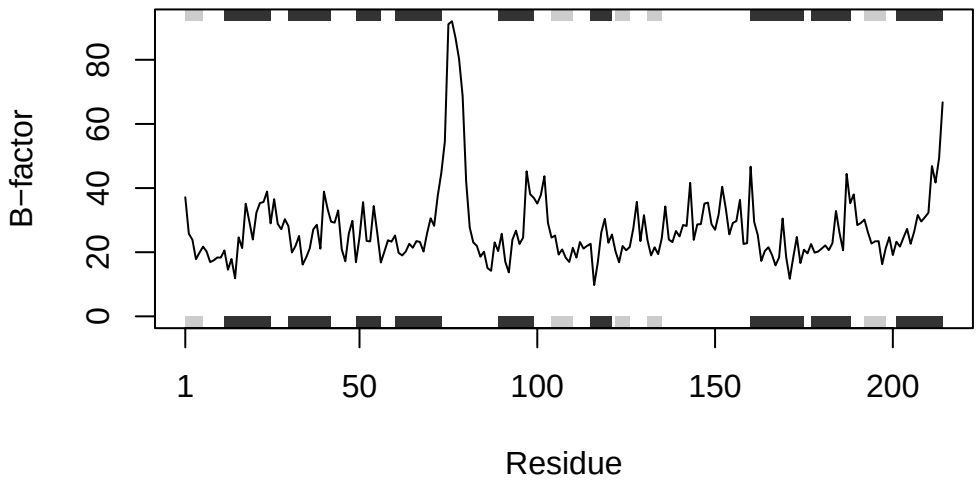
```
generate_protein("4AKE")
```

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



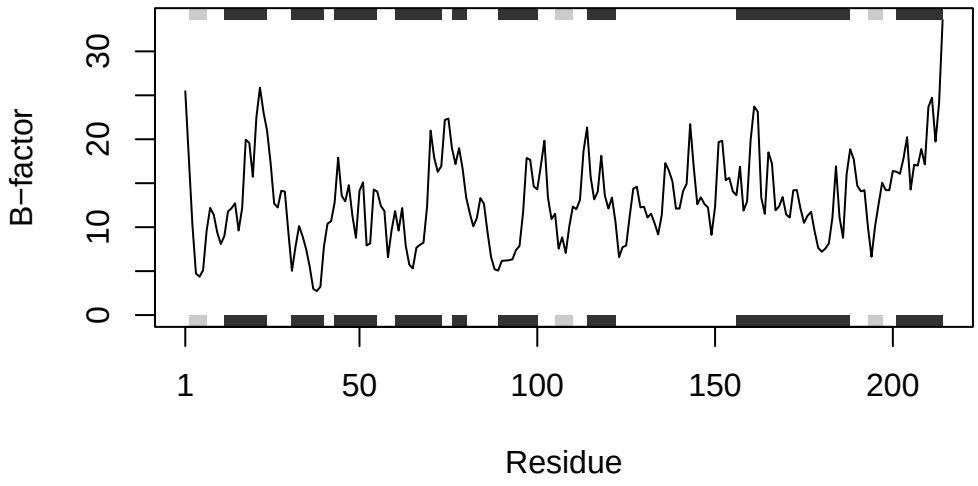

```
generate_protein("1AKE")
```

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




```
generate_protein ("1E4Y")
```

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



Documentation:

- Inputs: the function "generate_protein()" takes the input pdb_data, the character that indicates the PDB data of the protein to be examined further.
- What function does: this function fundamentally reads any PDB data for the chosen protein, using installed bio3d packages. In particular, it abstracts the carbon atom from chain A and searches the B-factor for these atoms. This function is supposed to create the plot of the B-factor.
- Outputs: the output of the function is a B-factor plot for the chosen protein. This plot indicates how flexible the several different parts of protein structures are, based on the B-factor of alpha carbon atoms.