Descriptions of All Code Written Thus Far

Senior Design

Group B
<table>
<thead>
<tr>
<th>Desolvation Thermodynamic Cycle</th>
<th>Thermodynamic Cycle from APBS website</th>
</tr>
</thead>
<tbody>
<tr>
<td>1. Call ABPS</td>
<td>8. Call ABPS new</td>
</tr>
<tr>
<td>2. APBS template</td>
<td>9. APBS template new</td>
</tr>
<tr>
<td>3. Conversion of PDB to PQR</td>
<td>10. Concatenate PQR</td>
</tr>
<tr>
<td>4. MySeq</td>
<td>----</td>
</tr>
<tr>
<td>5. Intrinsic Pka</td>
<td>11. Intrinsic PKa</td>
</tr>
<tr>
<td>6. Pka function</td>
<td>12. Same Pka function</td>
</tr>
<tr>
<td>7. Mean Pka function</td>
<td>13. Same Mean Pka function</td>
</tr>
</tbody>
</table>

### Description of each code

1. Calls to APBS software to calculate free energies for each individual protein, these values are then generated into a out file which will be inputted for the calculations of intrinsic Pka

2. This template file was generated to account for the 8 states of the thermodynamic cycle this file is inputted into call APBS script

3. This code converts a PDB to PQR, and will make two initial PQR files a neutral and charged case. This code will be inputted into intrinsic pKa code and these two PQR files will then be converted to account for all states of desolvation TC

4. MySeq function was written specifically for the desolvation Thermodynamic cycle so that would could convert the three letter amino acid code to a single letter amino acid code, these were based on PARSE naming scheme

5. Intrinsic Pka code incorporates all codes previously mentioned, to run intrinsic pKa calculations for each ionizable amino acids in protein of choice, the for loop accounts for the desolvation TC, and generates PQR and free energies based of that

6. pKa function was generated and implemented into intrinsic pKa code to take the free energies and calculate pKa values using pKa equation for Morikis et al, 2001

7. This function calculates the mean value for each individual amino acids, it takes the average value for each ionizable amino acids

8. . Calls to APBS software to calculate free energies for each individual protein, these values are then generated into a out file which will be inputted for the calculations of intrinsic Pka, this call APBS function using the thermodynamic cycle for APBS website

9. . This template file was generated to account for the 6 states of the thermodynamic cycle this file is inputted into call APBS script. These 6 states takes into account reference states

10. This code converts a PDB to PQR, and will make two initial PQR files a neutral and charged case. This code will be inputted into intrinsic pKa code and these two PQR files will then be converted to account for all states of APBS thermodynamic cycle

11. Intrinsic Pka code incorporates all codes previously mentioned, to run intrinsic pKa calculations for each ionizable amino acids in protein of choice, the for loop accounts for the APBS thermodynamic cycle, and generates PQR and free energies based of that
1: Call APBS for desolvation Thermodynamic cycle

```r
# ~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
# Functions that calls apbs to calculate free energies
# for ionizable amino acid residues
# ~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~

call_apbs <- function(in_file)
{

  bdp_file <- "1ly2_neutral.pqr"
  bp_file <- "Bound_protonated.pqr"
  fdp_file <- "Free_deprotonated.pqr"
  fp_file <- "Free_protonated.pqr"

  length <- 100
  width <- 100
  height <- 100


  in_file[12] <- paste(" fglen ", length, width, height, sep = " ")
  in_file[35] <- paste(" cglen ", length, width, height, sep = " ")
  in_file[36] <- paste(" fglen ", length, width, height, sep = " ")
  in_file[59] <- paste(" cglen ", length, width, height, sep = " ")
  in_file[60] <- paste(" fglen ", length, width, height, sep = " ")
  in_file[83] <- paste(" cglen ", length, width, height, sep = " ")
  in_file[84] <- paste(" fglen ", length, width, height, sep = " ")
  in_file[107] <- paste(" cglen ", length, width, height, sep = " ")
  in_file[108] <- paste(" fglen ", length, width, height, sep = " ")
  in_file[139] <- paste(" cglen ", length, width, height, sep = " ")
  in_file[131] <- paste(" fglen ", length, width, height, sep = " ")
  in_file[154] <- paste(" cglen ", length, width, height, sep = " ")
  in_file[155] <- paste(" fglen ", length, width, height, sep = " ")
  in_file[177] <- paste(" cglen ", length, width, height, sep = " ")
  in_file[178] <- paste(" fglen ", length, width, height, sep = " ")

  con <- file("infile.in", "w")
  writeLines(in_file, con, sep = " ")
  close(con)

  TC <- system(paste( "/apbs-1.2-mac-univ/bin/apbs", "infile.in"," >" ),

```

Code 1: Call APBS function generated for desolvation cycle
2: APBS Template for desolvation Thermodynamic cycle

read
  mol pqr bound_dp.pqr
  mol pqr bound_p.pqr
  mol pqr free_dp.pqr
  mol pqr free_p.pqr
end
elec name bdp
  mg-auto
dime 129 129 129
cglen 45.3322 54.9498 82.2633
glen 45.3322 52.3234 68.3902
cgcent mol 1
fgcent mol 1
mol 1
lpbe
bcfl sdh
pdie 20.0
ion charge 1 conc 0.15 radius 2.0
ion charge -1 conc 0.15 radius 2.0
sdie 78.54
srfm mol
chgm spl2
sdens 10.00
srad 1.40
swin 0.30
temp 298.15
calcenergy total
calcforce no
end
elec name bp
  mg-auto
dime 129 129 129
cglen 45.3322 54.9498 82.2633
glen 45.3322 52.3234 68.3902
cgcent mol 1
fgcent mol 1
mol 2
lpbe
bcfl sdh
pdie 20.0
sdie 78.54
ion charge 1 conc 0.15 radius 2.0
ion charge -1 conc 0.15 radius 2.0

Code 2: Shows a preview of template file generated for desolvation thermodynamic cycle
3: Conversion of PDB to PQR

```r
#~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
#This function creates two specific pqr files a netural
#pqr file were all amino acids partial charges are
#neutralized, and the charged where charge for ionizable
#amino acids are added, this function will then be used
#to concatenate the two pqr files and create pqr files
#for all states in the thermodynamic cycle for all
#ionizable amino acids
#~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~

Neu_Char_pdb <- function(pdb) 
{
  x <- pdb
  ##for neutral pdb
  x$atom[atom.select(x, resid = "ASP")$atom,4]<-sub("ASP", "ASH", x$atom[atom.select(x, resid = "ASP")$atom,4])
  x$atom[atom.select(x, resid = "GLU")$atom,4]<-sub("GLU", "GLH", x$atom[atom.select(x, resid = "GLU")$atom,4])
  x$atom[atom.select(x, resid = "LYS")$atom,4]<-sub("LYS", "LYN", x$atom[atom.select(x, resid = "LYS")$atom,4])
  x$atom[atom.select(x, resid = "ARG")$atom,4]<-sub("ARG", "AR0", x$atom[atom.select(x, resid = "ARG")$atom,4])
  write.pdb(pdb = x,file = "1ly2_neutral")
  pdb <- x
  ##for charged pdb
  x$atom[atom.select(x, resid = "HIS")$atom,4]<-sub("HIS", "HID", x$atom[atom.select(x, resid = "HIS")$atom,4])
  x$atom[atom.select(x, resid = "CYS")$atom,4]<-sub("CYS", "CYM", x$atom[atom.select(x, resid = "CYS")$atom,4])
  x$atom[atom.select(x, resid = "TYR")$atom,4]<-sub("TYR", "TYM", x$atom[atom.select(x, resid = "TYR")$atom,4])
  write.pdb(pdb = x,file = "1ly2_charged")

  pdb_list <- list(pdb,x)
  return(pdb_list)
}
```

**Code 3: Conversion of PDB to PQR and makes one neutral and one charged PQR file**
Code 4: Preview of code, which allows for the conversion from a single amino acid code to triplet code
5: Intrinsic pKa for desolvation thermodynamic cycle

#~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
#Cleans pdb file extracts line begin with atom
#~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
LY2 <- readLines("1LY2.pdb")
index1 <- grep("ATOM ", LY2)
cat(x[y], sep = "\n", file = "clean_1LY2.pdb")

#~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
#sources all functions written
#~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
source("cat_pdb.r")
source("neutral_charged.r")
source("call_apbs.r")

#~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
#converts pdb files to pqr files using python connection
#~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
system("python /Users/senior_design/pdb2pqr-1.5/pdb2pqr.py --ff parse 1ly2_charged.pdb 1ly2_charged.pqr")
system("python /Users/senior_design/pdb2pqr-1.5/pdb2pqr.py --ff parse 1ly2_neutral.pdb 1ly2_neutral.pqr")

#~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
#reads in our pqr files and these will be concatenated together
#~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
normal_pqr <- read.pqr("1ly2_charged.pq")
charged_pqr <- read.pqr("1ly2_neutral.pqr")

#~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
# parameters, extracts sequence of pqr file and creates a array
# of characters, k allows to extract length of pqr
#~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
k <- ( as.numeric(normal_pqr$atom[1,"resno"] ) )
end_of_seq <- length(seq.pdb(charged_pqr) ) - 1
seq <- seq.pdb(normal_pqr)

#~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
# runs a loop that will look for ionizable amino acids residues

Code 5: Code allows for the calculation of intrinsic pKa values for desolvation thermodynamic cycle
Code 5: Continued

6: pKa function

```r
pk <- function(AAdf)
  AAdf <- cbind(AAdf, "Pka"=as.numeric(6))
  AAdf[(AAdf[,1]=="E"),] <- c(AAdf[(AAdf[,1]=="E"),2] - log(4.3 - as.numeric(AAdf[(AAdf[,1]=="E"),3])/(2.303 * 0.008314 * 298), 10), AAdf[,3]]
  AAdf[(AAdf[,1]=="C"),] <- c(AAdf[(AAdf[,1]=="C"),2] - log(8.28 - as.numeric(AAdf[(AAdf[,1]=="C"),3])/(2.303 * 0.008314 * 298), 10), AAdf[,3]]
  AAdf[(AAdf[,1]=="K"),] <- c(AAdf[(AAdf[,1]=="K"),2] - log(10.5 - as.numeric(AAdf[(AAdf[,1]=="K"),3])/(2.303 * 0.008314 * 298), 10), AAdf[,3]]
  AAdf[(AAdf[,1]=="H"),] <- c(AAdf[(AAdf[,1]=="H"),2] - log(6.08 - as.numeric(AAdf[(AAdf[,1]=="H"),3])/(2.303 * 0.008314 * 298), 10), AAdf[,3]]
  AAdf[(AAdf[,1]=="R"),] <- c(AAdf[(AAdf[,1]=="R"),2] - log(12.0 - as.numeric(AAdf[(AAdf[,1]=="R"),3])/(2.303 * 0.008314 * 298), 10), AAdf[,3]]
  AAdf[(AAdf[,1]=="Y"),] <- c(AAdf[(AAdf[,1]=="Y"),2] - log(10.1 - as.numeric(AAdf[(AAdf[,1]=="Y"),3])/(2.303 * 0.008314 * 298), 10), AAdf[,3]]
  AAdf[(AAdf[,1]=="D"),] <- c(AAdf[(AAdf[,1]=="D"),2] - log(3.9 - as.numeric(AAdf[(AAdf[,1]=="D"),3])/(2.303 * 0.008314 * 298), 10), AAdf[,3]]
  return(AAdf)
```

Code 6: Function written to calculate pKa using equation from Morikis et al
7: Mean pKa function

```r
mean <- function(Pka)
{
    meandf <- NULL
    meandf <- rbind(meandf,"AvgG" = mean(as.numeric(Pka[,1])
    =="E"),"delta_G"), "Avg_Pka" = mean(as.numeric(Pka[,1])
    =="E"),"Pka"
    meandf <- rbind(meandf,"AvgG" = mean(as.numeric(Pka[,1])
    =="R"),"delta_G"), "Avg_Pka" = mean(as.numeric(Pka[,1])
    =="R"),"Pka"
    meandf <- rbind(meandf,"AvgG" = mean(as.numeric(Pka[,1])
    =="K"),"delta_G"), "Avg_Pka" = mean(as.numeric(Pka[,1])
    =="K"),"Pka"
    meandf <- rbind(meandf,"AvgG" = mean(as.numeric(Pka[,1])
    =="H"),"delta_G"), "Avg_Pka" = mean(as.numeric(Pka[,1])
    =="H"),"Pka"
    meandf <- rbind(meandf,"AvgG" = mean(as.numeric(Pka[,1])
    =="C"),"delta_G"), "Avg_Pka" = mean(as.numeric(Pka[,1])
    =="C"),"Pka"
    meandf <- rbind(meandf,"AvgG" = mean(as.numeric(Pka[,1])
    =="Y"),"delta_G"), "Avg_Pka" = mean(as.numeric(Pka[,1])
    =="Y"),"Pka"
    meandf <- rbind(meandf,"AvgG" = mean(as.numeric(Pka[,1])
    =="D"),"delta_G"), "Avg_Pka" = mean(as.numeric(Pka[,1])
    =="D"),"Pka"
    return(meandf)
}
```

**Code 7: Function to calculate average pKa value for each ionizable amino acid**
8: Call APBS for APBS Thermodynamic cycle

#~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
#Functions that calls apbs to calculate free energies
#for ionizable amino acid residues
#~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~

call_apbs2 <- function(in_file)
{

  bdp_file <- "1ly2_neutral.pqr"
  bp_file <- "Bound_protonated.pqr"
  fdp_file <- "Free_deprotonated.pqr"
  fp_file <- "Free_protonated.pqr"

  length <- 100
  width <- 100
  height <- 100


  in_file[13] <- paste("      cglen ",length,width,height, sep = " ")
  in_file[14] <- paste("      fglen ",length,width,height, sep = " ")
  in_file[37] <- paste("      cglen ",length,width,height, sep = " ")
  in_file[38] <- paste("      fglen ",length,width,height, sep = " ")
  in_file[61] <- paste("      cglen ",length,width,height, sep = " ")
  in_file[62] <- paste("      fglen ",length,width,height, sep = " ")
  in_file[85] <- paste("      cglen ",length,width,height, sep = " ")
  in_file[86] <- paste("      fglen ",length,width,height, sep = " ")
  in_file[109] <- paste("      cglen ",length,width,height, sep = " ")
  in_file[110] <- paste("      fglen ",length,width,height, sep = " ")
  in_file[133] <- paste("      cglen ",length,width,height, sep = " ")
  in_file[134] <- paste("      fglen ",length,width,height, sep = " ")

  con <- file("infile.in","w")
  writelines(in_file,con,sep = "\n")
  close(con)

  TC <- system(paste( "/apbs-1.2-mac-univ/bin/apbs", ""infile.in",">",
  "outfile.txt", sep = " ")
  outfile <- readLines("outfile.txt")

Code 8: Calls to APBS, and corresponds to thermodynamic cycle established by APBS
website
index <- grep("Global", outfile )
str_energy <- strsplit(outfile[index[length(index)]],split = "=")
char_number <- strsplit(str_energy[[1]],split = " ")
free_energy <- as.numeric(char_number[[2]][1])
return(free_energy)
}

9: APBS Template for APBS Thermodynamic cycle

read
  mol pqr bound_dp.pqr
  mol pqr bound_p.pqr
  mol pqr free_dp.pqr
  mol pqr free_p.pqr
end
elec name bdp
  mg-auto
dime 65 97 129
cglen 45.3322 54.9498 82.2633
fglen 45.3322 52.3134 68.3902
cgent mol 1
gcent mol 1
mol 1
lpbe
bcfl sdh
pdie 20.0
ion charge 1 conc 0.15 radius 2.0
ion charge -1 conc 0.15 radius 2.0
die 78.54
srfm mol
cghm spl2
sdens 10.00
srad 1.40
swin 0.30
temp 298.15
calce energy total
calcc force no
end
elec name bdp_vacum
  mg-auto
dime 65 97 129
cglen 45.3322 54.9498 82.2633
fglen 45.3322 52.3134 68.3902
cgent mol 1
gcent mol 1
mol 1
lpbe
bcfl sdh
pdie 20.0
ion charge 1 conc 0.0 radius 2.0
ion charge -1 conc 0.0 radius 2.0
die 20.0
srfm mol

Code 9: Preview of Template generated for APBS thermodynamic cycle
Concatenate to PQR files

source("cat_pdb.r")

# reads in our pqr files and these will be concatenated together
neutral_pqr <- read.pqr("clean.pqr")
charged_pqr <- read.pqr("test.pqr")

ek <- (as.numeric(neutral_pqr$atom[1,"resno"]))
end_of_seq <- length(seq.pdb(neutral_pqr)) - 1

# parameters, extracts sequence of pqr file and creates an array
# of characters, k allows to extract length of pqr

for ( i in seq )
{
    if ( i == "R" | i == "K" | i == "H" | i == "C" | i == "Y"
         | i == "D" | i == "E" )
    {
        Before <- trim.pdb( neutral_pqr, atom.select(neutral_pqr,
             resno = 1:(k - 1) ) )
        Free_protonated <- trim.pdb( charged_pqr, atom.select
             charged_pqr, resno = k )
        After <- trim.pdb( neutral_pqr, atom.select (neutral_pqr,
             resno = (k+1):end_of_seq ) )
        Free_deprotonated <- trim.pdb( neutral_pqr, atom.select
             neutral_pqr, resno = k))
        write.pqr(Free_protonated, file = "Free_protonated.pqr")
        Before_FP <- cat_pdb( Before, Free_protonated )
        Total <- cat_pdb(Before_FP, After)
    }
}

Code 10: Concatenates two PQR files
11: Intrinsic pKa for APBS thermodynamic cycle

```r
# Cleans pdb file extracts line begin with atom
# ___________________________________________________________
#LY2 <- readLines("1LY2.pdb")
#index1 <- grep("ATOM ", LY2)
#cat(x[index1], sep = "\n", file = "clean_1LY2.pdb")

# Open the connection for the in file used by APBS
# ___________________________________________________________
con <- file("apbs_template_new2.in", "r")
in_file <- readLines(con)
close(con)

# Sources all functions written
# ___________________________________________________________
source("cat_pdb.r")
source("neutral_charged.r")
source("call_apbs2.r")
source("our_seq.r")
source("Pka.r")
source("mean.r")
library(bio3d)

# Loads pdb and sequences according to the naming scheme
# ___________________________________________________________
LY2 <- read.pdb("/Users/kaila_bennett/Desktop/Senior_Design/1LY2.pdb")
Neu_Char_pdb(LY2)

# Converts pdb files to pqr files using python connection
# ___________________________________________________________
#system("python /Users/kaila_bennett/Desktop/Senior_Design/pdb2pqr-1.5/pdb2pqr.py --ff parse 1ly2_charged.pdb 1ly2_charged.pqr")
#system("python /Users/kaila_bennett/Desktop/Senior_Design/pdb2pqr-1.5/pdb2pqr.py --ff parse 1ly2_neutral.pdb 1ly2_neutral.pqr")
```

**Code 11: Calculates intrinsic pKa values for APBS thermodynamic cycle**
# reads in our pqr files and these will be concatenated together
neutral_pqr <- read.pqr("1ly2_1_neutral.pqr")
charged_pqr <- read.pqr("1ly2_charged.pqr")

# parameters, extracts sequence of pqr file and creates a array
# of characters, k allows to extract length of pqr
k <- as.numeric(neutral_pqr$atom[1,"resno"])
end_of_seq <- length(seq.pdb(neutral_pqr)) - 1
our_seq <- our_seq(LY2, end_of_seq)
AAdf <- NULL

# runs a loop that will look for ionizable amino acids residues
# when it finds one it will create 4 pqr files to account for
# each state in the thermodynamic cycle, and writes to file
# this will have APBS incorporated into for loop, k counter
# used to keep a numerical value
# run time is about 30 min.
for ( i in seq[1:15] )
{
    if ( i == "R" | i == "K" | i == "H" | i == "C" | i == "Y"
    | i == "D" | i == "E" )
    {
        Before <- trim.pdb( neutral_pqr, atom.select(neutral_pqr,
        resno = 1:(k - 1)) )
        Free_protonated <- trim.pdb( charged_pqr, atom.select
        (charged_pqr, resno = k))
        After <- trim.pdb( neutral_pqr, atom.select(neutral_pqr,
        resno = (k+1):end_of_seq))
        Free_deprotonated <- trim.pdb( neutral_pqr, atom.select
        (neutral_pqr, resno = k))
        write.pqr(Free_protonated, file =
        "Free_protonated.pqr")
        write.pqr(Free_deprotonated, file =
        "Free_deprotonated.pqr")
        Before_FP <- cat.pdb( Before,
        Free_protonated )
        Total <- cat.pdb(Before_FP, After)
        write.pqr(Total, file =
        "Bound_Protonated.pqr")
    }
}

Code 11: Continued
write.pqr(Free_deprotonated, file = "Free_deprotonated.pqr")

indice <- atom.select(neutral_pqr, resno = k)
neutral_pqr$atom[indice$atom,11] <- "0.00"
charged_pqr$atom[indice$atom,11] <- "0.00"
write.pqr(neutral_pqr, file = "Protein_Neu_AA.pqr")
write.pqr(charged_pqr, file = "Protein_Chr_AA.pqr")

bp <- read.pqr("Bound_Protonated.pqr")
bdp <- read.pqr("1ly2_1_neutral.pqr")
fp <- read.pqr("Free_protonated.pqr")
fdp <- read.pqr("Free_deprotonated.pqr")
delta_G <- call_apbs2(in_file)
AAdf <- rbind(AAdf, c("Resid"=i,"Resno" = k+1,"delta_G"=delta_G))
}
k <- k + 1
}

Pka <- NULL
Pka <- pKa(AAdf)

#-----------------------------------------------
#Mean values
#-----------------------------------------------

#-----------------------------------------------
#pka values
#-----------------------------------------------