# Comparative Structure Analysis

Jordan Prych (PID: A17080226)

## Table of contents



Here we run through a complete "pipeline" of structure analysis that begins with a single sequence identifier and ends in a PCA analysis.

Setup:

> Q10. Which of the packages above is found only on BioConductor and not CRAN?

msa

> Q11. Which of the above packages is not found on BioConductor or CRAN?

bio3d-view

> Q12. True or False? Functions from the devtools package can be used to install packages from GitHub and BitBucket?

True

```
library(bio3d)
```

### Step 1: Get a Sequence

retrieve a sequence for the protein we are interested in. We will take ADK "1ake_A"

```
id <- "1ake_A"
aa <- get.seq(id)
```

```
Warning in get.seq(id): Removing existing file: seqs.fasta

Fetching... Please wait. Done.
```

```
aa
```

```
              1        .         .         .         .         .         60
pdb|1AKE|A    MRIILLGAPGAGKGTQAQFIMEKYGIPQISTGDMLRAAVKSGSELGKQAKDIMDAGKLVT
              1        .         .         .         .         .         60

             61        .         .         .         .         .         120
pdb|1AKE|A    DELVIALVKERIAQEDCRNGFLLDGFPRTIPQADAMKEAGINVDYVLEFDVPDELIVDRI
             61        .         .         .         .         .         120

            121        .         .         .         .         .         180
pdb|1AKE|A    VGRRVHAPSGRVYHVKFNPPKVEGKDDVTGEELTTRKDDQEETVRKRLVEYHQMTAPLIG
            121        .         .         .         .         .         180

            181        .         .         .   214
pdb|1AKE|A    YYSKEAEAGNTKYAKVDGTKPVAEVRADLEKILG
            181        .         .         .   214

Call:
  read.fasta(file = outfile)

Class:
  fasta

Alignment dimensions:
  1 sequence rows; 214 position columns (214 non-gap, 0 gap)

+ attr: id, ali, call
```

Q13.How many amino acids are in this sequence, i.e. how long is this sequence?

214 AA long

### Step 2: BLAST Search

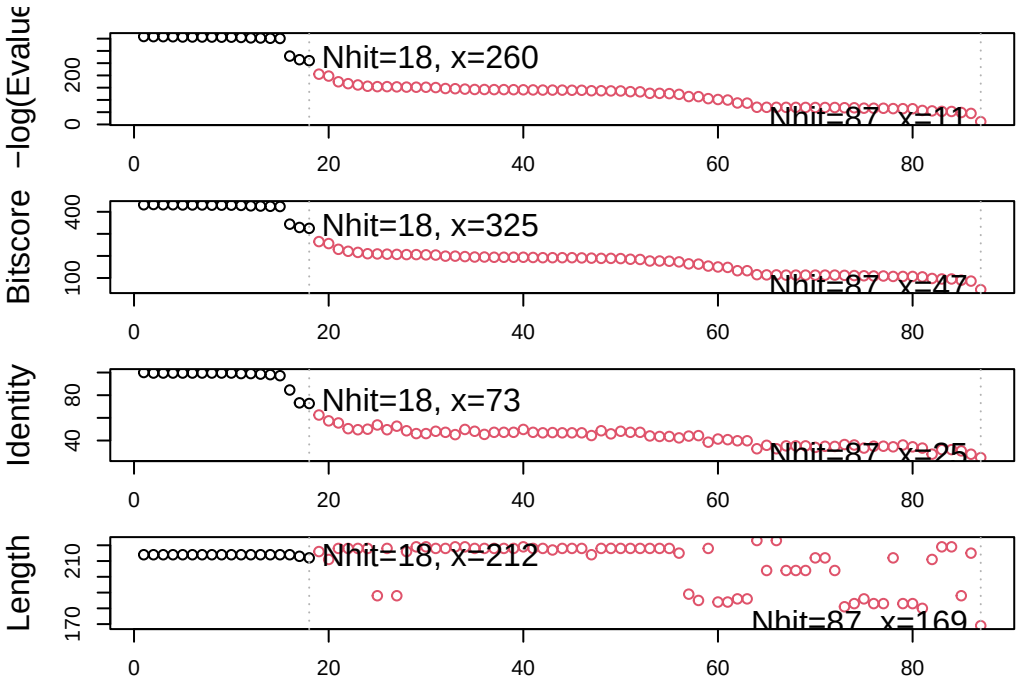
Run a BLAST search of the PDB for all related sequences to our input `aa`

```
blast <- blast.pdb(aa)
```

```
 Searching ... please wait (updates every 5 seconds) RID = V6Z839X9013
 ..................................
 Reporting 87 hits
```

```
hits <- plot(blast)
```

```
 * Possible cutoff values:    260 11
           Yielding Nhits:    18 87

 * Chosen cutoff value of:    260
           Yielding Nhits:    18
```



### Step 3: Download Structures

These are our "top hits" i.e. all the structures in the PDB database relates to our input sequences.

```
hits$pdb.id
```

```
 [1] "1AKE_A" "8BQF_A" "4X8M_A" "6S36_A" "8Q2B_A" "8RJ9_A" "6RZE_A" "4X8H_A"
 [9] "3HPR_A" "1E4V_A" "5EJE_A" "1E4Y_A" "3X2S_A" "6HAP_A" "6HAM_A" "8PVW_A"
[17] "4K46_A" "4NP6_A"
```

```
#Download related PDB files
files <- get.pdb(hits$pdb.id, path="pbds", split=TRUE, gzip=TRUE)
```

```
Warning in get.pdb(hits$pdb.id, path = "pbds", split = TRUE, gzip = TRUE):
pbds/1AKE.pdb exists. Skipping download
```

```
Warning in get.pdb(hits$pdb.id, path = "pbds", split = TRUE, gzip = TRUE):
pbds/8BQF.pdb exists. Skipping download
```

```
Warning in get.pdb(hits$pdb.id, path = "pbds", split = TRUE, gzip = TRUE):
pbds/4X8M.pdb exists. Skipping download
```

```
Warning in get.pdb(hits$pdb.id, path = "pbds", split = TRUE, gzip = TRUE):
pbds/6S36.pdb exists. Skipping download
```

```
Warning in get.pdb(hits$pdb.id, path = "pbds", split = TRUE, gzip = TRUE):
pbds/8Q2B.pdb exists. Skipping download
```

```
Warning in get.pdb(hits$pdb.id, path = "pbds", split = TRUE, gzip = TRUE):
pbds/8RJ9.pdb exists. Skipping download
```

```
Warning in get.pdb(hits$pdb.id, path = "pbds", split = TRUE, gzip = TRUE):
pbds/6RZE.pdb exists. Skipping download
```

```
Warning in get.pdb(hits$pdb.id, path = "pbds", split = TRUE, gzip = TRUE):
pbds/4X8H.pdb exists. Skipping download
```

```
Warning in get.pdb(hits$pdb.id, path = "pbds", split = TRUE, gzip = TRUE):
pbds/3HPR.pdb exists. Skipping download

Warning in get.pdb(hits$pdb.id, path = "pbds", split = TRUE, gzip = TRUE):
pbds/1E4V.pdb exists. Skipping download

Warning in get.pdb(hits$pdb.id, path = "pbds", split = TRUE, gzip = TRUE):
pbds/5EJE.pdb exists. Skipping download

Warning in get.pdb(hits$pdb.id, path = "pbds", split = TRUE, gzip = TRUE):
pbds/1E4Y.pdb exists. Skipping download

Warning in get.pdb(hits$pdb.id, path = "pbds", split = TRUE, gzip = TRUE):
pbds/3X2S.pdb exists. Skipping download

Warning in get.pdb(hits$pdb.id, path = "pbds", split = TRUE, gzip = TRUE):
pbds/6HAP.pdb exists. Skipping download

Warning in get.pdb(hits$pdb.id, path = "pbds", split = TRUE, gzip = TRUE):
pbds/6HAM.pdb exists. Skipping download

Warning in get.pdb(hits$pdb.id, path = "pbds", split = TRUE, gzip = TRUE):
pbds/8PVW.pdb exists. Skipping download

Warning in get.pdb(hits$pdb.id, path = "pbds", split = TRUE, gzip = TRUE):
pbds/4K46.pdb exists. Skipping download

Warning in get.pdb(hits$pdb.id, path = "pbds", split = TRUE, gzip = TRUE):
pbds/4NP6.pdb exists. Skipping download


  |
  |                                                                      |   0%
  |
  |====                                                                  |   6%
  |
  |========                                                              |  11%
  |
  |============                                                          |  17%
```

```
  |
  |================                                                      |  22%
  |
  |==================                                                    |  28%
  |
  |======================                                                |  33%
  |
  |==========================                                            |  39%
  |
  |==============================                                        |  44%
  |
  |==================================                                    |  50%
  |
  |======================================                                |  56%
  |
  |==========================================                            |  61%
  |
  |==============================================                        |  67%
  |
  |==================================================                    |  72%
  |
  |======================================================                |  78%
  |
  |==========================================================            |  83%
  |
  |================================================================      |  89%
  |
  |====================================================================  |  94%
  |
  |======================================================================| 100%
```

```
Warning in pdbsplit(pdb.files = names(rtn), ids = ids, path = file.path(path, :
unmatched pdb files: 4X8H
```

```
Warning in pdbsplit(pdb.files = names(rtn), ids = ids, path = file.path(path, :
unmatched ids: 4X8H_A
```

### Step 4: Align and Superpose

```
# Align releated PDBs
pbds <- pdbaln(files, fit = TRUE, exefile="msa")
```

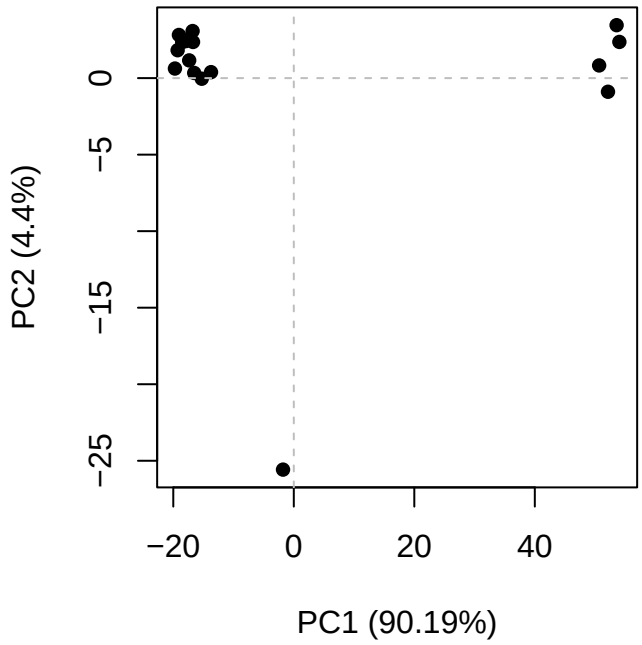
```
Reading PDB files:
pbds/split_chain/1AKE_A.pdb
pbds/split_chain/8BQF_A.pdb
pbds/split_chain/4X8M_A.pdb
pbds/split_chain/6S36_A.pdb
pbds/split_chain/8Q2B_A.pdb
pbds/split_chain/8RJ9_A.pdb
pbds/split_chain/6RZE_A.pdb
pbds/split_chain/3HPR_A.pdb
pbds/split_chain/1E4V_A.pdb
pbds/split_chain/5EJE_A.pdb
pbds/split_chain/1E4Y_A.pdb
pbds/split_chain/3X2S_A.pdb
pbds/split_chain/6HAP_A.pdb
pbds/split_chain/6HAM_A.pdb
pbds/split_chain/8PVW_A.pdb
pbds/split_chain/4K46_A.pdb
pbds/split_chain/4NP6_A.pdb
   PDB has ALT records, taking A only, rm.alt=TRUE
.   PDB has ALT records, taking A only, rm.alt=TRUE
..   PDB has ALT records, taking A only, rm.alt=TRUE
.   PDB has ALT records, taking A only, rm.alt=TRUE
.   PDB has ALT records, taking A only, rm.alt=TRUE
.   PDB has ALT records, taking A only, rm.alt=TRUE
.   PDB has ALT records, taking A only, rm.alt=TRUE
..   PDB has ALT records, taking A only, rm.alt=TRUE
....   PDB has ALT records, taking A only, rm.alt=TRUE
.   PDB has ALT records, taking A only, rm.alt=TRUE
.   PDB has ALT records, taking A only, rm.alt=TRUE
..

Extracting sequences

pdb/seq: 1   name: pbds/split_chain/1AKE_A.pdb
   PDB has ALT records, taking A only, rm.alt=TRUE
pdb/seq: 2   name: pbds/split_chain/8BQF_A.pdb
   PDB has ALT records, taking A only, rm.alt=TRUE
pdb/seq: 3   name: pbds/split_chain/4X8M_A.pdb
```

```
pdb/seq: 4   name: pbds/split_chain/6S36_A.pdb
   PDB has ALT records, taking A only, rm.alt=TRUE
pdb/seq: 5   name: pbds/split_chain/8Q2B_A.pdb
   PDB has ALT records, taking A only, rm.alt=TRUE
pdb/seq: 6   name: pbds/split_chain/8RJ9_A.pdb
   PDB has ALT records, taking A only, rm.alt=TRUE
pdb/seq: 7   name: pbds/split_chain/6RZE_A.pdb
   PDB has ALT records, taking A only, rm.alt=TRUE
pdb/seq: 8   name: pbds/split_chain/3HPR_A.pdb
   PDB has ALT records, taking A only, rm.alt=TRUE
pdb/seq: 9   name: pbds/split_chain/1E4V_A.pdb
pdb/seq: 10   name: pbds/split_chain/5EJE_A.pdb
   PDB has ALT records, taking A only, rm.alt=TRUE
pdb/seq: 11   name: pbds/split_chain/1E4Y_A.pdb
pdb/seq: 12   name: pbds/split_chain/3X2S_A.pdb
pdb/seq: 13   name: pbds/split_chain/6HAP_A.pdb
pdb/seq: 14   name: pbds/split_chain/6HAM_A.pdb
   PDB has ALT records, taking A only, rm.alt=TRUE
pdb/seq: 15   name: pbds/split_chain/8PVW_A.pdb
   PDB has ALT records, taking A only, rm.alt=TRUE
pdb/seq: 16   name: pbds/split_chain/4K46_A.pdb
   PDB has ALT records, taking A only, rm.alt=TRUE
pdb/seq: 17   name: pbds/split_chain/4NP6_A.pdb
```

### Step 5: PCA

Let's use our old friend `PCA` to make sense of the confusing, complicated structure relationships.

```
pca <- pca(pbds)
plot(pca, 1:2)
```




## RMSD ANALYSIS

RMSD is a common measure of structural distance used in structural biology.

```
rd <- rmsd(pbds, fit=T)
```

```
Warning in rmsd(pbds, fit = T): No indices provided, using the 182 non NA positions
```

```
range(rd)
```

```
[1] 0.000 5.546
```

Let's make a trajectory (or movie) of the main confrontational changes captured by PC1. We will use the `mktrj()`.

```
mktrj(pca, file="pca~results.pdb")
```

Figure 1: Main Confrontational Changes by PC1

Back of the envelope comparison of the PDB size to UniProt

```
uniprot <- 253206171
pdb <- 231029
pdb/uniprot *100
```

```
[1] 0.09124146
```

### AlphaFold with HIP-pr dimer analysis

```
results_dir1 <- "HIVprdimer_23119"
```

```
pdb_filesA <- list.files(path=results_dir1, pattern="*.pdb", full.names=TRUE)

basename(pdb_filesA)
```

```
[1] "HIVprdimer_23119_unrelaxed_rank_001_alphafold2_multimer_v3_model_1_seed_000.pdb"
[2] "HIVprdimer_23119_unrelaxed_rank_002_alphafold2_multimer_v3_model_5_seed_000.pdb"
[3] "HIVprdimer_23119_unrelaxed_rank_003_alphafold2_multimer_v3_model_4_seed_000.pdb"
[4] "HIVprdimer_23119_unrelaxed_rank_004_alphafold2_multimer_v3_model_2_seed_000.pdb"
[5] "HIVprdimer_23119_unrelaxed_rank_005_alphafold2_multimer_v3_model_3_seed_000.pdb"
```

```
library(bio3d)

pdbs3 <- pdbaln(pdb_filesA, fit=TRUE, exefile="msa")
```

```
Reading PDB files:
HIVprdimer_23119/HIVprdimer_23119_unrelaxed_rank_001_alphafold2_multimer_v3_model_1_seed_000
HIVprdimer_23119/HIVprdimer_23119_unrelaxed_rank_002_alphafold2_multimer_v3_model_5_seed_000
HIVprdimer_23119/HIVprdimer_23119_unrelaxed_rank_003_alphafold2_multimer_v3_model_4_seed_000
HIVprdimer_23119/HIVprdimer_23119_unrelaxed_rank_004_alphafold2_multimer_v3_model_2_seed_000
HIVprdimer_23119/HIVprdimer_23119_unrelaxed_rank_005_alphafold2_multimer_v3_model_3_seed_000
.....

Extracting sequences

pdb/seq: 1   name: HIVprdimer_23119/HIVprdimer_23119_unrelaxed_rank_001_alphafold2_multimer_v
pdb/seq: 2   name: HIVprdimer_23119/HIVprdimer_23119_unrelaxed_rank_002_alphafold2_multimer_v
pdb/seq: 3   name: HIVprdimer_23119/HIVprdimer_23119_unrelaxed_rank_003_alphafold2_multimer_v
pdb/seq: 4   name: HIVprdimer_23119/HIVprdimer_23119_unrelaxed_rank_004_alphafold2_multimer_v
pdb/seq: 5   name: HIVprdimer_23119/HIVprdimer_23119_unrelaxed_rank_005_alphafold2_multimer_v
```

```
pdbs3
```

```
                                1        .        .        .        .        50
[Truncated_Name:1]HIVprdimer   PQITLWQRPLVTIKIGGQLKEALLDTGADDTVLEEMSLPGRWKPKMIGGI
[Truncated_Name:2]HIVprdimer   PQITLWQRPLVTIKIGGQLKEALLDTGADDTVLEEMSLPGRWKPKMIGGI
[Truncated_Name:3]HIVprdimer   PQITLWQRPLVTIKIGGQLKEALLDTGADDTVLEEMSLPGRWKPKMIGGI
[Truncated_Name:4]HIVprdimer   PQITLWQRPLVTIKIGGQLKEALLDTGADDTVLEEMSLPGRWKPKMIGGI
[Truncated_Name:5]HIVprdimer   PQITLWQRPLVTIKIGGQLKEALLDTGADDTVLEEMSLPGRWKPKMIGGI
                               **************************************************
                                1        .        .        .        .        50

                               51        .        .        .        .        100
[Truncated_Name:1]HIVprdimer   GGFIKVRQYDQILIEICGHKAIGTVLVGPTPVNIIGRNLLTQIGCTLNFP
[Truncated_Name:2]HIVprdimer   GGFIKVRQYDQILIEICGHKAIGTVLVGPTPVNIIGRNLLTQIGCTLNFP
[Truncated_Name:3]HIVprdimer   GGFIKVRQYDQILIEICGHKAIGTVLVGPTPVNIIGRNLLTQIGCTLNFP
[Truncated_Name:4]HIVprdimer   GGFIKVRQYDQILIEICGHKAIGTVLVGPTPVNIIGRNLLTQIGCTLNFP
[Truncated_Name:5]HIVprdimer   GGFIKVRQYDQILIEICGHKAIGTVLVGPTPVNIIGRNLLTQIGCTLNFP
                               **************************************************
                               51        .        .        .        .        100

                              101        .        .        .        .        150
[Truncated_Name:1]HIVprdimer   QITLWQRPLVTIKIGGQLKEALLDTGADDTVLEEMSLPGRWKPKMIGGIG
[Truncated_Name:2]HIVprdimer   QITLWQRPLVTIKIGGQLKEALLDTGADDTVLEEMSLPGRWKPKMIGGIG
[Truncated_Name:3]HIVprdimer   QITLWQRPLVTIKIGGQLKEALLDTGADDTVLEEMSLPGRWKPKMIGGIG
[Truncated_Name:4]HIVprdimer   QITLWQRPLVTIKIGGQLKEALLDTGADDTVLEEMSLPGRWKPKMIGGIG
[Truncated_Name:5]HIVprdimer   QITLWQRPLVTIKIGGQLKEALLDTGADDTVLEEMSLPGRWKPKMIGGIG
                               **************************************************
                              101        .        .        .        .        150

                              151        .        .        .        .      198
[Truncated_Name:1]HIVprdimer   GFIKVRQYDQILIEICGHKAIGTVLVGPTPVNIIGRNLLTQIGCTLNF
[Truncated_Name:2]HIVprdimer   GFIKVRQYDQILIEICGHKAIGTVLVGPTPVNIIGRNLLTQIGCTLNF
[Truncated_Name:3]HIVprdimer   GFIKVRQYDQILIEICGHKAIGTVLVGPTPVNIIGRNLLTQIGCTLNF
[Truncated_Name:4]HIVprdimer   GFIKVRQYDQILIEICGHKAIGTVLVGPTPVNIIGRNLLTQIGCTLNF
[Truncated_Name:5]HIVprdimer   GFIKVRQYDQILIEICGHKAIGTVLVGPTPVNIIGRNLLTQIGCTLNF
                               *********************************************
                              151        .        .        .        .      198

Call:
  pdbaln(files = pdb_filesA, fit = TRUE, exefile = "msa")

Class:
```

```
  pdbs, fasta

Alignment dimensions:
  5 sequence rows; 198 position columns (198 non-gap, 0 gap)

+ attr: xyz, resno, b, chain, id, ali, resid, sse, call
```

Now we can calculate RMSD

```
rd2 <- rmsd(pdbs3, fit=TRUE)
```

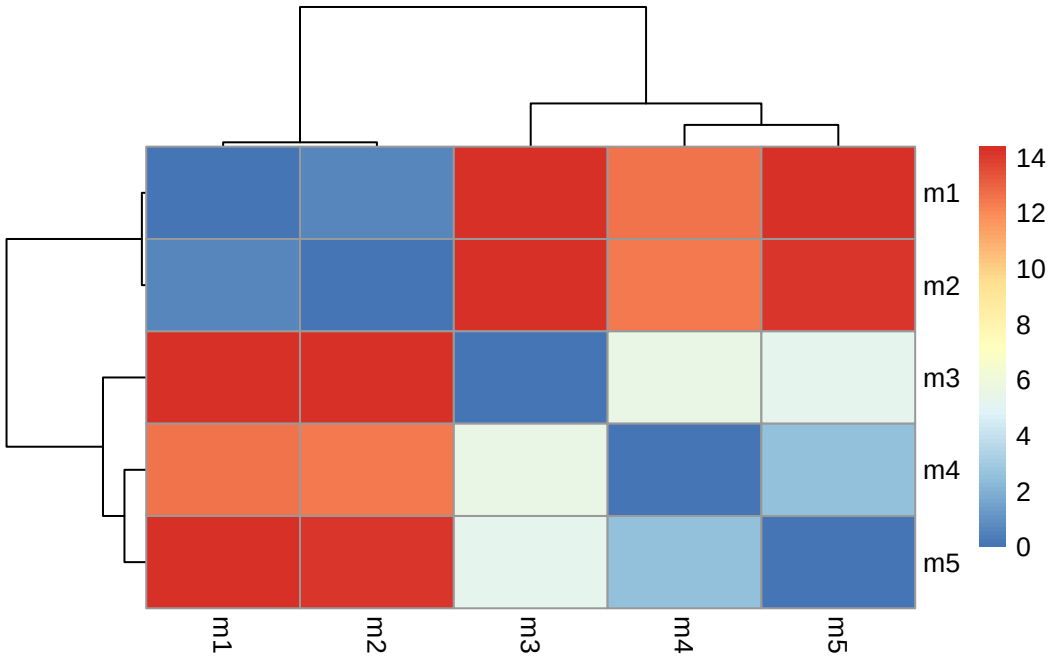
```
Warning in rmsd(pdbs3, fit = TRUE): No indices provided, using the 198 non NA positions
```

```
range(rd2)
```

```
[1]  0.000 14.376
```

Now we can draw a heatmap

```
library(pheatmap)

colnames(rd2) <- paste0("m",1:5)
rownames(rd2) <- paste0("m",1:5)
pheatmap(rd2)
```

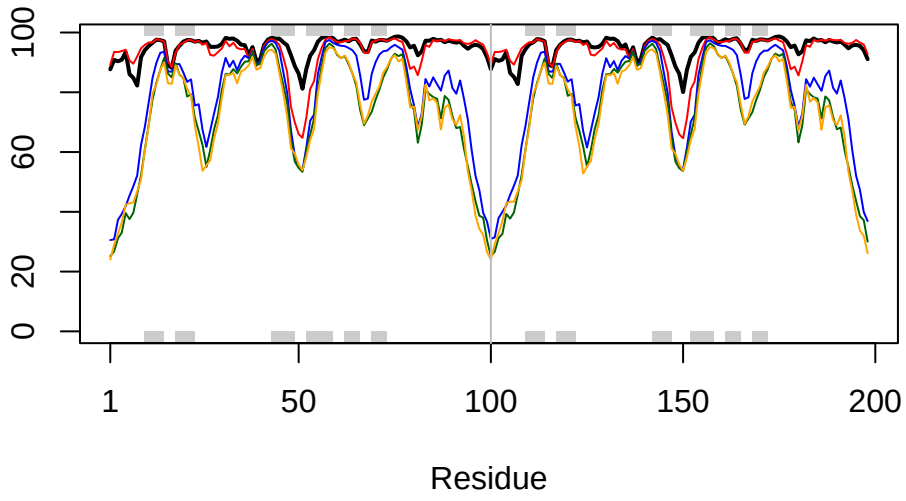



Now we can plot the pLDDT values across all models

```
pdb <- read.pdb("1hsg")
```

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

```
plotb3(pdbs3$b[1,], typ="l", lwd=2, sse=pdb)
points(pdbs3$b[2,], typ="l", col="red")
points(pdbs3$b[3,], typ="l", col="blue")
points(pdbs3$b[4,], typ="l", col="darkgreen")
points(pdbs3$b[5,], typ="l", col="orange")
abline(v=100, col="gray")
```




To improve the superimposition/fitting of our models:

```
core <- core.find(pdbs3)
```

```
 core size 197 of 198  vol = 4916.702
 core size 196 of 198  vol = 4311.481
 core size 195 of 198  vol = 4101.445
 core size 194 of 198  vol = 3907.124
 core size 193 of 198  vol = 3711.925
 core size 192 of 198  vol = 3546.511
 core size 191 of 198  vol = 3440.437
 core size 190 of 198  vol = 3317.571
 core size 189 of 198  vol = 3220.079
 core size 188 of 198  vol = 3142.057
 core size 187 of 198  vol = 3066.79
 core size 186 of 198  vol = 3015.892
 core size 185 of 198  vol = 2959.969
 core size 184 of 198  vol = 2913.74
 core size 183 of 198  vol = 2880.923
 core size 182 of 198  vol = 2848.081
 core size 181 of 198  vol = 2857.001
 core size 180 of 198  vol = 2871.24
 core size 179 of 198  vol = 2905.696
```

```
core size 178 of 198  vol = 2953.776
core size 177 of 198  vol = 3020.847
core size 176 of 198  vol = 3087.22
core size 175 of 198  vol = 3109.99
core size 174 of 198  vol = 3129.601
core size 173 of 198  vol = 3135.085
core size 172 of 198  vol = 3092.283
core size 171 of 198  vol = 3036.012
core size 170 of 198  vol = 2947.995
core size 169 of 198  vol = 2886.897
core size 168 of 198  vol = 2829.355
core size 167 of 198  vol = 2746.377
core size 166 of 198  vol = 2671.189
core size 165 of 198  vol = 2600.848
core size 164 of 198  vol = 2534.651
core size 163 of 198  vol = 2464.3
core size 162 of 198  vol = 2390.171
core size 161 of 198  vol = 2322.47
core size 160 of 198  vol = 2236.698
core size 159 of 198  vol = 2160.475
core size 158 of 198  vol = 2077.281
core size 157 of 198  vol = 2003.596
core size 156 of 198  vol = 1939.94
core size 155 of 198  vol = 1859.188
core size 154 of 198  vol = 1781.083
core size 153 of 198  vol = 1699.1
core size 152 of 198  vol = 1622.558
core size 151 of 198  vol = 1546.319
core size 150 of 198  vol = 1473.01
core size 149 of 198  vol = 1414.087
core size 148 of 198  vol = 1352.547
core size 147 of 198  vol = 1295.278
core size 146 of 198  vol = 1246.999
core size 145 of 198  vol = 1203.962
core size 144 of 198  vol = 1163.009
core size 143 of 198  vol = 1110.955
core size 142 of 198  vol = 1064.672
core size 141 of 198  vol = 1028.458
core size 140 of 198  vol = 986.121
core size 139 of 198  vol = 944.003
core size 138 of 198  vol = 895.914
core size 137 of 198  vol = 853.508
core size 136 of 198  vol = 827.977
```

```
core size 135 of 198  vol = 796.874
core size 134 of 198  vol = 772.763
core size 133 of 198  vol = 743.108
core size 132 of 198  vol = 707.65
core size 131 of 198  vol = 669.172
core size 130 of 198  vol = 634.655
core size 129 of 198  vol = 594.035
core size 128 of 198  vol = 559.154
core size 127 of 198  vol = 525.971
core size 126 of 198  vol = 493.19
core size 125 of 198  vol = 466.473
core size 124 of 198  vol = 438.433
core size 123 of 198  vol = 410.725
core size 122 of 198  vol = 401.38
core size 121 of 198  vol = 391.76
core size 120 of 198  vol = 362.084
core size 119 of 198  vol = 338.183
core size 118 of 198  vol = 312.338
core size 117 of 198  vol = 282.176
core size 116 of 198  vol = 262.215
core size 115 of 198  vol = 241.577
core size 114 of 198  vol = 225.151
core size 113 of 198  vol = 204.137
core size 112 of 198  vol = 185.038
core size 111 of 198  vol = 162.728
core size 110 of 198  vol = 146.181
core size 109 of 198  vol = 133.352
core size 108 of 198  vol = 123.207
core size 107 of 198  vol = 109.228
core size 106 of 198  vol = 98.824
core size 105 of 198  vol = 89.735
core size 104 of 198  vol = 81.206
core size 103 of 198  vol = 74.188
core size 102 of 198  vol = 67.042
core size 101 of 198  vol = 62.043
core size 100 of 198  vol = 58.432
core size 99 of 198  vol = 55.149
core size 98 of 198  vol = 51.114
core size 97 of 198  vol = 45.798
core size 96 of 198  vol = 41.161
core size 95 of 198  vol = 35.619
core size 94 of 198  vol = 29.784
core size 93 of 198  vol = 23.233
```

```
 core size 92 of 198  vol = 16.669
 core size 91 of 198  vol = 9.459
 core size 90 of 198  vol = 4.595
 core size 89 of 198  vol = 3.161
 core size 88 of 198  vol = 2.678
 core size 87 of 198  vol = 2.293
 core size 86 of 198  vol = 1.935
 core size 85 of 198  vol = 1.619
 core size 84 of 198  vol = 1.367
 core size 83 of 198  vol = 1.09
 core size 82 of 198  vol = 0.906
 core size 81 of 198  vol = 0.764
 core size 80 of 198  vol = 0.649
 core size 79 of 198  vol = 0.596
 core size 78 of 198  vol = 0.53
 core size 77 of 198  vol = 0.486
 FINISHED: Min vol ( 0.5 ) reached
```

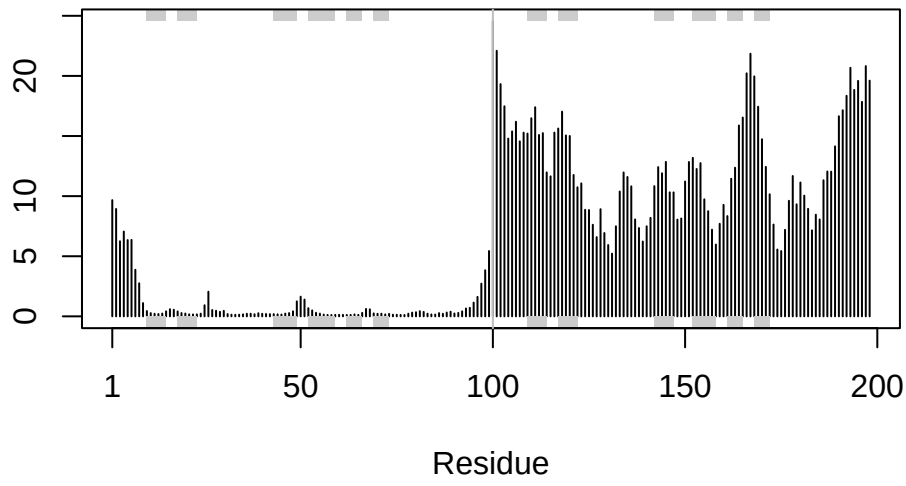
```
core.inds <- print(core, vol=0.5)
```

```
# 78 positions (cumulative volume <= 0.5 Angstrom^3)
  start end length
1    10  25     16
2    28  48     21
3    53  93     41
```

```
xyz <- pdbfit(pdbs3, core.inds, outpath="corefit_structures3")
```

Now we can examine the RMSF values between positions of the structure

```
rf <- rmsf(xyz)

plotb3(rf, sse=pdb)
abline(v=100, col="gray", ylab="RMSF")
```




### Predicted Alignment Error for Domains

```
library(jsonlite)

# Listing of all PAE JSON files
pae_files <- list.files(path=results_dir1,
                        pattern=".*model.*\\.json",
                        full.names = TRUE)
```

```
pae1 <- read_json(pae_files[1],simplifyVector = TRUE)
pae5 <- read_json(pae_files[5],simplifyVector = TRUE)

attributes(pae1)
```

```
$names
[1] "plddt"   "max_pae" "pae"     "ptm"     "iptm"
```

```
# Per-residue pLDDT scores
#  same as B-factor of PDB..
head(pae1$plddt)
```

```
[1] 87.69 90.81 90.38 90.88 93.44 86.06
```

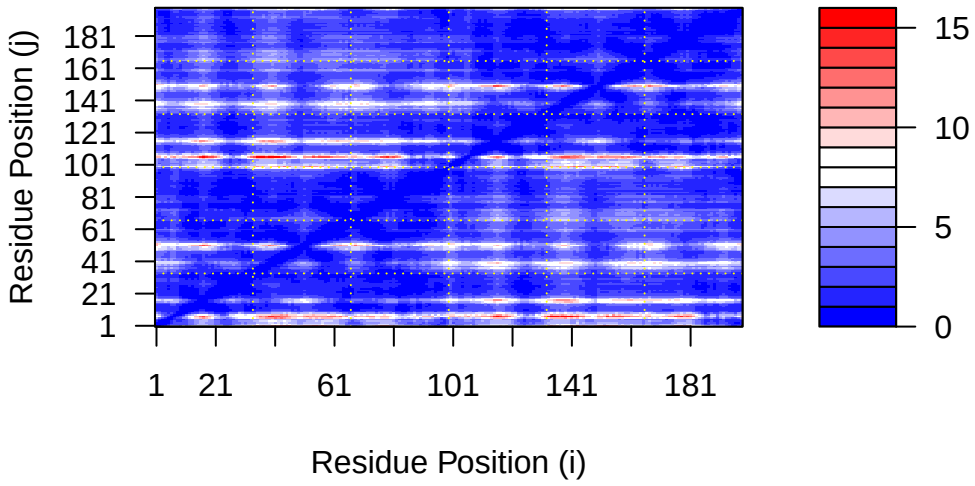
```
pae1$max_pae
```

```
[1] 15.47656
```

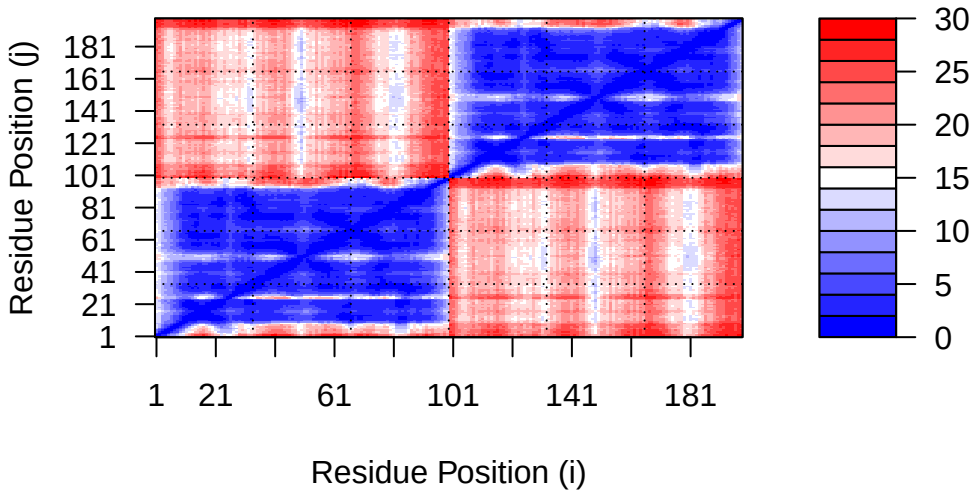
```
pae5$max_pae
```

```
[1] 29.32812
```

```
plot.dmat(pae1$pae,
          xlab="Residue Position (i)",
          ylab="Residue Position (j)")
```

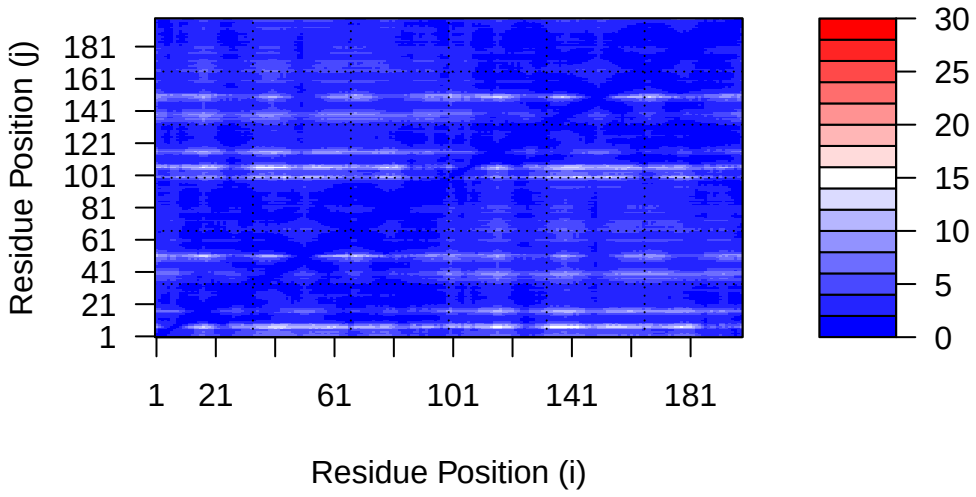



```
plot.dmat(pae5$pae,
          xlab="Residue Position (i)",
          ylab="Residue Position (j)",
          grid.col = "black",
          zlim=c(0,30))
```




Here is the model 1 plot but using the same data range as the plot for model 5

```
plot.dmat(pae1$pae,
          xlab="Residue Position (i)",
          ylab="Residue Position (j)",
          grid.col = "black",
          zlim=c(0,30))
```




### Residue Conservation from Alignment File

```
aln_file <- list.files(path=results_dir1,
                       pattern=".a3m$",
                        full.names = TRUE)
aln_file
```

```
[1] "HIVprdimer_23119/HIVprdimer_23119.a3m"
```

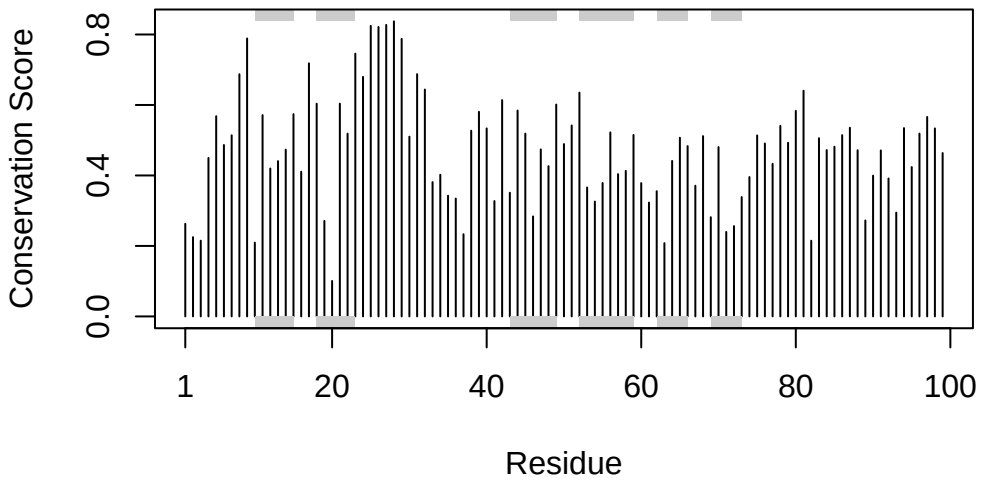
```
aln <- read.fasta(aln_file[1], to.upper = TRUE)
```

```
[1] " ** Duplicated sequence id's: 101 **"
[2] " ** Duplicated sequence id's: 101 **"
```

```
dim(aln$ali)
```

```
[1] 5378  132
```

```
sim <- conserv(aln)

plotb3(sim[1:99], sse=trim.pdb(pdb, chain="A"),
       ylab="Conservation Score")
```




```
con <- consensus(aln, cutoff = 0.9)
con$seq
```

```
  [1] "-" "-" "-" "-" "-" "-" "-" "-" "-" "-" "-" "-" "-" "-" "-" "-" "-" "-"
 [19] "-" "-" "-" "-" "-" "-" "D" "T" "G" "A" "-" "-" "-" "-" "-" "-" "-" "-"
 [37] "-" "-" "-" "-" "-" "-" "-" "-" "-" "-" "-" "-" "-" "-" "-" "-" "-" "-"
 [55] "-" "-" "-" "-" "-" "-" "-" "-" "-" "-" "-" "-" "-" "-" "-" "-" "-" "-"
 [73] "-" "-" "-" "-" "-" "-" "-" "-" "-" "-" "-" "-" "-" "-" "-" "-" "-" "-"
 [91] "-" "-" "-" "-" "-" "-" "-" "-" "-" "-" "-" "-" "-" "-" "-" "-" "-" "-"
[109] "-" "-" "-" "-" "-" "-" "-" "-" "-" "-" "-" "-" "-" "-" "-" "-" "-" "-"
[127] "-" "-" "-" "-" "-" "-"
```

For final visualization, we can map this conservation score of the occupancy column of a PDB file for Mol* viewing

```
m1.pdb <- read.pdb(pdb_filesA[1])
occ <- vec2resno(c(sim[1:99], sim[1:99]), m1.pdb$atom$resno)
write.pdb(m1.pdb, o=occ, file="m1_conserv.pdb")
```

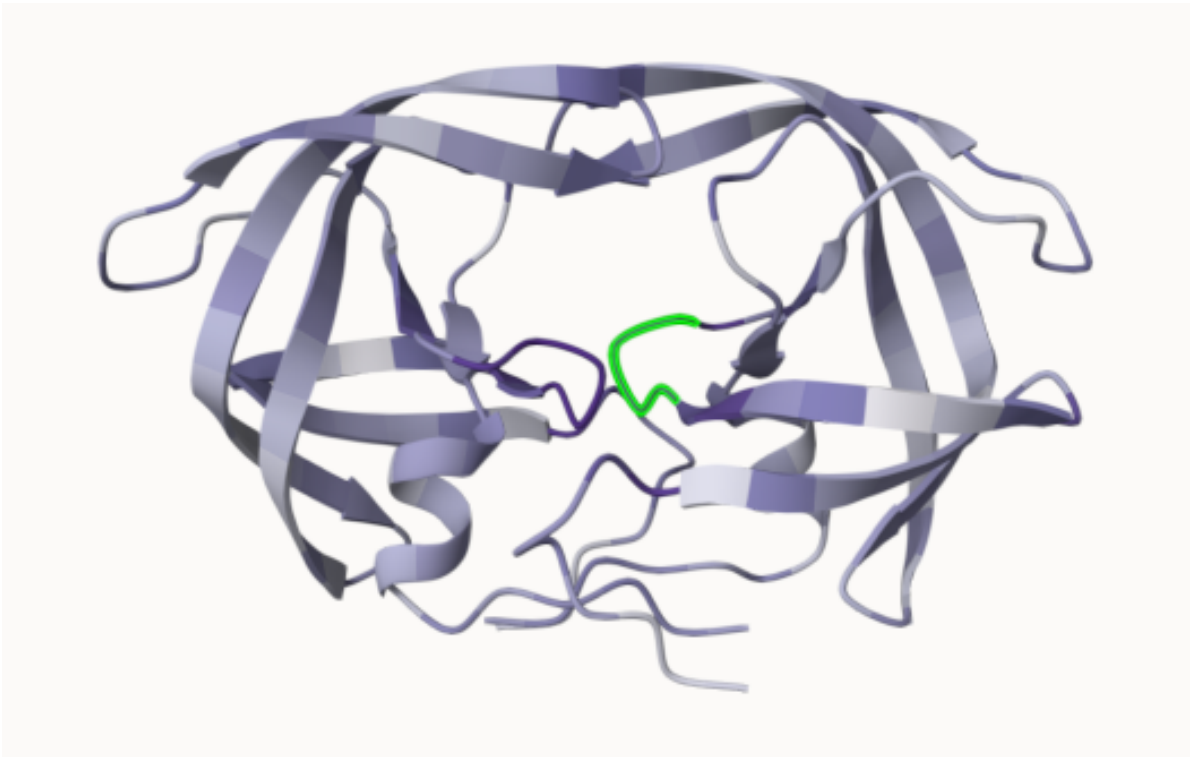


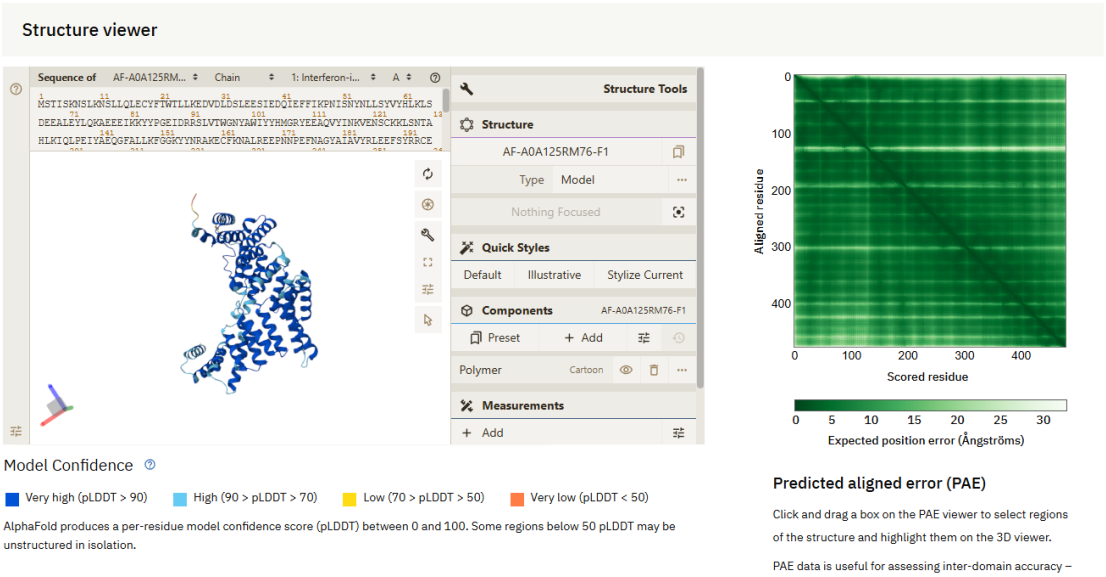
Figure 2: Dimer colored by sequence conservation. DTGA motif highlighted in green

### Alpha Fold With Novel Protein

First, take the sequence of the novel protein to search AFDB

The top hit was named “Interferon-induced protein with tetratricopeptide repeats 5” from Gallus gallus with ID A0A125RM76.

Visit corresponding AFDB structure page




Next, input the FAFSA sequence into AlphaFold2.

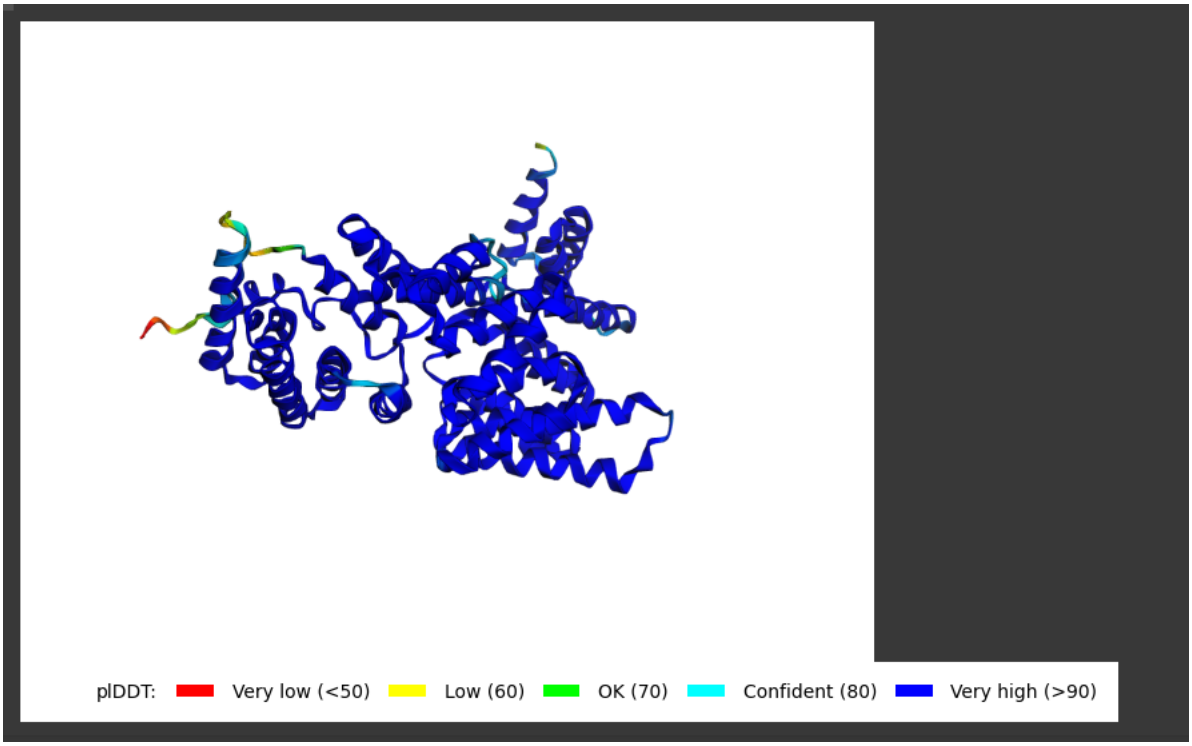



Figure 3: AlphaFold Structure model colored by IDDT Values

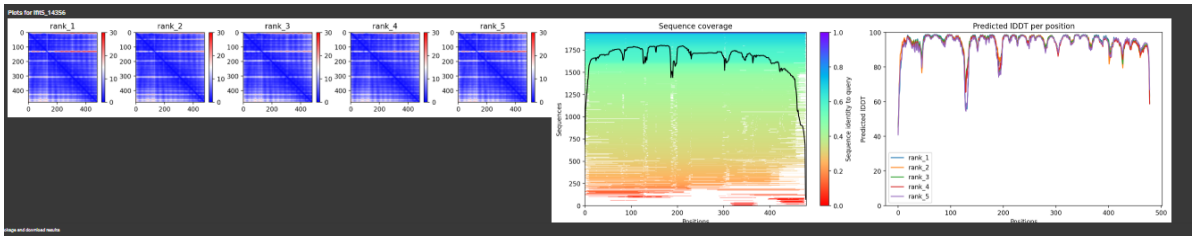
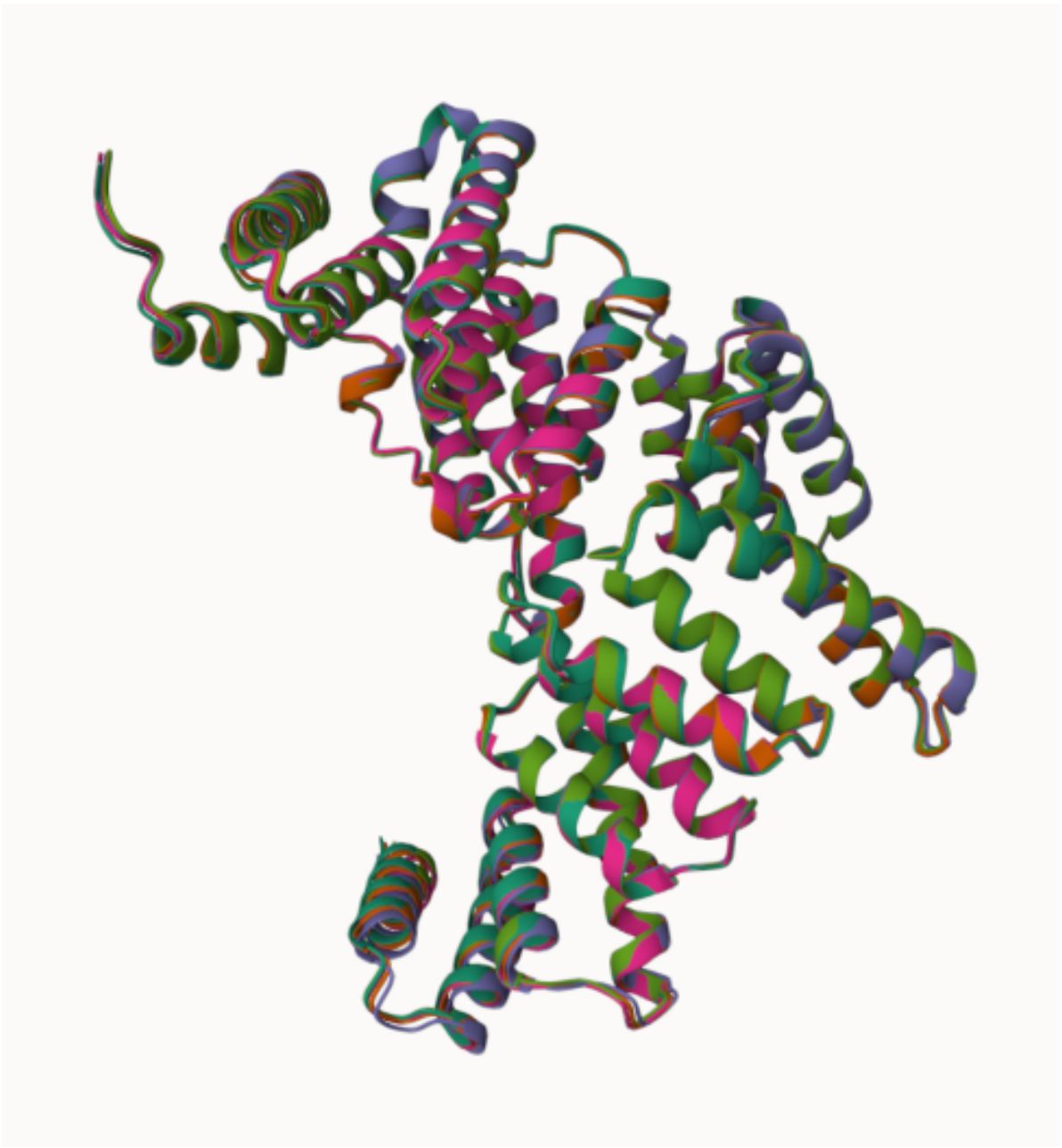


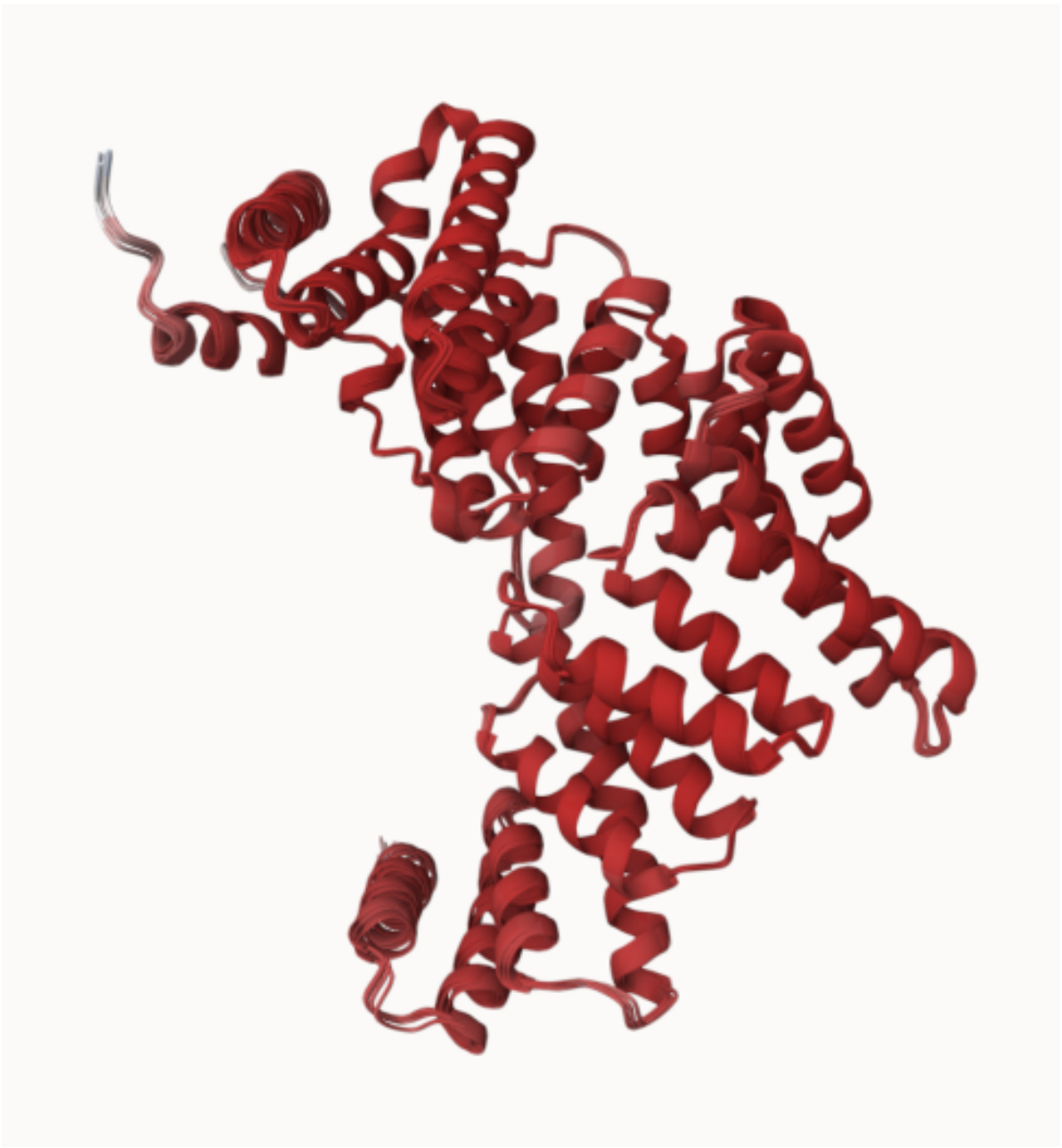

Figure 4: Sequence Coverage in MSA

## Visualization of the Models

we can use Mol* for visualization of your model PDB files



Next we can color by pLDDT scores



```
#IFIT5 results
results_dir <- "ifit5_14356"
```

```
pdb_files <- list.files(path=results_dir, pattern="*.pdb", full.names=TRUE)

basename(pdb_files)
```

```
[1] "ifit5_14356_unrelaxed_rank_001_alphafold2_ptm_model_5_seed_000.pdb"
[2] "ifit5_14356_unrelaxed_rank_002_alphafold2_ptm_model_3_seed_000.pdb"
[3] "ifit5_14356_unrelaxed_rank_003_alphafold2_ptm_model_4_seed_000.pdb"
[4] "ifit5_14356_unrelaxed_rank_004_alphafold2_ptm_model_2_seed_000.pdb"
[5] "ifit5_14356_unrelaxed_rank_005_alphafold2_ptm_model_1_seed_000.pdb"
```

```
library(bio3d)

pdbs2 <- pdbaln(pdb_files, fit=TRUE, exefile="msa")
```

```
Reading PDB files:
ifit5_14356/ifit5_14356_unrelaxed_rank_001_alphafold2_ptm_model_5_seed_000.pdb
ifit5_14356/ifit5_14356_unrelaxed_rank_002_alphafold2_ptm_model_3_seed_000.pdb
ifit5_14356/ifit5_14356_unrelaxed_rank_003_alphafold2_ptm_model_4_seed_000.pdb
ifit5_14356/ifit5_14356_unrelaxed_rank_004_alphafold2_ptm_model_2_seed_000.pdb
ifit5_14356/ifit5_14356_unrelaxed_rank_005_alphafold2_ptm_model_1_seed_000.pdb
.....

Extracting sequences

pdb/seq: 1   name: ifit5_14356/ifit5_14356_unrelaxed_rank_001_alphafold2_ptm_model_5_seed_000
pdb/seq: 2   name: ifit5_14356/ifit5_14356_unrelaxed_rank_002_alphafold2_ptm_model_3_seed_000
pdb/seq: 3   name: ifit5_14356/ifit5_14356_unrelaxed_rank_003_alphafold2_ptm_model_4_seed_000
pdb/seq: 4   name: ifit5_14356/ifit5_14356_unrelaxed_rank_004_alphafold2_ptm_model_2_seed_000
pdb/seq: 5   name: ifit5_14356/ifit5_14356_unrelaxed_rank_005_alphafold2_ptm_model_1_seed_000
```

```
pdbs2
```

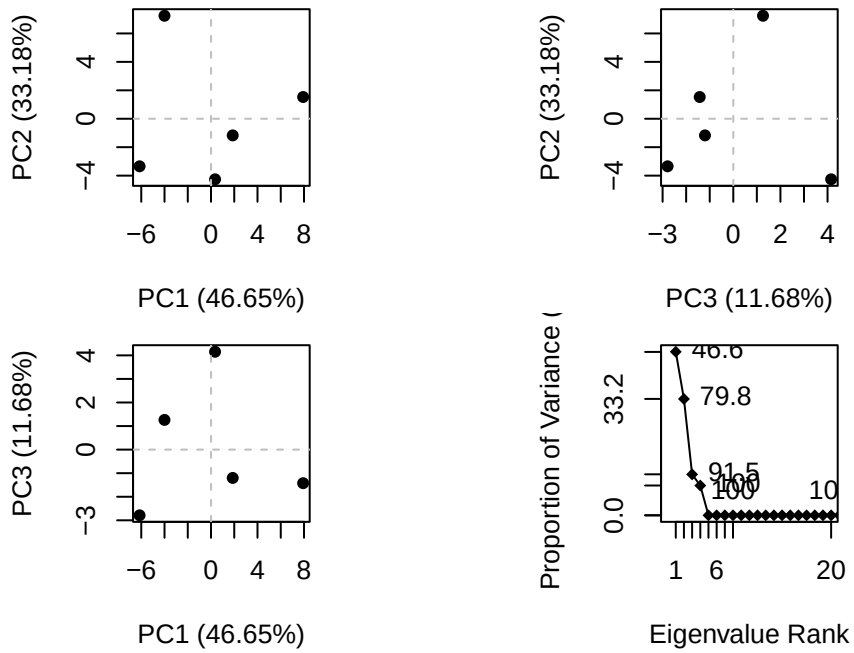
```
                                 1        .         .         .         .         50
[Truncated_Name:1]ifit5_1435   MSTISKNSLKNSLLQLECYFTWTLLKEDVDLDSLEESIEDQIEFFIKPNI
[Truncated_Name:2]ifit5_1435   MSTISKNSLKNSLLQLECYFTWTLLKEDVDLDSLEESIEDQIEFFIKPNI
[Truncated_Name:3]ifit5_1435   MSTISKNSLKNSLLQLECYFTWTLLKEDVDLDSLEESIEDQIEFFIKPNI
[Truncated_Name:4]ifit5_1435   MSTISKNSLKNSLLQLECYFTWTLLKEDVDLDSLEESIEDQIEFFIKPNI
[Truncated_Name:5]ifit5_1435   MSTISKNSLKNSLLQLECYFTWTLLKEDVDLDSLEESIEDQIEFFIKPNI
                               **************************************************
                                 1        .         .         .         .         50

                               51        .         .         .         .         100
[Truncated_Name:1]ifit5_1435   SNYNLLSYVYHLKLSDEEALEYLQKAEEEIKKYYPGEIDRRSLVTWGNYA
[Truncated_Name:2]ifit5_1435   SNYNLLSYVYHLKLSDEEALEYLQKAEEEIKKYYPGEIDRRSLVTWGNYA
[Truncated_Name:3]ifit5_1435   SNYNLLSYVYHLKLSDEEALEYLQKAEEEIKKYYPGEIDRRSLVTWGNYA
```

```
[Truncated_Name:4]ifit5_1435   SNYNLLSYVYHLKLSDEEALEYLQKAEEEIKKYYPGEIDRRSLVTWGNYA
[Truncated_Name:5]ifit5_1435   SNYNLLSYVYHLKLSDEEALEYLQKAEEEIKKYYPGEIDRRSLVTWGNYA
                               **************************************************
                              51        .         .         .         .        100

                             101        .         .         .         .        150
[Truncated_Name:1]ifit5_1435   WIYYHMGRYEEAQVYINKVENSCKKLSNTAHLKIQLPEIYAEQGFALLKF
[Truncated_Name:2]ifit5_1435   WIYYHMGRYEEAQVYINKVENSCKKLSNTAHLKIQLPEIYAEQGFALLKF
[Truncated_Name:3]ifit5_1435   WIYYHMGRYEEAQVYINKVENSCKKLSNTAHLKIQLPEIYAEQGFALLKF
[Truncated_Name:4]ifit5_1435   WIYYHMGRYEEAQVYINKVENSCKKLSNTAHLKIQLPEIYAEQGFALLKF
[Truncated_Name:5]ifit5_1435   WIYYHMGRYEEAQVYINKVENSCKKLSNTAHLKIQLPEIYAEQGFALLKF
                               **************************************************
                             101        .         .         .         .        150

                             151        .         .         .         .        200
[Truncated_Name:1]ifit5_1435   GGKYYNRAKECFKNALREEPNNPEFNAGYAIAVYRLEEFSYRRCEEVDSS
[Truncated_Name:2]ifit5_1435   GGKYYNRAKECFKNALREEPNNPEFNAGYAIAVYRLEEFSYRRCEEVDSS
[Truncated_Name:3]ifit5_1435   GGKYYNRAKECFKNALREEPNNPEFNAGYAIAVYRLEEFSYRRCEEVDSS
[Truncated_Name:4]ifit5_1435   GGKYYNRAKECFKNALREEPNNPEFNAGYAIAVYRLEEFSYRRCEEVDSS
[Truncated_Name:5]ifit5_1435   GGKYYNRAKECFKNALREEPNNPEFNAGYAIAVYRLEEFSYRRCEEVDSS
                               **************************************************
                             151        .         .         .         .        200

                             201        .         .         .         .        250
[Truncated_Name:1]ifit5_1435   LEPLKRALKLNPMDTYLLALLALKLQDSDQVDEAEKCIEEGMKKTPYLPY
[Truncated_Name:2]ifit5_1435   LEPLKRALKLNPMDTYLLALLALKLQDSDQVDEAEKCIEEGMKKTPYLPY
[Truncated_Name:3]ifit5_1435   LEPLKRALKLNPMDTYLLALLALKLQDSDQVDEAEKCIEEGMKKTPYLPY
[Truncated_Name:4]ifit5_1435   LEPLKRALKLNPMDTYLLALLALKLQDSDQVDEAEKCIEEGMKKTPYLPY
[Truncated_Name:5]ifit5_1435   LEPLKRALKLNPMDTYLLALLALKLQDSDQVDEAEKCIEEGMKKTPYLPY
                               **************************************************
                             201        .         .         .         .        250

                             251        .         .         .         .        300
[Truncated_Name:1]ifit5_1435   FLRYAAKFYRRKKELDKAQEVLERALEISPKSTFLLHQLGLCYRAKLYEL
[Truncated_Name:2]ifit5_1435   FLRYAAKFYRRKKELDKAQEVLERALEISPKSTFLLHQLGLCYRAKLYEL
[Truncated_Name:3]ifit5_1435   FLRYAAKFYRRKKELDKAQEVLERALEISPKSTFLLHQLGLCYRAKLYEL
[Truncated_Name:4]ifit5_1435   FLRYAAKFYRRKKELDKAQEVLERALEISPKSTFLLHQLGLCYRAKLYEL
[Truncated_Name:5]ifit5_1435   FLRYAAKFYRRKKELDKAQEVLERALEISPKSTFLLHQLGLCYRAKLYEL
                               **************************************************
                             251        .         .         .         .        300

                             301        .         .         .         .        350
[Truncated_Name:1]ifit5_1435   KNSTRYPPQDQIEELIQICISHFKVVTEQKPKFFSALIDLARMYAEANMY
```

```
[Truncated_Name:2]ifit5_1435   KNSTRYPPQDQIEELIQICISHFKVVTEQKPKFFSALIDLARMYAEANMY
[Truncated_Name:3]ifit5_1435   KNSTRYPPQDQIEELIQICISHFKVVTEQKPKFFSALIDLARMYAEANMY
[Truncated_Name:4]ifit5_1435   KNSTRYPPQDQIEELIQICISHFKVVTEQKPKFFSALIDLARMYAEANMY
[Truncated_Name:5]ifit5_1435   KNSTRYPPQDQIEELIQICISHFKVVTEQKPKFFSALIDLARMYAEANMY
                               **************************************************
                             301        .         .         .         .         350

                             351        .         .         .         .         400
[Truncated_Name:1]ifit5_1435   RKAEETFQKALNVNILTCSNKQEICYFYGNFLQYKKKSESEAIKYYKEGL
[Truncated_Name:2]ifit5_1435   RKAEETFQKALNVNILTCSNKQEICYFYGNFLQYKKKSESEAIKYYKEGL
[Truncated_Name:3]ifit5_1435   RKAEETFQKALNVNILTCSNKQEICYFYGNFLQYKKKSESEAIKYYKEGL
[Truncated_Name:4]ifit5_1435   RKAEETFQKALNVNILTCSNKQEICYFYGNFLQYKKKSESEAIKYYKEGL
[Truncated_Name:5]ifit5_1435   RKAEETFQKALNVNILTCSNKQEICYFYGNFLQYKKKSESEAIKYYKEGL
                               **************************************************
                             351        .         .         .         .         400

                             401        .         .         .         .         450
[Truncated_Name:1]ifit5_1435   KNGNYCFAEKIRQYLKRLLEKRIQGGLGGEDDFSTLGLIHKLDGEKLEAI
[Truncated_Name:2]ifit5_1435   KNGNYCFAEKIRQYLKRLLEKRIQGGLGGEDDFSTLGLIHKLDGEKLEAI
[Truncated_Name:3]ifit5_1435   KNGNYCFAEKIRQYLKRLLEKRIQGGLGGEDDFSTLGLIHKLDGEKLEAI
[Truncated_Name:4]ifit5_1435   KNGNYCFAEKIRQYLKRLLEKRIQGGLGGEDDFSTLGLIHKLDGEKLEAI
[Truncated_Name:5]ifit5_1435   KNGNYCFAEKIRQYLKRLLEKRIQGGLGGEDDFSTLGLIHKLDGEKLEAI
                               **************************************************
                             401        .         .         .         .         450

                             451        .         .       479
[Truncated_Name:1]ifit5_1435   ECYEKANEYNPDNEEYLSVLLELRLSLSS
[Truncated_Name:2]ifit5_1435   ECYEKANEYNPDNEEYLSVLLELRLSLSS
[Truncated_Name:3]ifit5_1435   ECYEKANEYNPDNEEYLSVLLELRLSLSS
[Truncated_Name:4]ifit5_1435   ECYEKANEYNPDNEEYLSVLLELRLSLSS
[Truncated_Name:5]ifit5_1435   ECYEKANEYNPDNEEYLSVLLELRLSLSS
                               ****************************
                             451        .         .       479

Call:
  pdbaln(files = pdb_files, fit = TRUE, exefile = "msa")

Class:
  pdbs, fasta

Alignment dimensions:
  5 sequence rows; 479 position columns (479 non-gap, 0 gap)
```

```
+ attr: xyz, resno, b, chain, id, ali, resid, sse, call
```

```
pca <- pca(pdbs2)
plot(pca)
```




RMSD is a standard measure of structural distance between coordinate sets. We can use `rmsd()` function to calculate the RMSD between all pairs and models.

```
rd <- rmsd(pdbs2, fit=TRUE)
```

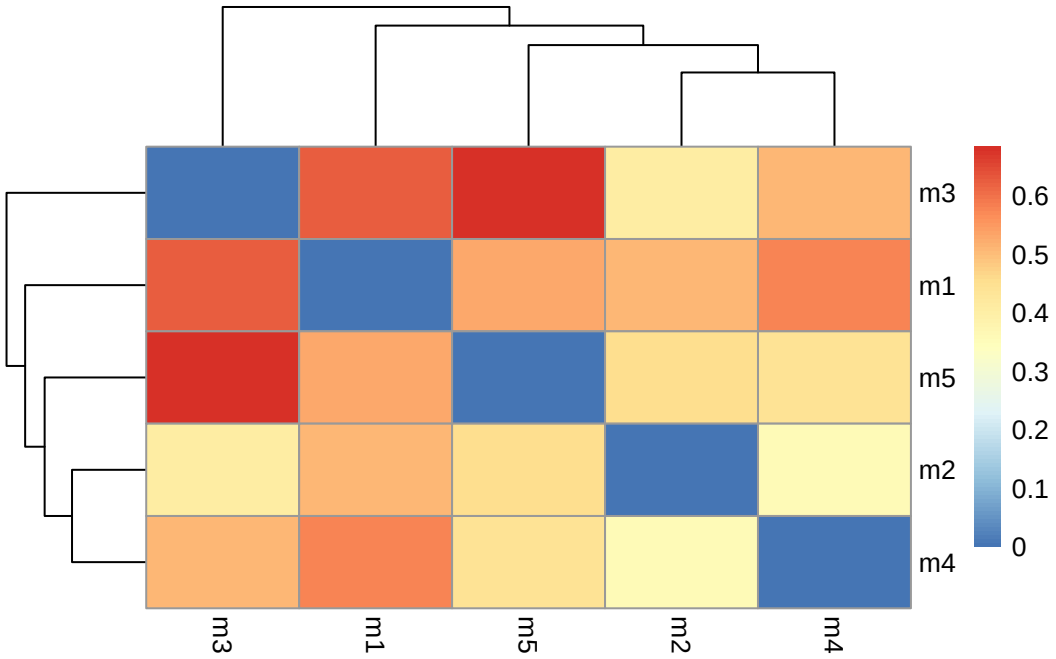
```
Warning in rmsd(pdbs2, fit = TRUE): No indices provided, using the 479 non NA positions
```

```
range(rd)
```

```
[1] 0.000 0.684
```

Draw a heatmap of these RMSD matrix values

```
library(pheatmap)

colnames(rd) <- paste0("m",1:5)
rownames(rd) <- paste0("m",1:5)
pheatmap(rd)
```

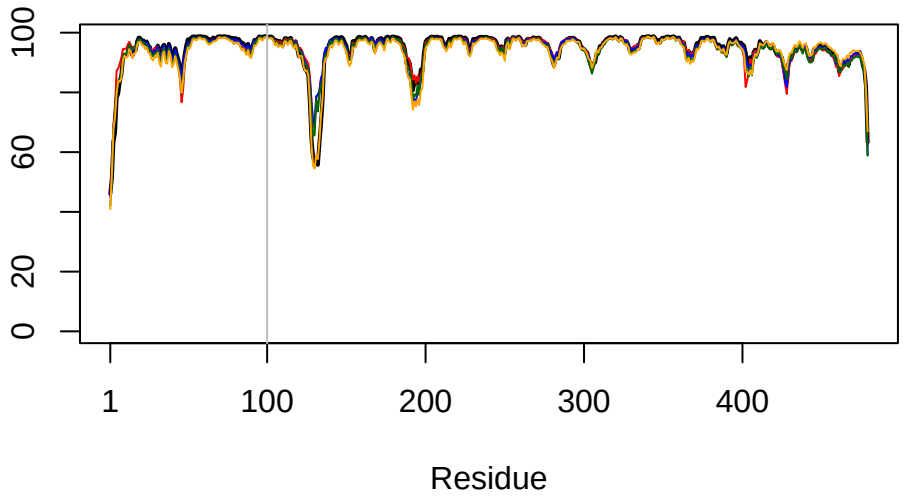



Let's plot the pLDDT values across all models. This information is stored in the B-factor column of each model and that this is stored in our aligned `pdbs` object as `pdbs$b` with a row per structure/model

```
plotb3(pdbs2$b[1,], typ="l", lwd=2, sse=pdb)
```

```
Warning in plotb3(pdbs2$b[1, ], typ = "l", lwd = 2, sse = pdb): Length of input
'sse' does not equal the length of input 'x'; Ignoring 'sse'
```

```
points(pdbs2$b[2,], typ="l", col="red")
points(pdbs2$b[3,], typ="l", col="blue")
points(pdbs2$b[4,], typ="l", col="darkgreen")
points(pdbs2$b[5,], typ="l", col="orange")
abline(v=100, col="gray")
```




```
core <- core.find(pdbs2)
```

```
 core size 478 of 479  vol = 6.063
 core size 477 of 479  vol = 5.75
 core size 476 of 479  vol = 5.467
 core size 475 of 479  vol = 5.286
 core size 474 of 479  vol = 5.1
 core size 473 of 479  vol = 4.95
 core size 472 of 479  vol = 4.801
 core size 471 of 479  vol = 4.654
 core size 470 of 479  vol = 4.505
 core size 469 of 479  vol = 4.361
 core size 468 of 479  vol = 4.223
 core size 467 of 479  vol = 4.078
 core size 466 of 479  vol = 3.954
 core size 465 of 479  vol = 3.841
 core size 464 of 479  vol = 3.709
 core size 463 of 479  vol = 3.588
 core size 462 of 479  vol = 3.467
 core size 461 of 479  vol = 3.352
 core size 460 of 479  vol = 3.244
 core size 459 of 479  vol = 3.126
```

```
core size 458 of 479  vol = 3.019
core size 457 of 479  vol = 2.91
core size 456 of 479  vol = 2.806
core size 455 of 479  vol = 2.718
core size 454 of 479  vol = 2.629
core size 453 of 479  vol = 2.543
core size 452 of 479  vol = 2.462
core size 451 of 479  vol = 2.393
core size 450 of 479  vol = 2.332
core size 449 of 479  vol = 2.282
core size 448 of 479  vol = 2.229
core size 447 of 479  vol = 2.181
core size 446 of 479  vol = 2.14
core size 445 of 479  vol = 2.099
core size 444 of 479  vol = 2.072
core size 443 of 479  vol = 2.029
core size 442 of 479  vol = 1.985
core size 441 of 479  vol = 1.944
core size 440 of 479  vol = 1.913
core size 439 of 479  vol = 1.887
core size 438 of 479  vol = 1.856
core size 437 of 479  vol = 1.824
core size 436 of 479  vol = 1.79
core size 435 of 479  vol = 1.759
core size 434 of 479  vol = 1.725
core size 433 of 479  vol = 1.689
core size 432 of 479  vol = 1.659
core size 431 of 479  vol = 1.63
core size 430 of 479  vol = 1.607
core size 429 of 479  vol = 1.579
core size 428 of 479  vol = 1.548
core size 427 of 479  vol = 1.523
core size 426 of 479  vol = 1.494
core size 425 of 479  vol = 1.472
core size 424 of 479  vol = 1.45
core size 423 of 479  vol = 1.425
core size 422 of 479  vol = 1.4
core size 421 of 479  vol = 1.377
core size 420 of 479  vol = 1.357
core size 419 of 479  vol = 1.334
core size 418 of 479  vol = 1.313
core size 417 of 479  vol = 1.295
core size 416 of 479  vol = 1.271
```

```
core size 415 of 479  vol = 1.247
core size 414 of 479  vol = 1.228
core size 413 of 479  vol = 1.207
core size 412 of 479  vol = 1.191
core size 411 of 479  vol = 1.168
core size 410 of 479  vol = 1.148
core size 409 of 479  vol = 1.133
core size 408 of 479  vol = 1.114
core size 407 of 479  vol = 1.092
core size 406 of 479  vol = 1.076
core size 405 of 479  vol = 1.06
core size 404 of 479  vol = 1.041
core size 403 of 479  vol = 1.032
core size 402 of 479  vol = 1.018
core size 401 of 479  vol = 1
core size 400 of 479  vol = 0.982
core size 399 of 479  vol = 0.964
core size 398 of 479  vol = 0.952
core size 397 of 479  vol = 0.934
core size 396 of 479  vol = 0.925
core size 395 of 479  vol = 0.908
core size 394 of 479  vol = 0.893
core size 393 of 479  vol = 0.876
core size 392 of 479  vol = 0.86
core size 391 of 479  vol = 0.845
core size 390 of 479  vol = 0.831
core size 389 of 479  vol = 0.82
core size 388 of 479  vol = 0.809
core size 387 of 479  vol = 0.799
core size 386 of 479  vol = 0.785
core size 385 of 479  vol = 0.772
core size 384 of 479  vol = 0.761
core size 383 of 479  vol = 0.749
core size 382 of 479  vol = 0.737
core size 381 of 479  vol = 0.727
core size 380 of 479  vol = 0.717
core size 379 of 479  vol = 0.708
core size 378 of 479  vol = 0.697
core size 377 of 479  vol = 0.687
core size 376 of 479  vol = 0.68
core size 375 of 479  vol = 0.673
core size 374 of 479  vol = 0.666
core size 373 of 479  vol = 0.66
```

```
 core size 372 of 479  vol = 0.652
 core size 371 of 479  vol = 0.643
 core size 370 of 479  vol = 0.636
 core size 369 of 479  vol = 0.627
 core size 368 of 479  vol = 0.62
 core size 367 of 479  vol = 0.615
 core size 366 of 479  vol = 0.607
 core size 365 of 479  vol = 0.601
 core size 364 of 479  vol = 0.595
 core size 363 of 479  vol = 0.589
 core size 362 of 479  vol = 0.583
 core size 361 of 479  vol = 0.576
 core size 360 of 479  vol = 0.571
 core size 359 of 479  vol = 0.566
 core size 358 of 479  vol = 0.56
 core size 357 of 479  vol = 0.554
 core size 356 of 479  vol = 0.548
 core size 355 of 479  vol = 0.544
 core size 354 of 479  vol = 0.539
 core size 353 of 479  vol = 0.534
 core size 352 of 479  vol = 0.527
 core size 351 of 479  vol = 0.521
 core size 350 of 479  vol = 0.515
 core size 349 of 479  vol = 0.51
 core size 348 of 479  vol = 0.505
 core size 347 of 479  vol = 0.5
 core size 346 of 479  vol = 0.495
 FINISHED: Min vol ( 0.5 ) reached
```

```
core.inds <- print(core, vol=0.5)
```

```
# 347 positions (cumulative volume <= 0.5 Angstrom^3)
   start end length
1     10  10      1
2     12  44     33
3     47 123     77
4    125 127      3
5    136 194     59
6    197 281     85
7    285 285      1
8    287 295      9
9    314 323     10
```

```
10   325 328      4
11   332 385     54
12   392 392      1
13   395 400      6
14   411 414      4
```

```
xyz <- pdbfit(pdbs2, core.inds, outpath="corefit_structures")
```

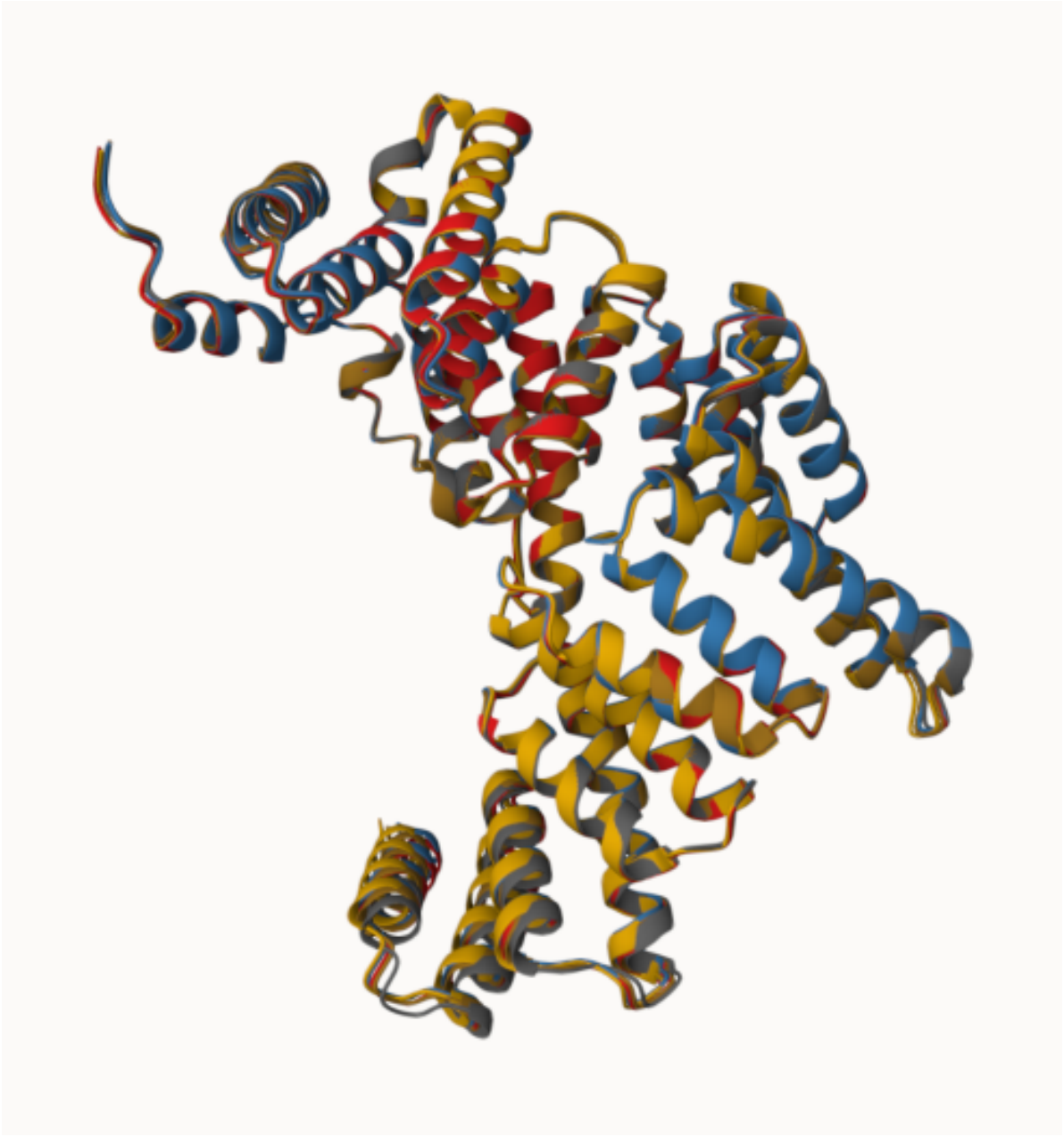


Figure 5: Core Superimposed structures colored by B-factor

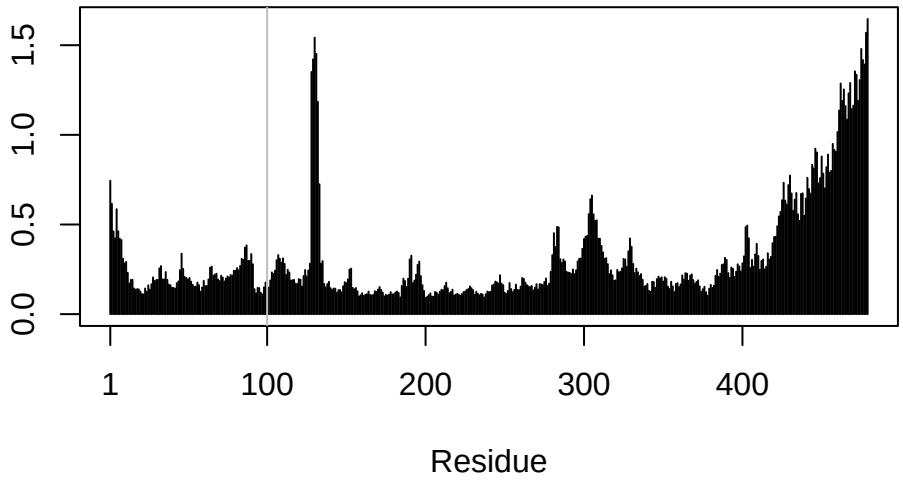
Now we can examine the RMSF between positions of the structure.

```
rf <- rmsf(xyz)
```

```
plotb3(rf, sse=pdb)
```

```
Warning in plotb3(rf, sse = pdb): Length of input 'sse' does not equal the
length of input 'x'; Ignoring 'sse'
```

```
abline(v=100, col="gray", ylab="RMSF")
```




Predicted Alignment Error for Domains

AlphaFold produces an output called PAE. We read these files that AlphaFold produces a useful inter-domain prediction for model 1 and 2

```
library(jsonlite)

# Listing of all PAE JSON files
pae_files <- list.files(path=results_dir,
                        pattern=".*model.*\\.json",
                        full.names = TRUE)
```

```
pae1 <- read_json(pae_files[1],simplifyVector = TRUE)
pae5 <- read_json(pae_files[5],simplifyVector = TRUE)

attributes(pae1)
```

```
$names
[1] "plddt"   "max_pae" "pae"     "ptm"
```

```
# Per-residue pLDDT scores
#  same as B-factor of PDB..
head(pae1$plddt)
```

```
[1] 46.03 51.16 63.62 66.38 78.19 79.62
```

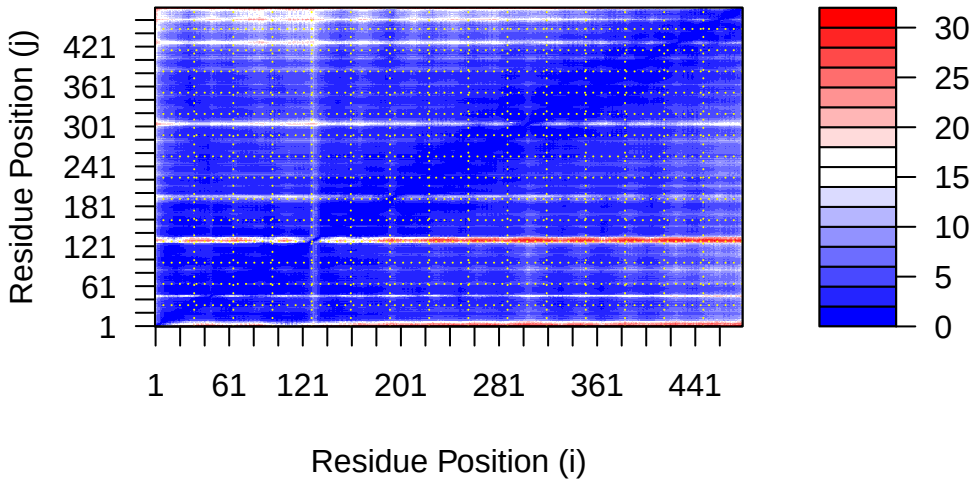
```
pae1$max_pae
```

```
[1] 30.98438
```

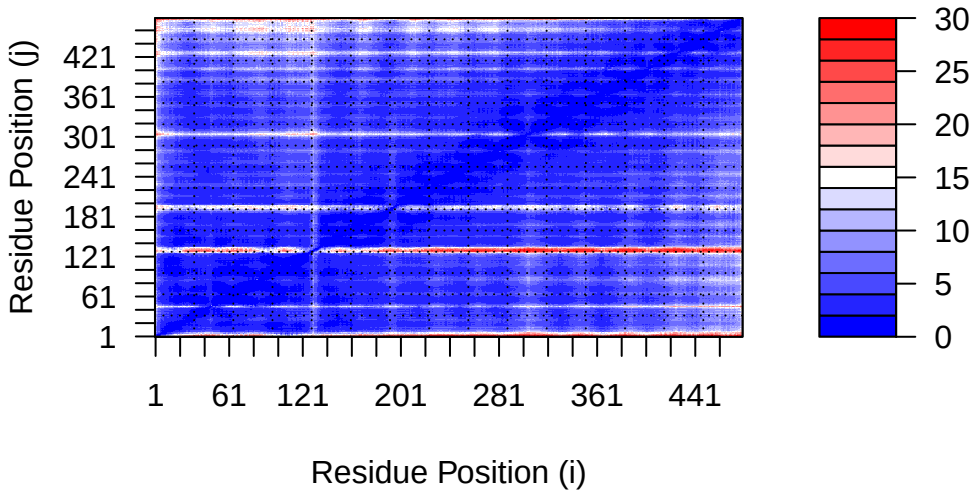
```
pae5$max_pae
```

```
[1] 31.1875
```

```
plot.dmat(pae1$pae,
          xlab="Residue Position (i)",
          ylab="Residue Position (j)")
```

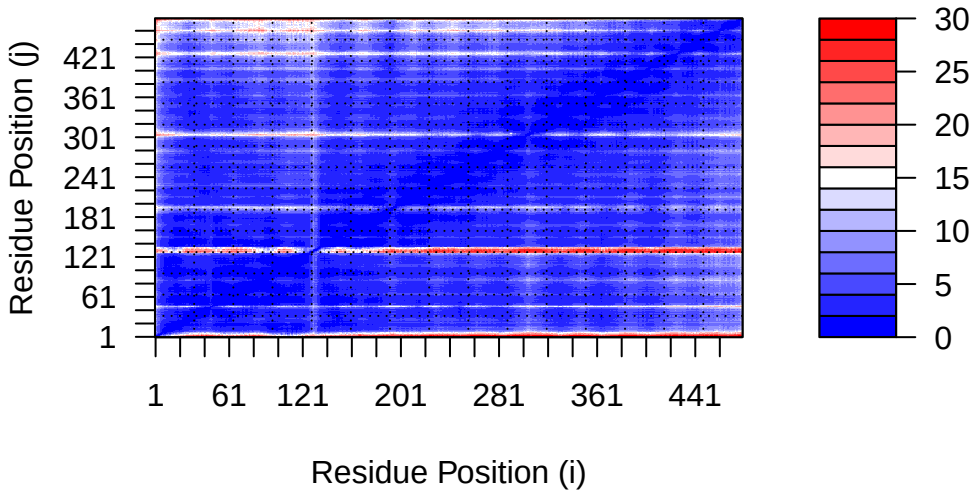


```
plot.dmat(pae5$pae,
          xlab="Residue Position (i)",
          ylab="Residue Position (j)",
          grid.col = "black",
          zlim=c(0,30))
```




Here is a plot using the same z range. Here is model 1 plot using the same data range for model 5

```
plot.dmat(pae1$pae,
          xlab="Residue Position (i)",
          ylab="Residue Position (j)",
          grid.col = "black",
          zlim=c(0,30))
```




Residue conservation from alignment file

```
aln_file <- list.files(path=results_dir,
                       pattern=".a3m$",
                        full.names = TRUE)
aln_file
```

```
[1] "ifit5_14356/ifit5_14356.a3m"
```

```
aln <- read.fasta(aln_file[1], to.upper = TRUE)
```

```
[1] " ** Duplicated sequence id's: 101 **"
```

```
dim(aln$ali)
```

```
[1] 1947  560
```

I am having an issue finding the PDB file for my novel protein. It does not pull up the same species.