# Class11:AlphaFold

Hugh (Trevor) Redford (PID:A17067426)

2025-02-11

## Custon Analysis of Resulting Models

```r
results_dir <- "hivprdimer_23119"

# File names for all PDB models
pdb_files <- list.files(path=results_dir,
pattern="*.pdb",
full.names = TRUE)

# Print our PDB file names
basename(pdb_files)
```

```
[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"
```

```r
 library(bio3d)
```

```r
# Read all data from Models
# and superpose/fit coords
pdbs <- pdbaln(pdb_files, 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
```

A quick view of model sequences - this should be a boring alignment in the sense that all sequences are the same.

```
pdbs
```

```
                               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_files, 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
```

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

```r
rd <- rmsd(pdbs, fit=T)
```

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

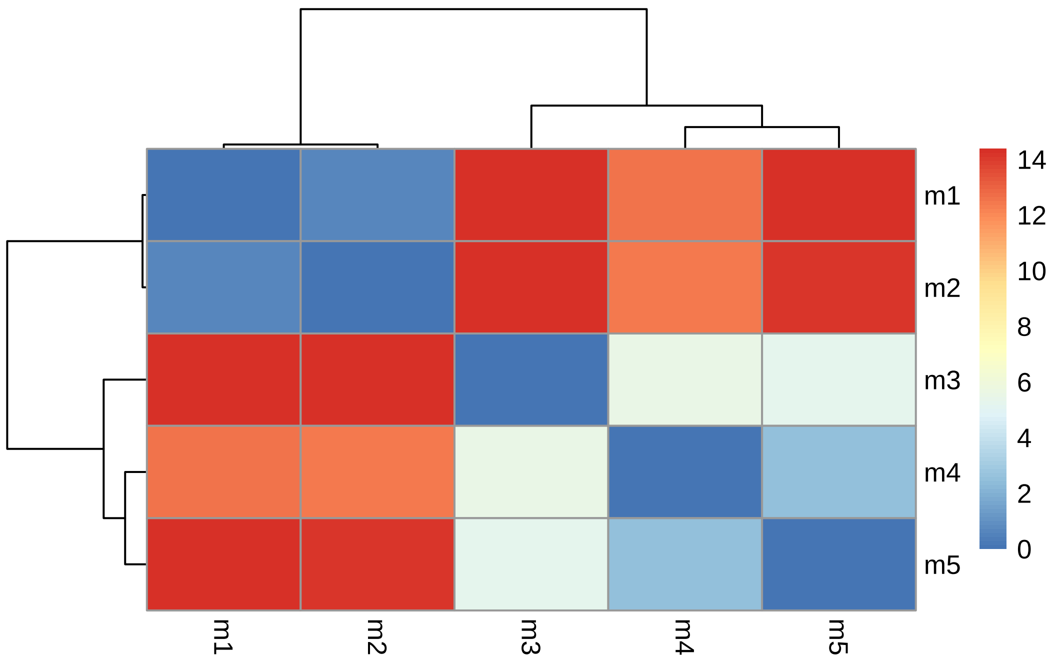
```r
range(rd)
```

```
[1]  0.000 14.376
```

Draw a heatmap of these RMSD matrix values

```r
library(pheatmap)
```

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



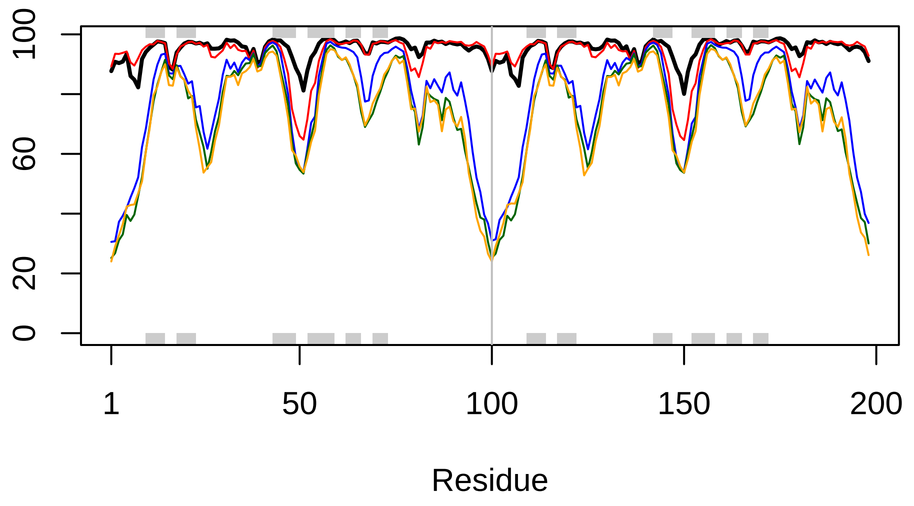
Now lets plot the pLDDT values across all models. Recall that this information is in the B-factor column of each model and that this is stored in our aligned pdbs object as pdbs$b with a row per structure/model.

```r
# Read a reference PDB structure
pdb <- read.pdb("1hsg")
```

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

You could optionally obtain secondary structure from a call to stride() or dssp() on any of the model structures.

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



We can improve the superposition/fitting of our models by finding the most consistent “rigid core” common across all the models. For this we will use the core.find() function:

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

```
 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
```

We can now use the identified core atom positions as a basis for a more suitable superposition and write out the fitted structures to a directory called corefit_structures:

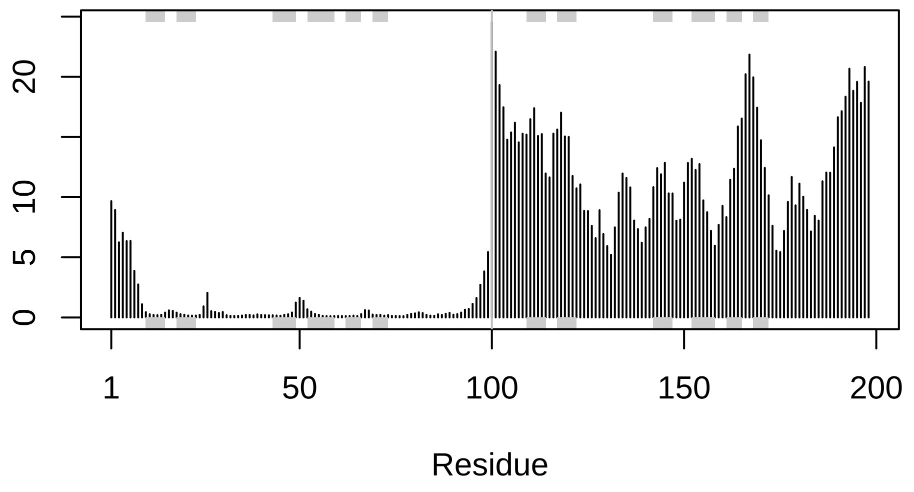
```r
core.inds <- print(core, vol=0.5)
```

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

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

Now we can examine the RMSF between positions of the structure. RMSF is an often used measure of conformational variance along the structure:

```r
rf <- rmsf(xyz)
plotb3(rf, sse=pdb)
abline(v=100, col="gray", ylab="RMSF")
```



Independent of the 3D structure, AlphaFold produces an output called Predicted Aligned Error (PAE). This is detailed in the JSON format result files, one for each model structure.

```
library(jsonlite)
```

```
# Load jsonlite package
library(jsonlite)

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

# Read the first and fifth JSON files correctly
pae1 <- fromJSON(pae_files[1], simplifyVector = TRUE)
pae5 <- fromJSON(pae_files[5], simplifyVector = TRUE)

# Print attributes to verify
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
```

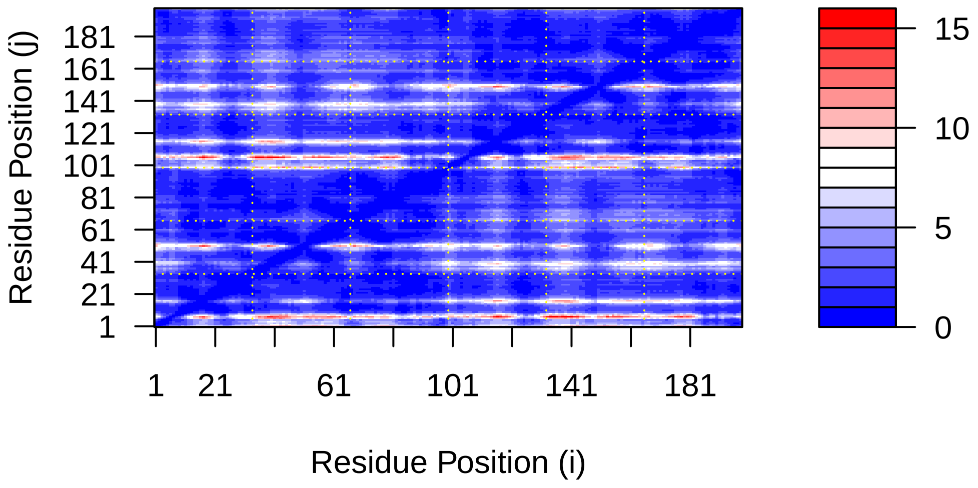
```
[1] 15.47656
```

```
pae5$max_pae
```

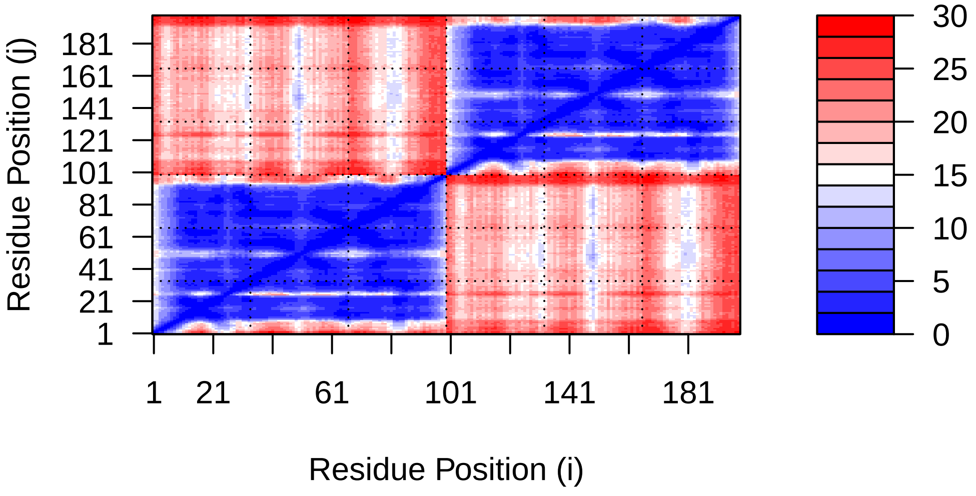
```
[1] 29.32812
```

We can plot the N by N (where N is the number of residues) PAE scores with ggplot or with functions from the Bio3D package:

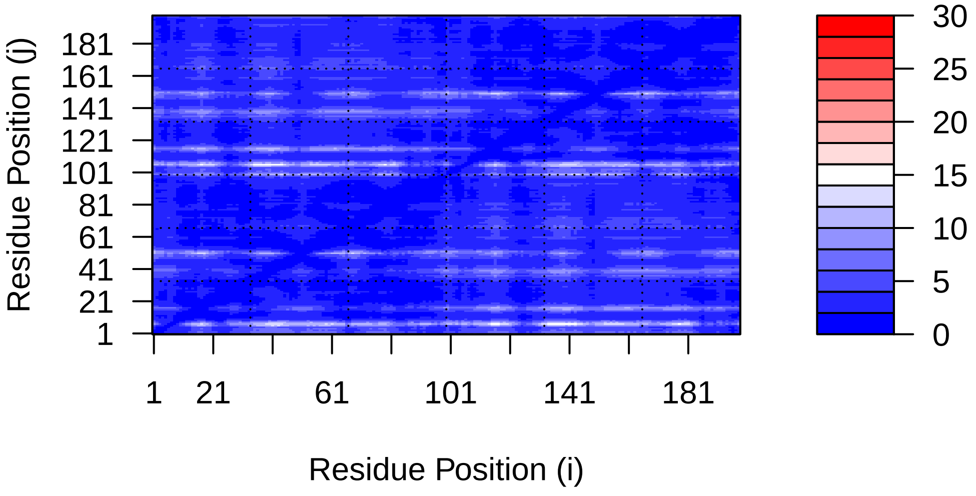
```
library(bio3d)

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))
```



```
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] "hivprdimer_23119/hivprdimer_23119.a3m"
```

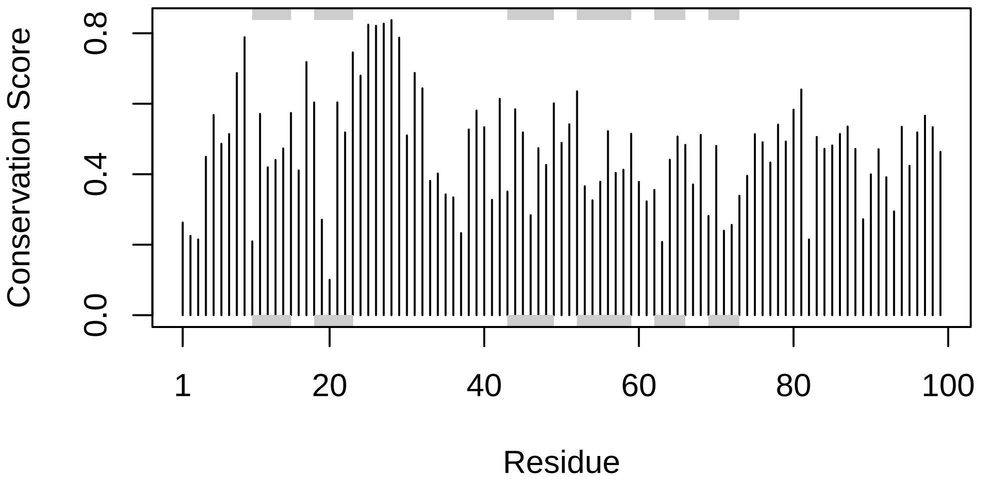
```r
aln <- read.fasta(aln_file[1], to.upper = TRUE)
```

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

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

```
[1] 5378  132
```

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



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

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

For a final visualization of these functionally important sites we can map this conservation score to the Occupancy column of a PDB file for viewing in molecular viewer programs such as Mol*, PyMol, VMD, chimera etc.

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