class 11: AlphaFold

Aigerim (PID: 09919142)

AlphaFold is a new bioinformatics method for structure prediction of sequence.

We can run AlphaFold in our computer by installing it or we can run in GoogleColab (without needing to install anything) via: https://github.com/sokrypton/ColabFold

```
pth <- "hiv1_dimer_23119/"
```

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


basename(pdb_files)
```

```
[1] "hiv1_dimer_23119_unrelaxed_rank_001_alphafold2_multimer_v3_model_5_seed_000.pdb"
[2] "hiv1_dimer_23119_unrelaxed_rank_002_alphafold2_multimer_v3_model_1_seed_000.pdb"
[3] "hiv1_dimer_23119_unrelaxed_rank_003_alphafold2_multimer_v3_model_4_seed_000.pdb"
[4] "hiv1_dimer_23119_unrelaxed_rank_004_alphafold2_multimer_v3_model_2_seed_000.pdb"
[5] "hiv1_dimer_23119_unrelaxed_rank_005_alphafold2_multimer_v3_model_3_seed_000.pdb"
```

```
library(bio3d)
```

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

```
Reading PDB files:
hiv1_dimer_23119/hiv1_dimer_23119_unrelaxed_rank_001_alphafold2_multimer_v3_model_5_seed_000
hiv1_dimer_23119/hiv1_dimer_23119_unrelaxed_rank_002_alphafold2_multimer_v3_model_1_seed_000
hiv1_dimer_23119/hiv1_dimer_23119_unrelaxed_rank_003_alphafold2_multimer_v3_model_4_seed_000
hiv1_dimer_23119/hiv1_dimer_23119_unrelaxed_rank_004_alphafold2_multimer_v3_model_2_seed_000
```

```
hiv1_dimer_23119/hiv1_dimer_23119_unrelaxed_rank_005_alphafold2_multimer_v3_model_3_seed_000
.....

Extracting sequences

pdb/seq: 1   name: hiv1_dimer_23119/hiv1_dimer_23119_unrelaxed_rank_001_alphafold2_multimer_v
pdb/seq: 2   name: hiv1_dimer_23119/hiv1_dimer_23119_unrelaxed_rank_002_alphafold2_multimer_v
pdb/seq: 3   name: hiv1_dimer_23119/hiv1_dimer_23119_unrelaxed_rank_003_alphafold2_multimer_v
pdb/seq: 4   name: hiv1_dimer_23119/hiv1_dimer_23119_unrelaxed_rank_004_alphafold2_multimer_v
pdb/seq: 5   name: hiv1_dimer_23119/hiv1_dimer_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]hiv1_dimer  PQITLWQRPLVTIKIGGQLKEALLDTGADDTVLEEMSLPGRWKPKMIGGI
[Truncated_Name:2]hiv1_dimer  PQITLWQRPLVTIKIGGQLKEALLDTGADDTVLEEMSLPGRWKPKMIGGI
[Truncated_Name:3]hiv1_dimer  PQITLWQRPLVTIKIGGQLKEALLDTGADDTVLEEMSLPGRWKPKMIGGI
[Truncated_Name:4]hiv1_dimer  PQITLWQRPLVTIKIGGQLKEALLDTGADDTVLEEMSLPGRWKPKMIGGI
[Truncated_Name:5]hiv1_dimer  PQITLWQRPLVTIKIGGQLKEALLDTGADDTVLEEMSLPGRWKPKMIGGI
                              **************************************************
                              1        .         .         .         .         50

                             51        .         .         .         .         100
[Truncated_Name:1]hiv1_dimer  GGFIKVRQYDQILIEICGHKAIGTVLVGPTPVNIIGRNLLTQIGCTLNFP
[Truncated_Name:2]hiv1_dimer  GGFIKVRQYDQILIEICGHKAIGTVLVGPTPVNIIGRNLLTQIGCTLNFP
[Truncated_Name:3]hiv1_dimer  GGFIKVRQYDQILIEICGHKAIGTVLVGPTPVNIIGRNLLTQIGCTLNFP
[Truncated_Name:4]hiv1_dimer  GGFIKVRQYDQILIEICGHKAIGTVLVGPTPVNIIGRNLLTQIGCTLNFP
[Truncated_Name:5]hiv1_dimer  GGFIKVRQYDQILIEICGHKAIGTVLVGPTPVNIIGRNLLTQIGCTLNFP
                              **************************************************
                             51        .         .         .         .         100

                            101        .         .         .         .         150
[Truncated_Name:1]hiv1_dimer  QITLWQRPLVTIKIGGQLKEALLDTGADDTVLEEMSLPGRWKPKMIGGIG
[Truncated_Name:2]hiv1_dimer  QITLWQRPLVTIKIGGQLKEALLDTGADDTVLEEMSLPGRWKPKMIGGIG
[Truncated_Name:3]hiv1_dimer  QITLWQRPLVTIKIGGQLKEALLDTGADDTVLEEMSLPGRWKPKMIGGIG
[Truncated_Name:4]hiv1_dimer  QITLWQRPLVTIKIGGQLKEALLDTGADDTVLEEMSLPGRWKPKMIGGIG
[Truncated_Name:5]hiv1_dimer  QITLWQRPLVTIKIGGQLKEALLDTGADDTVLEEMSLPGRWKPKMIGGIG
                              **************************************************
                            101        .         .         .         .         150
```

```
                            151        .         .         .         .       198
[Truncated_Name:1]hiv1_dimer   GFIKVRQYDQILIEICGHKAIGTVLVGPTPVNIIGRNLLTQIGCTLNF
[Truncated_Name:2]hiv1_dimer   GFIKVRQYDQILIEICGHKAIGTVLVGPTPVNIIGRNLLTQIGCTLNF
[Truncated_Name:3]hiv1_dimer   GFIKVRQYDQILIEICGHKAIGTVLVGPTPVNIIGRNLLTQIGCTLNF
[Truncated_Name:4]hiv1_dimer   GFIKVRQYDQILIEICGHKAIGTVLVGPTPVNIIGRNLLTQIGCTLNF
[Truncated_Name:5]hiv1_dimer   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.

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

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

```
range(rd)
```

```
[1]  0.000 14.507
```

Draw a heatmap of these RMSD matrix values

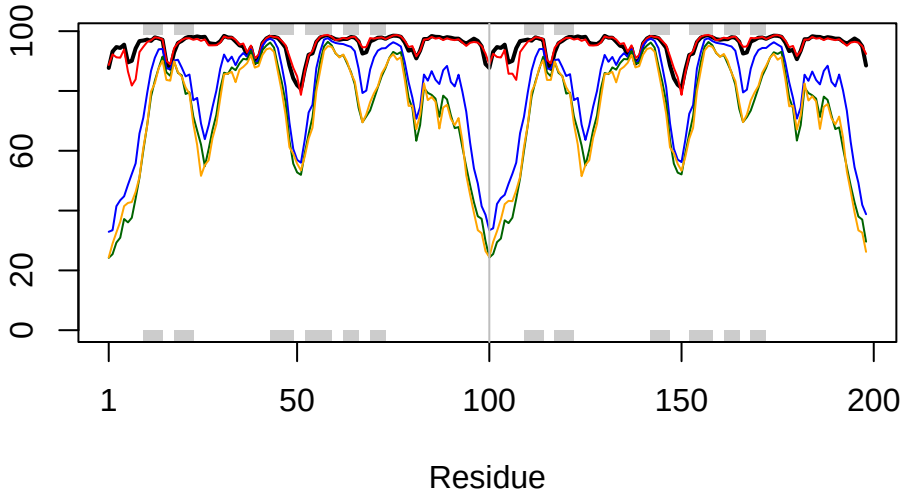
```
#library(pheatmap)

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

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.

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

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

```
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")
```



Predicted Alignment Error for domains

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.

Below we read these files and see that AlphaFold produces a useful inter-domain prediction for model 1 (and 2) but not for model 5 (or indeed models 3, 4, and 5):

```
library(jsonlite)


pae_files <- list.files(path=pth,
                        pattern=".*model.*\\.json",
                        full.names = TRUE)
```

For example purposes lets read the 1st and 5th files (you can read the others and make similar plots).

```
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"
```

```
head(pae1$plddt)
```

```
[1] 87.69 93.19 94.69 94.38 95.50 89.56
```

```
pae1$max_pae
```

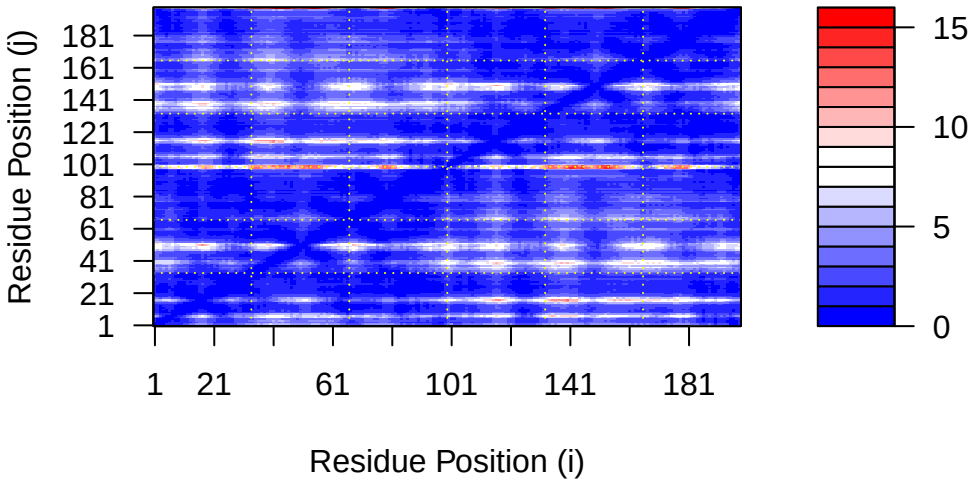
```
[1] 15.89844
```

```
pae5$max_pae
```

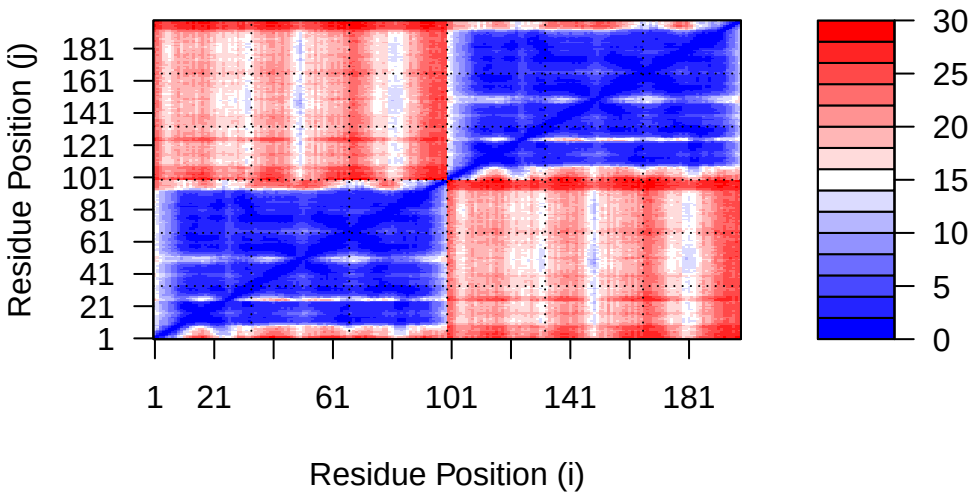
```
[1] 29.25
```

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:

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

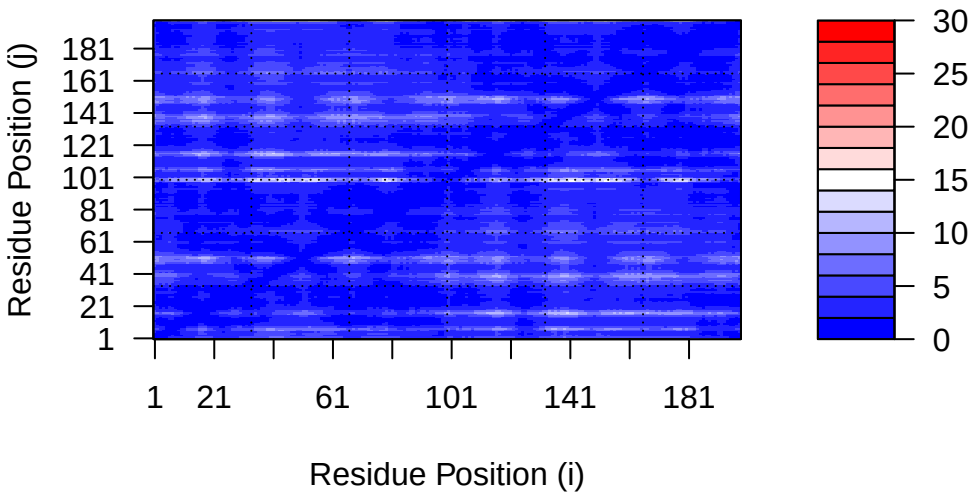



```
plot.dmat(pae5$pae,
          xlab="Residue Position (i)",
          ylab="Residue Position (j)",
          grid.col = "black",
          zlim=c(0,30))
```



We should really plot all of these using the same z range. Here is the model 1 plot again but this time 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=pth,
                       pattern=".a3m$",
                        full.names = TRUE)
aln_file
```

```
[1] "hiv1_dimer_23119/hiv1_dimer_23119.a3m"
```

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

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

How many sequences in this alignment

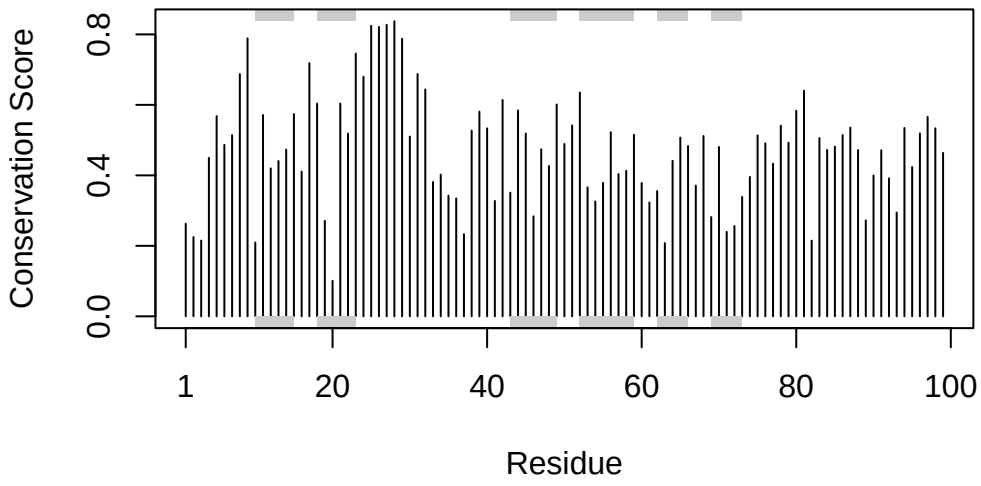
```
dim(aln$ali)
```

```
[1] 5378  132
```

We can score residue conservation in the alignment with the conserv() function.

```
sim <- conserv(aln)
```

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



Note the conserved Active Site residues D25, T26, G27, A28. These positions will stand out if we generate a consensus sequence with a high cutoff value:

```
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.

```
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")
```