Analysis of TCR:pMHC complexes having the same antigen but distinct CDR3beta

Mikhail Ignatov, Mikhail Shugay

July 30, 2016

This analysis shows that while TCRs recognizing the same antigen can have highly dissimilar CDR3 sequences, their spatial patterns of CDR3:antigen interactions (namely, maps of pairwise amino acid distances) observed in structural data are extremely similar.

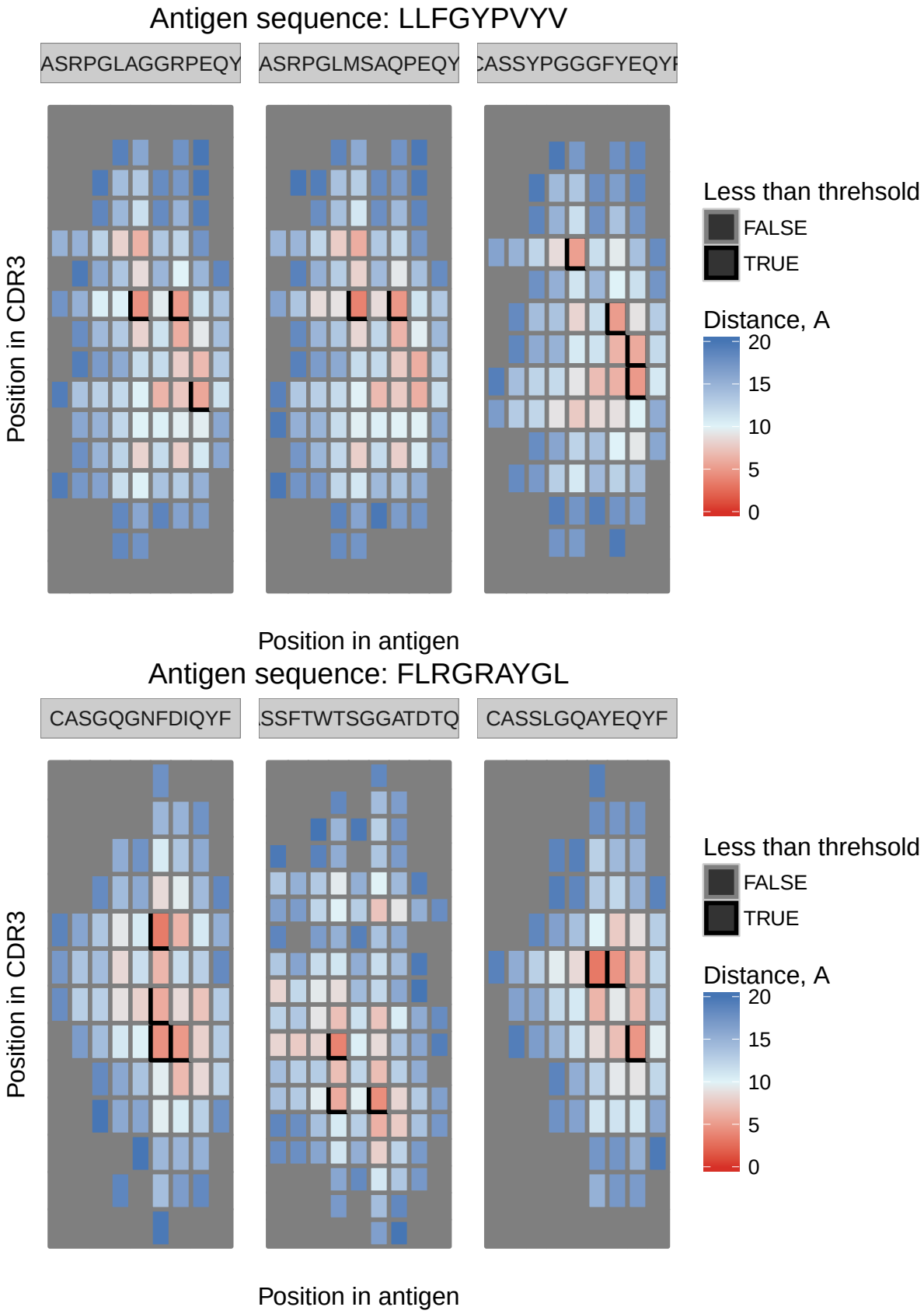
Start with loading our master table.

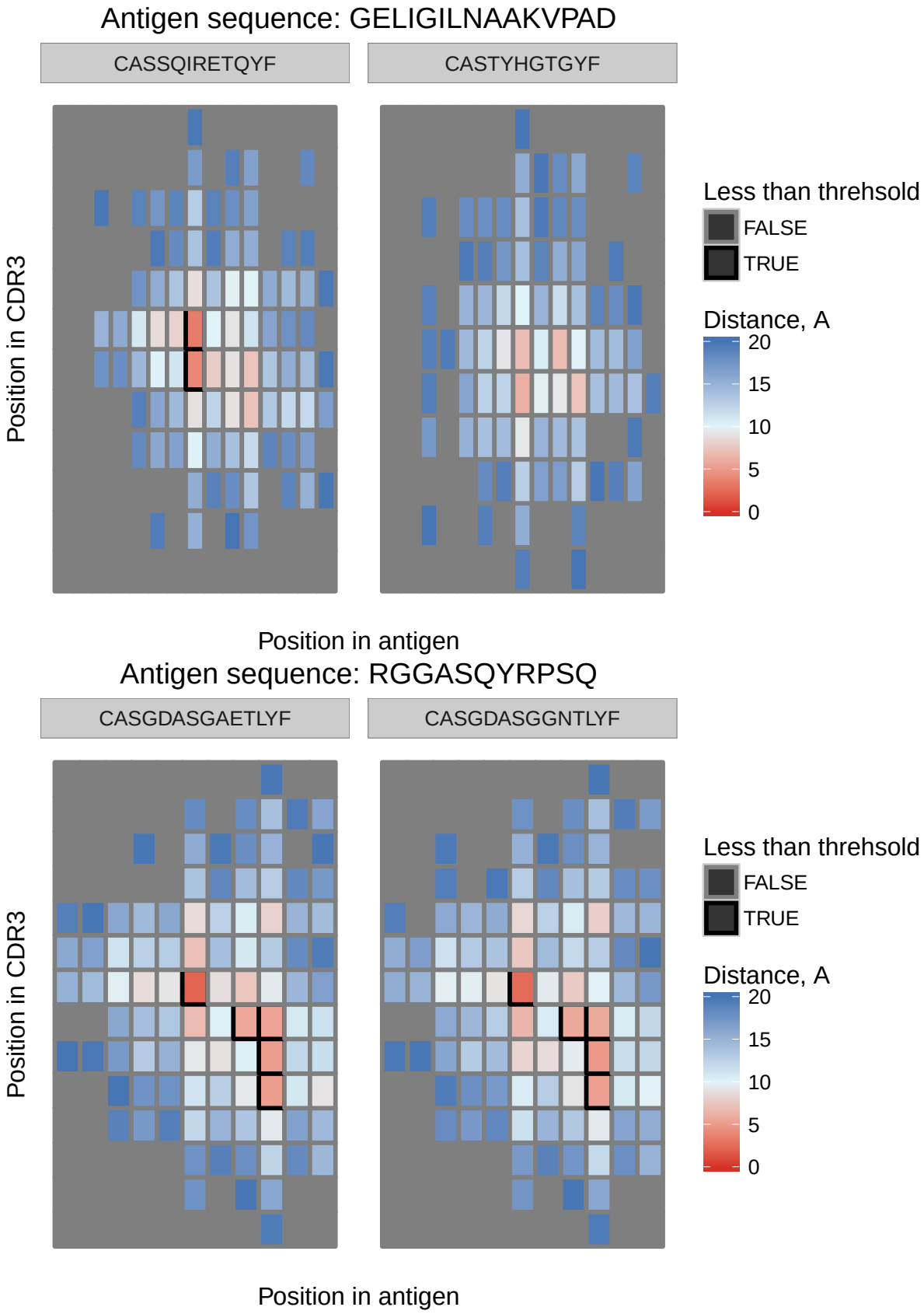
```
df <- read.table("../result/structure.txt", header=T, sep="\t")
```

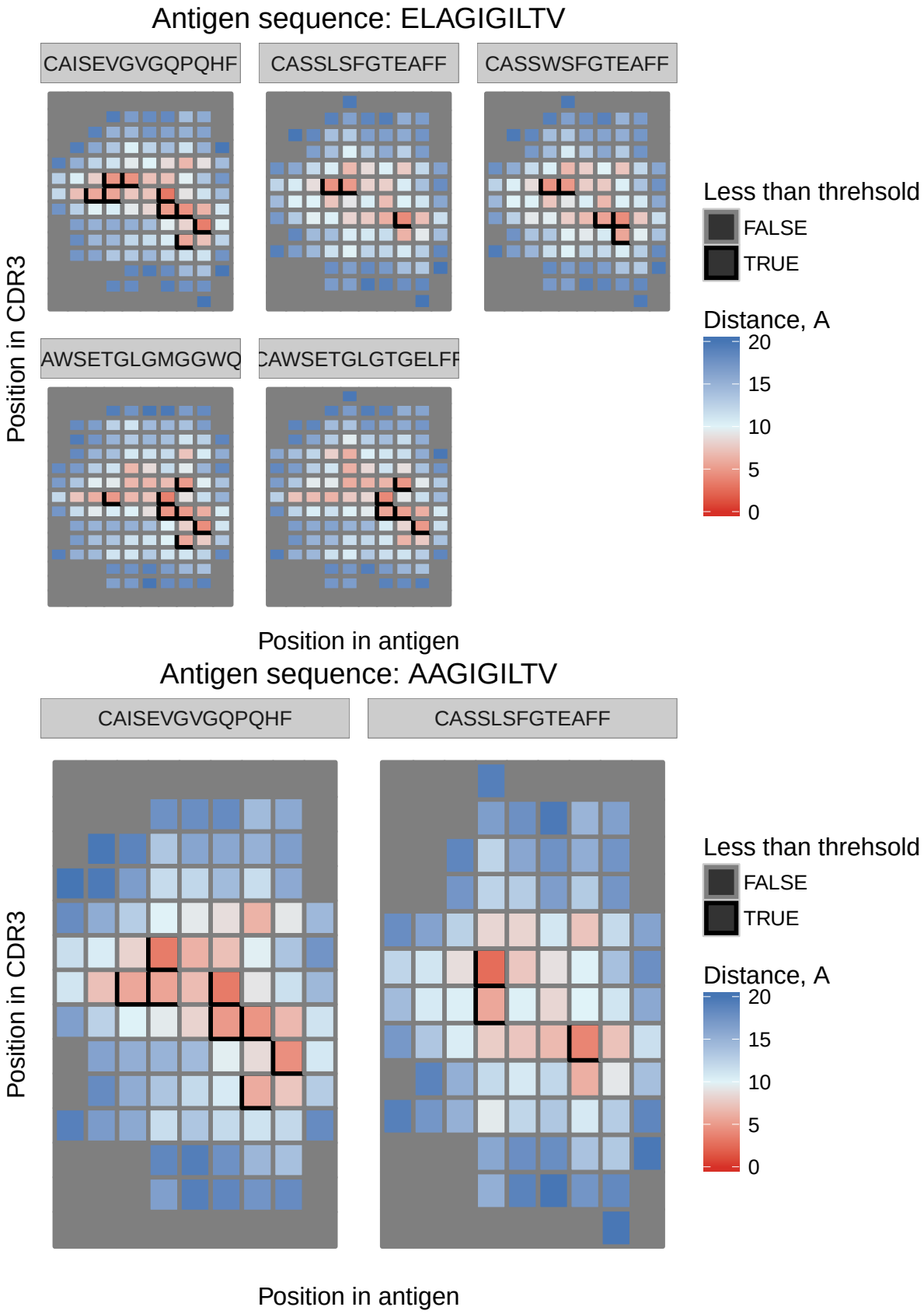
Select complexes for further analysis. We select complexes that have at least 2 distinct CDR3 beta with the same antigen and no duplicate CDR3 beta.

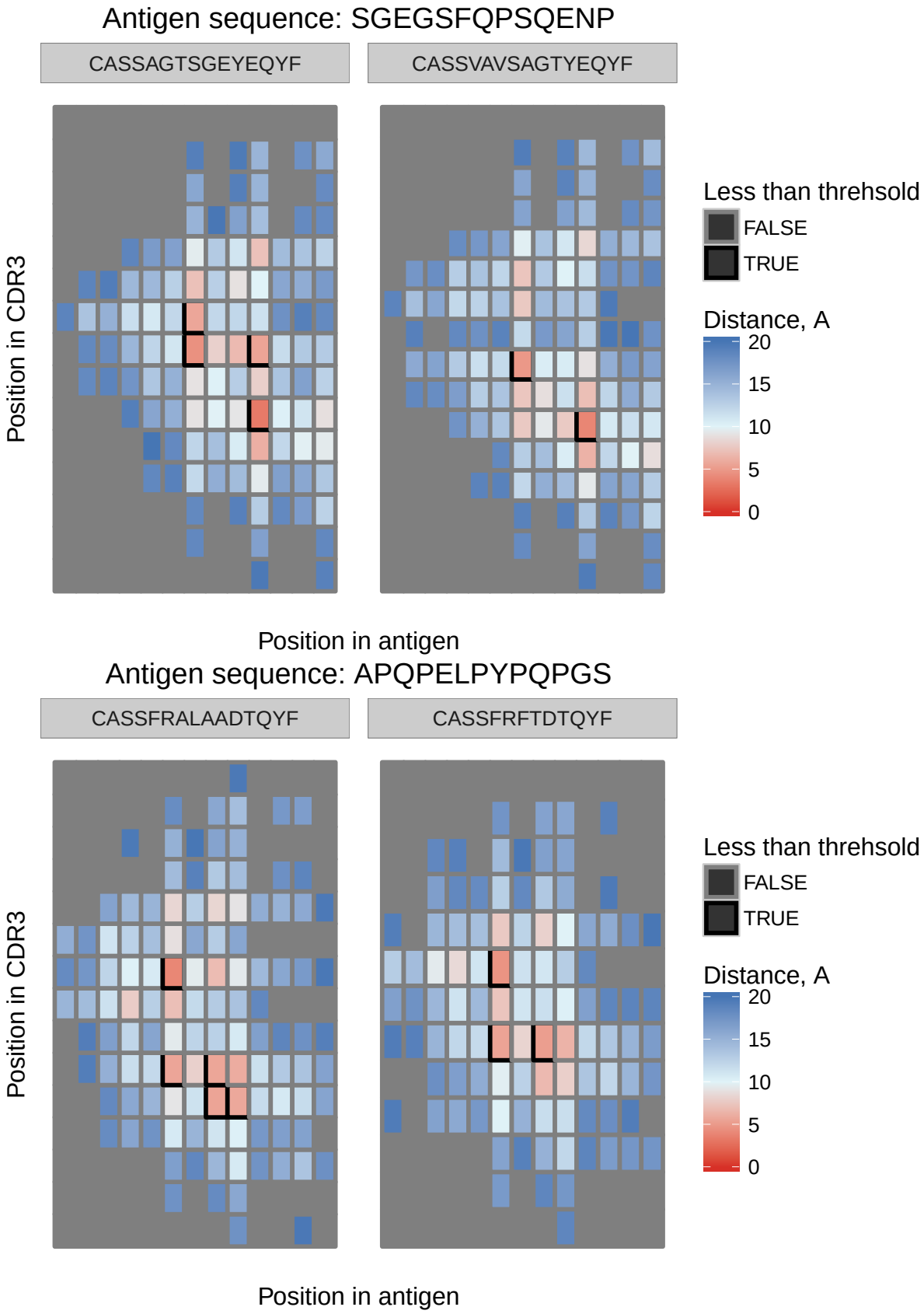
```
df <- subset(df, grepl("TRBV", tcr_v_allele) & tcr_region == "CDR3" &
               !(pdb_id %in% c("5d2l", "5d2n"))) # remove two complexes that do not have CDR3 contacts

library(plyr)

df.s <- ddply(df, .(antigen_seq), summarize,
              ncdr = length(unique(tcr_region_seq)), ncompl = length(unique(pdb_id)))

df.ss <- subset(df.s, ncdr > 1 & ncdr == ncompl)

df.m <- subset(df, antigen_seq %in% df.ss$antigen_seq)
```

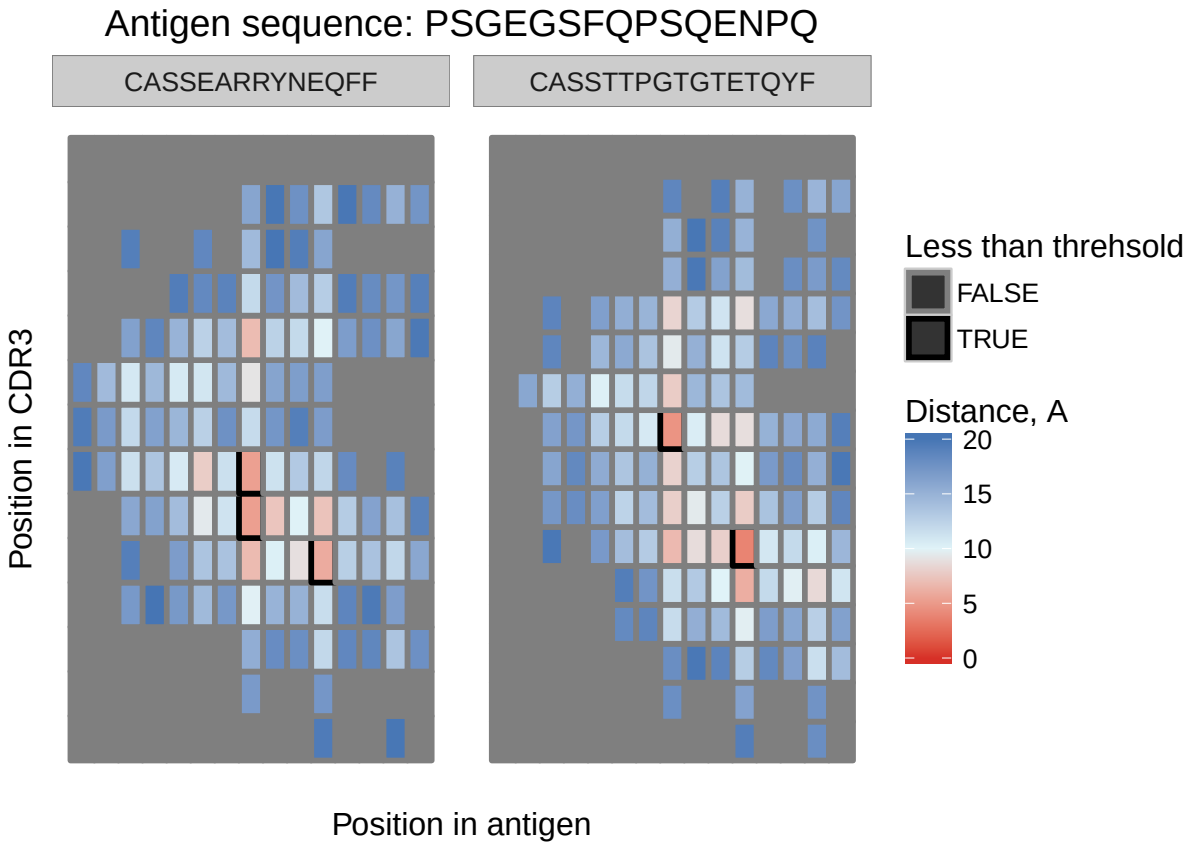
Next, we will draw contact maps, i.e. heatmaps of pairwise distances between CDR3beta and antigen residues as computed from structural data.


Antigen sequence: LLFGYPVYV
ASRPGLAGGRPEQY
ASRPGLMSAQPEQY
CASSYPGGGFYEQYF
Position in CDR3
Position in antigen
Less than threhsold
FALSE
TRUE
Distance, A
20
15
10
5
0
Antigen sequence: FLRGRAYGL
CASGQGNFDIQYF
SSFTWTSGGATDTQ
CASSLGQAYEQYF
Position in CDR3
Position in antigen
Less than threhsold
FALSE
TRUE
Distance, A
20
15
10
5
0


Antigen sequence: GELIGILNAAKVPAD
CASSQIRETQYF
CASTYHGTGYF
Position in CDR3
Position in antigen
Less than threhsold
FALSE
TRUE
Distance, A
20
15
10
5
0
Antigen sequence: RGGASQYRPSQ
CASGDASGAETLYF
CASGDASGGNTLYF
Position in CDR3
Position in antigen
Less than threhsold
FALSE
TRUE
Distance, A
20
15
10
5
0


Antigen sequence: ELAGIGILTV
CAISEVGVGQPQHF
CASSLSFGTEAFF
CASSWSFGTEAFF
AWSETGLGMGGWQ
CAWSETGLGTGELFF
Position in CDR3
Position in antigen
Less than threhsold
FALSE
TRUE
Distance, A
20
15
10
5
0
Antigen sequence: AAGIGILTV
CAISEVGVGQPQHF
CASSLSFGTEAFF
Position in CDR3
Position in antigen
Less than threhsold
FALSE
TRUE
Distance, A
20
15
10
5
0


Antigen sequence: SGEGSFQPSQENP
CASSAGTSGEYEQYF
CASSVAVSAGTYEQYF
Position in CDR3
Position in antigen
Less than threhsold
FALSE
TRUE
Distance, A
20
15
10
5
0
Antigen sequence: APQPELPYPQPGS
CASSFRALAADTQYF
CASSFRFTDTQYF
Position in CDR3
Position in antigen
Less than threhsold
FALSE
TRUE
Distance, A
20
15
10
5
0



Next compute distances by aligning and computing distance matrix RMSD. While the matrices above look similar within CDR3beta subset recognizing the same antigen and different between different antigens, it is well known that computing similarity between two heatmaps is a complex task (see SO thread and this paper). So, instead of implementing all those complex stats here we'll simply compute RMSD between matrices:

- We will allow `-5:5` offset for the CDR3 dimension (antigen dimension/columns are fixed)
- We will select the best distance among those offsets
- We will compute RMSD for `1/distance` to give more weight for closer residues we are really interested in

  Note: we can use exp(-distance) to get the same result & matrices will be very similar to energy matrices, we can also get similar results with energy matrices. But for consistency (the MS is based on distances and <=6A threshold) we'll stick to distances here.

Some helper functions to align/compute distance between contact matrices.

```
library(Biostrings)
```

```
## Loading required package: BiocGenerics
```

```
## Loading required package: parallel
```

```
##
## Attaching package: 'BiocGenerics'
```

```
## The following objects are masked from 'package:parallel':
##
##     clusterApply, clusterApplyLB, clusterCall, clusterEvalQ,
##     clusterExport, clusterMap, parApply, parCapply, parLapply,
##     parLapplyLB, parRapply, parSapply, parSapplyLB

## The following objects are masked from 'package:stats':
##
##     IQR, mad, xtabs

## The following objects are masked from 'package:base':
##
##     anyDuplicated, append, as.data.frame, as.vector, cbind,
##     colnames, do.call, duplicated, eval, evalq, Filter, Find, get,
##     grep, grepl, intersect, is.unsorted, lapply, lengths, Map,
##     mapply, match, mget, order, paste, pmax, pmax.int, pmin,
##     pmin.int, Position, rank, rbind, Reduce, rownames, sapply,
##     setdiff, sort, table, tapply, union, unique, unlist, unsplit

## Loading required package: S4Vectors

## Loading required package: stats4

##
## Attaching package: 'S4Vectors'

## The following object is masked from 'package:plyr':
##
##     rename

## Loading required package: IRanges

##
## Attaching package: 'IRanges'

## The following object is masked from 'package:plyr':
##
##     desc

## Loading required package: XVector

##
## Attaching package: 'XVector'

## The following object is masked from 'package:plyr':
##
##     compact
```

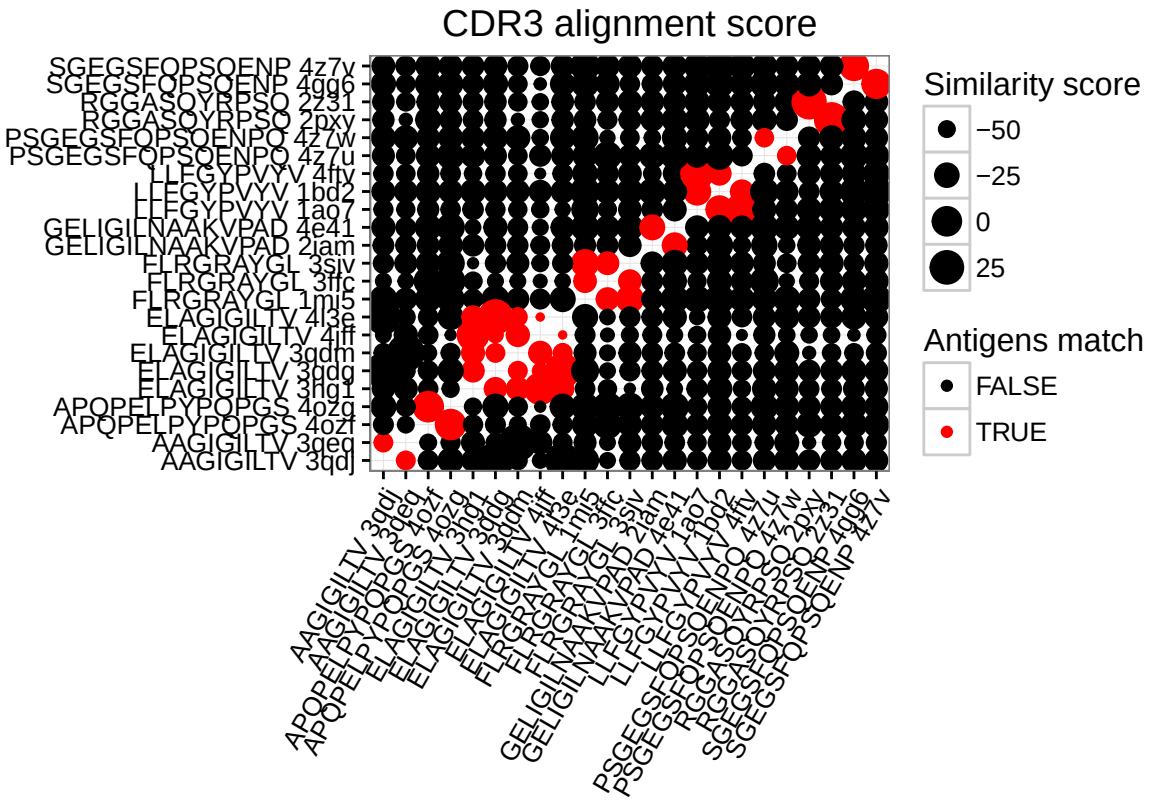
```
# Global pairwise alignment with default parameters

get_full_seq <- function(tcr_data) {
  as.character(tcr_data$tcr_region_seq[1])
}

get_aln_score <- function(tcr_data1, tcr_data2) {
  pairwiseAlignment(get_full_seq(tcr_data1), get_full_seq(tcr_data2), scoreOnly = T)
}

# Computing RMSD between inverted AA pairwise distance matrices

get_pos_dist_mat <- function(tcr_data, offset = 0) {
  data.frame(pos_tcr = tcr_data$pos_tcr + offset,
             pos_antigen = tcr_data$pos_antigen,
             distance = tcr_data$distance)
}

get_matrix_score <- function(tcr_data1, tcr_data2, offset_range = 5) {
  dmin <- 99999
  for (offset in -offset_range:offset_range) {
    d1 <- get_pos_dist_mat(tcr_data1, offset)
    d2 <- get_pos_dist_mat(tcr_data2)

    dd <- merge(d1, d2, by = c("pos_tcr", "pos_antigen"))
    dmin <- min(dist(rbind(1/dd$distance.x, 1/dd$distance.y))[1] / sqrt(nrow(dd)), dmin)
  }
  -dmin
}
```

The computation itself (a bit time consuming, haven't optimized).

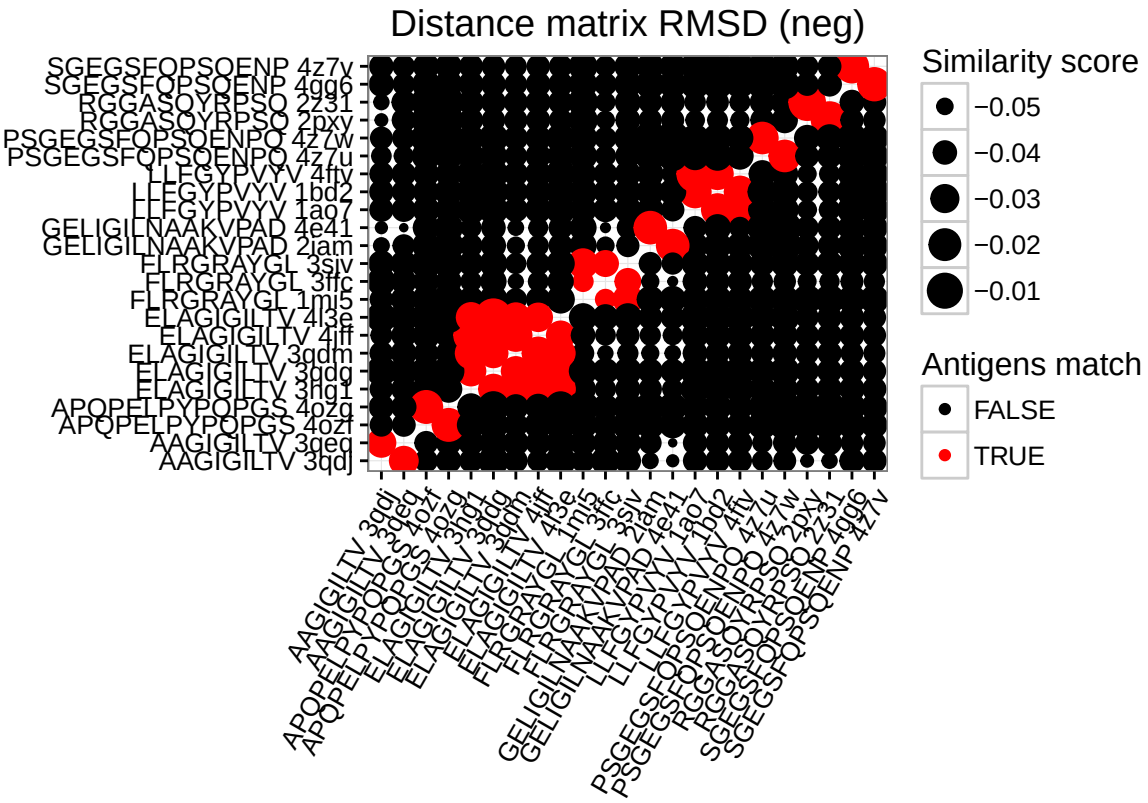
```
pdb_ids <- unique(df.m$pdb_id)

df.d <- expand.grid(pdb_ids, pdb_ids)
colnames(df.d) <- c("pdb_id_1", "pdb_id_2")

df.d <- subset(df.d, pdb_id_1 != pdb_id_2)

df.d <- ddply(df.d, .(pdb_id_1, pdb_id_2), transform,
      dist_aln = get_aln_score(subset(df, pdb_id==pdb_id_1), subset(df, pdb_id==pdb_id_2)),
      dist_mat = get_matrix_score(subset(df, pdb_id==pdb_id_1), subset(df, pdb_id==pdb_id_2)))

df.d <- merge(df.d,
              unique(data.frame(pdb_id_1=df.m$pdb_id, antigen_seq_1=df.m$antigen_seq)),
              all.x = T, all.y = F)

df.d <- merge(df.d,
              unique(data.frame(pdb_id_2=df.m$pdb_id, antigen_seq_2=df.m$antigen_seq)),
              all.x = T, all.y = F)
```

Plot similarity maps computed using CDR3 alignment and contact matrix correlation. Highlight TCR beta variants that recognize the same antigen.

```
ggplot(df.d, aes(x=paste(antigen_seq_1, pdb_id_1),
                 y=paste(antigen_seq_2, pdb_id_2),
                 size = dist_aln,
                 color = antigen_seq_1==antigen_seq_2)) +
  scale_size_continuous("Similarity score") +
  scale_color_manual("Antigens match", values = c("black", "red")) +
  labs(title="CDR3 alignment score", x="", y="") +
  geom_point() +
  theme_bw() +
  theme(axis.text.x = element_text(angle=60, hjust=1))
```




```
ggplot(df.d, aes(x=paste(antigen_seq_1, pdb_id_1),
                 y=paste(antigen_seq_2, pdb_id_2),
                 size = dist_mat,
                 color = antigen_seq_1==antigen_seq_2)) +
  scale_size_continuous("Similarity score") +
  scale_color_manual("Antigens match", values = c("black", "red")) +
  labs(title="Distance matrix RMSD (neg)", x="", y="") +
  geom_point() +
  theme_bw() +
  theme(axis.text.x = element_text(angle=60, hjust=1))
```



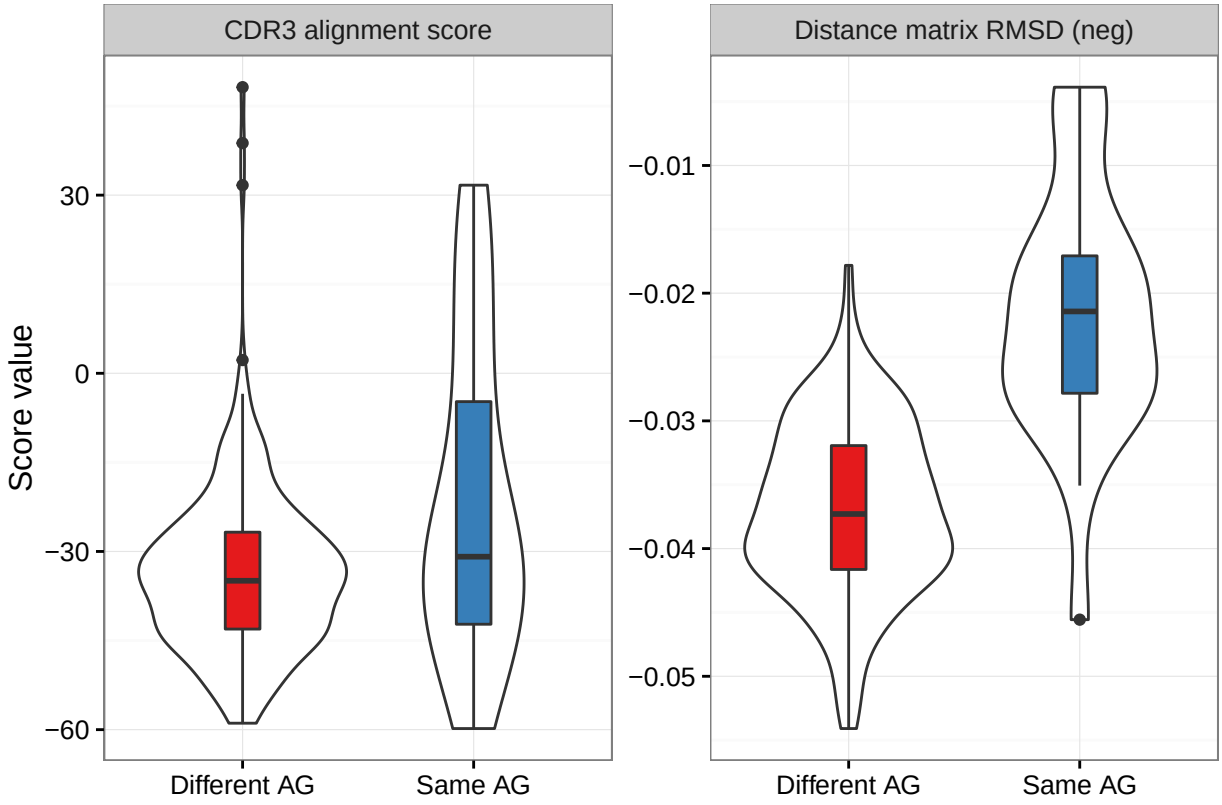
Its clear from the previous plot that CDR3 alignment scores cannot be used to match CDR3beta with the same antigen. On the other hand, RMSD values computed for CDR3:antigen distance matrices can distinguish complexes with the same antigen.

```
library(reshape2)
df.dd <- melt(df.d)
```

```
## Using pdb_id_2, pdb_id_1, antigen_seq_1, antigen_seq_2 as id variables
```

```
# first - get rid of duplicates
dedupl <- function(df1) {
  df1$pdb_id_1 <- as.integer(df1$pdb_id_1)
  df1$pdb_id_2 <- as.integer(df1$pdb_id_2)

  df1[1:2] <- t(apply(df1[1:2], 1, sort))
  df1[!duplicated(df1[1:2]),]
}

df.dd <- rbind(dedupl(subset(df.dd, variable=="dist_aln")),
               dedupl(subset(df.dd, variable=="dist_mat")))

# rename a bit

df.dd$same_ag <- ifelse(df.dd$antigen_seq_1==df.dd$antigen_seq_2, "Same AG", "Different AG")
df.dd$variable <- revalue(df.dd$variable, c("dist_aln"="CDR3 alignment score",
                                            "dist_mat"="Distance matrix RMSD (neg)"))

ggplot(df.dd, aes(x=same_ag, group=same_ag, y = value)) +
```

```
        geom_violin() +
        geom_boxplot(width=0.2, aes(fill=same_ag)) +
        scale_fill_brewer(guide=F, palette = "Set1") +
        facet_wrap(~variable, scales = "free") +
        xlab("") + ylab("Score value") +
        theme_bw()
```



Perform Mann-Whitney-Wilcoxon test for both scores. Its non-parametric so the fact that we have different scales/distributions for our scores is not a problem.

```
df.dda <- subset(df.dd, variable == "CDR3 alignment score")
df.ddr <- subset(df.dd, variable == "Distance matrix RMSD (neg)")
wilcox.test(value ~ same_ag, data=df.dda)
```

```
##
##  Wilcoxon rank sum test with continuity correction
##
## data:  value by same_ag
## W = 1959.5, p-value = 0.07647
## alternative hypothesis: true location shift is not equal to 0
```

```
wilcox.test(value ~ same_ag, data=df.ddr)
```

```
##
##  Wilcoxon rank sum test with continuity correction
##
```

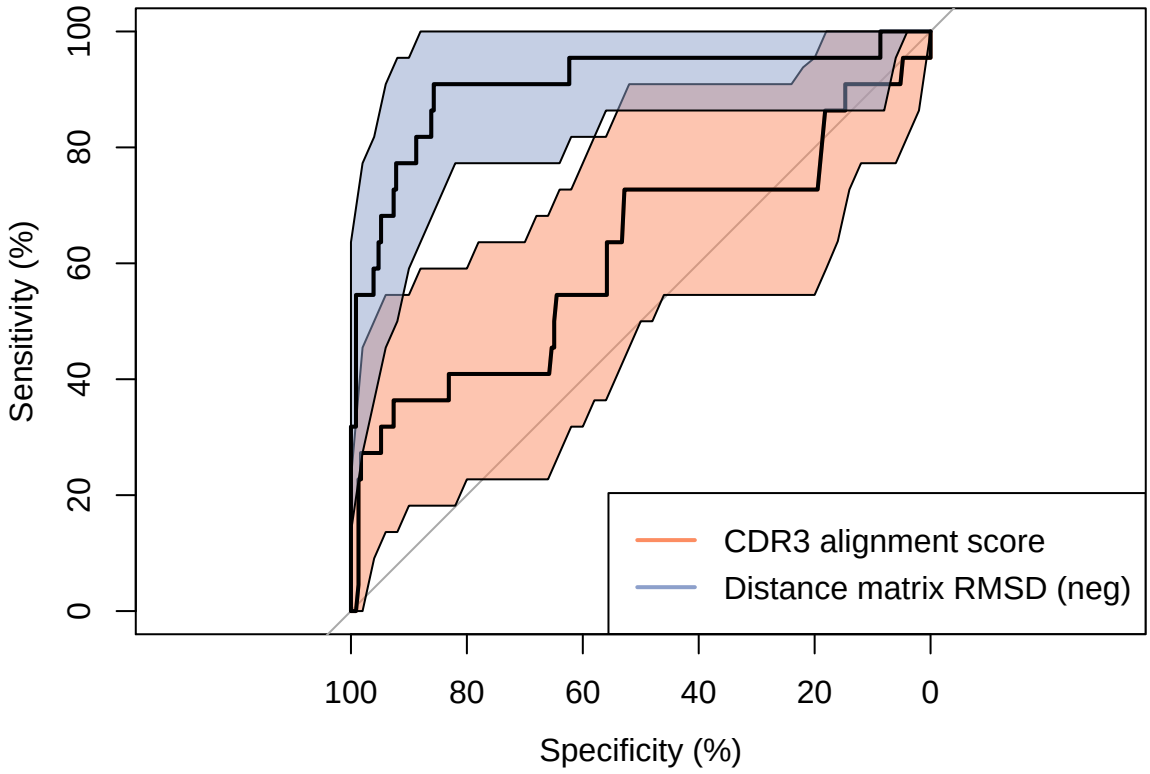
```
## data:  value by same_ag
## W = 466, p-value = 2.53e-10
## alternative hypothesis: true location shift is not equal to 0
```

Finally, lets plot ROC curves for both scores.

```
library(pROC)
```

```
## Type 'citation("pROC")' for a citation.

##
## Attaching package: 'pROC'

## The following objects are masked from 'package:IRanges':
##
##     cov, var

## The following objects are masked from 'package:stats':
##
##     cov, smooth, var
```

```
rocobj1 <- plot.roc(df.dda$same_ag, df.dda$value,  percent=T, col="#fc8d62", ci=T)
rocobj2 <- lines.roc(df.ddr$same_ag, df.ddr$value, percent=T, col="#8da0cb", ci=T)

sens.ci <- ci.se(rocobj1, specificities=seq(0, 100, 2))
plot(sens.ci, type="shape", col = alpha("#fc8d62", 0.5))
sens.ci <- ci.se(rocobj2, specificities=seq(0, 100, 2))
plot(sens.ci, type="shape", col = alpha("#8da0cb", 0.5))

legend("bottomright", legend=c("CDR3 alignment score", "Distance matrix RMSD (neg)"),
       col=c("#fc8d62", "#8da0cb"), lwd=2)
```


Sensitivity (%)
100
80
60
40
20
0
100
80
60
40
20
0
Specificity (%)
CDR3 alignment score
Distance matrix RMSD (neg)

QED.