Hallmarks of TCR:peptide:MHC interactions inferred from structural data mining.

Mikhail Shugay

February 29, 2016

Structural data used in the study

TCR:peptide:MHC complexe entries were obtained from PDB by a batch query with corresponding keywords. Complex records were then automatically annotated using in-house scripts that performed:

- TCR, MHC and antigen flags were assigned to chain records
- Antigen and host species were inferred
- MHC alleles were assigned using blast search against a database of MHC protein sequences manually assembled from public databases
- TCR partitioning into CDR and Framework regions was performed using custom IgBlast wrapper

This dataset was then used to generate a flat table with annotated TCR:antigen amino acid pairs.

```
library(plyr)
library(ggplot2)
library(reshape2)
library(gplots)

df <- read.table("../result/structure.txt", header=T, sep="\t")
df$energy[is.na(df$energy)] <- 0

df <- ddply(df, .(tcr_v_allele), transform,
            tcr_chain = factor(substr(as.character(tcr_v_allele[1]), 1, 3)))
```

The total number of complexes that were successfully annotated was

```
length(levels(df$pdb_id))
```

```
[1] 103
```

and the total number of amino acid pairs was

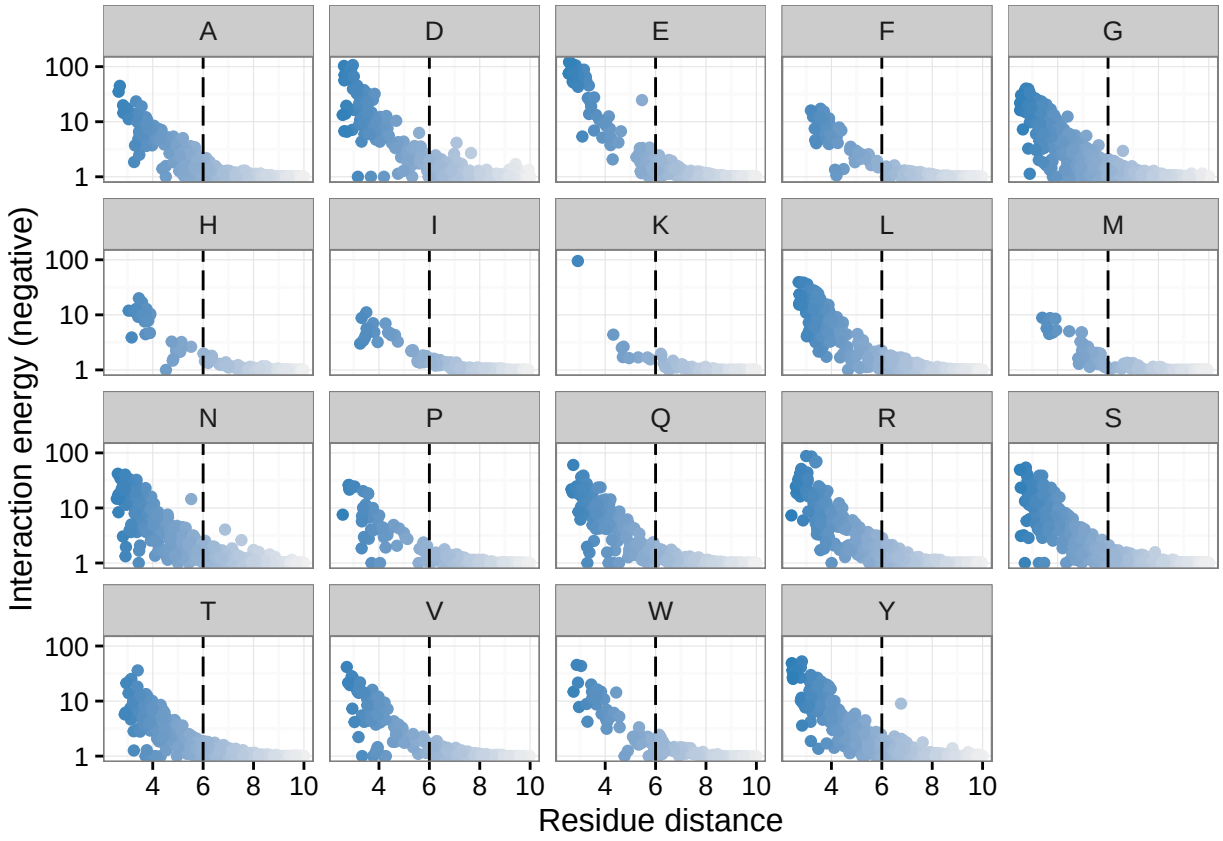
```
nrow(df)
```

```
[1] 51904
```

Selecting distance threshold for amino acid contacts

GROMACS software was used to calculate point energies for amino acid contacts using TCR:peptide:MHC structures. Each amino acid pair record in the database was then assigned with an interaction energy value using in-house scripts. Distances between residues were computed as the minimal distance between a pair of atoms using Bio.PDB python package. Interaction energies grouped by CDR3 amino acid are plotted against residue distances below. Selected distance threshold for contacting residues is shown as a vertical line, the same value is used further throughout the manuscript.

```
DIST_THRESHOLD = 6

ggplot(subset(df, distance <= 10 & energy <= 0),
       aes(x=distance, y=-energy+1, colour=distance)) +
  geom_point() + geom_vline(xintercept = DIST_THRESHOLD, linetype = "longdash") +
  scale_y_log10(name = "Interaction energy (negative)") + xlab("Residue distance") +
  scale_colour_gradient(guide = FALSE, low = "#2c7fb8", high = "#f0f0f0") +
  facet_wrap(~aa_tcr) + theme_bw()
```



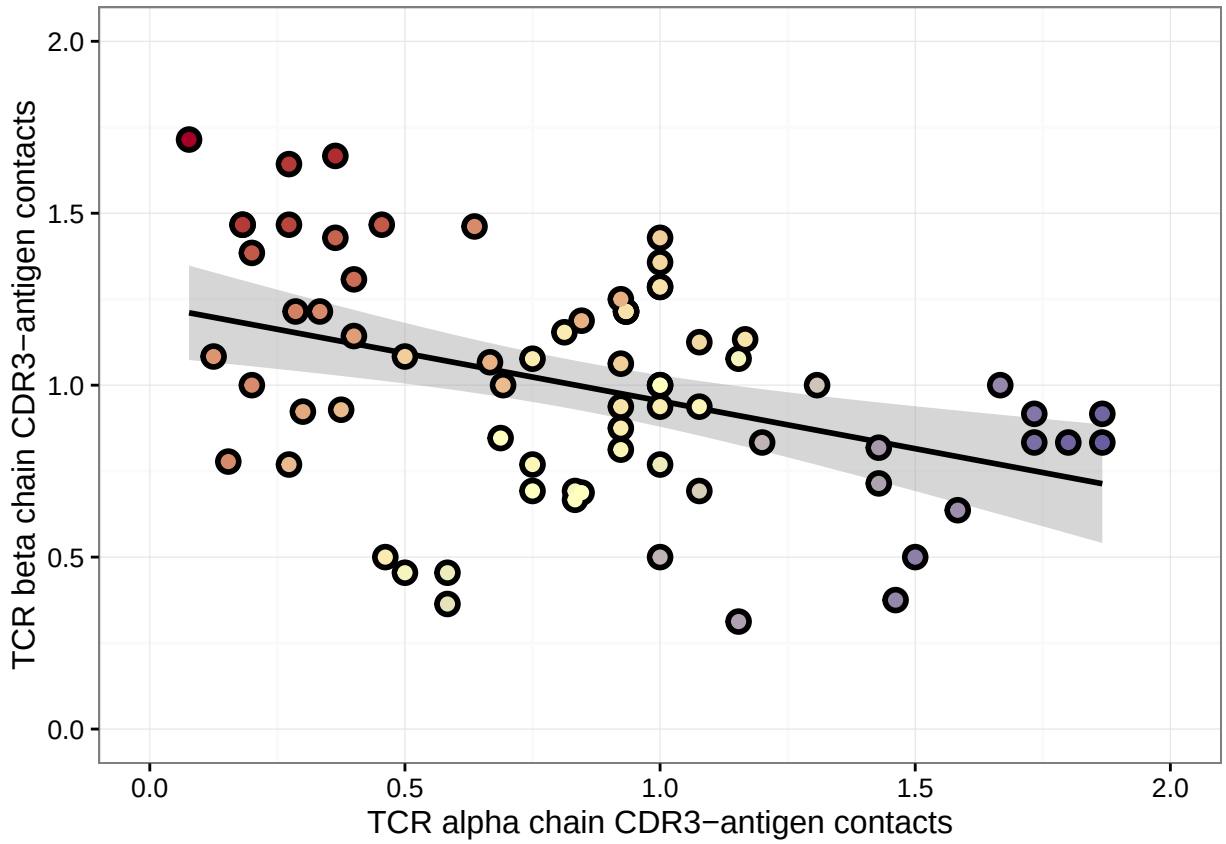
Studying distribution of TCR:antigen contact residues

Number of TCR alpha and beta chain contacts is inversely correlated

The first hypothesis we've tested was the correlation between number of antigen contacts with TCR alpha and beta chains within the same TCR:peptide:MHC complex. We've focused on CDR3 region as the one having a critical role in antigen specificity and MHCI molecules. Based on previous observations of Yokosuka et al. 2002 and Turner et al. 1997, we have hypothesized that while the number of CDR3-antigen contacts per complex is quite stable, CDR3 regions of alpha and beta chains "compete" for antigen binding, resulting in TCRs in which one of the chains has a dominant role in antigen binding. Indeed, an inverse correlation between number of TCR alpha and beta CDR3-antigen contacts was observed as shown in the plot below.

```
df.s <- ddply(subset(df, tcr_region == "CDR3" & mhc_type=="MHCI"),
              .(pdb_id, tcr_chain),
              summarize,
              csum=sum(distance <= DIST_THRESHOLD),
```

```
              len = mean(len_tcr),
              mhc_a_allele = mhc_a_allele[1]
              )
df.s <- subset(df.s, csum > 0)
df.s <- merge(subset(df.s, tcr_chain=="TRA"), subset(df.s, tcr_chain=="TRB"),
              by="pdb_id", suffixes = c(".TRA",".TRB"))

ggplot(df.s, aes(x=csum.TRA / len.TRA, y=csum.TRB / len.TRB)) +
  geom_smooth(method="lm", color="black") +
  geom_point(size=4) +
  geom_point(size=2, aes(color=csum.TRA-csum.TRB)) +
  scale_color_gradient2(guide = FALSE, low = "#a50026", mid = "#ffffbf", high = "#313695") +
  scale_x_continuous(name = "TCR alpha chain CDR3-antigen contacts", limits=c(0,2)) +
  scale_y_continuous(name = "TCR beta chain CDR3-antigen contacts", limits=c(0,2)) +
  #facet_wrap(~mhc_a_allele.TRA) +
  theme_bw()
```



The statistical significance of the observed dependency is given below.

```
summary(lm(formula = csum.TRA ~ csum.TRB, data = df.s))
```

```
##
## Call:
## lm(formula = csum.TRA ~ csum.TRB, data = df.s)
##
## Residuals:
```

```
##      Min       1Q   Median       3Q      Max
## -10.8601  -4.5618  -0.6146   4.3854  15.0171
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)  19.1581     2.2485   8.520 2.01e-12 ***
## csum.TRB     -0.5614     0.1523  -3.686 0.000446 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 6.464 on 70 degrees of freedom
## Multiple R-squared:  0.1625, Adjusted R-squared:  0.1506
## F-statistic: 13.58 on 1 and 70 DF,  p-value: 0.0004462
```

```
summary(lm(formula = csum.TRA / len.TRA ~ csum.TRB / len.TRB, data = df.s))
```

```
##
## Call:
## lm(formula = csum.TRA/len.TRA ~ csum.TRB/len.TRB, data = df.s)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -0.79459 -0.32244 -0.01047  0.31006  0.87869
##
## Coefficients:
##                   Estimate Std. Error t value Pr(>|t|)
## (Intercept)       1.364102   0.160208   8.515 2.29e-12 ***
## csum.TRB         -0.014631   0.044744  -0.327    0.745
## csum.TRB:len.TRB -0.001630   0.002781  -0.586    0.560
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.4246 on 69 degrees of freedom
## Multiple R-squared:  0.1928, Adjusted R-squared:  0.1694
## F-statistic:  8.24 on 2 and 69 DF,  p-value: 0.000618
```

Distribution of total number of CDR3 contacts per complex is given below.

```
df.s <- ddply(df.s, .(pdb_id),
              summarize, csum=csum.TRA+csum.TRB)
summary(df.s)
```

```
##      pdb_id        csum
##  1ao7   : 1   Min.   :11.00
##  1bd2   : 1   1st Qu.:20.75
##  1g6r   : 1   Median :26.00
##  1kj2   : 1   Mean   :25.25
##  1mi5   : 1   3rd Qu.:29.25
##  1mwa   : 1   Max.   :39.00
##  (Other):66
```

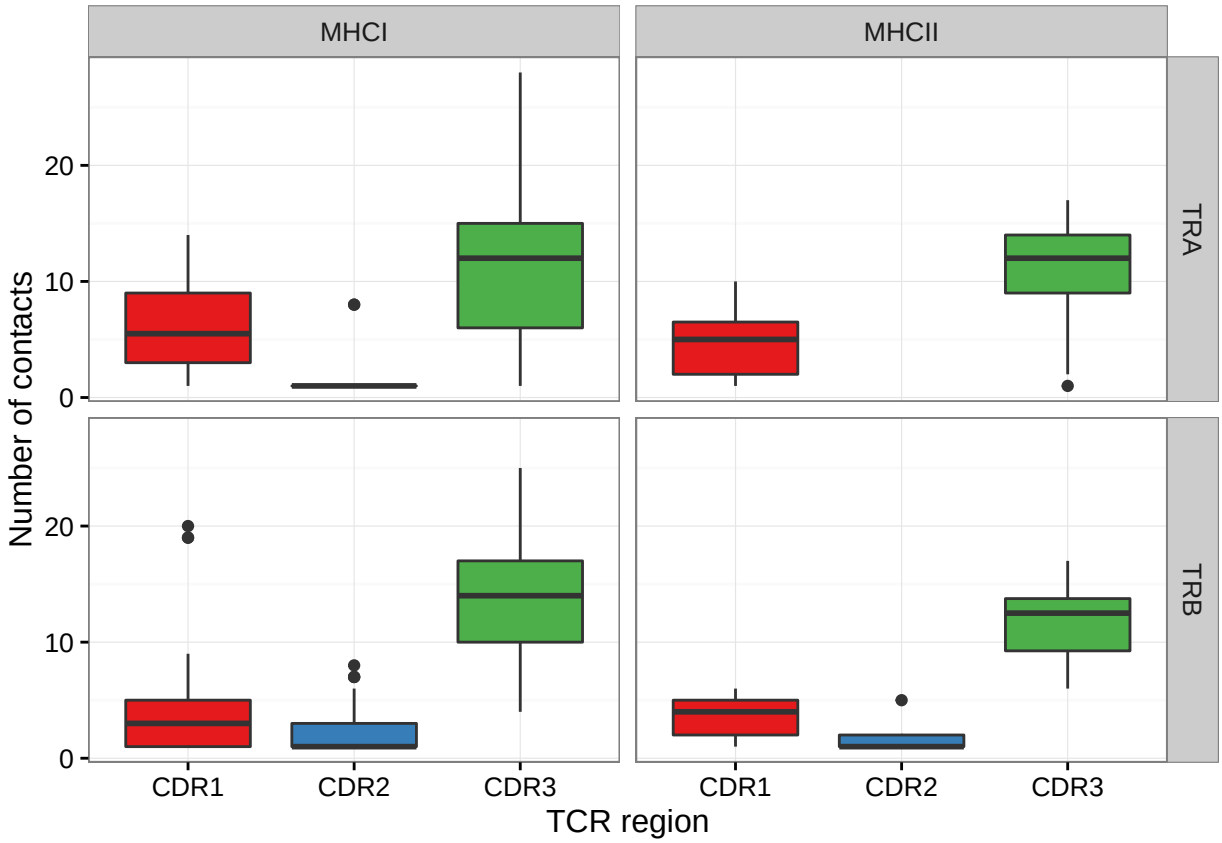
TCR region and MHC contact preferences

Next, we've extended our analysis on other TCR regions (CDR1 and 2 in germline) and MHCII. Surprisingly, CDR1, but not CDR2, appears to confer a substantial number of TCR-antigen contacts, comparable to that of CDR3.

> Given the apparent role of CDR1 in peptide recognition, we present a hypothesis (yet to be tested) stating that the observed germline Variable (V) segment restriction and cross-reactivity to MHC alleles Garcia 2012 can be at least partially explained by thymic selection due to the CDR1-encoded specificity to the self-peptide pool presented by a certain MHC. Note that this hypothesis is strengthened by the study of Cole et al 2014 demonstrating that TCR-peptide specificity overrides affinity-enhancing TCR-MHC interactions.

Additionally, it appears that MHCI complexes have a higer number of TCR-antigen contacts than MHCII complexes. Our observations are summarized in the figure below.

```
df.r <- ddply(df, .(pdb_id, tcr_chain, tcr_region, mhc_type), summarize,
              csum=sum(distance <= DIST_THRESHOLD))

ggplot(subset(df.r, csum > 0), aes(x=tcr_region,group=tcr_region,y=csum,fill=tcr_region)) +
  geom_boxplot() + xlab("TCR region") + ylab("Number of contacts") +
  scale_fill_brewer(guide = F, palette = "Set1") +
  facet_grid(tcr_chain~mhc_type) + theme_bw()
```



Statistical significance of results described in this section is provided below.

```
a <- aov(csum~tcr_region + tcr_chain + mhc_type, df.r)
summary(a)
```

```
##              Df Sum Sq Mean Sq F value  Pr(>F)
## tcr_region    2  12607    6304 340.264 < 2e-16 ***
## tcr_chain     1     10      10   0.538 0.46345
## mhc_type      1    202     202  10.914 0.00101 **
## Residuals   612  11338      19
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
TukeyHSD(a, "mhc_type")
```

```
##   Tukey multiple comparisons of means
##     95% family-wise confidence level
##
## Fit: aov(formula = csum ~ tcr_region + tcr_chain + mhc_type, data = df.r)
##
## $mhc_type
##                diff       lwr        upr     p adj
## MHCII-MHCI -1.30093 -2.074274 -0.5275864 0.0010102
```

```
TukeyHSD(a, "tcr_region")
```

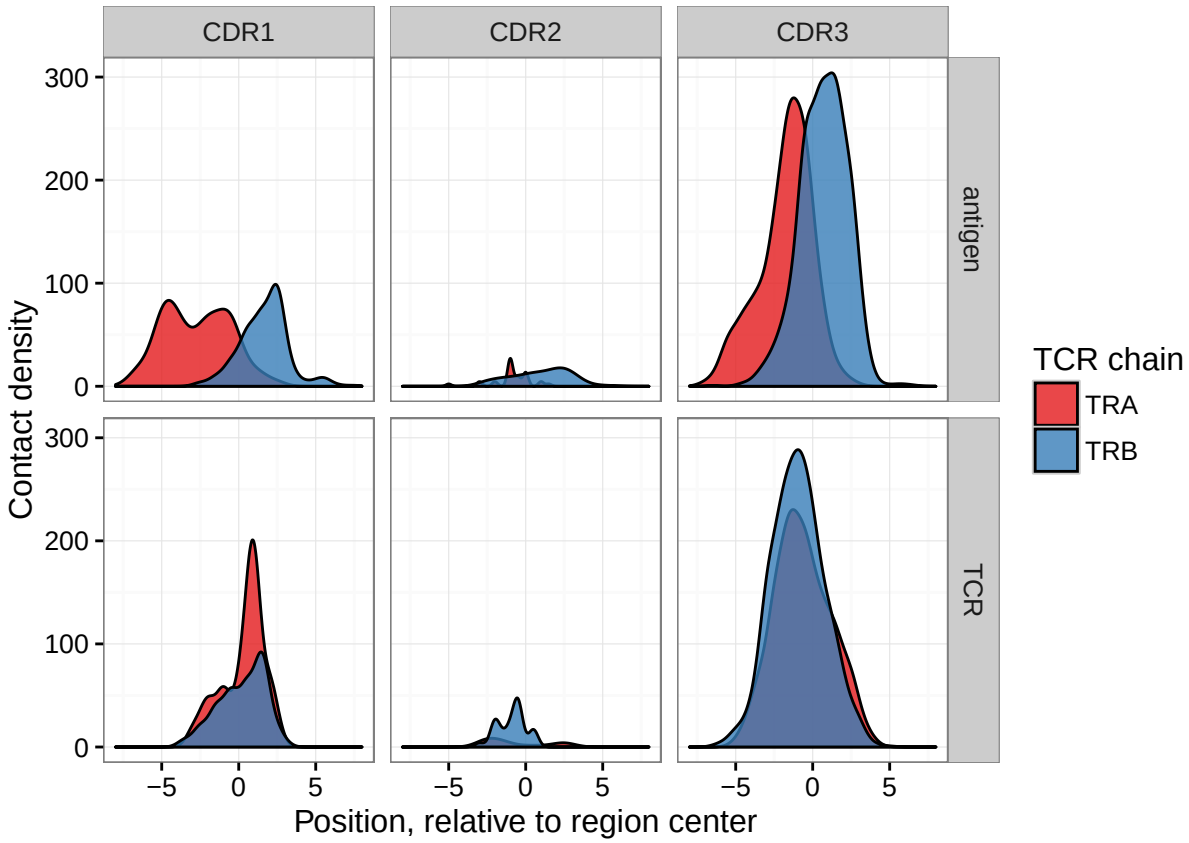
```
##   Tukey multiple comparisons of means
##     95% family-wise confidence level
##
## Fit: aov(formula = csum ~ tcr_region + tcr_chain + mhc_type, data = df.r)
##
## $tcr_region
##                diff       lwr       upr p adj
## CDR2-CDR1 -3.317641 -4.315244 -2.320039     0
## CDR3-CDR1  7.490291  6.493903  8.486680     0
## CDR3-CDR2 10.807933  9.810330 11.805535     0
```

Contact residues are tightly clustered on TCR and antigen length

We have next studied the distribution of interacting residues by their position on antigen and CDR sequences. Below are the plots of contact distribution grouped by TCR region conferring the contact (columns) and contact residue parent sequence (rows). CDR contacts tend to cluster near the center of corresponding region. Note the clear difference between TCR alpha and beta contacts that are closer to **N** and **C** terminus of the antigen peptide respectively.

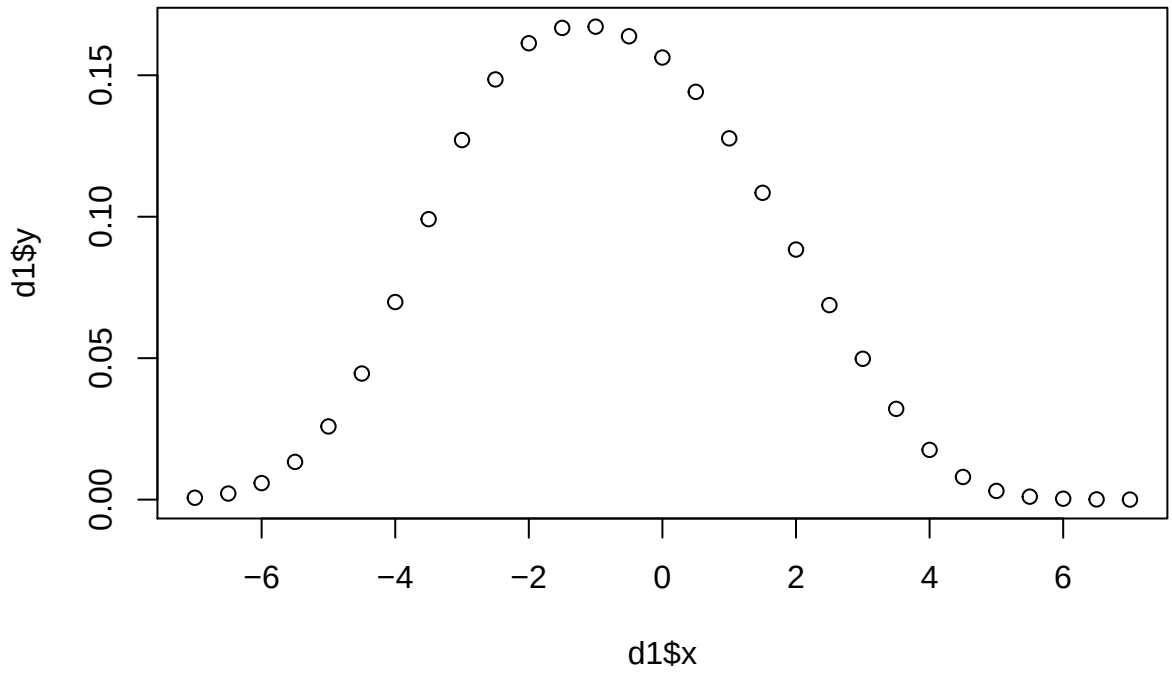
```
df.p1 <- ddply(df, .(pdb_id, tcr_chain, tcr_region, mhc_type, pos_tcr), summarize,
               pos_norm = mean(pos_tcr - len_tcr / 2),
               contacts = sum(distance <= DIST_THRESHOLD))
df.p1$pos_tcr <- NULL
df.p1$sequence <- "TCR"

df.p2 <- ddply(df, .(pdb_id, tcr_chain, tcr_region, mhc_type, pos_antigen), summarize,
               pos_norm = mean(pos_antigen - len_antigen / 2),
```

```
              contacts = sum(distance <= DIST_THRESHOLD))
df.p2$pos_antigen <- NULL
df.p2$sequence <- "antigen"

df.p <- rbind(df.p1, df.p2)

ggplot(subset(df.p, contacts > 0),
       aes(x=pos_norm, weight=contacts, fill=tcr_chain)) +
  geom_density(alpha=0.8) +
  scale_fill_brewer(name = "TCR chain", palette = "Set1") +
  ylab("Contact density") +
  scale_x_continuous(name = "Position, relative to region center", limits=c(-8,8)) +
  facet_grid(sequence~tcr_region) +
  theme_bw()
```



Concat profile:

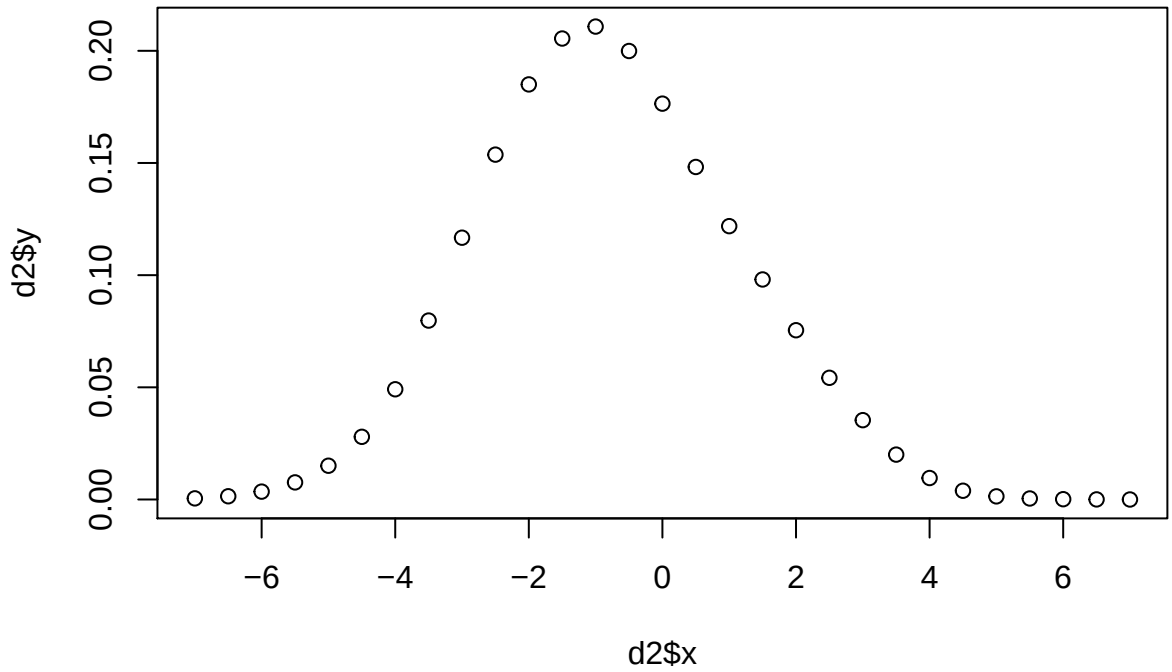
```
library(stats)

df.pp <- subset(df.p, sequence == "TCR" & tcr_region == "CDR3")
wt <- ifelse(df.pp$contacts == 0, 0, 1)
wt <- wt / sum(wt)
d1 <- density(df.pp$pos_norm, weights = wt, from=-7, to=7, n=29)
plot(d1$x, d1$y)
```




```
print(data.frame(x=d1$x, y=d1$y))
```

```
##       x           y
## 1  -7.0 6.661484e-04
## 2  -6.5 2.154686e-03
## 3  -6.0 5.845091e-03
## 4  -5.5 1.332682e-02
## 5  -5.0 2.586417e-02
## 6  -4.5 4.456489e-02
## 7  -4.0 6.983724e-02
## 8  -3.5 9.913621e-02
## 9  -3.0 1.270954e-01
## 10 -2.5 1.485377e-01
## 11 -2.0 1.613228e-01
## 12 -1.5 1.667353e-01
## 13 -1.0 1.671752e-01
## 14 -0.5 1.637894e-01
## 15  0.0 1.562721e-01
## 16  0.5 1.441522e-01
## 17  1.0 1.276963e-01
## 18  1.5 1.084490e-01
## 19  2.0 8.840289e-02
## 20  2.5 6.875496e-02
## 21  3.0 4.978266e-02
## 22  3.5 3.207937e-02
## 23  4.0 1.757951e-02
## 24  4.5 8.005986e-03
## 25  5.0 3.072727e-03
## 26  5.5 1.044496e-03
## 27  6.0 3.181196e-04
## 28  6.5 7.924924e-05
## 29  7.0 1.428670e-05
```

```
wt <- df.pp$contacts
wt <- wt / sum(wt)
d2 <- density(df.pp$pos_norm, weights = wt, from=-7, to=7, n=29)
plot(d2$x, d2$y)
```




```
print(data.frame(x=d2$x, y=d2$y))
```

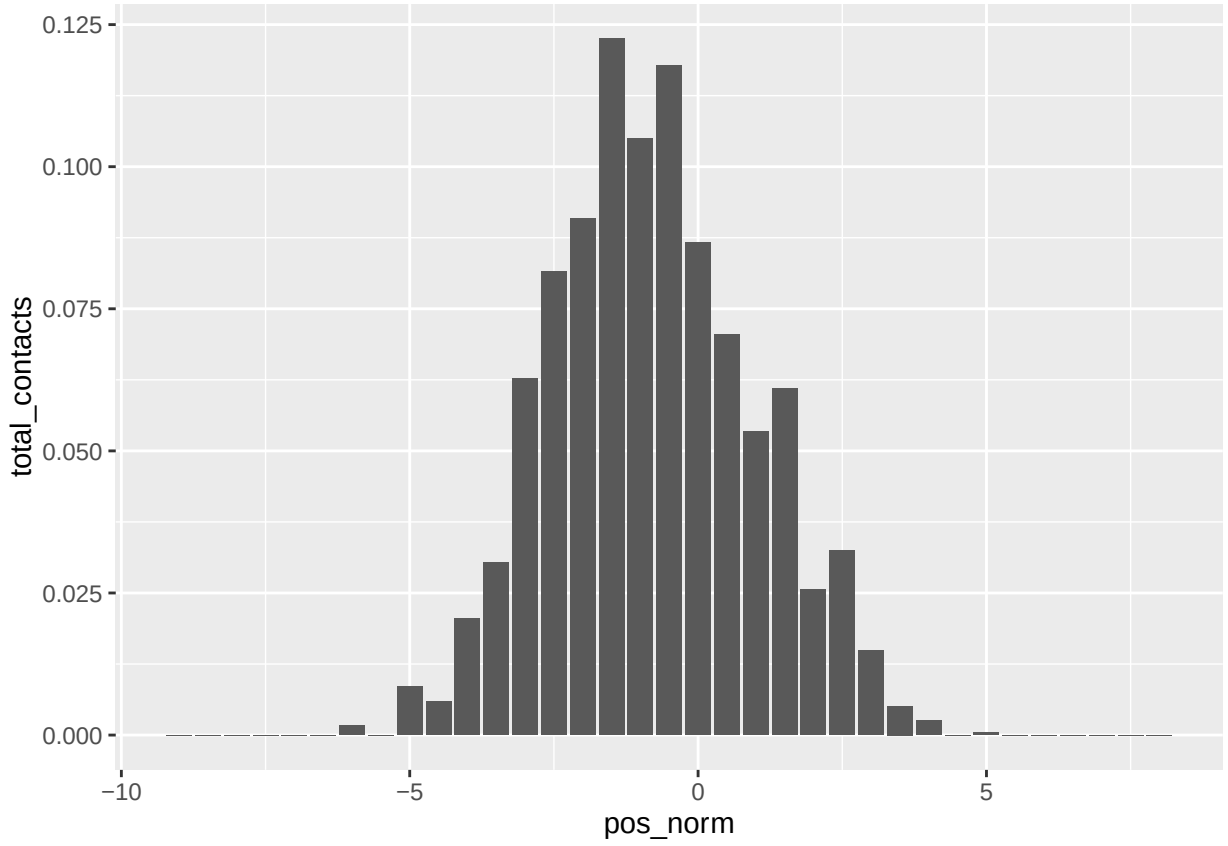
```
##       x           y
## 1  -7.0 4.969965e-04
## 2  -6.5 1.436694e-03
## 3  -6.0 3.511817e-03
## 4  -5.5 7.624702e-03
## 5  -5.0 1.505454e-02
## 6  -4.5 2.794098e-02
## 7  -4.0 4.913465e-02
## 8  -3.5 7.976974e-02
## 9  -3.0 1.166962e-01
## 10 -2.5 1.537248e-01
## 11 -2.0 1.850292e-01
## 12 -1.5 2.054815e-01
## 13 -1.0 2.108031e-01
## 14 -0.5 1.999189e-01
## 15  0.0 1.764592e-01
## 16  0.5 1.482253e-01
## 17  1.0 1.218442e-01
## 18  1.5 9.808052e-02
## 19  2.0 7.544367e-02
## 20  2.5 5.424554e-02
## 21  3.0 3.536574e-02
## 22  3.5 2.000912e-02
## 23  4.0 9.575709e-03
```

```
## 24  4.5 3.900203e-03
## 25  5.0 1.393621e-03
## 26  5.5 4.576403e-04
## 27  6.0 1.377419e-04
## 28  6.5 3.421721e-05
## 29  7.0 6.165308e-06
```

```
df.pp <- ddply(subset(df.p, sequence == "TCR" & tcr_region == "CDR3"),
               .(pos_norm), summarize,
               total_contacts = sum(contacts))

df.pp$total_contacts <- df.pp$total_contacts / sum(df.pp$total_contacts)

print(df.pp)
```

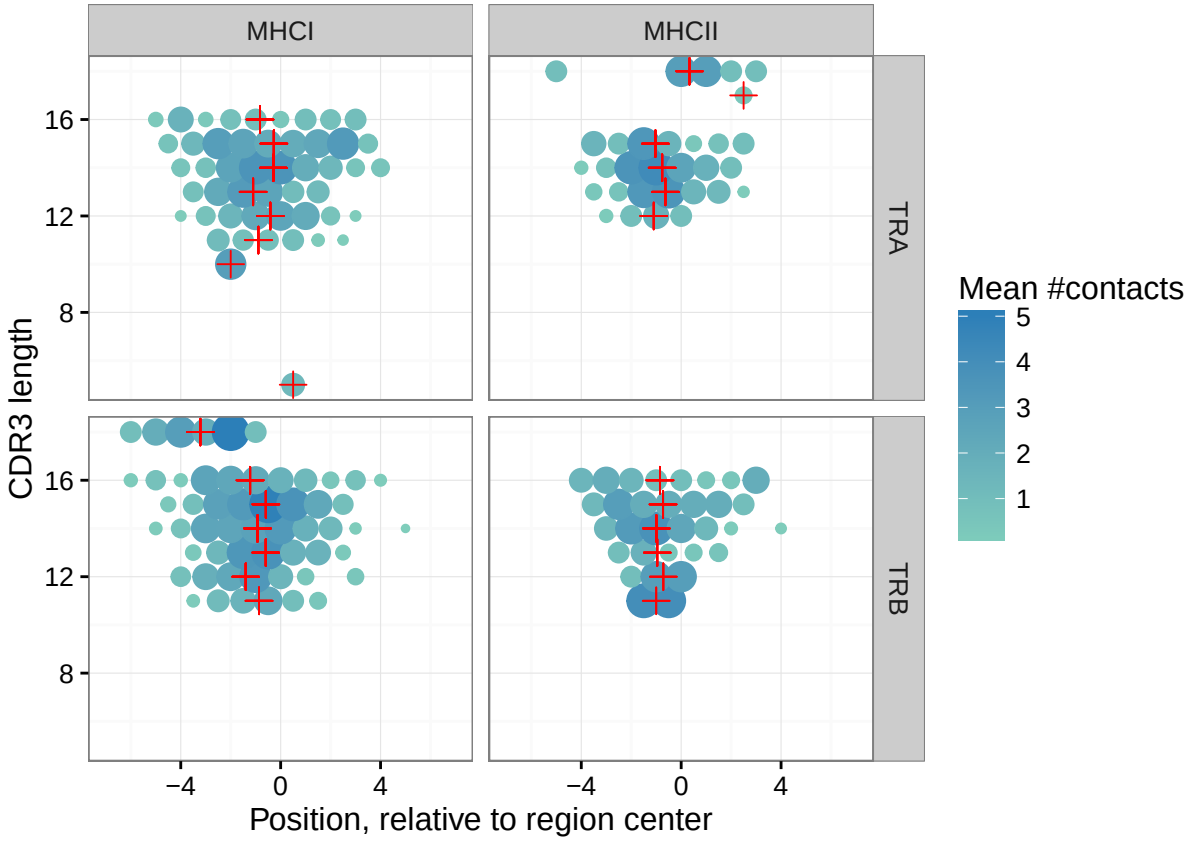
```
##    pos_norm total_contacts
## 1      -9.0   0.0000000000
## 2      -8.5   0.0000000000
## 3      -8.0   0.0000000000
## 4      -7.5   0.0000000000
## 5      -7.0   0.0000000000
## 6      -6.5   0.0000000000
## 7      -6.0   0.0017072130
## 8      -5.5   0.0000000000
## 9      -5.0   0.0085360649
## 10     -4.5   0.0059752454
## 11     -4.0   0.0204865557
## 12     -3.5   0.0303030303
## 13     -3.0   0.0627400768
## 14     -2.5   0.0815194195
## 15     -2.0   0.0909090909
## 16     -1.5   0.1224925309
## 17     -1.0   0.1049935980
## 18     -0.5   0.1177976953
## 19      0.0   0.0866410585
## 20      0.5   0.0704225352
## 21      1.0   0.0533504055
## 22      1.5   0.0610328638
## 23      2.0   0.0256081946
## 24      2.5   0.0324370465
## 25      3.0   0.0149381135
## 26      3.5   0.0051216389
## 27      4.0   0.0025608195
## 28      4.5   0.0000000000
## 29      5.0   0.0004268032
## 30      5.5   0.0000000000
## 31      6.0   0.0000000000
## 32      6.5   0.0000000000
## 33      7.0   0.0000000000
## 34      7.5   0.0000000000
## 35      8.0   0.0000000000
```

```
ggplot(df.pp, aes(x=pos_norm, y=total_contacts)) + geom_bar(stat="identity")
```



TODO: relative contact position density and CDR3 length

```
df.p1 <- ddply(subset(df, tcr_region == "CDR3"), .(pdb_id, len_tcr, tcr_chain, mhc_type, pos_tcr), summa
              contacts = sum(distance <= DIST_THRESHOLD))

df.p1 <- ddply(df.p1, .(len_tcr, tcr_chain, mhc_type, pos_tcr), summarize,
               contacts = mean(contacts))

df.p1 <- ddply(df.p1, .(len_tcr, tcr_chain, mhc_type), transform,
               mean_contacts_pos = sum(pos_tcr * contacts / sum(contacts)))

ggplot(subset(df.p1, contacts > 0),
       aes(y=len_tcr)) +
  geom_point(aes(x=pos_tcr - len_tcr / 2, size = contacts, color=contacts)) +
  geom_point(aes(x=mean_contacts_pos - len_tcr / 2), color="red", shape =3, size=3) +
  ylab("CDR3 length") +
  scale_x_continuous(name = "Position, relative to region center", limits=c(-7,7)) +
  scale_color_gradient(name = "Mean #contacts", low="#7fcdbb", high="#2c7fb8") +
  scale_size(guide=F)+
  facet_grid(tcr_chain ~ mhc_type) +
  theme_bw()
```



Features of contacting amino acids

This section of the manuscript deals with amino acid features of TCR and antigen contact residues, such as physical properties and pairing preference. Positional information is not used, however, we restrict our analysis to CDR3 region.

Physical properties of contacting TCR amino acids

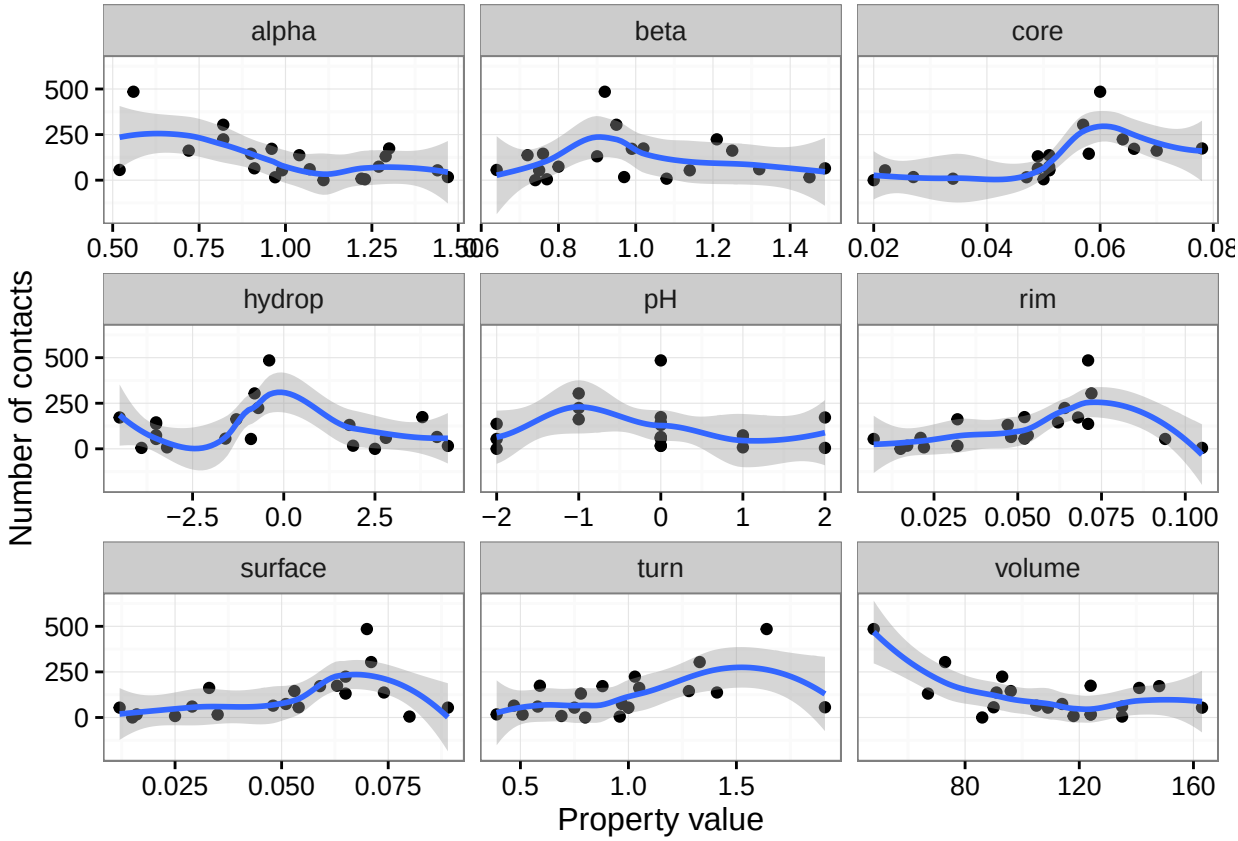
The list of structural and basic physical properties of amino acids was taken from Elhanati et al 2015 (FIG. 13).

```
aa_prop <- read.table("aa_properties.txt", header=T)

df.c <- subset(df, distance <= DIST_THRESHOLD & tcr_region == "CDR3")
df.c$tmp <- 1

# make sure all aa combinations here
tmp <- data.frame(expand.grid(aa_tcr = levels(aa_prop$aa), aa_antigen = levels(aa_prop$aa)))
tmp$count <- 0
df.c <- ddply(df.c, .(aa_tcr, aa_antigen), summarize, count = sum(tmp))
df.c <- rbind(df.c, tmp)
df.c <- ddply(df.c, .(aa_tcr, aa_antigen), summarize, count = sum(count))
```

We have observed a clear trend between number of contacts and certain properties of CDR3 amino acids. Notably, the **core** property derived in Martin et al 2012 as the frequency of a given amino acid in the core

region of protein-protein interacting interfaces had the best correlation with the number of contacts.

```
df.cc <- ddply(df.c, .(aa_tcr), summarize, count = sum(count))
colnames(df.cc) <- c("aa", "contacts")

df.cc2 <- merge(df.cc, subset(aa_prop, !(property %in% c("charge", "polar"))), all=T)

ggplot(df.cc2, aes(x=value,y=contacts)) +
  geom_point() + geom_smooth() + ylab("Number of contacts") + xlab("Property value") +
  facet_wrap(~property, scales="free_x") + theme_bw()
```



Spearman correlation coefficients for each of these properties are given below.

```
spearmentt <- function(r,n) r*sqrt((n-2)/(1-r^2))
calcpval <- function(r) {
  pval <- pt(spearmentt(r, 20), 20)
  min(pval, 1-pval)
}

df.cor <- ddply(df.cc2, .(property), summarize, r = cor(contacts, value, method="spearman"))
df.cor$pvalue <- sapply(df.cor$r, function(r) calcpval(r))

print(df.cor)
```

```
##   property           r       pvalue
## 1    alpha -0.47705041 1.607714e-02
```

```
## 2     beta  0.10755924 3.255925e-01
## 3     core  0.83777995 1.202374e-06
## 4   hydrop -0.04039283 4.327722e-01
## 5       pH -0.18314810 2.192830e-01
## 6      rim  0.44804230 2.306291e-02
## 7  surface  0.44168548 2.486421e-02
## 8     turn  0.47837536 1.580293e-02
## 9   volume -0.27505683 1.194846e-01
```

Pairwise interaction preferences

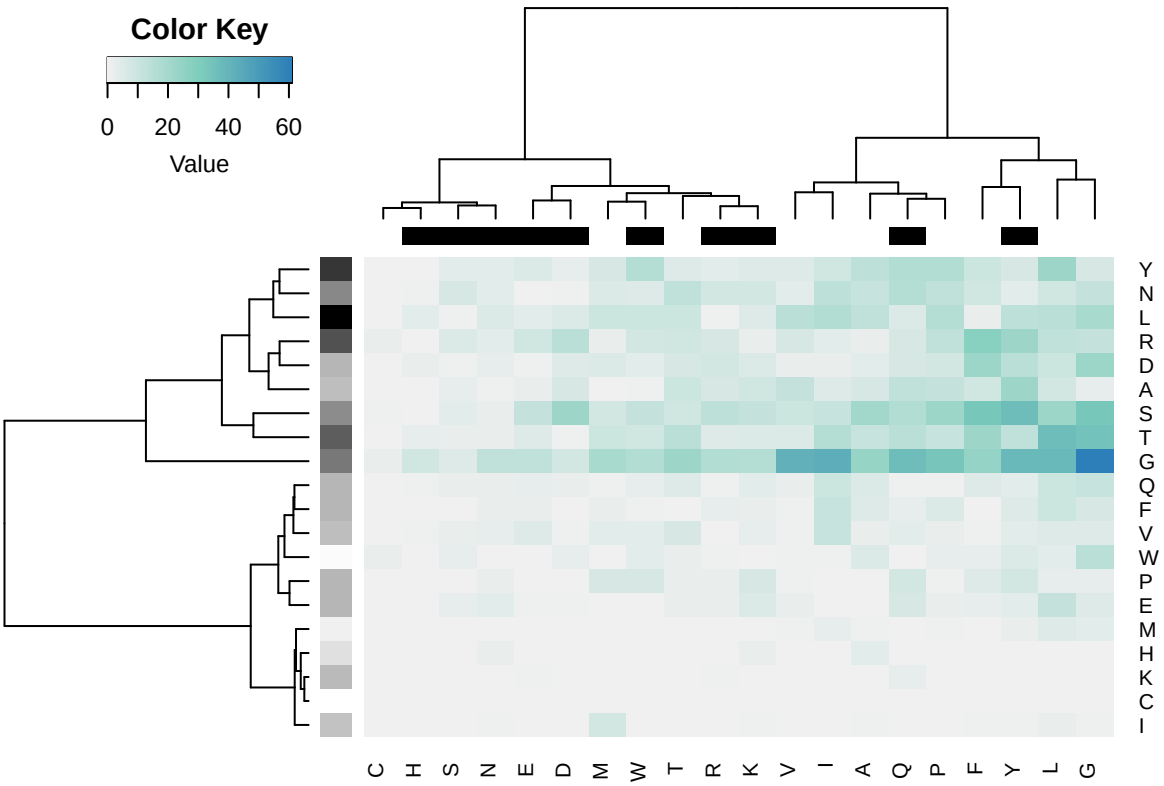
Next, we've constructed and analyzed the CDR3-antigen amino acid contact matrix. The matrix is shown as a heatmap below, with row and column dendrograms constructed using hierarchical clustering (Ward's method). Note that here *rows* and *columns* corresponding to CDR3 and antigen amino acids respectively.

The analysis results demonstrate clear clustering of CDR3 amino acids based on the **core** property, as shown by left color panel (*white* and *black* correspond to low and high **core** property values respectively). Notably, antigen amino acids were also nicely separated based on amino acid polarity, as shown top color panel (*black* corresponds to **polar** amino acids).

TODO Compare with MJ matrix

```
df.cc <- dcast(df.c, aa_tcr ~ aa_antigen, fun.aggregate = mean, value.var="count")

contact_mat <- as.matrix(df.cc[,2:ncol(df.cc)])
rownames(contact_mat) <- df.cc$aa_tcr

contact_mat[is.na(contact_mat)] <- 0

colgen <- function(aas, p, low, mid, hi) {
  panel <- colorpanel(100, low, mid, hi)
  df.1 <- merge(data.frame(aa=aas), subset(aa_prop, property == p))
  df.1$value <- as.integer((df.1$value - min(df.1$value)) / (max(df.1$value) - min(df.1$value)) * 99 + 1
  return(panel[df.1$value])
}

heatmap.2(contact_mat, col=colorpanel(100, "#f0f0f0", "#7fcdbb", "#2c7fb8"),
          hclustfun = function(d) hclust(d, method="ward"),
          ColSideColors = colgen(colnames(contact_mat), "polar", "white", "white", "black"),
          RowSideColors = colgen(rownames(contact_mat), "core", "white", "grey", "black"),
          density.info = "none", trace="none")
```

```
## The "ward" method has been renamed to "ward.D"; note new "ward.D2"
## The "ward" method has been renamed to "ward.D"; note new "ward.D2"
```



Raw number of contacts for CDR3 residues:

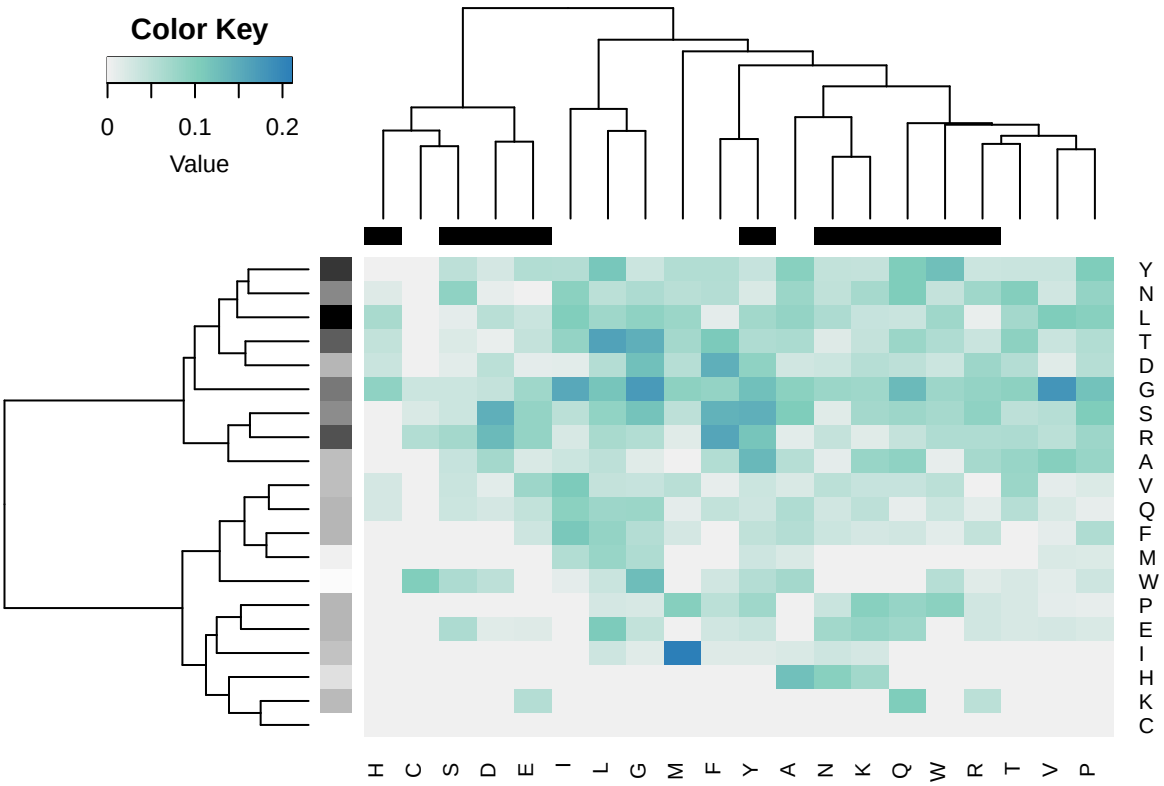
```
rowSums(contact_mat)
```

```
##   A   C   D   E   F   G   H   I   K   L   M   N   P   Q   R   S   T   V
## 131   0 137  54  60 485   8  16   5 174  17 145  56  74 172 304 224  65
##   W   Y
##  54 162
```

Normalize contact matrix:

```
contact_mat_norm <- contact_mat
rSum <- rowSums(contact_mat_norm)
cSum <- colSums(contact_mat_norm)
contact_mat_norm<-sweep(contact_mat_norm, 1, sqrt(rSum), `/`)
contact_mat_norm<-sweep(contact_mat_norm, 2, sqrt(cSum), `/`)
contact_mat_norm[is.na(contact_mat_norm)]<-0

heatmap.2(contact_mat_norm, col=colorpanel(100, "#f0f0f0", "#7fcdbb", "#2c7fb8"),
          hclustfun = function(d) hclust(d, method="ward"),
          ColSideColors = colgen(colnames(contact_mat), "polar", "white", "white", "black"),
          RowSideColors = colgen(rownames(contact_mat), "core", "white", "grey", "black"),
          density.info = "none", trace="none")
```

```
## The "ward" method has been renamed to "ward.D"; note new "ward.D2"
## The "ward" method has been renamed to "ward.D"; note new "ward.D2"
```



Normalized number of contacts for CDR3 residues:

```
rowSums(contact_mat_norm)
```

```
##         A         C         D         E         F         G         H
## 0.9963461 0.0000000 0.9924522 0.6492570 0.6367620 1.9566866 0.2913702
##         I         K         L         M         N         P         Q
## 0.3769470 0.2113282 1.1796838 0.2959306 1.0805276 0.6617497 0.7505089
##         R         S         T         V         W         Y
## 1.2082606 1.5127824 1.2368154 0.7512995 0.6843378 1.1163172
```

Application of findings to HIV escape epitope data

Finally, we have applied our findings to study escape epitope variants from NIAID HIV databases database. The database was filtered to retain only variants with a single amino acid substitution. Escape variants were selected according to **E** (documented escape) and **NSF** (non-susceptible form) flags, and were further partitioned into three groups:

- **hla** variants that abrogate MHC binding (**DHB** flag, diminished HLA binding or increased off-rate)
- **tcr** variants that do not bind or show decreased binding to TCR (**TCR** flag).
- **other** unclassified escape variants

As only 8,9,10 and 11-mer epitopes had reported **tcr** escape variants, we've limited our analysis to this group.

```
df.epi <- read.csv("ctl_variant.csv", header=T)

df.epi <- df.epi[grep("^[ACDEFGHIKLMNPQRSTVWY]\\d+[ACDEFGHIKLMNPQRSTVWY]$", df.epi$Mutation_epitope),]
```

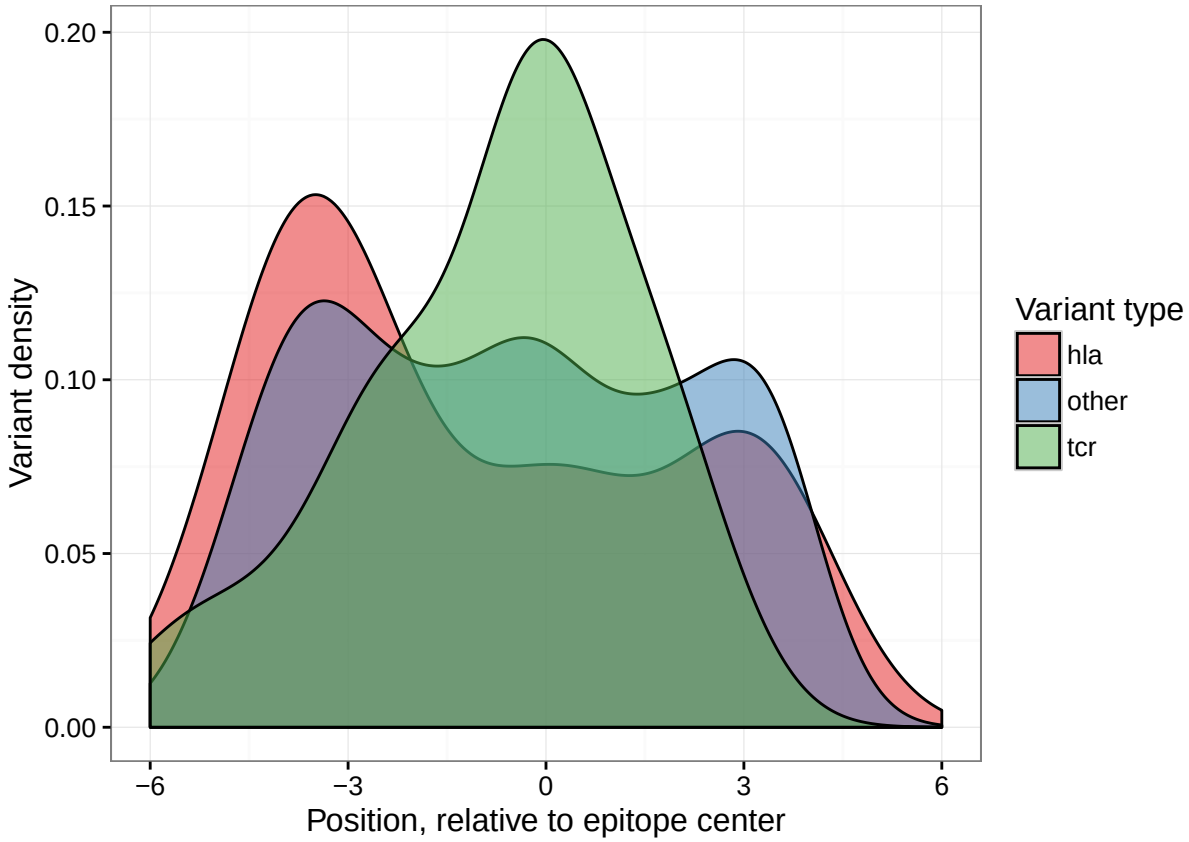
```
df.epi <- df.epi[grep("^[ACDEFGHIKLMNPQRSTVWY]+$", df.epi$Epitope),]
len <- function(x) nchar(as.character(x))

df.epi$codes <- sapply(df.epi$Mutation_Type_Code, function(x) strsplit(as.character(x),", "))

isescape <- function(x) "E" %in% x || "LE" %in% x || "IE" %in% x || "NSF" %in% x

df.epi$type <- sapply(df.epi$codes,
                      function(x) ifelse("TCR" %in% x, "tcr",
                                  ifelse("DHB" %in% x, "hla",
                                  ifelse(isescape(x), "other",
                                         "non-escape"
                                         ))))

df.epi <- subset(df.epi, type != "non-escape")

df.epi <- data.frame(seq = df.epi$Epitope,
                     aa_from = as.character(sapply(df.epi$Mutation_epitope,
                                                   function(x) substr(x,1,1))),
                     aa_to = as.character(sapply(df.epi$Mutation_epitope,
                                                 function(x) substr(x,len(x),len(x)))),
                     pos = sapply(df.epi$Mutation_epitope,
                                  function(x) as.integer(substr(x,2,len(x)-1))),
                     type = as.factor(df.epi$type),
                     elen = sapply(df.epi$Epitope,
                                   function(x) len(x))
                     )

df.epi <- unique(df.epi)
df.epi <- subset(df.epi, elen %in% c(8, 9, 10, 11))

df.epi$pos_adj <- apply(df.epi, 1, function(x) as.numeric(x[4])-(as.numeric(x[6]) / 2 + 1))

summary(df.epi)
```

```
##          seq         aa_from        aa_to          pos            type
##  SLYNTVATL : 23   K      : 49   A      : 40   Min.   : 1.000   hla  : 63
##  FLKEKGGL  : 15   V      : 39   R      : 39   1st Qu.: 2.250   other:342
##  KRWIILGLNK: 11   T      : 38   V      : 36   Median : 5.000   tcr  : 17
##  KEKGGLEGL : 10   R      : 35   K      : 34   Mean   : 4.964
##  AVDLSHFLK :  8   L      : 33   T      : 31   3rd Qu.: 7.000
##  HPVHAGPVA :  7   E      : 32   L      : 28   Max.   :11.000
##  (Other)   :348   (Other):196   (Other):214
##       elen           pos_adj
##  Min.   : 8.000   Min.   :-5.5000
##  1st Qu.: 9.000   1st Qu.:-3.0000
##  Median : 9.000   Median :-0.5000
##  Mean   : 9.201   Mean   :-0.6363
##  3rd Qu.:10.000   3rd Qu.: 1.5000
##  Max.   :11.000   Max.   : 4.5000
##
```

As expected, **tcr**-mediate escape mutations are clustered in the central part of antigen (showing a two-peak

picture resembling TCR contact positioning plot above), while **hla**-mediate escape mutations are clustered on **N** and **C** termini.

```
ggplot(df.epi, aes(x=pos_adj, fill=type)) +
  geom_density(alpha=0.5) +
  xlab("Position, relative to epitope center") +
  ylab("Variant density") +
  scale_fill_brewer(name = "Variant type", palette = "Set1") +
  scale_x_continuous(limits=c(-6,6)) + theme_bw()
```



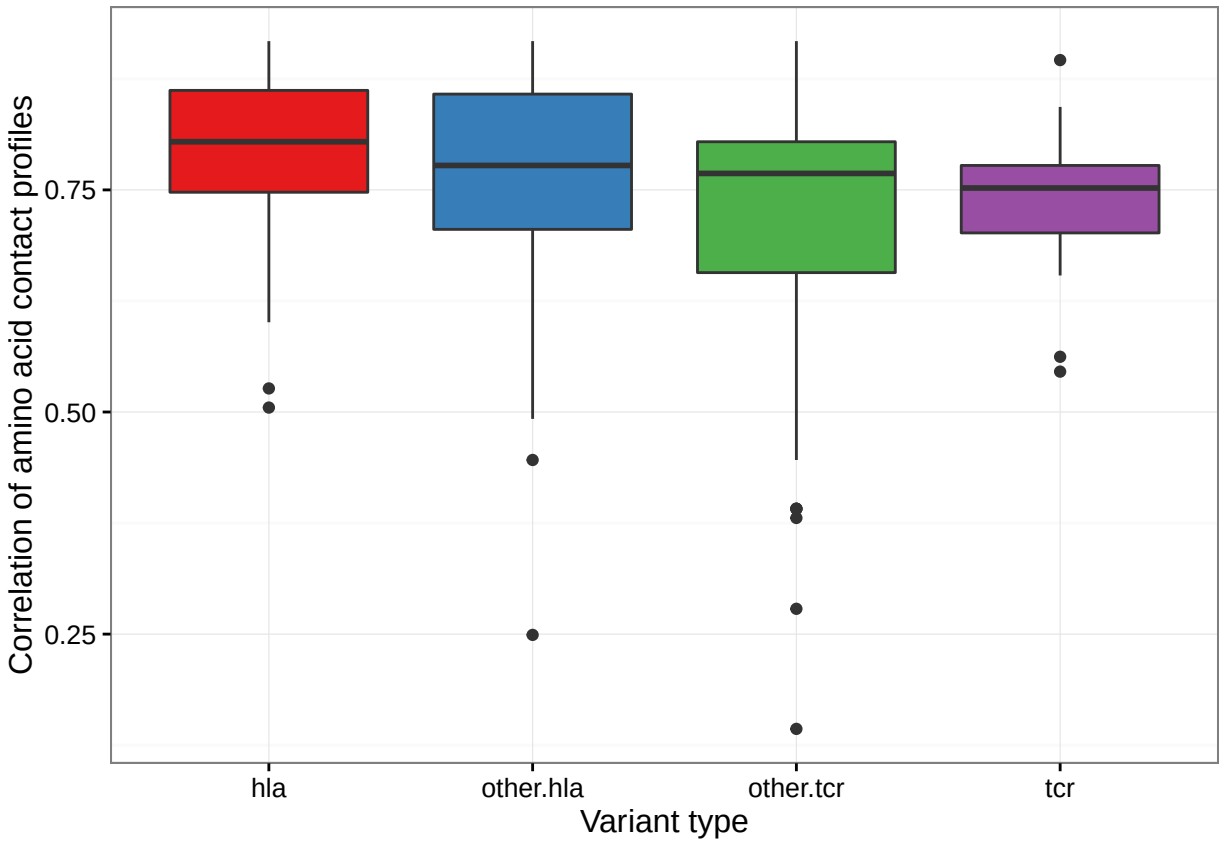
Next, we have tested the hypothesis that **tcr**-mediated escape variants are biased in the direction of mutations that substantially change TCR recognition profile (*TRP*, i.e. a column from the contact amino acid matrix described above). We have therefore computed correlation coefficients between *TRPs* of original and substituted amino acids and compared them between **hla**, **tcr** and **other** mutation groups. We have also taken an advantage of positioning data and partitioned the **other** group into **other.tcr** and **other.hla** based on whether the variant was in *[-2, 2]* region in respect to antigen center or not.

```
calc_dist <- function(row) cor(contact_mat[,row[2]], contact_mat[,row[3]])

df.epi$type2 <- as.factor(apply(df.epi, 1, function(x) ifelse(x[5] == "other",
                                                              ifelse(abs(as.numeric(x[7])) <= 2, "other.tcr", "othe
                                                              x[5])))

summary(df.epi)
```

```
##          seq         aa_from        aa_to          pos            type
```

```
##  SLYNTVATL : 23   K      : 49   A      : 40   Min.   : 1.000   hla  : 63
##  FLKEKGGL  : 15   V      : 39   R      : 39   1st Qu.: 2.250   other:342
##  KRWIILGLNK: 11   T      : 38   V      : 36   Median : 5.000   tcr  : 17
##  KEKGGLEGL : 10   R      : 35   K      : 34   Mean   : 4.964
##  AVDLSHFLK :  8   L      : 33   T      : 31   3rd Qu.: 7.000
##  HPVHAGPVA :  7   E      : 32   L      : 28   Max.   :11.000
##  (Other)   :348   (Other):196   (Other):214
##       elen           pos_adj              type2
##  Min.   : 8.000   Min.   :-5.5000   hla      : 63
##  1st Qu.: 9.000   1st Qu.:-3.0000   other.hla:190
##  Median : 9.000   Median :-0.5000   other.tcr:152
##  Mean   : 9.201   Mean   :-0.6363   tcr      : 17
##  3rd Qu.:10.000   3rd Qu.: 1.5000
##  Max.   :11.000   Max.   : 4.5000
##
```

```
df.epi$dist <- apply(df.epi, 1, calc_dist)

ggplot(df.epi, aes(x = type2, fill=type2, group=type2, y=dist)) +
  geom_boxplot() + xlab("Variant type") +
  ylab("Correlation of amino acid contact profiles") +
  scale_fill_brewer(guide=F, palette = "Set1") +
  theme_bw()
```



There was indeed a clear trend showing that **hla**-mediated escape mutations resulted in similar *TRPs*, while **tcr**-mediated mutations had least similar *TRPs*. Statistical details are given below.

```
kruskal.test(dist ~ type2, df.epi)
```

```
##
##  Kruskal-Wallis rank sum test
##
## data:  dist by type2
## Kruskal-Wallis chi-squared = 15.962, df = 3, p-value = 0.001154
```

```
wilcox.test(subset(df.epi, type2 == "other.hla")$dist, subset(df.epi, type2 == "other.tcr")$dist)
```

```
##
##  Wilcoxon rank sum test with continuity correction
##
## data:  subset(df.epi, type2 == "other.hla")$dist and subset(df.epi, type2 == "other.tcr")$dist
## W = 16952, p-value = 0.005678
## alternative hypothesis: true location shift is not equal to 0
```

```
wilcox.test(subset(df.epi, type2 == "hla")$dist, subset(df.epi, type2 == "tcr")$dist)
```

```
##
##  Wilcoxon rank sum test with continuity correction
##
## data:  subset(df.epi, type2 == "hla")$dist and subset(df.epi, type2 == "tcr")$dist
## W = 728, p-value = 0.02377
## alternative hypothesis: true location shift is not equal to 0
```

The analysis described here can be further extended to immunogenic and non-immunogenic mutated self-peptides from cancer peptidomes once a database of cancer epitopes similar to NIAID HIV databases will become available.