ch VDJdb sandbox-2

Load data from Aleksey (substitution and msubstitution matrices made from pairwise alignments for sequences from *Homo Sapiens*). There are two types of substitution and msubstitution matrices. For first type both matches and mismathces in pairwise alignments were used, while for second (with "_diagonal"added at the end of the name) only matches were used.

Code from Mikhail ("VDJdb sandbox-1") has been used to make function *"info_from_matrices"*.

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

```
##
## Attaching package: 'gplots'
```

```
## The following object is masked from 'package:stats':
##
##     lowess
```

```
library(plyr)
library(MASS)
library(stringr)

read_subst_matrix <- function(inp, sub, ins, del, tot) {
  suffix <- paste(sub, ins, del, tot, sep = "_")

  .df <- read.csv(paste(inp, "/msubstitution_matrix_out_teach_", suffix, ".csv", sep = ""),
                  stringsAsFactors = F)
  .df <- melt(.df)
  .df$type <- "outer"

  .df.1 <- read.csv(paste(inp, "/substitution_matrix_out_teach_", suffix, ".csv", sep = ""),
                    stringsAsFactors = F)
  .df.1 <- melt(.df.1)
  .df.1$type <- "inner"

  .df <- rbind(.df, .df.1)

  colnames(.df) <- c("from", "to", "count", "type")

  .df$dataset <- suffix

  subset(.df, from != "C" & to != "C")
}

df <- data.frame()

for (s in 1:6) {
  df <- rbind(df, read_subst_matrix('seqdata/HomoSapiens/occurencematrices', s, 0, 0, s))
}
```

```
## Using X as id variables
```

```
## Using X as id variables
```

```
## Using X as id variables
## Using X as id variables
## Using X as id variables
## Using X as id variables
## Using X as id variables
## Using X as id variables
## Using X as id variables
## Using X as id variables
## Using X as id variables
## Using X as id variables
```

Here $C_{ij}^{(s)}$ and $C_{ij}^{(d)}$ amino acid substitution counts for CDR3 alignments with same (s) and distinct (d) antigens. $C_{ii}^{(s,d)}$ represents the number of times amino acid gets unchanged.

Protect against $\log 0$

$$C_{ij}^{(s,d)} \leftarrow C_{ij}^{(s,d)} + 1$$

Probability of not changing amino acid

$$P_{ii}^{(s,d)} = \frac{C_{ii}^{(s,d)}}{\sum_k C_{ik}^{(s,d)}}$$

Probability of $i \to j, i \neq j$ is

$$P_{ij}^{(s,d)} = (1 - P_{ii}^{(s,d)}) \frac{C_{ij}^{(s,d)}}{\sum_{k \neq i} C_{ik}^{(s,d)}}$$

Symmetrize as follows

$$P_{ij}^{(s,d)} = \frac{P_{ij}^{(s,d)}}{P_{ij}^{(s,d)} + P_{ji}^{(s,d)}} P_{ij}^{(s,d)} + \frac{P_{ji}^{(s,d)}}{P_{ij}^{(s,d)} + P_{ji}^{(s,d)}} P_{ji}^{(s,d)}$$

Compute odds ratios as

$$Q_{ij} = \log_{10} \frac{P_{ij}^{(s)}}{P_{ij}^{(d)}}$$

Perform calculations, cluster amino acids based on substitution odds ratios

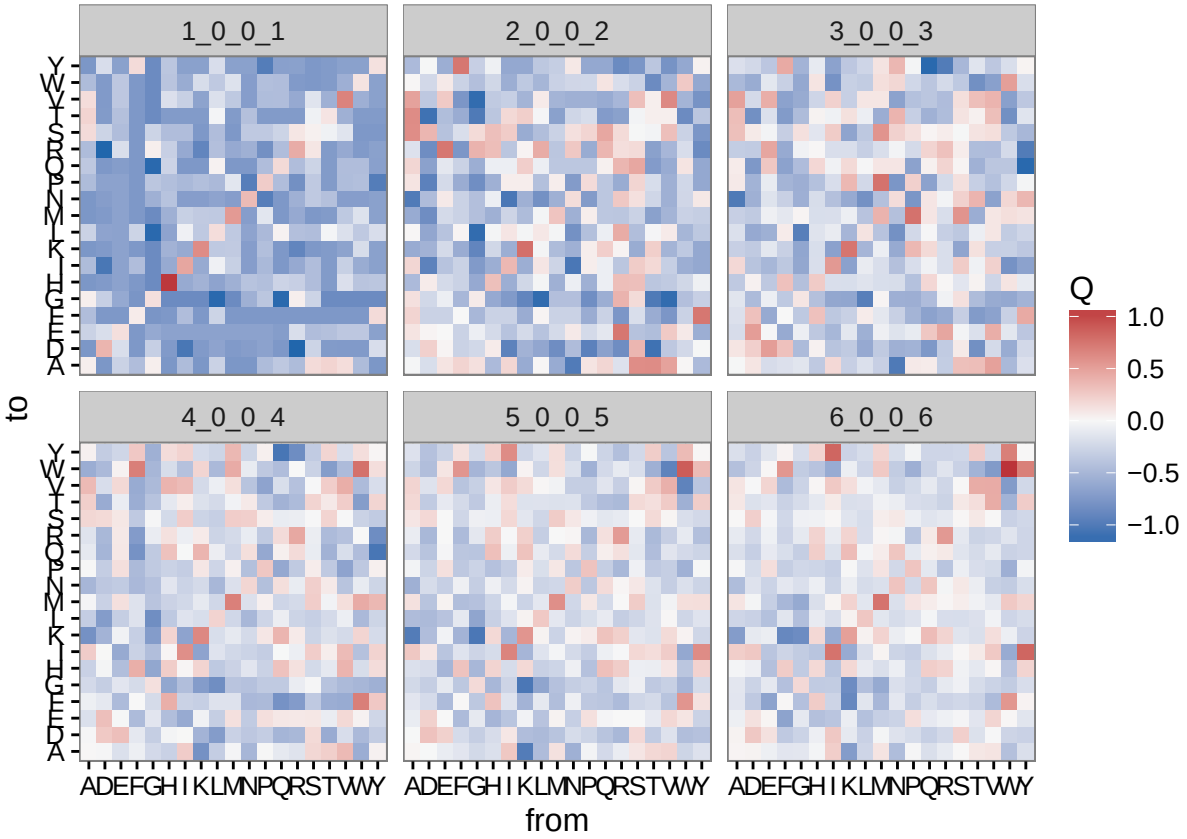
```
df$count <- df$count + 1

# Pii
df <- ddply(df, .(from, type, dataset), transform, P = count / sum(count))

# Pij
df <- ddply(df, .(from, type, dataset), transform, P = ifelse(from == to, P,
                          (1 - P[which(from == to)]) * count / sum(count[which(from != to)])))

df.inv <- data.frame(from = df$to, to = df$from, type = df$type, count = df$count, P = df$P, dataset = d
df <- merge(df, df.inv, by = c("from", "to", "type", "count", "dataset"))
```

```
df$P <- mapply(function(x, y) min(x, y), df$P.x, df$P.y)

df <- ddply(df, .(from, to, dataset), summarize, Q = log10(P[which(type=="inner")] / P[which(type=="out

df$to <- as.character(df$to)
```

Comparing different search scope settings

Compare different substitution counts. First, heatmaps (not very informative)

```
# Heatmaps
ggplot(df, aes(from, to, fill = Q)) +
  geom_tile() +
  facet_wrap(~dataset) +
  scale_fill_gradient2(midpoint = 0, low = "#2166ac", mid = "#f7f7f7", high = "#b2182b") +
  theme_bw()
```



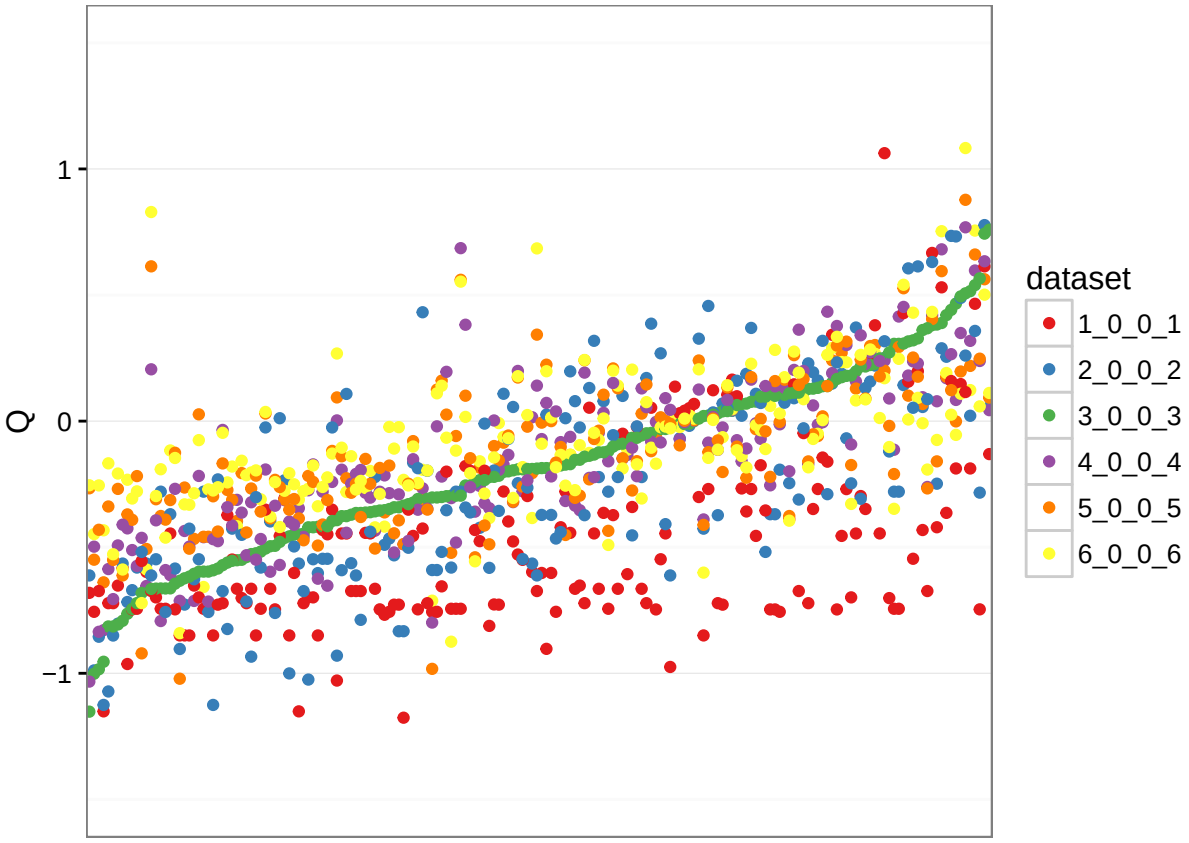
Write scores

```
write.table(df, "scores.txt", quote=F, sep="\t", row.names=F)
```

Correlation between scores

```
df.1 <- subset(df, from >= to) # remove duplicates
df.1$pattern <- interaction(df.1$from, df.1$to)
df.1$pattern <- factor(df.1$pattern, levels = arrange(subset(df.1, dataset == "3_0_0_3"), Q)$pattern)
```

```
ggplot(df.1, aes(pattern, Q, color = dataset)) +
  geom_point() + scale_color_brewer(palette = "Set1") +
  scale_y_continuous(limits=c(-1.5, 1.5)) +
  scale_x_discrete(breaks = c()) +
  theme_bw() +
  theme(axis.title.x=element_blank(),
        axis.text.x=element_blank(),
        axis.ticks.x=element_blank())
```



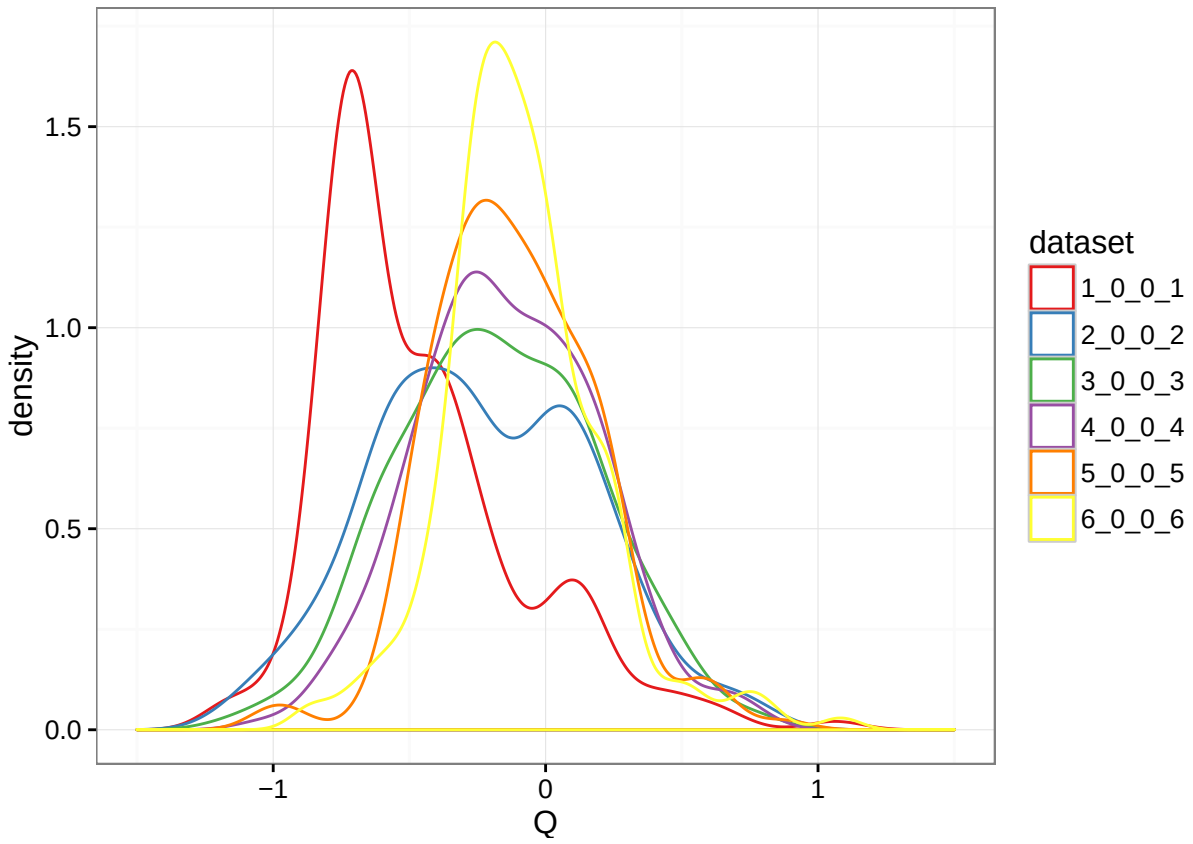

```
df.cor <- data.frame()

for (d1 in unique(df.1$dataset)) {
  for (d2 in unique(df.1$dataset)) {
    if (d1 != d2) {
      df.cor <- rbind(df.cor, data.frame(
        d1 = d1, d2 = d2, r = cor(arrange(subset(df.1, dataset==d1), pattern)$Q,
                arrange(subset(df.1, dataset==d2), pattern)$Q,
                method = "spearman"))
       )
    }
  }
}

print(df.cor)
```

```
##         d1      d2         r
## 1  1_0_0_1 2_0_0_2 0.6585702
```

```
## 2  1_0_0_1 3_0_0_3 0.5340793
## 3  1_0_0_1 4_0_0_4 0.5055900
## 4  1_0_0_1 5_0_0_5 0.4829618
## 5  1_0_0_1 6_0_0_6 0.4549245
## 6  2_0_0_2 1_0_0_1 0.6585702
## 7  2_0_0_2 3_0_0_3 0.7506390
## 8  2_0_0_2 4_0_0_4 0.5979946
## 9  2_0_0_2 5_0_0_5 0.5774168
## 10 2_0_0_2 6_0_0_6 0.5024605
## 11 3_0_0_3 1_0_0_1 0.5340793
## 12 3_0_0_3 2_0_0_2 0.7506390
## 13 3_0_0_3 4_0_0_4 0.8125531
## 14 3_0_0_3 5_0_0_5 0.7005970
## 15 3_0_0_3 6_0_0_6 0.6440241
## 16 4_0_0_4 1_0_0_1 0.5055900
## 17 4_0_0_4 2_0_0_2 0.5979946
## 18 4_0_0_4 3_0_0_3 0.8125531
## 19 4_0_0_4 5_0_0_5 0.9006153
## 20 4_0_0_4 6_0_0_6 0.8245473
## 21 5_0_0_5 1_0_0_1 0.4829618
## 22 5_0_0_5 2_0_0_2 0.5774168
## 23 5_0_0_5 3_0_0_3 0.7005970
## 24 5_0_0_5 4_0_0_4 0.9006153
## 25 5_0_0_5 6_0_0_6 0.9045860
## 26 6_0_0_6 1_0_0_1 0.4549245
## 27 6_0_0_6 2_0_0_2 0.5024605
## 28 6_0_0_6 3_0_0_3 0.6440241
## 29 6_0_0_6 4_0_0_4 0.8245473
## 30 6_0_0_6 5_0_0_5 0.9045860
```

Distribution of Q-scores, $1, 0, 0, 1$ seems very conservative

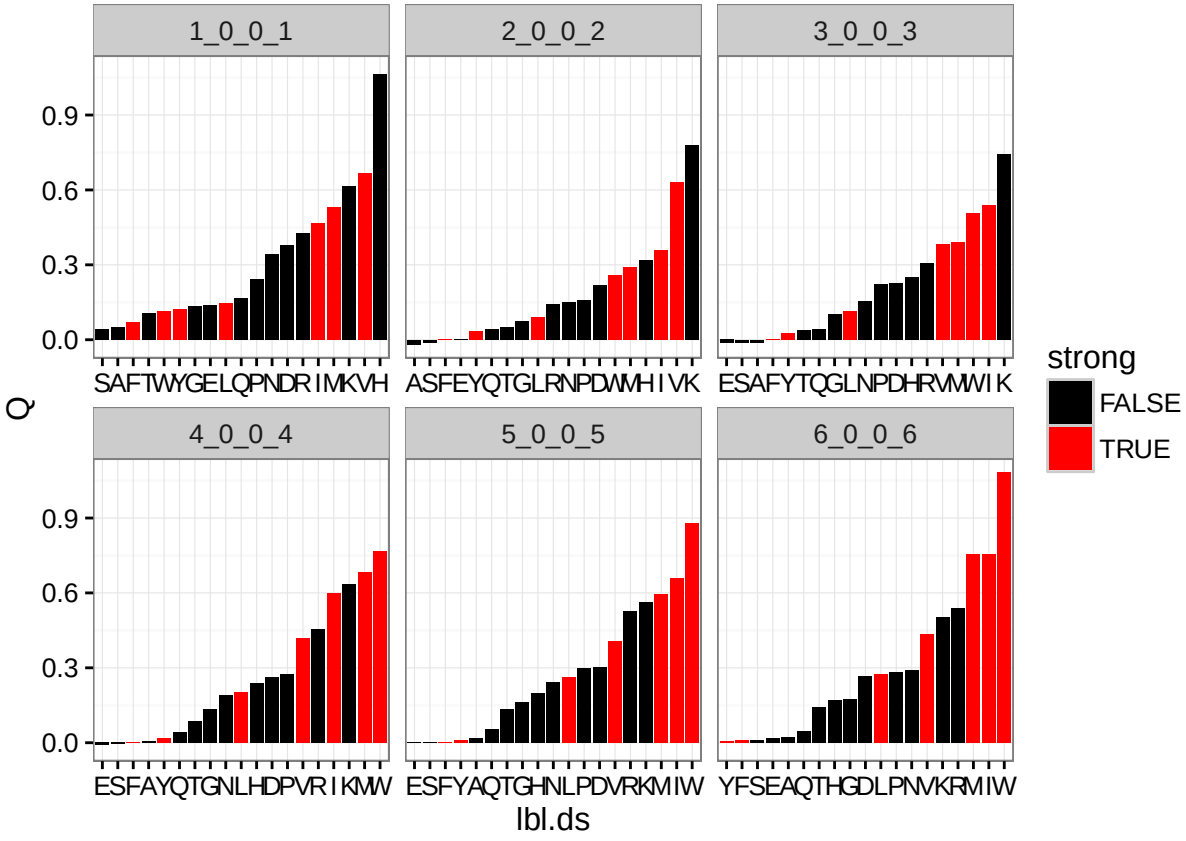
```
ggplot(df.1, aes(x=Q, color=dataset)) +
  geom_density() + scale_x_continuous(limits=c(-1.5, 1.5)) +
  scale_color_brewer(palette = "Set1") + theme_bw()
```



Non-replacement probabilities

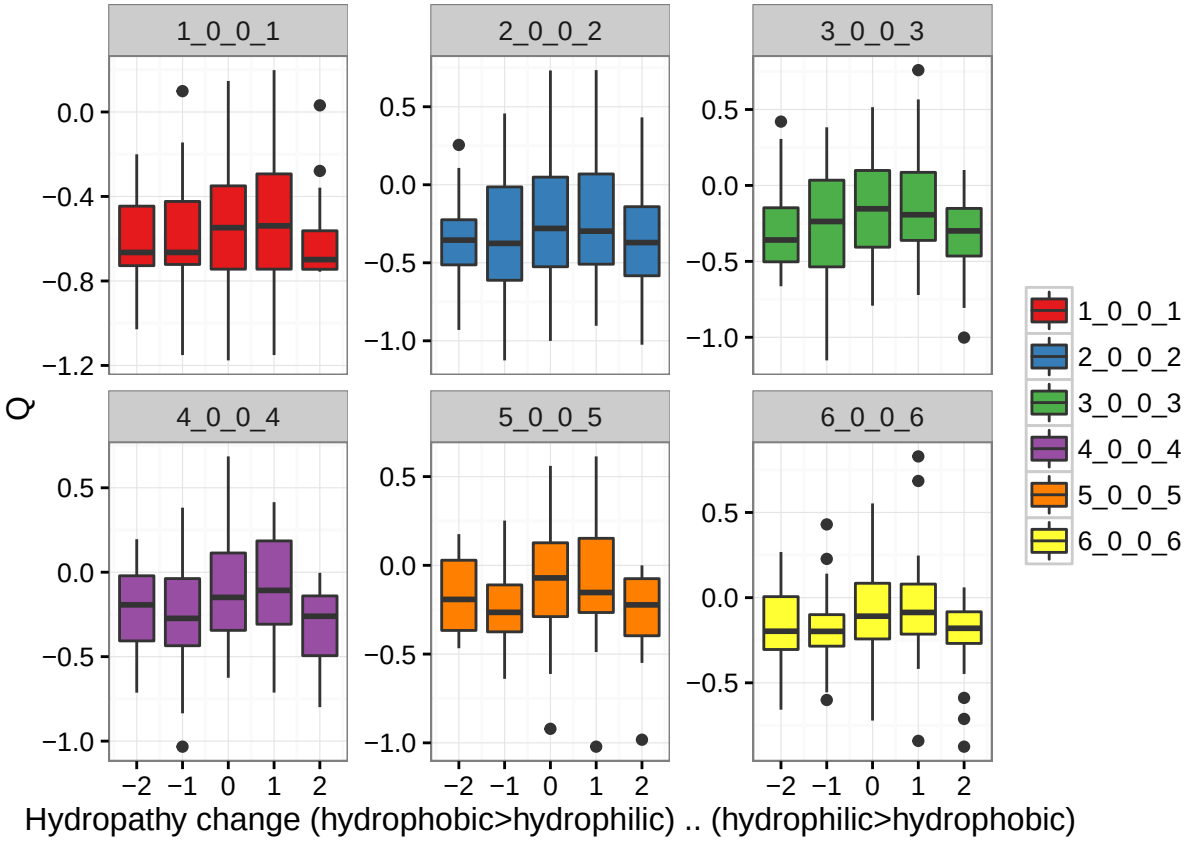
```
df.diag <- subset(df.1, from == to)

# Useful faceted axis reordering magick
df.diag$strong <- df.diag$from %in% c("F", "I", "L", "M", "V", "W", "Y")
df.diag$lbl.ds <- paste(df.diag$from, df.diag$dataset, sep = ".")
df.diag$lbl.ds <- reorder(df.diag$lbl.ds, df.diag$Q, )

ggplot(df.diag, aes(x = lbl.ds, y = Q, fill=strong)) +
  geom_bar(stat="identity") +
  scale_fill_manual(values = c("black", "red")) +
  facet_wrap(~dataset, scales="free_x") +
  scale_x_discrete(labels=function(x) sapply(strsplit(x,"[.]"),"[",1)) +
  theme_bw()
```

```
## Warning: Stacking not well defined when ymin != 0
```



Replacement probabilities and hydropathy change

```
aa.classes <- data.frame(aa = strsplit("I,V,L,F,C,M,A,W,G,T,S,Y,P,H,N,D,Q,E,K,R", ",")[[1]],
                         hydrop = c(rep("hydrophobic", 8), rep("neutral", 6),
                                    rep("hydrophilic", 6)))

aa.classes$hydrop <- factor(aa.classes$hydrop, c("hydrophobic", "neutral", "hydrophilic"))

df.2 <- subset(df.1, from != to)

df.2 <- merge(df.2, aa.classes, by.x = "from", by.y = "aa")
df.2 <- merge(df.2, aa.classes, by.x = "to", by.y = "aa")

hydrop_toint <- function(x) {
  ifelse(x == "hydrophobic", 1, ifelse(x == "neutral", 0, -1))
}

df.2$hydrop.change <- with(df.2, hydrop_toint(hydrop.y) - hydrop_toint(hydrop.x))

ggplot(df.2, aes(x=hydrop.change, group = interaction(hydrop.change, dataset), y=Q,
                 fill=dataset)) +
  geom_boxplot() +
  xlab("Hydropathy change (hydrophobic>hydrophilic) .. (hydrophilic>hydrophobic)") +
  ylab("Q") + scale_fill_brewer("", palette = "Set1") +
  facet_wrap(~dataset, scales="free_y") +
  theme_bw()
```




```
a <- aov(Q ~ I(abs(hydrop.change)) * dataset, df.2)
summary(a)
```

```
##                                 Df Sum Sq Mean Sq F value   Pr(>F)
## I(abs(hydrop.change))            1   1.93   1.932  20.777 5.79e-06 ***
## dataset                          5  20.35   4.071  43.779  < 2e-16 ***
## I(abs(hydrop.change)):dataset    5   0.10   0.020   0.213    0.957
## Residuals                     1014  94.29   0.093
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

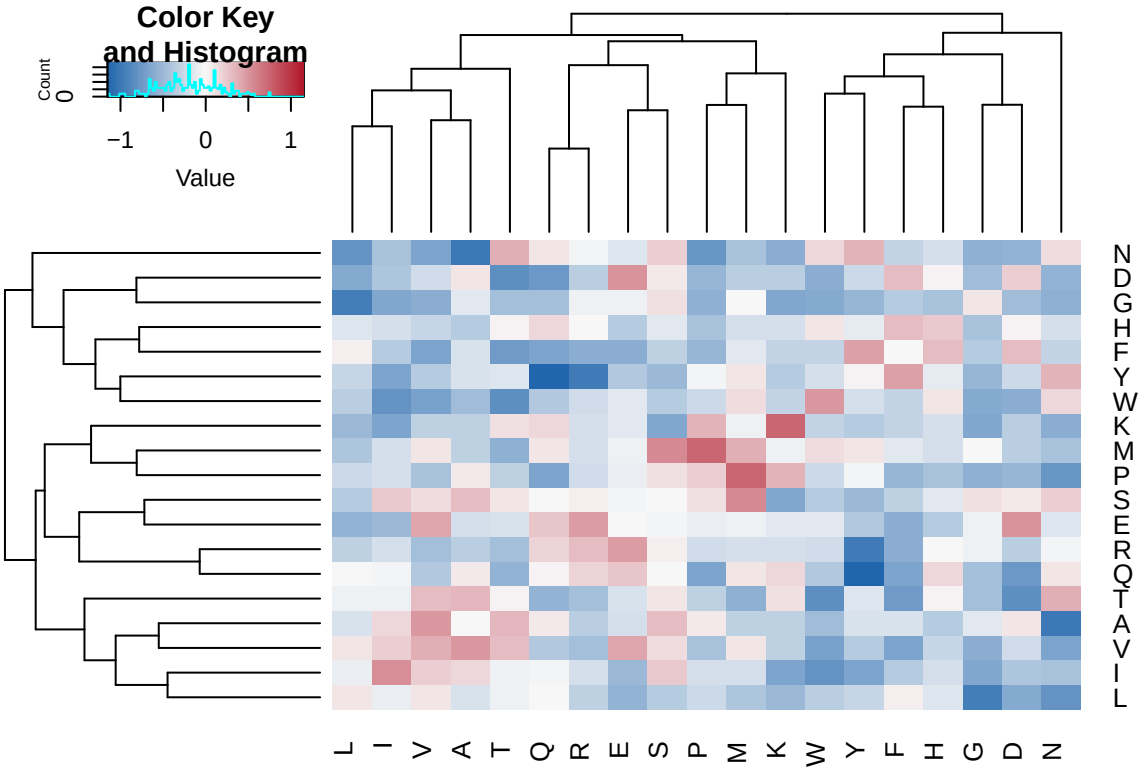
Obsolete

Cluster amino acids. Hereafter using $3,0,0,3$

```
df.m <- dcast(subset(df, dataset=="3_0_0_3"), from ~ to)
```

```
## Using Q as value column: use value.var to override.
```

```
rownames(df.m) <- df.m$from
df.m$from <- NULL
df.m <- data.matrix(df.m)

df.m <- df.m[order(rownames(df.m)), order(colnames(df.m))]

heatmap.2(df.m, scale="none", symm = T, trace = "none",
          col=colorpanel(100, "#2166ac", "#f7f7f7", "#b2182b"))
```

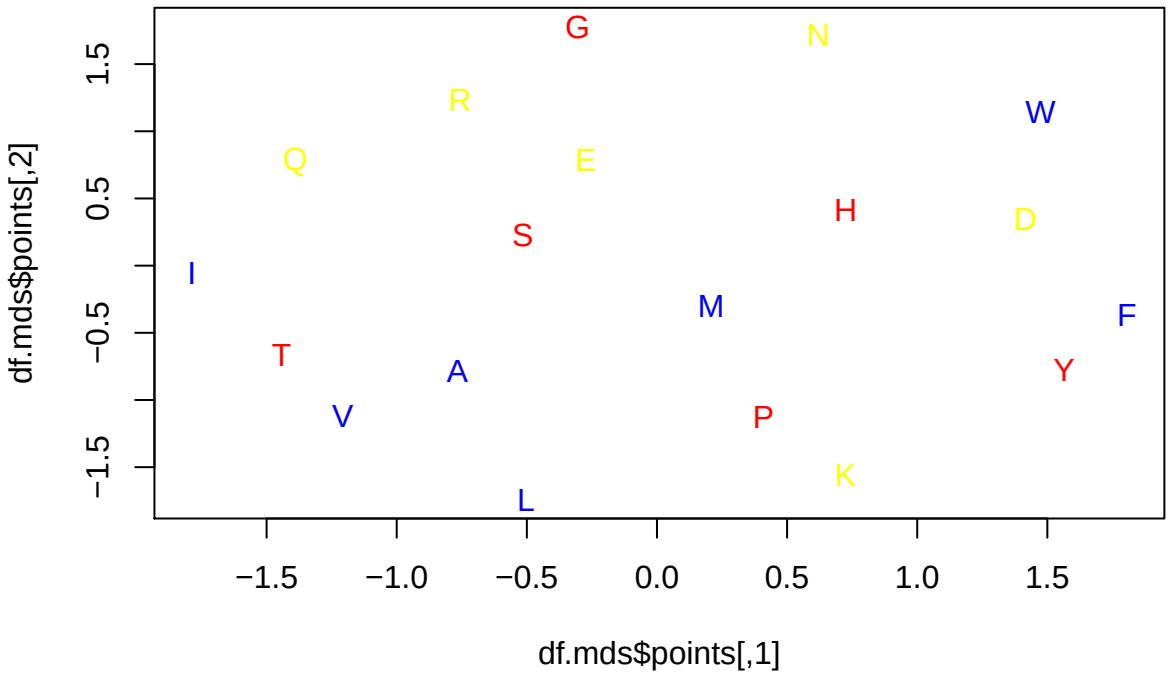


Some MDS analysis

```
# hydrophobic, neutral, hydrophilic
colors <- c(rep("blue", 8), rep("red", 6), rep("yellow", 6))
names(colors) <- strsplit("I V L F C M A W G T S Y P H N D Q E K R", " ")[[1]]

df.mds <- isoMDS(as.dist(-df.m + 2), k = 2)
```

```
## initial  value 34.894898
## iter   5 value 29.543927
## iter  10 value 28.371253
## final  value 28.053694
## converged
```

```
plot(df.mds$points, type = "n")
text(df.mds$points, labels = rownames(df.m), col = colors[rownames(df.m)])
```


G
N
R
W
Q
E
H
D
S
I
M
F
T
A
Y
V
P
K
L
1.5
0.5
−0.5
−1.5
df.mds$points[,2]
−1.5
−1.0
−0.5
0.0
0.5
1.0
1.5
df.mds$points[,1]