

T-cell repertoire annotation and motif discovery

A RepSeq data analysis tutorial in R

Mikhail Shugay

29 October 2018, 2nd SCAIR meeting

RepSeq sample annotation

Here is the layout of our experiment, datasets were selected from *Emerson et al. Nat Genet 2017*.

Samples:

(B35+)	HIP02877	A*26	A*33	B*14	B*35	CMV-
(CMV+)	HIP13994	A*02	A*02	B*07	B*44	CMV+

Controls:

(Control-1)	HIP03484	A*02	A*02	B*07	B*58	CMV-
(Control-2)	HIP03592	A*02	A*32	B*07	B*39	CMV-
(Control-3)	HIP04532	A*02	A*24	B*07	B*51	CMV-
(Control-4)	HIP04576	A*02	A*30	B*07	B*18	CMV-

Compute some basic statistics using VDJtools.

```
run_java("vdjtools",  
         "CalcBasicStats data/control.txt.gz data/CMV+.txt.gz data/B35+.txt.gz output/",  
         T)
```

Number of reads and clonotypes per sample:

```
df.stats <- fread("output/basicstats.txt")  
df.stats
```

```
##   sample_id metadata_blank   count diversity mean_frequency  
## 1:   control                . 10881045   913905  1.094206e-06  
## 2:    CMV+                  .  3819906   187639  5.329382e-06  
## 3:    B35+                  .   899992    63737  1.568947e-05  
##   geomean_frequency nc_diversity nc_frequency mean_cdr3nt_length  
## 1:    6.754538e-07           0    0.0000000         43.16485  
## 2:    2.593038e-06          33513    0.1621477         44.70757  
## 3:    7.764453e-06           0    0.0000000         43.34150  
##   mean_insert_size mean_ndn_size convergence  
## 1:         3.079917         11.25928    1.112407  
## 2:         4.183891         12.30924    1.036891  
## 3:         2.717946         10.46588    1.027992
```

Annotate samples using VDJmatch. The following arguments are used:

- `match` runs routine that matches samples against VDJdb
- `-S human` sets species
- `-R TRB` sets receptor chain
- `-O 1,0,1` sets the search scope - number of substitutions, indels and total number of mutations. Here we'll just allow a single substitution. Note that allowing indels can make results quite messy (need to use correct scoring with `-A` argument)
- `--min-epi-size 30` will select VDJdb epitopes that have at least 30 unique TCR records

```
run_java("vdjmatch",
  "match -S human -R TRB -O 1,0,1 --min-epi-size 30 data/control.txt.gz data/CMV+.txt.gz data/B35+.txt.gz",
  T)
```

Lets explore annotation results. Load and quality-filter VDJdb annotations

```
# Read in data
list("control", "CMV+", "B35+") %>%
  lapply(function(x)
    "output/vdjdbc.{x}.txt" %>%
    str_glue() %>%
    fread() %>%
    mutate(sample_id = x)) %>%
  rbindlist() %>%
  mutate(mhc.a = str_split_fixed(mhc.a, "[:,]", 2)[,1]) %>%
  group_by(cdr3aa, antigen.epitope, antigen.species,
    mhc.a, sample_id, vdjdbc.score, reference.id) %>%
  summarise(freq = sum(freq), count = sum(count)) %>%
  ungroup -> df.vdjdbc

df.vdjdbc %>%
  head
```

```
## # A tibble: 6 x 9
##   cdr3aa antigen.epitope antigen.species mhc.a sample_id vdjdbc.score
##   <chr>   <chr>           <chr>         <chr> <chr>         <int>
## 1 CAAAG~ GILGFVFTL      InfluenzaA    HLA~ control         0
## 2 CAAGG~ FLYNLLTRV      HomoSapiens   HLA~ B35+          0
## 3 CAAGG~ FLYNLLTRV      HomoSapiens   HLA~ control         0
## 4 CAAGG~ ELAGIGILTV     HomoSapiens   HLA~ control         0
## 5 CAAGL~ LLWNGPMAV     YellowFeverVir~ HLA~ control         1
## 6 CAAGR~ MLNIPSINV       CMV           HLA~ control         0
## # ... with 3 more variables: reference.id <chr>, freq <dbl>, count <int>
```

```
nrow(df.vdjdbc)
```

```
## [1] 115127
```

```
# Select unambiguous assignments
df.vdjdbc.good <- df.vdjdbc %>%
  select(cdr3aa, antigen.epitope, mhc.a, vdjdbc.score, reference.id) %>%
  unique %>%
  group_by(cdr3aa) %>%
  mutate(vdjdbc.score.max = max(vdjdbc.score)) %>%
  filter(vdjdbc.score == vdjdbc.score.max) %>%
  # In case of ties select the one with max # publications
  group_by(cdr3aa) %>%
  mutate(num.pub = str_count(reference.id, ","),
    num.pub.max = max(num.pub)) %>%
  filter(num.pub == num.pub.max) %>%
  # Remove all remaining ambiguous cases
  group_by(cdr3aa) %>%
  mutate(n.epitopes = length(unique(antigen.epitope))) %>%
  filter(n.epitopes == 1) %>%
  ungroup
```

```

# Apply filter
df.vdjdb <- df.vdjdb %>%
  merge(df.vdjdb.good)

# Some naming fixes
df.vdjdb <- df.vdjdb %>%
  mutate(
    epi.name = paste(
      substr(str_split_fixed(mhc.a, "[,:]", 2)[,1], 5, 10),
      substr(antigen.epitope, 1, 3)),
    antigen.species = ifelse(
      startsWith(antigen.species, "DENV"),
      "DengueVirus",
      antigen.species))

nrow(df.vdjdb)

```

```
## [1] 66736
```

```

# Split control
df.vdjdb.control <- df.vdjdb %>%
  filter(sample_id == "control")
df.vdjdb <- df.vdjdb %>%
  filter(sample_id != "control")

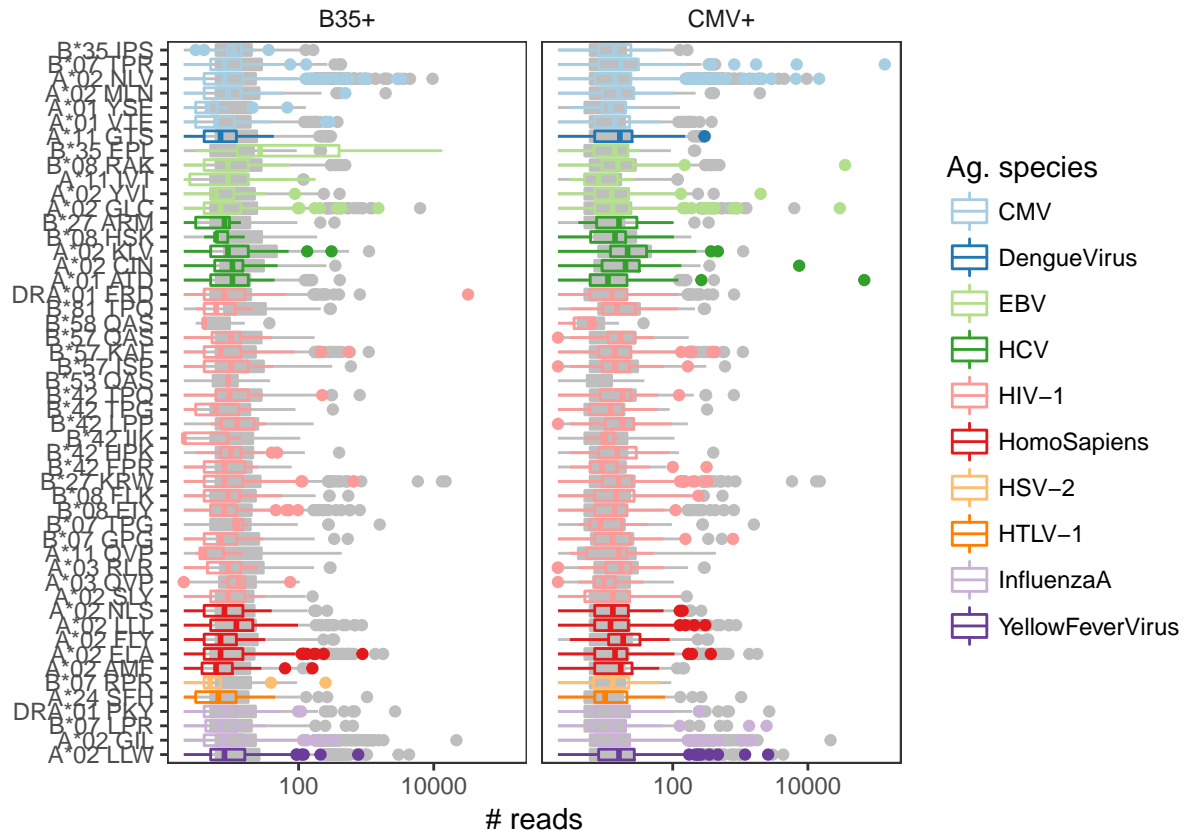
```

Plot all VDJdb annotations

```

df.vdjdb %>%
  ggplot(aes(x = fct_reorder2(epi.name,
                             freq,
                             as.integer(as.factor(antigen.species))),
             y = count,
             color = antigen.species)) +
  geom_boxplot(data = df.vdjdb.control %>% select(-sample_id),
              color = "grey", fill = "grey") +
  geom_boxplot(fill = NA) +
  coord_flip() +
  scale_y_log10("# reads") + xlab("") +
  scale_color_brewer("Ag. species", palette = "Paired") +
  facet_wrap(~sample_id) +
  theme_bw() +
  theme(panel.grid = element_blank(),
        strip.background = element_blank())

```

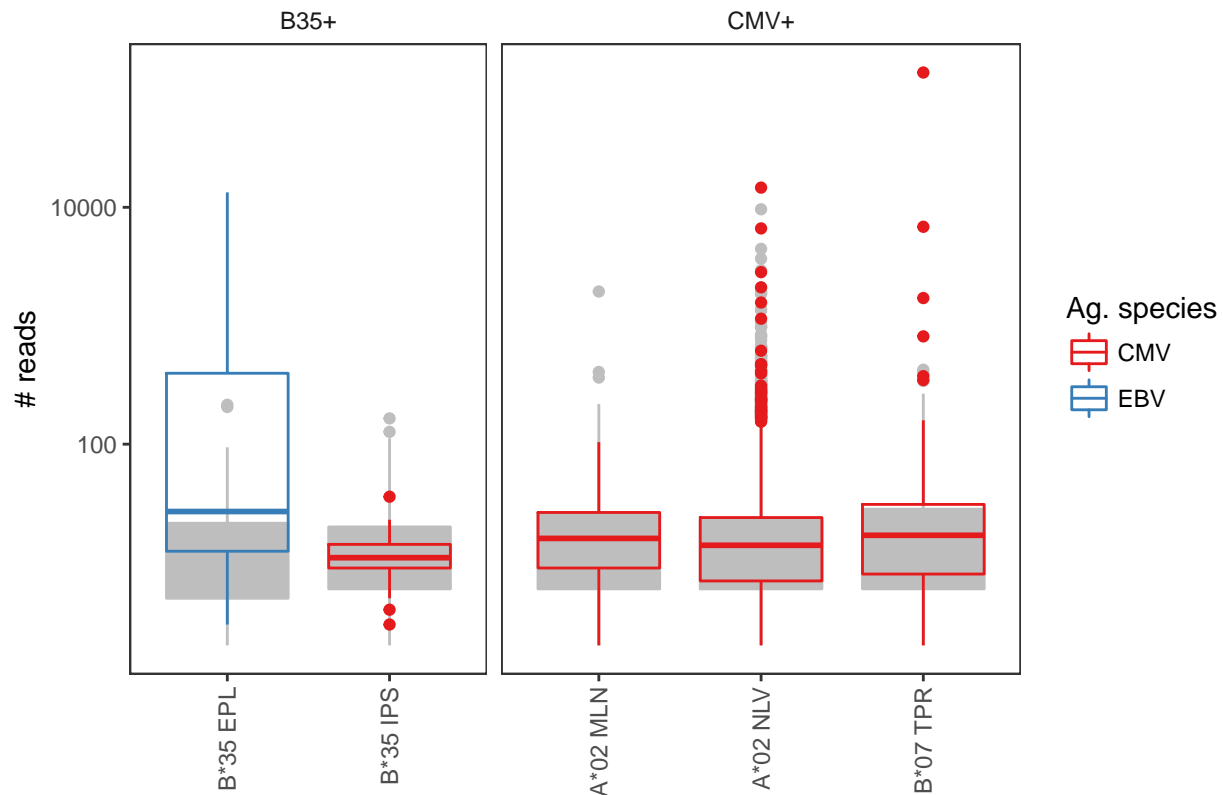


Zoom in/filter results based on donor HLA haplotype knowledge.

```
df.vdjdbc.f <- df.vdjdbc %>%
  filter(
    (sample_id == "B35+" & startsWith(mhc.a, "HLA-A*26")) |
    (sample_id == "B35+" & startsWith(mhc.a, "HLA-A*33")) |
    (sample_id == "B35+" & startsWith(mhc.a, "HLA-B*14")) |
    (sample_id == "B35+" & startsWith(mhc.a, "HLA-B*35")) |
    (sample_id == "CMV+" & startsWith(mhc.a, "HLA-A*02") & antigen.species == "CMV") |
    (sample_id == "CMV+" & startsWith(mhc.a, "HLA-B*07") & antigen.species == "CMV") |
    (sample_id == "CMV+" & startsWith(mhc.a, "HLA-B*44") & antigen.species == "CMV")
  )

df.vdjdbc.c <- df.vdjdbc.control %>%
  mutate(sample_id = "B35+") %>%
  filter(startsWith(mhc.a, "HLA-A*26") |
    startsWith(mhc.a, "HLA-A*33") |
    startsWith(mhc.a, "HLA-B*14") |
    startsWith(mhc.a, "HLA-B*35") ) %>%
  rbind(
    df.vdjdbc.control %>%
      mutate(sample_id = "CMV+") %>%
      filter(startsWith(mhc.a, "HLA-A*02") & antigen.species == "CMV" |
        startsWith(mhc.a, "HLA-B*07") & antigen.species == "CMV" |
        startsWith(mhc.a, "HLA-B*44") & antigen.species == "CMV"
      )
  )
```

```
df.vdjdb.f %>%
  ggplot(aes(x = fct_reorder2(epi.name,
                             freq,
                             as.integer(as.factor(antigen.species))),
            y = count,
            color = antigen.species)) +
  geom_boxplot(data = df.vdjdb.c,
              color = "grey", fill = "grey") +
  geom_boxplot(fill = NA) +
  scale_y_log10("# reads") + xlab("") +
  scale_color_brewer("Ag. species", palette = "Set1") +
  facet_grid(~sample_id, scales = "free", space = "free") +
  theme_bw() +
  theme(panel.grid = element_blank(),
        axis.text.x = element_text(angle = 90, vjust = 0.5, hjust = 1),
        strip.background = element_blank())
```



Searching for “expanded” TCR groups

We will not look at the actual number of reads per clonotype here, but do it the other way. We will search for groups of homologous TCR sequences that are unlikely to be found in the sample simply by chance. Here we run TCR neighbourhood enrichment test (TCRNET) to select TCR groups enriched in the memory compartment.

- CalcDegreeStats runs TCRNET routine
- -o 1,0,1 sets the search scope - match with one substitution

- -g2 vj compute the number of clonotypes with the same V/J combination, corrects for differential V/J usage
- -b data/control.txt.gz specifies the control (background dataset)

```
run_java("vdjtools",
        "CalcDegreeStats -o 1,0,1 -g2 vj -b data/control.txt.gz data/CMV+.txt.gz data/B35+.txt.gz output/CMV+.txt.gz"
        T)
```

Let's have a look at TCRNET P-values, correct them and select enriched clonotypes

```
# Load all data
list("CMV+", "B35+") %>%
  lapply(function(x)
    "output/tcrnet.{x}.txt" %>%
      str_glue() %>%
      fread() %>%
      mutate(sample_id = x)) %>%
  rbindlist(fill = T) -> df.tcrnet
```

```
# Have a glance on output table
df.tcrnet %>%
  head
```

```
##      count      freq
## 1: 256397 0.06712129
## 2: 137460 0.03598518
## 3:  66664 0.01745174
## 4:  63072 0.01651140
## 5:  57317 0.01500482
## 6:  45167 0.01182411
##
##                                     cdr3nt
## 1: TCGCCAGCAGCCAAAGATTGGGGGACAGACTCCCTATTCTCTGGAAACACCATATATTTT
## 2:          TGTGCCAGCAGCCTCCAGACAGGGTTGAACACTGAAGCTTTCTTT
## 3:          TGTGCCAGCAGCTTAGTGGGGGGCGCGGGGAGCAGTACTTC
## 4:          TGTGCCAGCCCCTGAGCTAAATTAGAGAGCAGTACTTC
## 5:          TGTGCCAGCAGTTTATCGATTGCGAGGGCGGGCACTGAAGCTTTCTTT
## 6:          TGTGCCAGCAGTTTAGAAATCGCCGTGAACACTGAAGCTTTCTTT
##
##      cdr3aa      v      d      j VEnd DStart DEnd JStart
## 1: CASSQDWGTDLSFGNTIYF TRBV4-3 TRBD1 TRBJ1-3 18 21 28 38
## 2:  CASSLQTLNTEAFF TRBV7-9 TRBD1 TRBJ1-1 12 17 24 25
## 3:  CASSLVGGAGEQYF TRBV7-9 TRBD1 TRBJ2-7 16 18 26 30
## 4:  CASP*A_IREQYF TRBV6-4 TRBD1 TRBJ2-7 9 9 14 26
## 5:  CASSLSIRRAGTEAFF TRBV28 TRBD2 TRBJ1-1 16 23 28 32
## 6:  CASSLEIAVNTEAFF TRBV28 . TRBJ1-1 15 -37 -37 25
##
##      degree.s group.count.s group2.count.s degree.c group.count.c
## 1: 1 154126 43 0 913905
## 2: 2 154126 850 26 913905
## 3: 4 154126 1422 12 913905
## 4: -1 -1 -1 -1 -1
## 5: 1 154126 445 0 913905
## 6: 1 154126 445 2 913905
##
##      group2.count.c p.value.g p.value.g2 sample_id
## 1: 201 1.0000000 1.0000000 CMV+
## 2: 5609 0.9515390 0.9306782 CMV+
## 3: 7264 0.2016678 0.2738291 CMV+
## 4: -1 1.0000000 1.0000000 CMV+
```

```
## 5:          5895 1.0000000  1.0000000      CMV+
## 6:          5895 1.0000000  1.0000000      CMV+
```

```
# Remove singletons, correct P-values
df.tcrnet <- df.tcrnet %>%
  group_by(sample_id) %>%
  mutate(p.adj = p.adjust(p.value.g2),
         fold = (degree.s + 1) / group.count.s /
               (degree.c + 1) * group.count.c) %>%
  ungroup
```

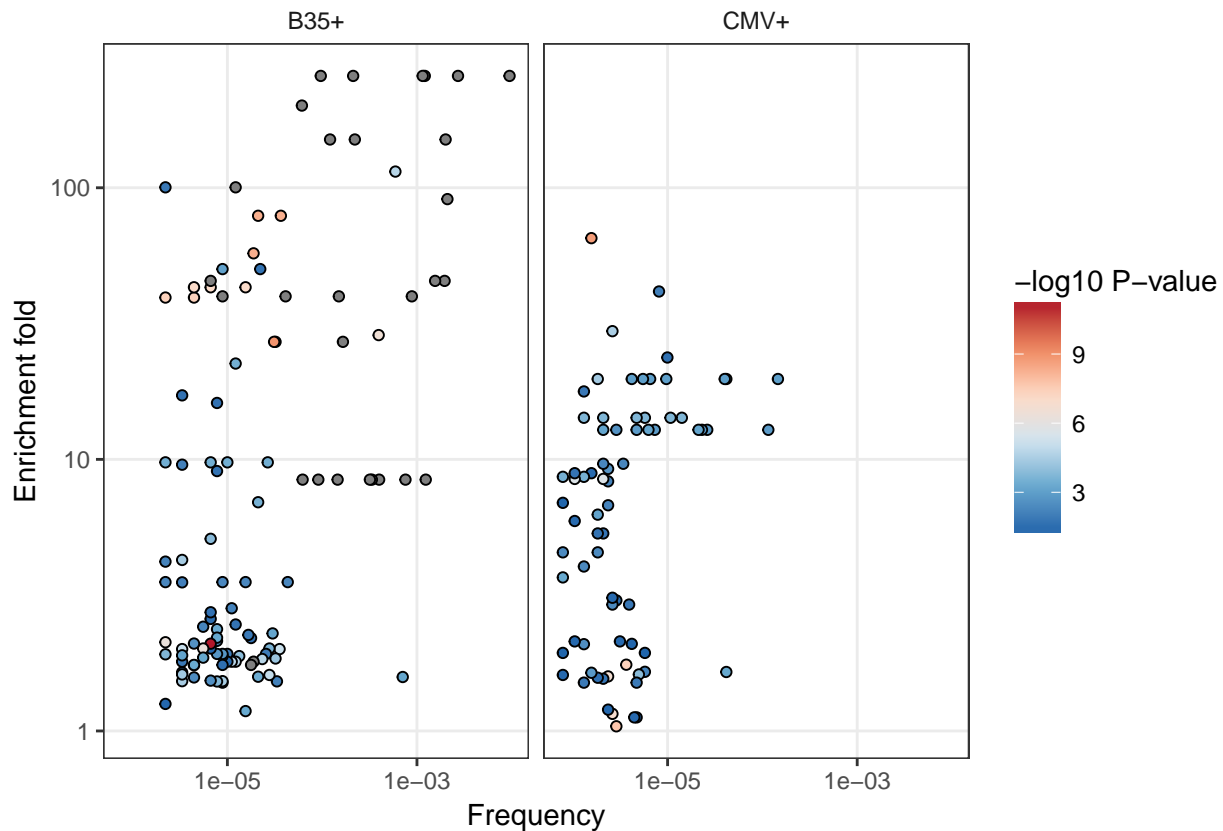
```
# Select enriched variants
df.tcrnet.e <- df.tcrnet %>%
  filter(p.adj < 0.05)
```

```
df.tcrnet.e %>%
  group_by(sample_id) %>%
  summarise(count = n())
```

```
## # A tibble: 2 x 2
##   sample_id count
##   <chr>      <int>
## 1 B35+       114
## 2 CMV+        74
```

Not much correlation between enrichment fold and clonotype frequency

```
# Volcano-like plot
df.tcrnet.e %>%
  ggplot(aes(x = freq, y = fold, fill = -log10(p.adj))) +
  geom_point(shape = 21) +
  scale_x_log10("Frequency") +
  scale_y_log10("Enrichment fold") +
  scale_fill_distiller("-log10 P-value", palette = "RdBu") +
  facet_wrap(~sample_id) +
  theme_bw() +
  theme(panel.grid.minor = element_blank(),
        strip.background = element_blank())
```



```
df.tcrnet.e %>%
  group_by(sample_id) %>%
  do(cor.test(.$fold, .$freq, method = "spearman") %>% tidy)
```

```
## Warning in cor.test.default(.$fold, .$freq, method = "spearman"): Cannot
## compute exact p-value with ties

## Warning in cor.test.default(.$fold, .$freq, method = "spearman"): Cannot
## compute exact p-value with ties

## # A tibble: 2 x 6
## # Groups:   sample_id [2]
##   sample_id estimate statistic p.value method alternative
##   <chr>      <dbl>    <dbl>   <dbl> <fct>      <fct>
## 1 B35+      0.419   143457. 3.49e-6 Spearman's rank corr~ two.sided
## 2 CMV+      0.305    46915. 8.18e-3 Spearman's rank corr~ two.sided
```

Extracting enriched groups of homologous TCRs

Compute graph with 1 substitution allowed. Here we'll use all clonotypes (except singletons) that are neighbours of enriched clonotypes.

```
# Hamming distance
find_pairs <- function(x, y) {
  res <- stringdistmatrix(x, y,
    method = "hamming",
    useNames = "strings",
    nthread = CORES) %>%
```



```

    melt %>%
    filter(value == 1) %>%
    select(-value)
colnames(res) <- c("from.cdr3", "to.cdr3")
res
}

# Graph data frame
df.tcrnet.e %>%
  .$sample_id %>%
  unique %>%
  as.list %>%
  lapply(function(x)
    find_pairs(df.tcrnet.e %>% filter(sample_id == x) %>% .$cdr3aa %>% unique,
              df.tcrnet %>% filter(sample_id == x) %>% .$cdr3aa %>% unique) %>%
    mutate(sample_id = x)
  ) %>%
  rbindlist -> df.graph

df.graph %>%
  head

```

```

##           from.cdr3           to.cdr3 sample_id
## 1:  CASSLQGYEQYF    CASSLAGYEQYF    CMV+
## 2: CASSLLGQASSYEQYF CASSLEGQASSYEQYF    CMV+
## 3: CASSLEGQASTYEQYF CASSLEGQASSYEQYF    CMV+
## 4:  CASSYSPGGTQYF    CASSQSPGGTQYF    CMV+
## 5:  CASSQSPGGTQYF    CASSQSPGGIQYF    CMV+
## 6:  CASSLGPSYEQYF    CASSLGQSYEQYF    CMV+

```

Layout and plot graphs. Highlight connected components/clusters

```

# graph layout/component naming function
layout_graph <- function(graph) {
  set.seed(42)

  gg <- graph %>%
    select(-sample_id) %>%
    graph_from_data_frame %>%
    simplify

  cc <- clusters(gg)

  coords <- gg %>%
    layout_with_graphopt(niter = 3000, charge = 0.005)

  data.frame(cdr3aa = names(V(gg)),
             x = coords[,1],
             y = coords[,2],
             stringsAsFactors = F) %>%
    merge(
      data.frame(cdr3aa = names(cc$membership),
                 cid = cc$membership,
                 cid2 = paste0(graph$sample_id[1], "_C", cc$membership)))
}

```

```

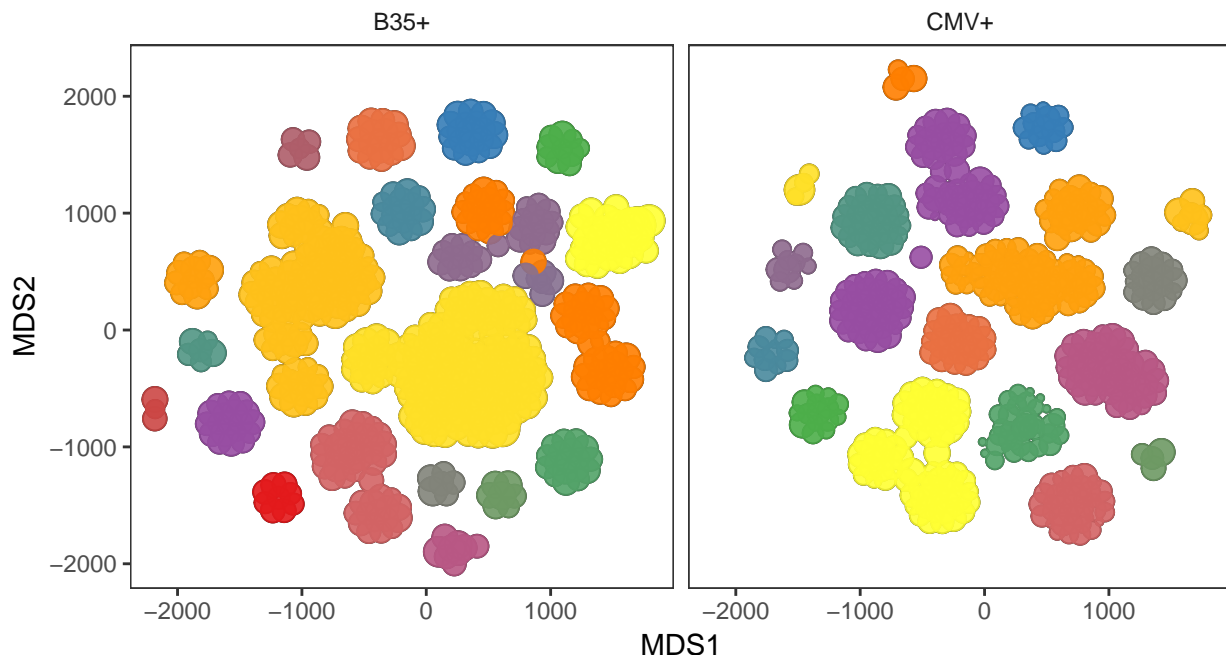
# apply to both samples
df.mds <- df.graph %>%
  group_by(sample_id) %>%
  do(layout_graph()) %>%
  ungroup %>%
  merge(df.tcrnet %>%
    group_by(cdr3aa, sample_id) %>%
    summarise(freq = sum(freq)),
    by = c("cdr3aa", "sample_id"))

## Warning in bind_rows(x, .id): Unequal factor levels: coercing to character
## Warning in bind_rows(x, .id): binding character and factor vector,
## coercing into character vector

## Warning in bind_rows(x, .id): binding character and factor vector,
## coercing into character vector

# plot 2D graph layout colored by connected component
df.mds %>%
  ggplot(aes(x = x, y = y,
             size = log10(freq))) +
  geom_point(shape = 21) +
  geom_point(aes(color = as.integer(factor(cid))), alpha = 0.9) +
  xlab("MDS1") + ylab("MDS2") +
  scale_color_distiller(guide = F, palette = "Set1") +
  scale_size(guide = F) +
  facet_wrap(~sample_id) +
  theme_bw() +
  theme(aspect = 1,
        panel.grid.major = element_blank(),
        panel.grid.minor = element_blank(),
        strip.background = element_blank())

```

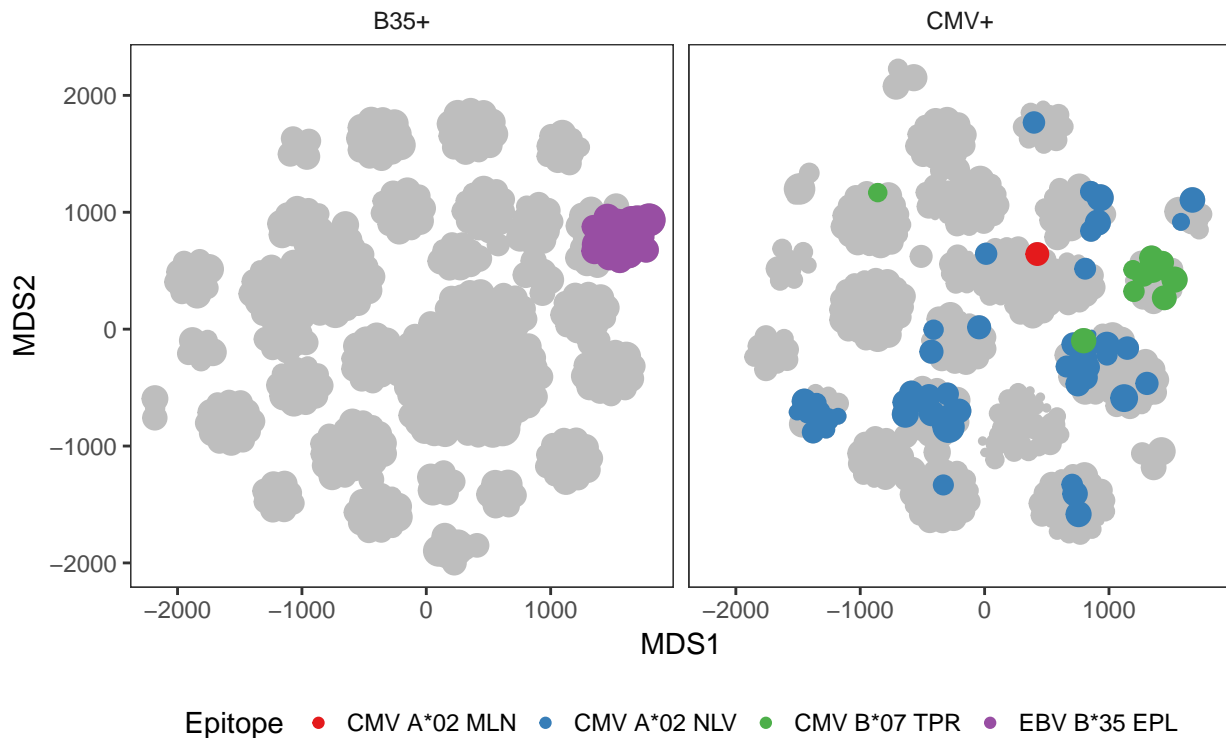


Combining TCRNET results and VDJdb annotations

Color graph by annotations

```
# append annotations
df.mds.ag.freq <- df.mds %>%
  merge(df.vdjdb.f %>%
    mutate(emi.name = paste(antigen.species, emi.name)) %>%
    select(cdr3aa, emi.name, sample_id) %>% unique,
    all.x = T, by = c("cdr3aa", "sample_id"))

# plot graph layout colored by annotation
ggplot(df.mds.ag.freq %>% filter(!is.na(emi.name)),
  aes(x = x, y = y, color = factor(emi.name),
    size = log10(freq)
  )) +
  geom_point(data = df.mds.ag.freq, color = "grey") +
  geom_point() +
  xlab("MDS1") + ylab("MDS2") +
  scale_color_brewer("EpiTope", palette = "Set1") +
  scale_size(guide = F) +
  facet_wrap(~sample_id) +
  theme_bw() +
  theme(aspect = 1,
    legend.position = "bottom",
    panel.grid.major = element_blank(),
    panel.grid.minor = element_blank(),
    strip.background = element_blank())
```



Get IDs of interesting clusters

```
df.mds.ag.freq %>%
  group_by(cid2) %>%
  mutate(total = n()) %>%
  filter(!is.na(epi.name)) %>%
  group_by(cid2, epi.name) %>%
  # fraction annotated of all nodes in component
  summarise(fraction.annot = n() / total[1]) %>%
  arrange(-fraction.annot)
```

```
## # A tibble: 13 x 3
## # Groups:   cid2 [11]
##   cid2      epi.name      fraction.annot
##   <chr>    <chr>          <dbl>
## 1 B35+_C1 EBV B*35 EPL        0.719
## 2 CMV+_C13 CMV A*02 NLV        0.6
## 3 CMV+_C11 CMV B*07 TPR        0.421
## 4 CMV+_C3  CMV A*02 NLV        0.286
## 5 CMV+_C8  CMV A*02 NLV        0.268
## 6 CMV+_C1  CMV A*02 NLV        0.15
## 7 CMV+_C6  CMV A*02 NLV        0.107
## 8 CMV+_C17 CMV A*02 NLV        0.0769
## 9 CMV+_C7  CMV A*02 NLV        0.0769
## 10 CMV+_C4  CMV A*02 NLV        0.0594
## 11 CMV+_C15 CMV B*07 TPR        0.0303
## 12 CMV+_C8  CMV B*07 TPR        0.0179
## 13 CMV+_C4  CMV A*02 MLN        0.00990
```

Plotting motifs

```
# fetching sequences
get_seqs_cid <- function(cc) {
  df.mds.ag.freq %>%
    filter(cid2 == cc) %>%
    .$cdr3aa
}

# multiple sequence alignment
align_seqs <- function(seqs, cons = F) {
  x <- seqs %>% AAStrngSet %>% msa(method = "ClustalW")

  if (cons) {
    return(msaConsensusSequence(.x))
  } else {
    return(x %>%
      as.matrix %>%
      melt %>%
      mutate(seq_id = Var1, base_id = Var2, aa = value) %>%
      select(-Var1, -Var2, -value) %>%
      group_by(seq_id) %>%
      mutate(seq = paste0(aa[base_id], collapse = "")) %>%
      ungroup)
  }
}

## Plotting
```

```

# plots a grid of AAs from multiple alignment
plot_seqgrid <- function(seqs) {
  seqs %>%
    align_seqs %>%
    ggplot(aes(x=base_id, y=seq_id)) +
    geom_text(aes(label=aa), size = 3) +
    scale_x_continuous("", breaks = c(),
                      expand = c(0.105, 0)) +
    theme_logo() +
    theme(legend.position = 'none')
}

# plots sequence logo from multiple alignment
plot_seqlogo <- function(seqs) {
  seqs %>% align_seqs %>% .$seq %>% unique %>% ggseqlogo +
    theme(legend.position = 'none')
}

# plots graph using igraph
plot_seqgraph <- function(cc, epitope) {
  set.seed(42)
  ss <- (df.mds.ag.freq %>%
    filter(cid2 == cc) %>%
    .$sample_id)[1]

  seqs <- get_seqs_cid(cc)

  df.graph %>%
    filter(sample_id == ss, to.cdr3 %in% seqs | from.cdr3 %in% seqs) %>%
    select(to.cdr3, from.cdr3) %>%
    unique %>%
    as.matrix %>%
    network -> nn

  seqs_annot <- df.mds.ag.freq %>%
    filter(epi.name == epitope & cid2 == cc) %>%
    .$cdr3aa

  grp <- ifelse(network.vertex.names(nn) %in% seqs_annot, "g1", "g2")
  nn %v% "group" <- grp
  clrs <- c("black", "red")
  names(clrs) <- c("g2", "g1")

  nn %>% ggnet2(color = "group",
               size = 5,
               color.palette = clrs,
               legend.position = "none") +
    ggtitle(paste(cc, epitope))
}

# make all plots
plot_cid_full <- function(cc) {
  plotlist <- cc %>% strsplit(",") %>% lapply(function(x)

```

```

    plot_seqgraph(x[1], x[2])
  )
  #plotlist <- c(plotlist,
  #              cc %>% as.list %>% lapply(function(x)
  #              x %>% get_seqs_cid %>% plot_seqgrid
  #              )
  #              )
  plotlist <- c(plotlist,
                cc %>% strsplit(",") %>% lapply(function(x)
                x[1] %>% get_seqs_cid %>% plot_seqlogo
                )
                )

  plotlist
}

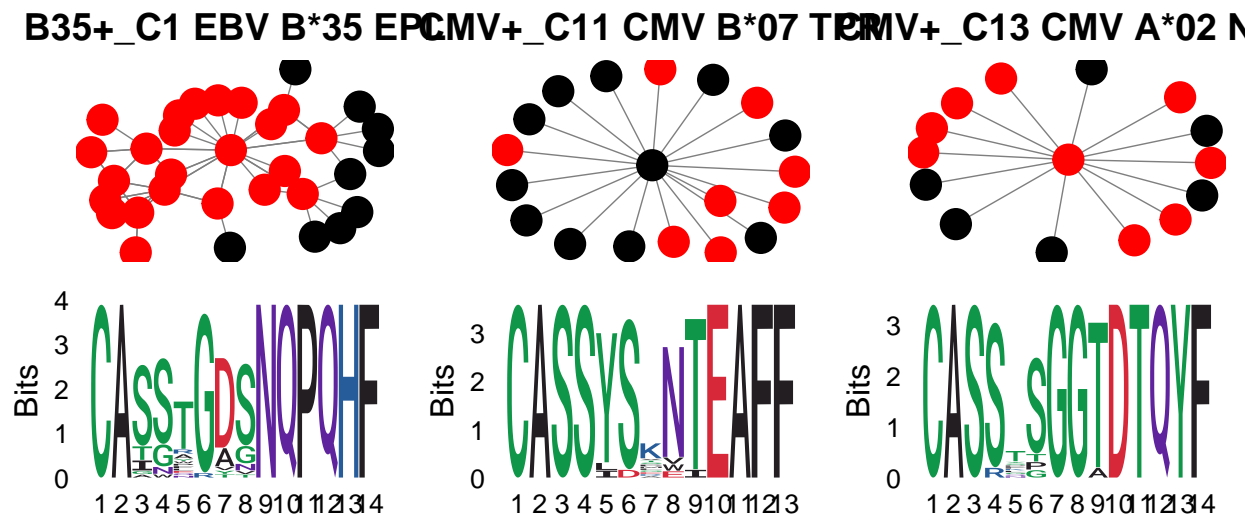
plot_grid(plotlist = plot_cid_full(c("B35+_C1,EBV B*35 EPL",
                                     "CMV+_C11,CMV B*07 TPR",
                                     "CMV+_C13,CMV A*02 NLV")),
          ncol = 3, nrow = 3, align = 'v')

```

```

## use default substitution matrix
## use default substitution matrix
## use default substitution matrix

```



Something we have missed

```

get_top_clonotypes <- function(allele) {
  df.vdjdb.f %>%
    filter(sample_id == "CMV+") %>%
    filter(startsWith(mhc.a, allele)) %>%
    select(count, freq, cdr3aa) %>%
    arrange(-count) %>%
    head(10)
}

get_top_clonotypes("HLA-B*07")

```

##	count	freq	cdr3aa
## 1	137472	3.598832e-02	CASSLQTGLNTEAFF
## 2	6846	1.792191e-03	CASSPSRNTTEAFF
## 3	1713	4.484404e-04	CASSPHRNTTEAFF
## 4	813	2.128325e-04	CASSFRQGIDTGELFF
## 5	375	9.816995e-05	CASSYSSGELFF
## 6	348	9.110172e-05	CASSYSHGELFF
## 7	159	4.162406e-05	CASSLRDGINTGELFF
## 8	154	4.031513e-05	CASSLRQGANTGELFF
## 9	143	3.743548e-05	CASSYSRNTTEAFF
## 10	143	3.743548e-05	CASSYSRNTTEAFF

```
get_top_clonotypes("HLA-A*02")
```

##	count	freq	cdr3aa
## 1	14664	0.0038388379	CASSLGQDTQYF
## 2	6633	0.0017364302	CASSSVNEQFF
## 3	2834	0.0007419031	CASLQGNTEAFF
## 4	2110	0.0005523696	CASSSVGGYTF
## 5	1570	0.0004110049	CASSLAGYEYF
## 6	1146	0.0003000074	CASSPTGNYGYTF
## 7	615	0.0001609987	CASSQEGSQPHF
## 8	473	0.0001238250	CASSYSADTGELFF
## 9	472	0.0001235632	CASSLDILSYNEQFF
## 10	463	0.0001212072	CASSLAPGATNEKLFF