Immune repertoire annotation: a RepSeq data analysis tutorial

Mikhail Shugay

5 December 2017

Dataset description

We have 16 human TCR beta repertoire samples from two donors:

- One donor is CMV+, another is CMV-
- Cells come from PBMCs, FACS-sorted, TCR beta chain sequenced using 5'RACE
- T-cell populations: CD4 and CD8
- T-cell phenotypes: memory and naive
- Each donor/population/phenotype combination comes in two replicas (independent blood draws)

Each sample is a **clonotype** (a combination of Variable segment id and amino acid sequence of CDR3 region) frequency table. The `count` column contains the number of reads.

Samples are named $s_1 \cdots s_{16}$, the metadata is missing.

> Clonotype tables were built using 10000 random reads taken from real samples from Qi et al. PNAS 2014 study, CDR3 nucleotide sequences and other general information was discarded.

Dataset layout is shown in Figure 1 below.

Analysis outline

The main goal is to recover sample metadata, i.e. label samples with donor, population and phenotype where possible.

The following basic repertoire characteristics will be considered:

Repertoire diversity measures include

- `Observed diversity`, the number of unique **clonotypes**, D_{obs}.
- `Chao1 index`, the estimate of total diversity (including unobserved species), $D_{Chao} = D_{obs} + \frac{c_1^2}{2c_2}$, where $c_{1,2}$ is the number of clonotypes supported by 1 or 2 reads.
- `Normalized Shannon index`, the entropy of clonotype frequency distribution, $D_{Shannon} = -\frac{1}{\log D_{obs}} \sum f \log f$.

Segment usage, the profile of frequencies of clonotypes having a specific Variable segment ($p_n(v_i)$, where n is the sample id and i is the V segment id). These frequency profiles contain information about the VDJ rearrangement profile and CDR3-independent selection. Variable segments are known to interact with MHC molecule itself, as well as with the presented peptides via CDR1/2. Here we'll use the Pearson correlation coefficient to compare two V usage profiles, $\rho(p_n, p_m)$.

However, a better measure for this kind of variables is the Jensen-Shannon divergence: $D_{JS}(p_n, p_m) = 0.5D_{KL}(p_n, q) + 0.5D_{KL}(p_m, q)$, where $q = 0.5p_n + 0.5p_m$ and $D_{KL}(p, q) = \sum_i p(v_i) \log \frac{p(v_i)}{q(v_i)}$.

Repertoire overlap computed as $s_{nm} = \sum_i \sqrt{f_i^{(n)} f_i^{(m)}}$ (Bhattacharyya similarity measure), where $f_i^{(n)}$ is the frequency of ith clonotype in sample n, $f_i^{(n)} = 0$ if the clonotype is absent.

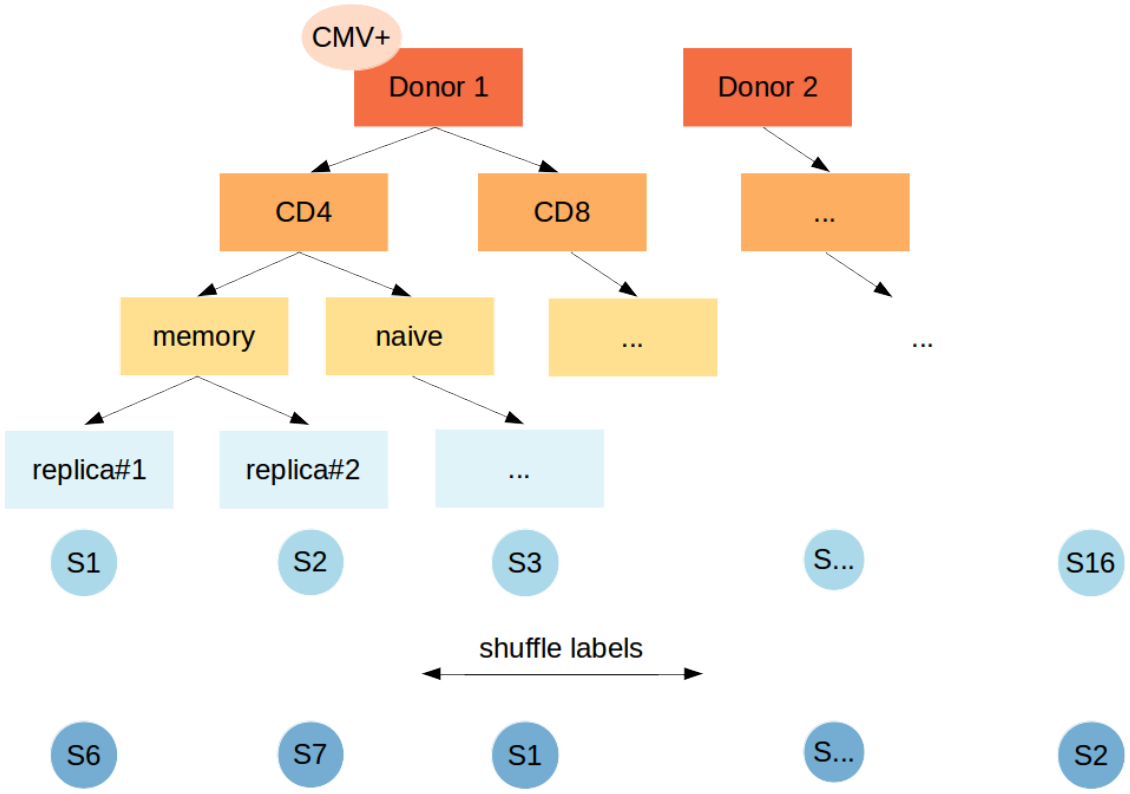


Figure 1: **Dataset layout.** You are going to perform comparative analysis of immune repertoires and retrieve sample labels.

Repertoire annotation is performed by matching samples against VDJdb, a manually curated database containing T-cell receptor sequences with known antigen specificity (i.e. ability to recognize a certain epitope presented by a certain MHC molecule).

Analysis

Load required R libraries:

Loading datasets

Load all 16 samples, minor pre-processing

```
sample_names = paste0("s", 1:16)

datasets = as.list(paste0("datasets/", sample_names, ".txt")) %>%
  # read dataset and append name
  lapply(function(x) fread(x) %>%
           mutate(sample_id = str_split_fixed(x, "[./]", 3)[2])) %>%
  # bind together
  rbindlist() %>%
  # compute frequencies
  group_by(sample_id) %>%
  mutate(freq = count / sum(count)) %>%
  ungroup

# proper ordering of sample names factor
datasets$sample_id = factor(datasets$sample_id, levels = sample_names)

head(datasets)
```

```
## # A tibble: 6 x 6
##   v        j       cdr3aa          count sample_id   freq
##   <chr>    <chr>   <chr>           <int> <fct>      <dbl>
## 1 TRBV20-1 TRBJ1-6 CSASPGTGLSPLHF    229 s1        0.0229
## 2 TRBV9    TRBJ2-7 CASSVASGGSYEQYF   161 s1        0.0161
## 3 TRBV6-5  TRBJ2-7 CASSTLVGGTYEQYF   114 s1        0.0114
## 4 TRBV2    TRBJ2-1 CASSPRPDNEQFF      68 s1        0.0068
## 5 TRBV18   TRBJ2-2 CASVGQGWTGELFF     25 s1        0.0025
## 6 TRBV29-1 TRBJ2-1 CSVRSNNEQFF        20 s1        0.002
```

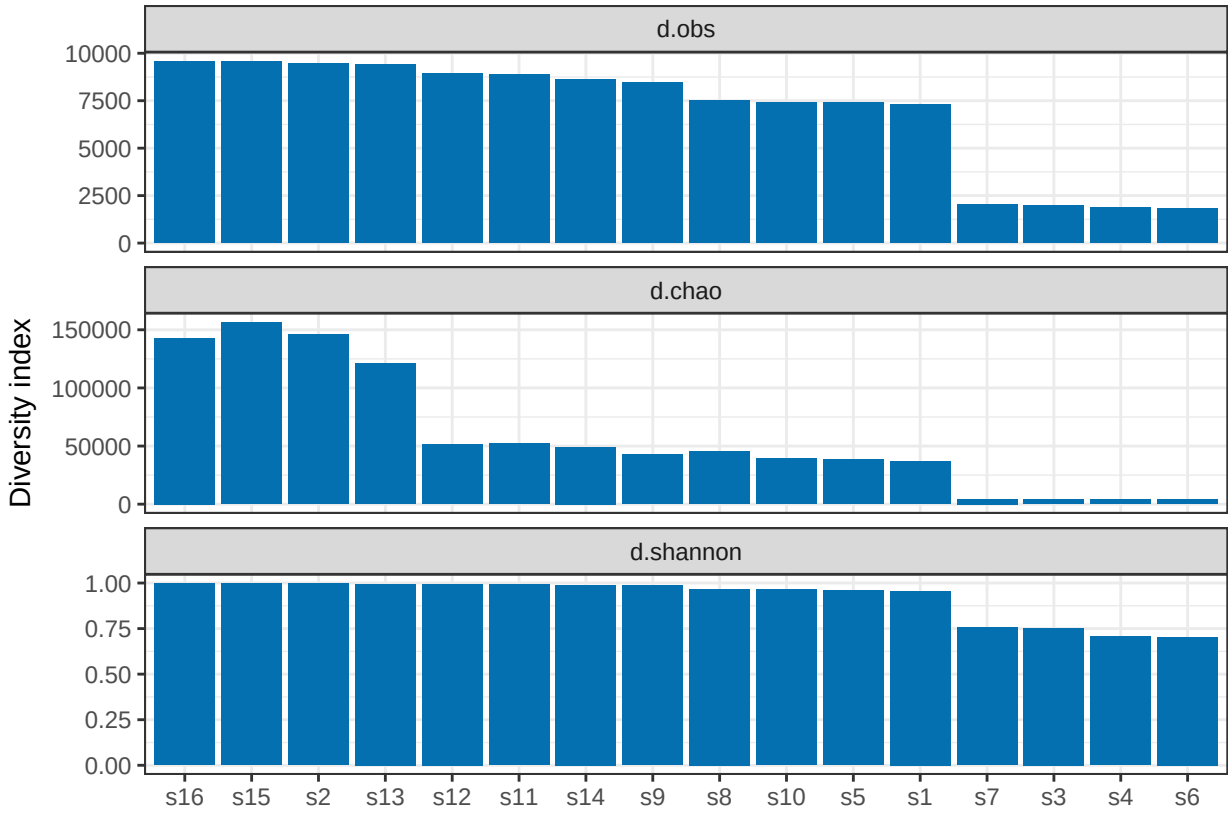
Computing diversity

Compute three aforementioned indices

```
diversity = datasets %>%
  group_by(sample_id) %>%
  summarise(d.obs = n(),
            d.chao = d.obs + sum(count == 1) ^ 2 / 2 / sum(count == 2),
            d.shannon = -sum(freq * log(freq))/log(d.obs))
```

Plot diversity values

```
diversity %>%
  melt %>%
  # set what values we are going to plot
  # fct_reorder reorders sample id by value
  ggplot(aes(x=fct_reorder2(sample_id, variable, value), y=value)) +
  # we'll make a bar plot
  geom_bar(stat = "identity", fill = "#0570b0") +
  # show each index on different subplot
  facet_wrap(~variable, scales = "free_y", ncol = 1) +
  xlab("") + ylab("Diversity index") +
  theme_bw()
```



Computing Variable segment usage profile

Here we do not count the total frequency of V in terms of number of reads, rather we use the fraction of unique clonotypes that have a given V, i.e. $f_V = \frac{\#hasV}{D_{obs}}$. We also scale these frequency values so that for each V the f_V has zero mean and unit standard deviation across samples, i.e. $f'_V = \frac{f_V - \mu_V}{\sigma_V}$.

All these steps are done to remove bias from clonal expansions and intrinsic V usage bias of the VDJ rearrangement machinery (see Murugan et al. PNAS 2012).

Summarise our data by Variable segment.

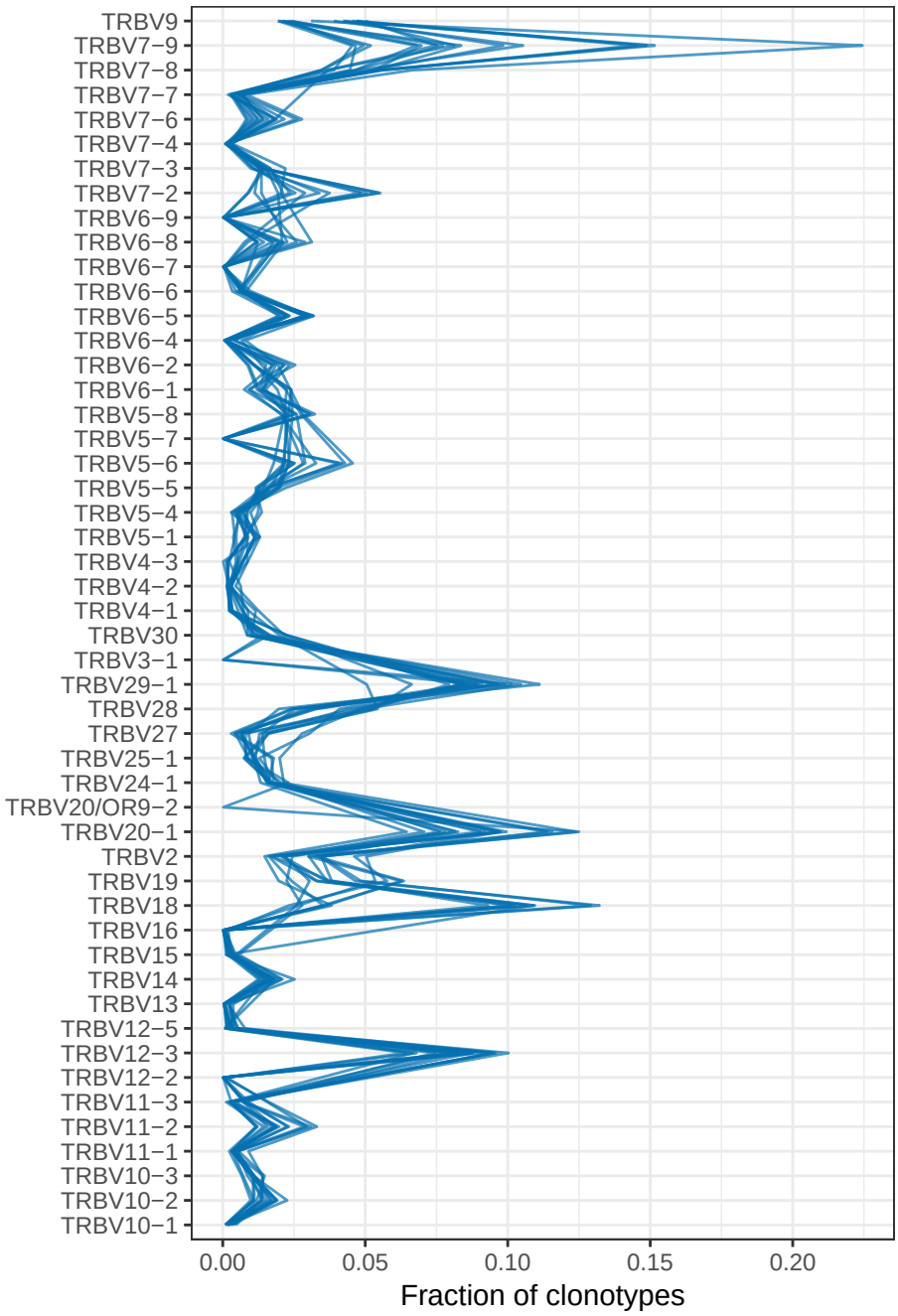
```
vusage = datasets %>%
  group_by(sample_id, v) %>%
  summarise(freq = n()) %>%
  group_by(sample_id) %>%
  mutate(freq = (freq + 1/2)/(sum(freq)+1))
```

```
head(vusage)
```

```
## # A tibble: 6 x 3
## # Groups:   sample_id [1]
##   sample_id v           freq
##   <fct>     <chr>      <dbl>
## 1 s1        TRBV10-1 0.00172
## 2 s1        TRBV10-2 0.0137
## 3 s1        TRBV10-3 0.0101
## 4 s1        TRBV11-1 0.00336
## 5 s1        TRBV11-2 0.0189
## 6 s1        TRBV11-3 0.00446
```

Unscaled V usage values show a nice correlation across samples:

```
ggplot(vusage, aes(x=v, group=sample_id, y=freq)) +
  geom_line(alpha=0.7, color= "#0570b0") +
  coord_flip() +
  xlab("") + ylab("Fraction of clonotypes") +
  theme_bw()
```



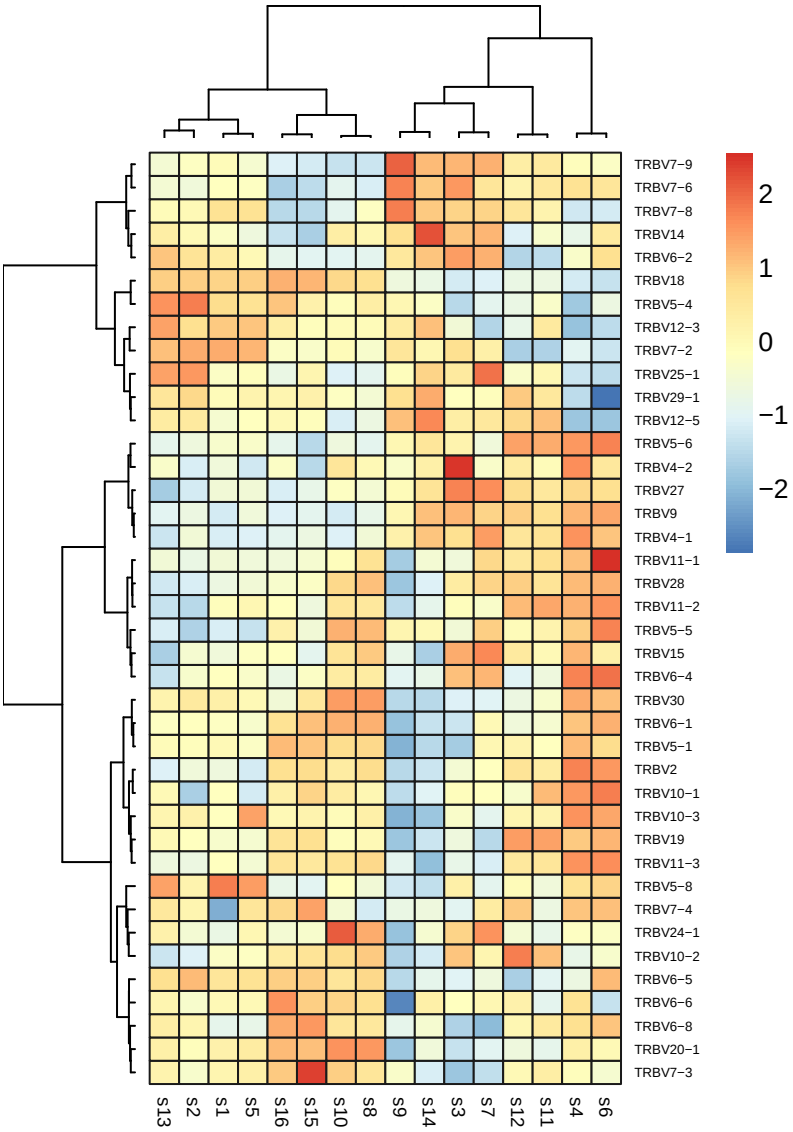
Now we'll groom the data a bit and select only those V that are present in all samples.

```
# select V present in all samples
good_v = vusage %>%
  group_by(v) %>%
  summarise(samples = length(unique(sample_id))) %>%
  filter(samples == 16) %>%
  .$v

vusage = vusage %>%
  filter(v %in% good_v)
```

```
# create V usage matrix
mat_vusage = vusage %>%
  dcast(sample_id ~ v)
rownames(mat_vusage) = mat_vusage$sample_id
mat_vusage$sample_id = NULL
mat_vusage = as.matrix(mat_vusage)
```

Draw a heatmap and perform clustering

```
aheatmap(log10(t(mat_vusage)),
         distfun = "pearson",
         hclustfun = "ward",
         scale = "row",
         border_color = "grey10")
```



Computing sample overlap

Overlap clonotype lists and compute distances (this may take a while)

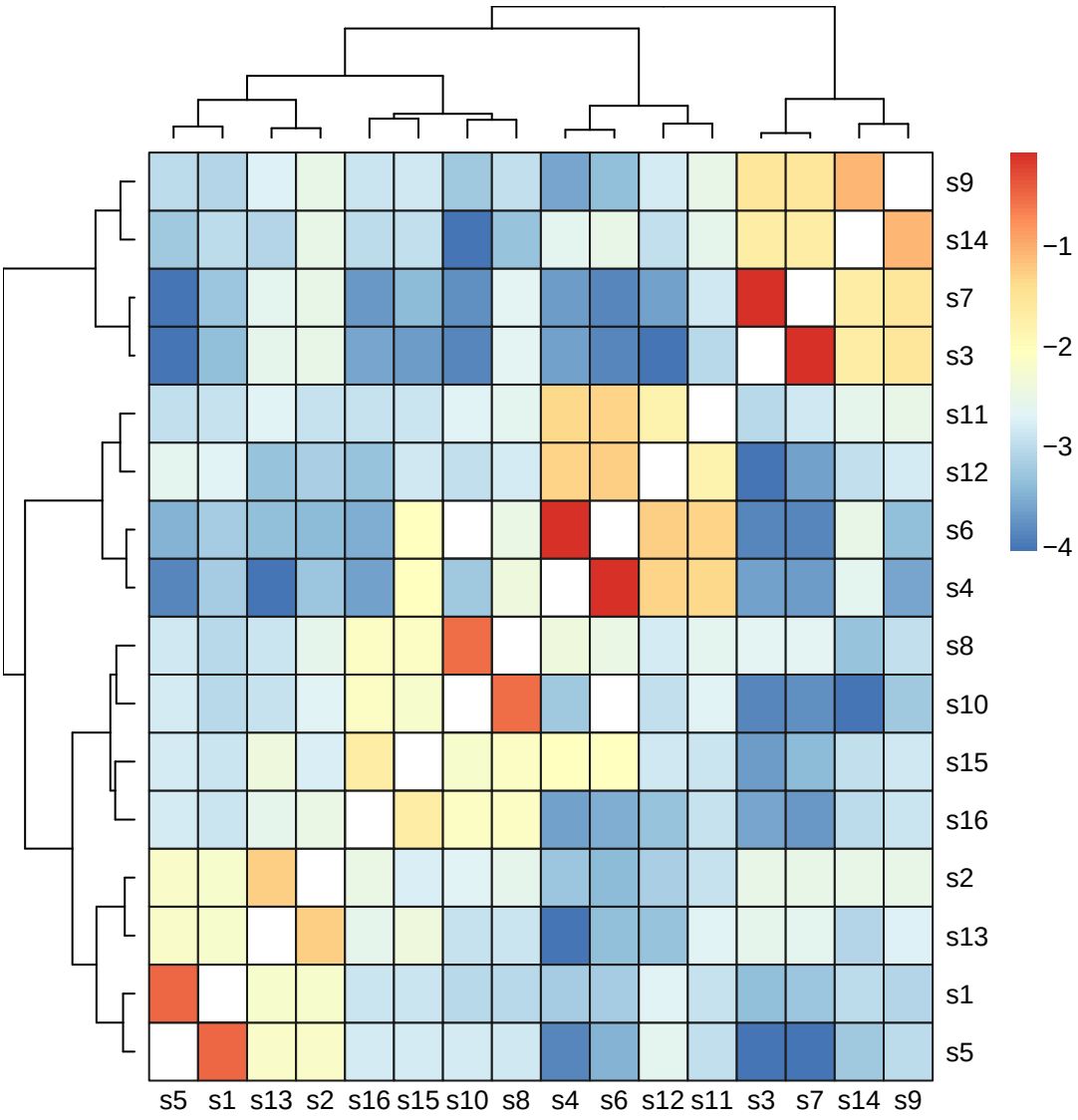
```
overlap_pair = function(id1, id2) {
  # get datasets
  .df1 = datasets %>%
    filter(sample_id == id1) %>%
    mutate(freq1 = freq) %>%
    select(v, cdr3aa, freq1)
  .df2 = datasets %>%
    filter(sample_id == id2) %>%
    mutate(freq2 = freq) %>%
    select(v, cdr3aa, freq2)

  # merge them by (v, cdr3aa)
  merge(.df1, .df2, by = c("v", "cdr3aa"), all = T) %>%
    # replace NAs (not found) with zeros
    mutate(id1 = id1, id2 = id2,
           freq1 = ifelse(is.na(freq1), 0, freq1),
           freq2 = ifelse(is.na(freq2), 0, freq2)) %>%
    # compute overlap
    group_by(id1, id2) %>%
    summarise(d = sum(sqrt(freq1 * freq2))) %>%
    as.data.table
}

# List all sample pairs
sample_pairs = expand.grid(id1=sample_names, id2=sample_names) %>%
  mutate(id1 = as.character(id1), id2 = as.character(id2)) %>%
  # remove duplicates
  filter(id1 > id2)
# wrap to list
sample_pair_list = with(sample_pairs,
                        mapply(c, id1, id2, SIMPLIFY = F))

# compute distances in parallel
distances = sample_pair_list %>%
  mclapply(function(x) overlap_pair(x[[1]], x[[2]]),
           mc.cores = 4) %>%
  rbindlist
```

Plot a heatmap with a dendrogram (we'll use a log_{10} scale)

```
# convert to matrix
distances.mat = distances %>%
  rbind(data.table(id1 = sample_names, id2 = sample_names, d = 0)) %>%
  dcast(id1~id2,fill = 0)
rownames(distances.mat) = distances.mat$id1
distances.mat$id1 = NULL
distances.mat = as.matrix(distances.mat)
# symmetrize
tmp = t(distances.mat)
diag(tmp) = NA
distances.mat = distances.mat + tmp
```

```
aheatmap(log10(distances.mat-1e-5),
         hclustfun = "ward",
         border_color = "grey10")
```

```
## Warning in is(x, "ExpressionSet"): NaNs produced
```



Annotating the repertoire

Load VDJdb data, select specific *Homo Sapiens* CD8 TRB sequences

```
vdjdb = fread("datasets/vdjdb.slim.txt") %>%
  filter(species == "HomoSapiens", gene == "TRB", mhc.class == "MHCI") %>%
  mutate(cdr3aa = cdr3, hla = str_split_fixed(mhc.a, "[-:]", 3)[,2]) %>%
  select(cdr3aa, antigen.epitope, hla, antigen.species)
```

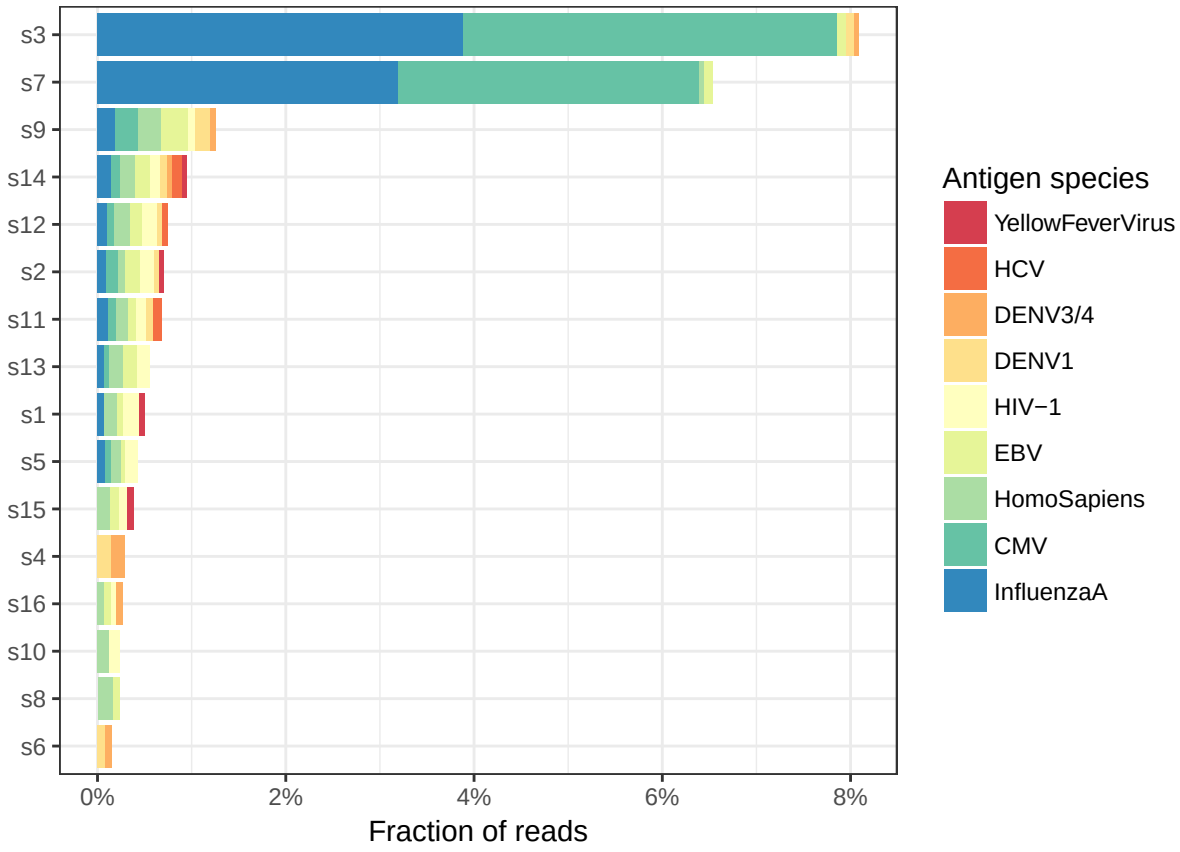
Annotate: merge to our data

```
annotated = datasets %>%
  merge(vdjdb)

head(annotated)
```

```
##         cdr3aa        v       j count sample_id  freq antigen.epitope  hla
## 1  CAISESYEQYF TRBV10-3 TRBJ2-7     1       s11 1e-04       NLVPMVATV A*02
## 2  CASFLNTEAFF  TRBV6-5 TRBJ1-1     1       s15 1e-04       ISPRTLNAW B*57
## 3 CASGQGQETQYF TRBV12-3 TRBJ2-5     1       s14 1e-04      KLVALGINAV A*02
## 4    CASGTEAFF  TRBV6-2 TRBJ1-1     1        s3 1e-04      ELAGIGILTV A*02
## 5    CASGTEAFF  TRBV6-5 TRBJ1-1     1        s7 1e-04      ELAGIGILTV A*02
## 6 CASNGQNYGYTF  TRBV6-8 TRBJ1-2     1        s8 1e-04     KAFSPEVIPMF B*57
##   antigen.species
## 1             CMV
## 2           HIV-1
## 3             HCV
## 4     HomoSapiens
## 5     HomoSapiens
## 6           HIV-1
```

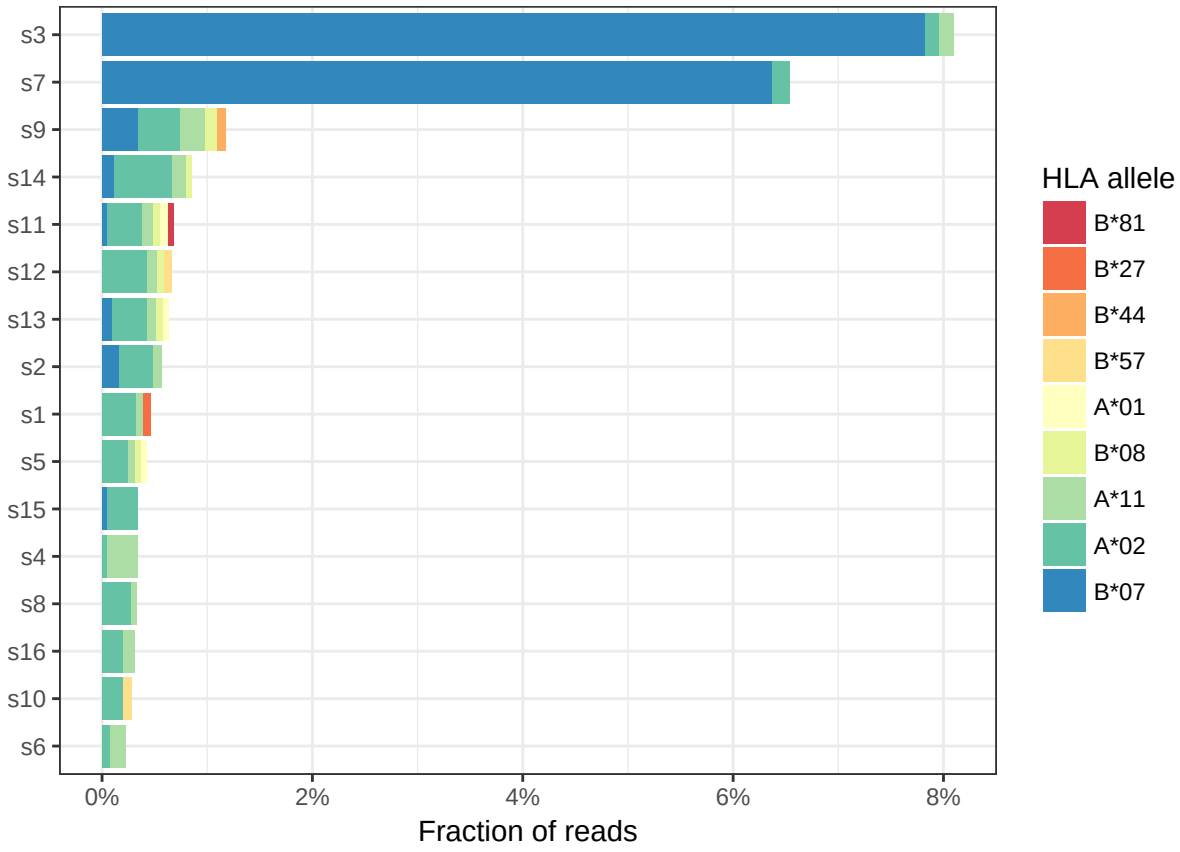
Plot parent species of putative antigen recognized by TCRs versus TCR frequency

```
annotated %>%
  group_by(sample_id, antigen.species) %>%
  summarise(freq = sum(freq)) %>%
  # simplify plot by filtering all HLAs represented by less than 5 reads
  filter(freq > 4e-4) %>%
  ggplot(aes(x=fct_reorder(sample_id, freq, fun = sum), y = freq,
             fill = fct_reorder(antigen.species, freq, fun = sum))) +
  geom_bar(stat="identity") +
  scale_fill_brewer("Antigen species", palette = "Spectral") +
  xlab("") + scale_y_continuous("Fraction of reads", labels = percent) +
  coord_flip() +
  theme_bw()
```



Same for HLA

```
annotated %>%
  group_by(sample_id, hla) %>%
  summarise(freq = sum(freq)) %>%
  # simplify plot by filtering all HLAs represented by less than 5 reads
  filter(freq > 4e-4) %>%
  ggplot(aes(x=fct_reorder(sample_id, freq, fun = sum), y = freq,
             fill = fct_reorder(hla, freq, fun = sum))) +
  geom_bar(stat="identity") +
  scale_fill_brewer("HLA allele", palette = "Spectral") +
  coord_flip() +
  xlab("") + scale_y_continuous("Fraction of reads", labels = percent) +
  theme_bw()
```



Concluding remarks

Using the results obtained above, one can deduce sample labels, namely guess

- Biological replicas of the same donor/population/phenotype
- CMV+ and CMV- donors
- One of the HLAs of the CMV+ donor
- Naive and memory cells
- CD4 and CD8 cells