T-cell repertoire re-capture in vaccination time-course

2022-10-24

Load data

```
meta <- read_tsv("../data/yellow_fever_vac/metadata.txt") %>%
  filter(replica == "F1") %>%
  select(-replica)

data <- meta %>%
  group_by(donor, time) %>%
  group_modify(~fread(paste0("../data/yellow_fever_vac/",
                             .x$file.name)))
```

Some processing

```
data <- data %>%
  group_by(donor, time, cdr3nt, cdr3aa, v) %>%
  summarise(count = sum(count)) %>%
  ungroup
```

```
## `summarise()` has grouped output by 'donor', 'time', 'cdr3nt', 'cdr3aa'. You
## can override using the `.groups` argument.
```

```
data.div <- data %>%
  group_by(donor, time) %>%
  summarise(div = length(unique(paste(cdr3nt, v))))
```

```
## `summarise()` has grouped output by 'donor'. You can override using the
## `.groups` argument.
```

Merge datasets to check sharing between 'before' and 'after' time points

```
data.m <- data %>%
  group_by(donor, time) %>%
  group_modify(~left_join(.x,
                          data %>%
                            mutate(time.next = time) %>%
                            select(-time) %>%
                            filter(.y$time < time.next) %>%
                            mutate(found = T) %>%
                            select(v, cdr3nt, cdr3aa, time.next, found)))
```

```
## Joining with `by = join_by(cdr3nt, cdr3aa, v)`
## Joining with `by = join_by(cdr3nt, cdr3aa, v)`
## Joining with `by = join_by(cdr3nt, cdr3aa, v)`
## Joining with `by = join_by(cdr3nt, cdr3aa, v)`
## Joining with `by = join_by(cdr3nt, cdr3aa, v)`
## Joining with `by = join_by(cdr3nt, cdr3aa, v)`
## Joining with `by = join_by(cdr3nt, cdr3aa, v)`
## Joining with `by = join_by(cdr3nt, cdr3aa, v)`
```

```
## Joining with `by = join_by(cdr3nt, cdr3aa, v)`
## Joining with `by = join_by(cdr3nt, cdr3aa, v)`
## Joining with `by = join_by(cdr3nt, cdr3aa, v)`
## Joining with `by = join_by(cdr3nt, cdr3aa, v)`
## Joining with `by = join_by(cdr3nt, cdr3aa, v)`
## Joining with `by = join_by(cdr3nt, cdr3aa, v)`
## Joining with `by = join_by(cdr3nt, cdr3aa, v)`
## Joining with `by = join_by(cdr3nt, cdr3aa, v)`
## Joining with `by = join_by(cdr3nt, cdr3aa, v)`
## Joining with `by = join_by(cdr3nt, cdr3aa, v)`
## Joining with `by = join_by(cdr3nt, cdr3aa, v)`
## Joining with `by = join_by(cdr3nt, cdr3aa, v)`
## Joining with `by = join_by(cdr3nt, cdr3aa, v)`
## Joining with `by = join_by(cdr3nt, cdr3aa, v)`
## Joining with `by = join_by(cdr3nt, cdr3aa, v)`
## Joining with `by = join_by(cdr3nt, cdr3aa, v)`
## Joining with `by = join_by(cdr3nt, cdr3aa, v)`
## Joining with `by = join_by(cdr3nt, cdr3aa, v)`
## Joining with `by = join_by(cdr3nt, cdr3aa, v)`
## Joining with `by = join_by(cdr3nt, cdr3aa, v)`
## Joining with `by = join_by(cdr3nt, cdr3aa, v)`
## Joining with `by = join_by(cdr3nt, cdr3aa, v)`
```

```
data.m <- data.m %>%
  mutate(found = !is.na(found)) %>%
  mutate(quantile = case_when(
    count == 1 ~ "Singleton",
    count == 2 ~ "Doubleton",
    count == 3 ~ "Tripleton",
    T ~ "Large")) %>%
  mutate(quantile = factor(
    quantile,
    c("Singleton",
      "Doubleton",
      "Tripleton",
      "Large")))
```

Load annotations - A02LLW-specific clonotypes, mark them and clonotypes that differ by 1 mm in CDR3 amino acid sequences in the dataset as specific

```
annot <- read_tsv("../example/annotations.txt")

#Compute distances between strings:
get_distances <- function(aa.seq.1, aa.seq.2, threshold = 1,
                          method = "hamming", ...) {
  stringdistmatrix(unique(aa.seq.1), unique(aa.seq.2),
                   method = method,
                   useNames = T, ...) %>%
    melt %>%
    filter(value <= threshold) %>%
    rename(aa.seq = Var1, aa.seq.db = Var2, dist = value) %>%
    mutate(aa.seq = as.character(aa.seq), aa.seq.db = as.character(aa.seq.db))
}

#An optimized routine that splits by length and processes in chunks(hamming only):
```

```
get_1mm_pairs <- function(aa.seq, aa.seq.db, chunks = 64) {
  d <- tibble(aa.seq = unique(aa.seq)) %>%
    mutate(len = nchar(aa.seq),
           chunk.id = rep(1:chunks, length.out = length(unique(aa.seq))))

  db <- tibble(aa.seq.db = unique(aa.seq.db)) %>%
    mutate(len.db = nchar(aa.seq.db))

  d %>%
    group_by(chunk.id, len) %>%
    group_modify(~ get_distances(.x$aa.seq, db %>%
                                   filter(len.db == .y$len) %>%
                                   .$aa.seq.db)) %>%
    ungroup %>%
    select(-chunk.id, -len)
}

annot <- get_1mm_pairs(data$cdr3aa, annot$cdr3aa) %>%
  rename(cdr3aa = aa.seq) %>%
  select(cdr3aa) %>%
  mutate(specificity = "A02LLW")
```

For annotations - capture probability at day 15 compared to day 0

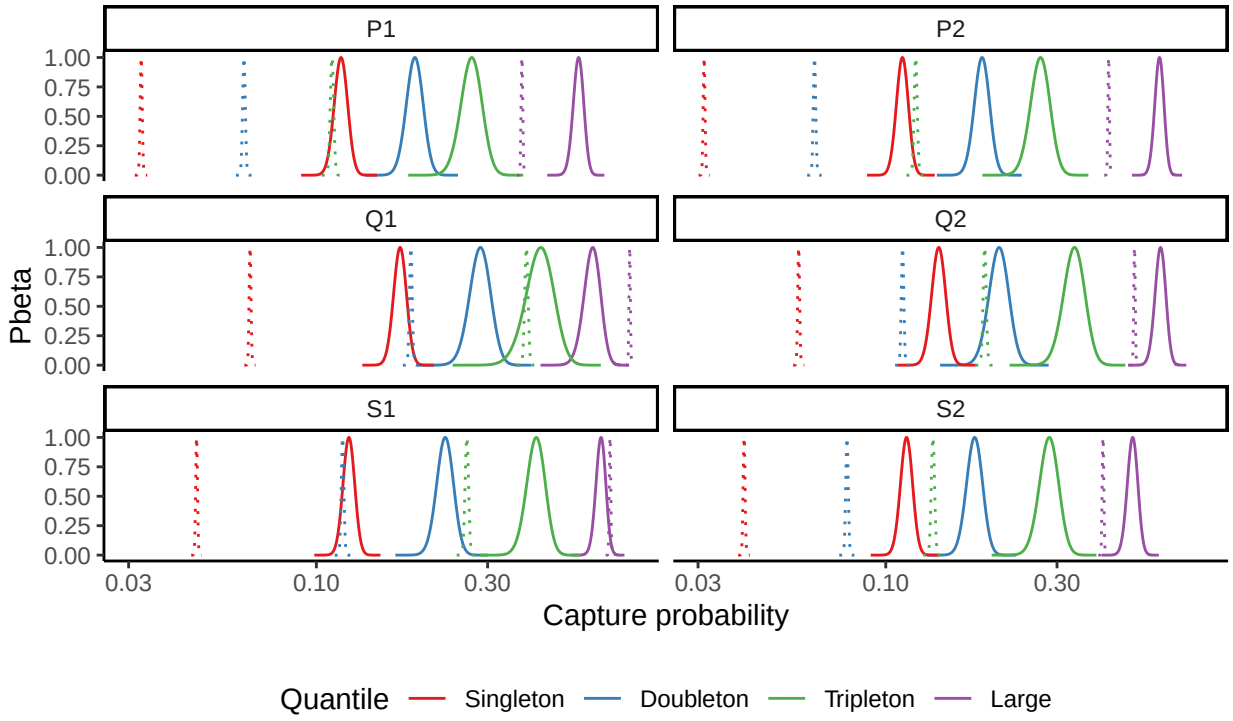
```
data.s.s <- data.m %>%
  mutate(specific = cdr3aa %in% annot$cdr3aa) %>%
  group_by(time, time.next, donor, quantile, specific) %>%
  summarise(alpha = sum(found), beta = sum(!found)) %>%
  group_by(time, donor, quantile, specific) %>% # append data for missing clonotypes
  mutate(beta = sum(beta)) %>%
  filter(!is.na(time.next)) %>%
  ungroup
```

```
## `summarise()` has grouped output by 'time', 'time.next', 'donor', 'quantile'.
## You can override using the `.groups` argument.
```

```
data.b <- data.s.s %>%
  filter(time == 15, time.next == 45) %>%
  merge(tibble(p = 0:10000/10000)) %>%
  filter(p > 0.0001) %>%
  group_by(donor, specific, quantile) %>%
  mutate(Pbeta = dbeta(p, alpha, beta)) %>%
  ungroup

p1 <- data.b %>%
  filter(Pbeta > 1e-5) %>%
  mutate(specificity = ifelse(specific, "A02_LLW", "Unknown"),
         time = as.character(time)) %>%
  group_by(donor, specific, quantile) %>%
  mutate(height = Pbeta / max(Pbeta)) %>%
  ggplot(aes(x = p,
             y = height,
             group = paste(specificity, quantile, donor, time),
             linetype = specificity,
             color = quantile)) +
```

```
  geom_line() +
  scale_color_brewer("Quantile", palette = "Set1") +
  scale_x_log10("Capture probability") +
  scale_linetype_manual("Specificity", values = c("solid", "dotted")) +
  ylab("Pbeta") +
  facet_wrap(.~donor, ncol = 2) +
  theme_classic() +
  theme(legend.position = "bottom",
        legend.box="vertical")
p1
```



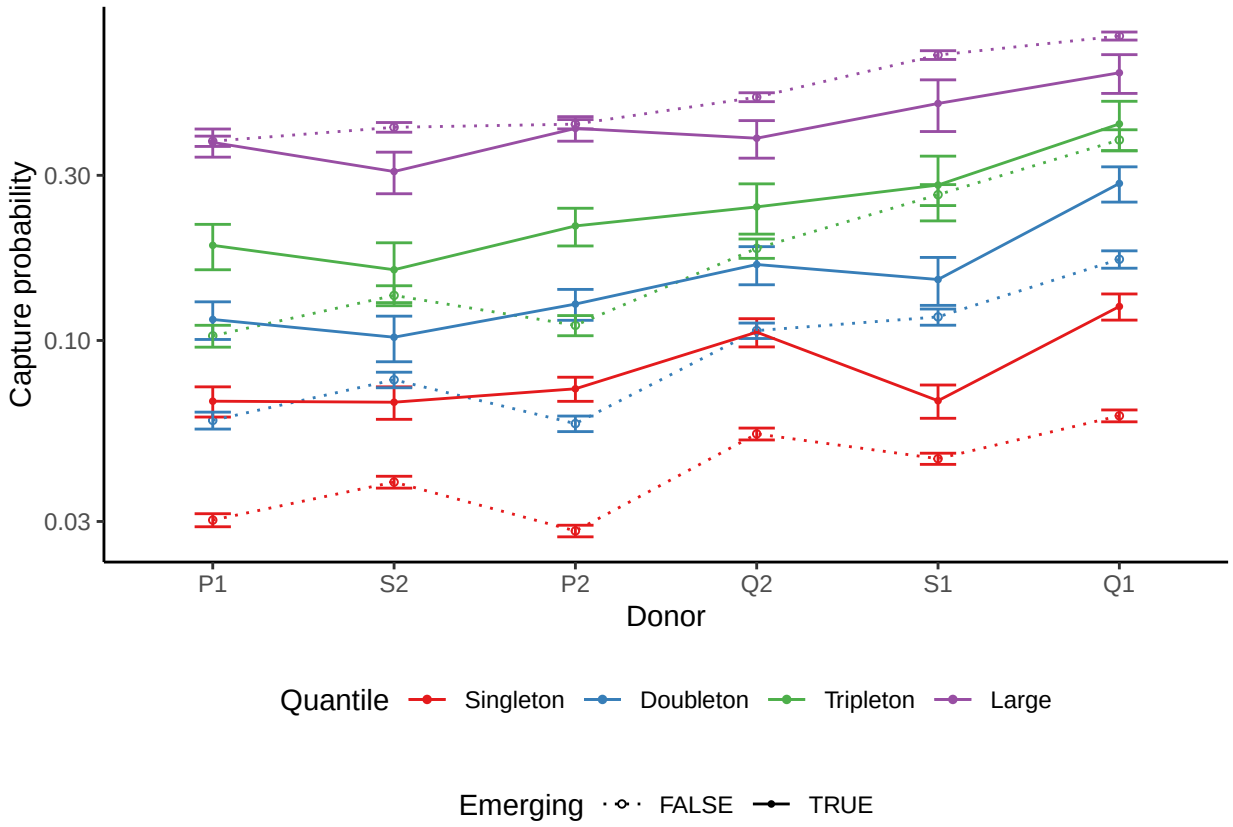
For emerging clonotypes - found in 7 but absent in 0

```
cdrs7 <- data %>% filter(time == 7) %>% .$cdr3aa %>% unique
cdrs0 <- data %>% filter(time == 0) %>% .$cdr3aa %>% unique
cdrs.emerging <- setdiff(cdrs7, cdrs0)
```

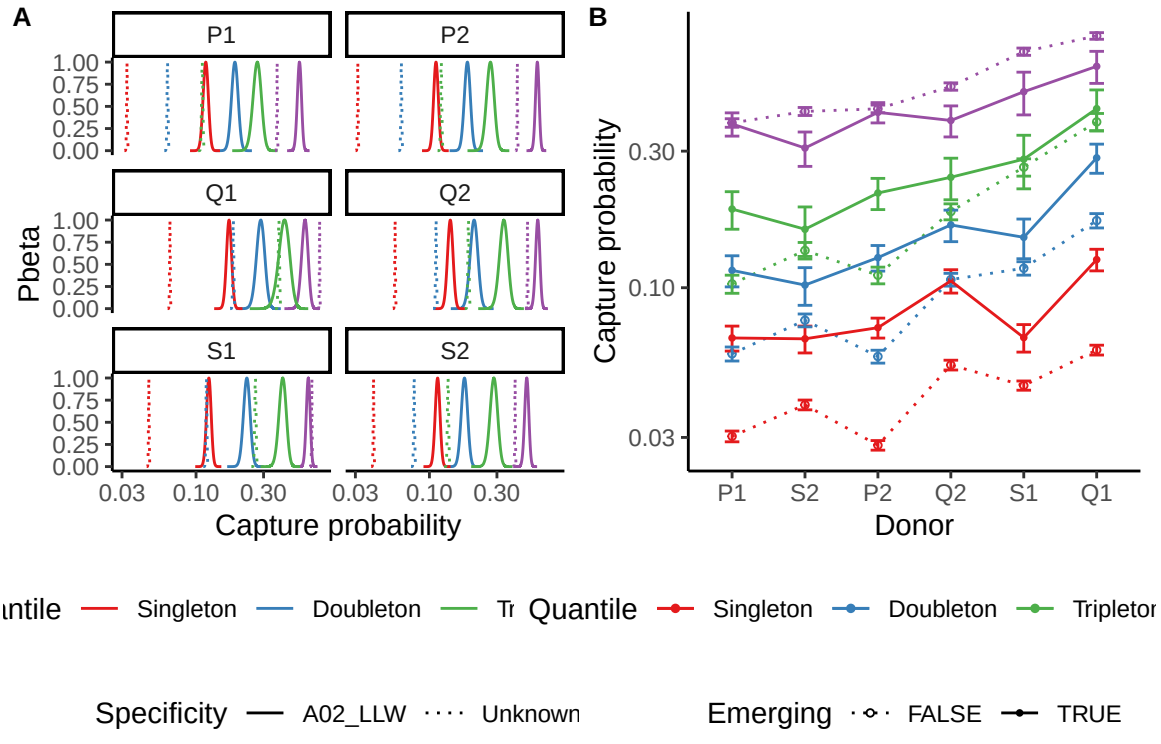
```
data.s.e <- data.m %>%
  mutate(emerging = cdr3aa %in% cdrs.emerging) %>%
  group_by(time, time.next, donor, quantile, emerging) %>%
  summarise(alpha = sum(found), beta = sum(!found)) %>%
  group_by(time, donor, quantile, emerging) %>% # append data for missing clonotypes
  mutate(beta = sum(beta)) %>%
  filter(!is.na(time.next)) %>%
  ungroup %>%
  left_join(data.div) %>%
  left_join(data.div %>%
              rename(time.next = time, div.next = div))
```

```
## `summarise()` has grouped output by 'time', 'time.next', 'donor', 'quantile'.
## You can override using the `.groups` argument.
## Joining with `by = join_by(time, donor)`
## Joining with `by = join_by(time.next, donor)`
```

```
p2 <- data.s.e %>%
  filter(time == 15, time.next == 45) %>%
  mutate(alpha = alpha + 1, beta = beta + 1) %>%
  mutate(p = alpha / (alpha + beta),
         sdp = sqrt(alpha * beta / (alpha + beta) ^ 2 / (alpha + beta + 1))) %>%
  ggplot(aes(x = donor %>% fct_reorder(p),
             y = p,
             linetype = emerging,
             shape = emerging,
             color = quantile)) +
  geom_point(size = 1) +
  geom_errorbar(aes(ymin = p - 6 * sdp, ymax = p + 6 * sdp),
                linetype = "solid", width = 0.2) +
  geom_line(aes(group = paste(emerging, quantile))) +
  scale_y_log10("Capture probability") +
  xlab("Donor") +
  scale_linetype_manual("Emerging", values = c("dotted", "solid")) +
  scale_color_brewer("Quantile", palette = "Set1") +
  scale_shape_manual("Emerging", values = c(1, 16)) +
  theme_classic() +
  theme(legend.position = "bottom", legend.box="vertical")
p2
```



```
plot_grid(p1, p2,
          labels = c("A", "B"),
          label_size = 10) -> fig4
fig4
```




```
ggsave("../figures/fig4.pdf", fig4)
```

```
## Saving 6 x 4 in image
```

```
data.s.e %>%
  filter(time == 15, time.next == 45) %>%
  mutate(logP = log(alpha / (alpha + beta))) %>%
  arrange(donor) %>%
  group_by(quantile) %>%
  group_modify(~ t.test(.x$logP[which(!.x$emerging)], .x$logP[which(.x$emerging)],
                        paired = T) %>% tidy)
```

```
## # A tibble: 4 x 9
## # Groups:   quantile [4]
##   quantile  estimate statistic  p.value parameter conf.low conf.high method
##   <fct>        <dbl>     <dbl>    <dbl>     <dbl>    <dbl>     <dbl> <chr>
## 1 Singleton   -0.676     -8.41 0.000390         5  -0.883    -0.469  Paired t-t~
## 2 Doubleton   -0.490     -5.60 0.00251          5  -0.715    -0.265  Paired t-t~
## 3 Tripleton   -0.313     -2.98 0.0308           5  -0.583    -0.0429 Paired t-t~
## 4 Large        0.195      3.43 0.0187           5   0.0488    0.342  Paired t-t~
## # i 1 more variable: alternative <chr>
```