Applying VDJtrack algorithm to Yellow Fever Virus vaccination data

M.S.

2023-03-25

Load clonotype annotations, metadata and data from the Yellow Fever Vaccination time course (0th day = before, 15th day = after)

```
annot <- read_tsv("example/annotations.txt")
meta <- read_tsv("example/metadata.txt")
data <- meta %>%
  group_by(donor, time) %>%
  group_modify(~ read_tsv(paste0("example/", .x$file.name)))
```

A little trick - annotate with 1 amino acid substitution, expand the annotation

```
#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))
}

annot <- annot %>%
  group_by(group) %>%
  group_modify(~get_1mm_pairs(data$cdr3aa, .x$cdr3aa) %>%
```

```
                rename(cdr3aa = aa.seq) %>%
                select(cdr3aa)) %>%
  ungroup %>%
  select(-chunk.id, -len)
```

```
## Adding missing grouping variables: `chunk.id`, `len`
```

Annotate data by size (singleton/doubleton/...) and according to Yellow Fever Virus antigen specificity

```
data <- data %>%
  mutate(quantile = case_when(
    count == 1 ~ "singleton",
    count == 2 ~ "doubleton",
    .default = "3+"
  )) %>%
  mutate(quantile = factor(quantile, levels = c("singleton",
                                                "doubleton",
                                                "3+"))) %>%
  left_join(annot) %>%
  mutate(group = ifelse(is.na(group), "unknown", group)) %>%
  ungroup
```

```
## Joining with `by = join_by(cdr3aa)`
```

```
## Warning in left_join(., annot): Each row in `x` is expected to match at most 1 row in `y`.
## i Row 276 of `x` matches multiple rows.
## i If multiple matches are expected, set `multiple = "all"` to silence this
##   warning.
```

Identify clonotypes that appeared (spawned) in `after` dataset but were absent in `before` time point

```
# count is the frequency in the 'after' dataset
data %>%
  filter(time == "before") %>%
  select(-time) %>%
  left_join(data %>%
              filter(time == "after") %>%
              select(-time, -count, -quantile, -group) %>%
              mutate(spawned = T)) %>%
  mutate(spawned = !is.na(spawned)) -> data.m
```

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

```
## Warning in left_join(., data %>% filter(time == "after") %>% select(-time, : Each row in `x` is expe
## i Row 308 of `x` matches multiple rows.
## i If multiple matches are expected, set `multiple = "all"` to silence this
##   warning.
```

Calculate sample sizes and compute sampling statistics

```
data.s0 <- data %>%
  group_by(donor, time, group) %>%
  summarise(diversity = n()) %>%
  dcast(donor + group ~ time) %>%
  rename(div.after = after, div.before = before)
```

```
## `summarise()` has grouped output by 'donor', 'time'. You can override using the
## `.groups` argument.
## Using diversity as value column: use value.var to override.
```

```
data.s <- data.m %>%
  group_by(donor, group, quantile) %>%
  summarise(spawned = sum(spawned), total = n())
```

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

```
data.vdjtrack <- inner_join(data.s, data.s0)
```

```
## Joining with `by = join_by(donor, group)`
```

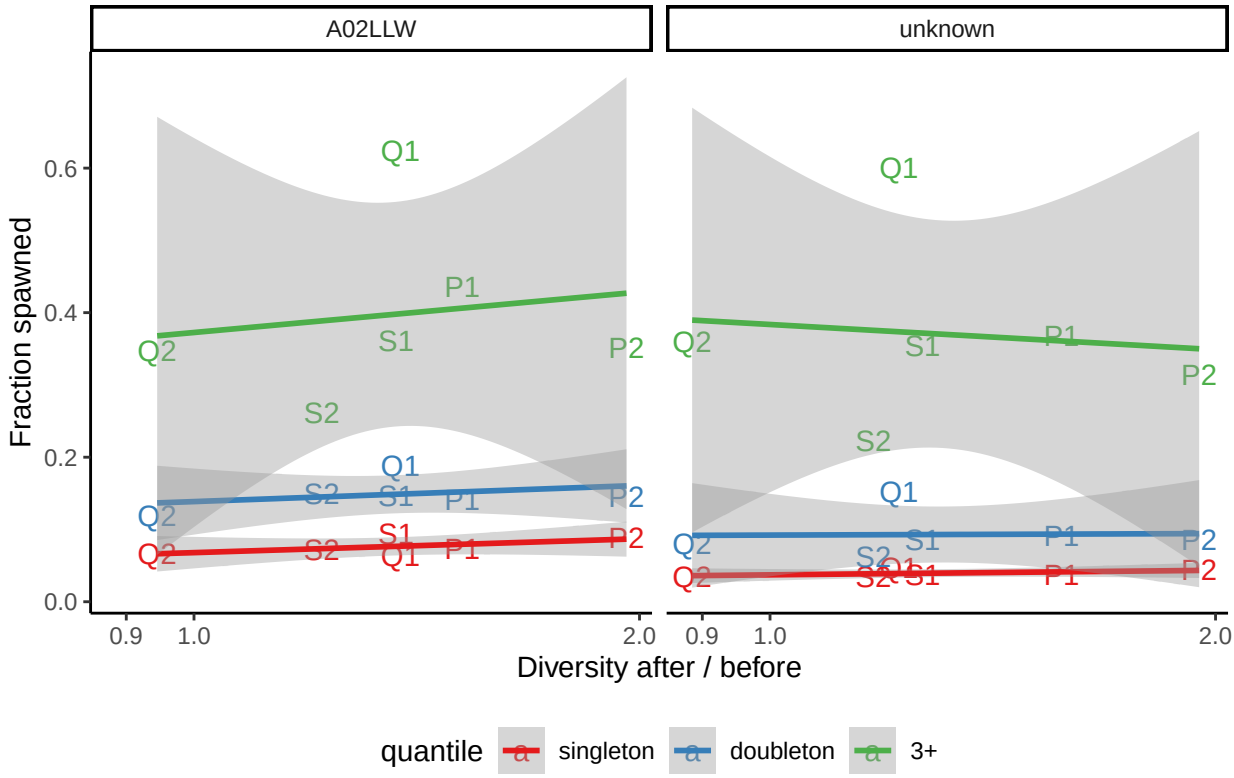
```
data.vdjtrack
```

```
## # A tibble: 36 x 7
## # Groups:   donor, group [12]
##    donor group   quantile  spawned  total div.after div.before
##    <chr> <chr>   <fct>       <int>  <int>     <int>      <int>
##  1 P1    A02LLW  singleton     282   3849      7554       4976
##  2 P1    A02LLW  doubleton     119    841      7554       4976
##  3 P1    A02LLW  3+            187    428      7554       4976
##  4 P1    unknown singleton   19089 500790    975618     619748
##  5 P1    unknown doubleton    8049  87773    975618     619748
##  6 P1    unknown 3+          11491  31185    975618     619748
##  7 P2    A02LLW  singleton     319   3602      9827       5014
##  8 P2    A02LLW  doubleton     143    980      9827       5014
##  9 P2    A02LLW  3+            210    596      9827       5014
## 10 P2    unknown singleton   21505 487949   1269794     651283
## # ... with 26 more rows
```

Plot results

```
data.vdjtrack %>%
  ggplot(aes(x = div.after / div.before,
             y = spawned / total,
             color = quantile)) +
  geom_text(aes(label = donor)) +
  geom_smooth(method = "lm") +
  scale_x_log10("Diversity after / before") +
  ylab("Fraction spawned") +
  scale_color_brewer(palette = "Set1") +
  facet_wrap(~group) +
  theme_classic() +
  theme(aspect = 1, legend.position = "bottom")
```

```
## `geom_smooth()` using formula = 'y ~ x'
```



Compute statistics

```
mdl <- lm(spawned.frac ~ quantile + group + div.ratio,
          data = data.vdjtrack %>%
            mutate(div.ratio = log(div.after / div.before),
                   spawned.frac = log(spawned / total)
            ))
summary(mdl)
```

```
##
## Call:
## lm(formula = spawned.frac ~ quantile + group + div.ratio, data = data.vdjtrack %>%
##     mutate(div.ratio = log(div.after/div.before), spawned.frac = log(spawned/total)))
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -0.52612 -0.16034 -0.03415  0.11838  0.70380
##
## Coefficients:
##                    Estimate Std. Error t value Pr(>|t|)
## (Intercept)         -2.7531     0.1050 -26.233  < 2e-16 ***
## quantiledoubleton    0.7425     0.1069   6.944 8.64e-08 ***
## quantile3+           1.9088     0.1069  17.852  < 2e-16 ***
## groupunknown        -0.4029     0.0877  -4.594 6.84e-05 ***
## div.ratio            0.1658     0.1871   0.886    0.382
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.2619 on 31 degrees of freedom
## Multiple R-squared:  0.918,  Adjusted R-squared:  0.9074
```

```
## F-statistic: 86.71 on 4 and 31 DF,  p-value: 2.243e-16
#END
```