annotate

2022-10-17

Load data

```
data <- read_tsv("sample.txt.gz")
```

```
## Rows: 4540462 Columns: 7
```

```
## -- Column specification --------------------------------------------------------
## Delimiter: "\t"
## chr (5): sample.id, replica, nt.seq, best.V.gene, aa.seq
## dbl (2): time.point, clone.count
```

```
##
## i Use `spec()` to retrieve the full column specification for this data.
## i Specify the column types or set `show_col_types = FALSE` to quiet this message.
```

```
database <- read_tsv("database.txt.gz")
```

```
## Rows: 405 Columns: 2
```

```
## -- Column specification --------------------------------------------------------
## Delimiter: "\t"
## chr (2): aa.seq.db, epitope
```

```
##
## i Use `spec()` to retrieve the full column specification for this data.
## i Specify the column types or set `show_col_types = FALSE` to quiet this message.
```

Data structure is better be kept the way specified below

```
glimpse(data)
```

```
## Rows: 4,540,462
## Columns: 7
## $ sample.id   <chr> "S2", "S2", "S2", "S2", "S2", "S2", "S2", "S2", "S2", "S2"~
## $ replica     <chr> "F1", "F1", "F1", "F1", "F1", "F1", "F1", "F1", "F1", "F1"~
## $ time.point  <dbl> 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0~
## $ clone.count <dbl> 5160, 2317, 2154, 2073, 1583, 915, 845, 811, 673, 673, 666~
## $ nt.seq      <chr> "CGTGCCAGCAGCGCCCGGACTAGCGGGAGTAGGGACAATGAGCAGTTCTTC", "TG~
## $ best.V.gene <chr> "TRBV7-3*00", "TRBV6-2*00", "TRBV27*00", "TRBV20-1*00", "T~
## $ aa.seq      <chr> "RASSARTSGSRDNEQFF", "CASSYRGTAWETQYF", "CASRPLLDRNNEQFF",~
```

```
glimpse(database)
```

```
## Rows: 405
## Columns: 2
## $ aa.seq.db <chr> "CSVVDAAPGANVLTF", "CAWSPGPVNEQFF", "CSARASYEQYF", "CASSDSGT~
## $ epitope   <chr> "LLWNGPMAV", "LLWNGPMAV", "LLWNGPMAV", "LLWNGPMAV", "LLWNGPM~
```

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))
}

with(database, get_distances(aa.seq.db, aa.seq.db)) %>% head
```

```
##              aa.seq         aa.seq.db dist
## 1   CSVVDAAPGANVLTF   CSVVDAAPGANVLTF    0
## 2     CAWSPGPVNEQFF     CAWSPGPVNEQFF    0
## 3       CSARASYEQYF       CSARASYEQYF    0
## 4     CASSDSGTDTQYF     CASSDSGTDTQYF    0
## 5    CASSFGTGRAGYTF    CASSFGTGRAGYTF    0
## 6 CASSDWGGTGRGPEAFF CASSDWGGTGRGPEAFF    0
```

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))
}

with(database, get_1mm_pairs(aa.seq.db, aa.seq.db)) %>% head
```

```
## # A tibble: 6 x 5
## # Groups:   chunk.id, len [4]
##   chunk.id   len aa.seq          aa.seq.db        dist
##      <int> <int> <chr>           <chr>           <dbl>
## 1        1    11 CASSLMYEQYF     CASSLMYEQYF         0
## 2        1    12 CASSEGIYGYTF    CASSEGIYGYTF        0
## 3        1    13 CATTGGSGYEQYF   CATTGGSGYEQYF       0
## 4        1    13 CASSGSSGYEQYF   CASSGSSGYEQYF       0
## 5        1    13 CSASHRAGNEQYF   CSASHRAGNEQYF       0
## 6        1    15 CSVVDAAPGANVLTF CSVVDAAPGANVLTF     0
```

Now the general routine for tables in original format. Sample table should come first, database should come second.

```
get_1mm_annot <- function(d, db) {
  pairs <- get_1mm_pairs(d$aa.seq, db$aa.seq.db) %>%
```

```
    inner_join(db)
  d %>%
    left_join(pairs) %>%
    select(-chunk.id, -len)
}

get_1mm_annot(data %>% head(100000), database) %>%
  filter(!is.na(epitope)) %>% head
```

```
## Joining, by = "aa.seq.db"
```

```
## Joining, by = "aa.seq"
```

```
## # A tibble: 6 x 10
##   sample.id replica time.point clone.count nt.seq   best.V.gene aa.seq aa.seq.db
##   <chr>     <chr>        <dbl>       <dbl> <chr>    <chr>       <chr>  <chr>
## 1 S2        F1               0         257 TGTGCCA~ TRBV12-4*00 CASSL~ CASSLAGS~
## 2 S2        F1               0         257 TGTGCCA~ TRBV12-4*00 CASSL~ CASSLGGA~
## 3 S2        F1               0         106 TGTGCCA~ TRBV12-4*00 CASSL~ CASSLAGS~
## 4 S2        F1               0         106 TGTGCCA~ TRBV12-4*00 CASSL~ CASSLGGA~
## 5 S2        F1               0          61 TGTGCCA~ TRBV3-1*00  CASSQ~ CASSSAGA~
## 6 S2        F1               0          57 TGTGCCA~ TRBV12-4*00 CASSP~ CASSPGPT~
## # ... with 2 more variables: dist <dbl>, epitope <chr>
```

Compute final table

```
system.time({data.ann <- get_1mm_annot(data, database)})
```

```
## Joining, by = "aa.seq.db"
```

```
## Joining, by = "aa.seq"
```

```
##    user  system elapsed
## 113.929   3.620 117.692
```

```
glimpse(data.ann)
```

```
## Rows: 4,548,994
## Columns: 10
## $ sample.id   <chr> "S2", "S2", "S2", "S2", "S2", "S2", "S2", "S2", "S2", "S2"~
## $ replica     <chr> "F1", "F1", "F1", "F1", "F1", "F1", "F1", "F1", "F1", "F1"~
## $ time.point  <dbl> 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0~
## $ clone.count <dbl> 5160, 2317, 2154, 2073, 1583, 915, 845, 811, 673, 673, 666~
## $ nt.seq      <chr> "CGTGCCAGCAGCGCCCGGACTAGCGGGAGTAGGGACAATGAGCAGTTCTTC", "TG~
## $ best.V.gene <chr> "TRBV7-3*00", "TRBV6-2*00", "TRBV27*00", "TRBV20-1*00", "T~
## $ aa.seq      <chr> "RASSARTSGSRDNEQFF", "CASSYRGTAWETQYF", "CASRPLLDRNNEQFF",~
## $ aa.seq.db   <chr> NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, NA~
## $ dist        <dbl> NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, NA~
## $ epitope     <chr> NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, NA~
```