VDJbet motif frequency - inverse k-mer frequency (MF-IKF) snippet

M.S.

2025-03-14

Load the 11 IMGT ‘Physicochemical’ classes of the 20 common amino acids (https://imgt.org/IMGTeducation/Aide-memoire/_UK/aminoacids/IMGTclasses.html#refs)

```
aa_table <- read_tsv("../assets/aa_classes.tsv")
```

```
## Rows: 24 Columns: 3
## ── Column specification ────────────────────────────────────────────────────────
## Delimiter: "\t"
## chr (3): aa, class_imgt, aa_imgt
##
## 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.
```

```
aa_dict <- aa_table$aa_imgt
names(aa_dict) <- aa_table$aa
aa_dict[c("D", "E", "K", "R", "F", "W")]
```

```
##   D   E   K   R   F   W
## "c" "c" "b" "b" "F" "W"
```

```
aa_translate <- function(seqs) {
  seqs |>
    sapply(\(seq)
           aa_dict[str_split_1(seq, pattern = "")] |>
             str_flatten()
    )
}
aa_translate(c("DEDE", "DRDWF"))
```

```
##    DEDE   DRDWF
##  "cccc" "cbcWF"
```

Kmerize function

```
get_kmer <- function(seq, pos, k = K, xpos = -1) {
  kmer <- substr(seq, pos, pos + k - 1)
  if (xpos > 0) {
    substr(kmer, xpos, xpos) <- "X"
  }
  kmer
}

get_kmers_1 <- function(seq, k = K, mask = 1:k, imgt_aa = T) {
  if (is.na(seq)) {
    return(tibble())
  }
  len <- nchar(seq)
  if (len < k) {
    return(tibble())
  }
  tbl <- expand_grid(mask = mask,
                     pos = 1:(len - k + 1))
  tbl$k <- k
  if (imgt_aa) {
    seqt <- aa_translate(seq)
  }
  tbl$seq <- seq
  tbl |>
    rowwise() |>
    mutate(kmer = get_kmer(seqt, pos, k, mask)) |>
    ungroup()
}

get_kmers_1("CASSLAPGATNEKLFF")
```

```
## # A tibble: 52 × 5
##     mask   pos     k seq              kmer
##    <int> <int> <dbl> <chr>            <chr>
##  1     1     1     4 CASSLAPGATNEKLFF Xlhh
##  2     1     2     4 CASSLAPGATNEKLFF Xhhl
##  3     1     3     4 CASSLAPGATNEKLFF Xhll
##  4     1     4     4 CASSLAPGATNEKLFF XllP
##  5     1     5     4 CASSLAPGATNEKLFF XlPG
##  6     1     6     4 CASSLAPGATNEKLFF XPGl
##  7     1     7     4 CASSLAPGATNEKLFF XGlh
##  8     1     8     4 CASSLAPGATNEKLFF Xlhm
##  9     1     9     4 CASSLAPGATNEKLFF Xhmc
## 10     1    10     4 CASSLAPGATNEKLFF Xmcb
## # ℹ 42 more rows
```

```
get_kmers <- function(seqs, k = K, mask = 1:k, imgt_aa = T) {
  as.list(seqs) |>
    lapply(\(seq) get_kmers_1(seq, k, mask, imgt_aa)) |>
    bind_rows() |>
    ungroup()
}

get_kmers(c("CAS",
            "CASSLA",
            "PGATNEKLFF"))
```

```
## # A tibble: 40 × 5
##     mask   pos     k seq    kmer
##    <int> <int> <dbl> <chr>  <chr>
##  1     1     1     4 CASSLA Xlhh
##  2     1     2     4 CASSLA Xhhl
##  3     1     3     4 CASSLA Xhll
##  4     2     1     4 CASSLA sXhh
##  5     2     2     4 CASSLA lXhl
##  6     2     3     4 CASSLA hXll
##  7     3     1     4 CASSLA slXh
##  8     3     2     4 CASSLA lhXl
##  9     3     3     4 CASSLA hhXl
## 10     4     1     4 CASSLA slhX
## # ℹ 30 more rows
```

Compute information content and frequency of K-mers across antigens, taking into account V and junction_aa_len: * IKF: smaller - more ubiquous K-mers like CASS/KLFF coming from V/J, larger - rare K-mers that are more likely to distinguish between antigens * MF: motif frequency in a given antigen, either K-mer, or K-mer+V, or K-mer+V+len frequency * confidence: for a given epitope k-mer should be seen at least 5 times, at least 3 times with a given V, at least 2 times with a given V for a given len

```
get_kmer_freqs <- function(seqs, k = K, mask = 1:k, imgt_aa = T) {
  get_kmers(seqs, k, mask, imgt_aa) |>
    group_by(k, mask, kmer) |>
    summarise(x_lva = length(unique(seq)),
              .groups = "drop")
}

data.vdjdb <- read_tsv("../assets/vdjdb-2025-02-21/vdjdb_full.txt") |>
  group_by(antigen.epitope) |>
  mutate(variants = length(unique(cdr3.beta)),
         junction_aa_len = nchar(cdr3.beta)) |>
  ungroup() |>
  filter(species == "HomoSapiens",
         antigen.epitope != "KLGGALQAK", # certified bad by 10X
         !is.na(cdr3.beta),
         variants >= 30,
         junction_aa_len >= 10) |>
  mutate(v.beta = str_split_fixed(v.beta, fixed("*"), 2)[,1]) |>
  select(junction_aa = cdr3.beta,
         v_call = v.beta,
         junction_aa_len,
         antigen = antigen.epitope,
         antigen_host = antigen.species) |>
  unique()
```

```
## Rows: 78826 Columns: 33
## ── Column specification ────────────────────────────────────────────────────
## Delimiter: "\t"
## chr (33): cdr3.alpha, v.alpha, j.alpha, cdr3.beta, v.beta, d.beta, j.beta, s...
##
## 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.vdjdb.kmer <- data.vdjdb |>
  group_by(antigen, antigen_host, v_call, junction_aa_len) |>
  group_modify(~get_kmer_freqs(.$junction_aa)) |>
  group_by(kmer, antigen, antigen_host, v_call) |>
  mutate(x_va = sum(x_lva)) |>
  group_by(kmer, antigen, antigen_host) |>
  mutate(x_a = sum(x_lva)) |>
  group_by(kmer) |>
  mutate(x = sum(x_lva)) |>
  left_join(data.vdjdb |>
              group_by(antigen, antigen_host, v_call, junction_aa_len) |>
              summarise(X_lva = length(unique(junction_aa))) |>
              group_by(antigen, antigen_host, v_call) |>
              mutate(X_va = sum(X_lva)) |>
              group_by(antigen, antigen_host) |>
              mutate(X_a = sum(X_lva)) |>
              ungroup() |>
              mutate(X = sum(X_lva))) |>
  group_by(kmer) |>
  mutate(MF_lva = x_lva / x,
         MF_va = x_va / x,
         MF_a = x_a / x, # frequency in antigen
         IKF_lva = mean(log(X_lva / x_lva)),
         IKF_va = mean(log(X_va / x_va)),
         IKF_a = mean(log(X_a / x_a)),
         IKF = log(X / x)) |> # information across antigens
  mutate(confidence = x_lva >= 2 & x_va >= 3 & x_a >= 5)
```

```
## `summarise()` has grouped output by 'antigen', 'antigen_host', 'v_call'. You
## can override using the `.groups` argument.
## Joining with `by = join_by(antigen, antigen_host, v_call, junction_aa_len)`
```

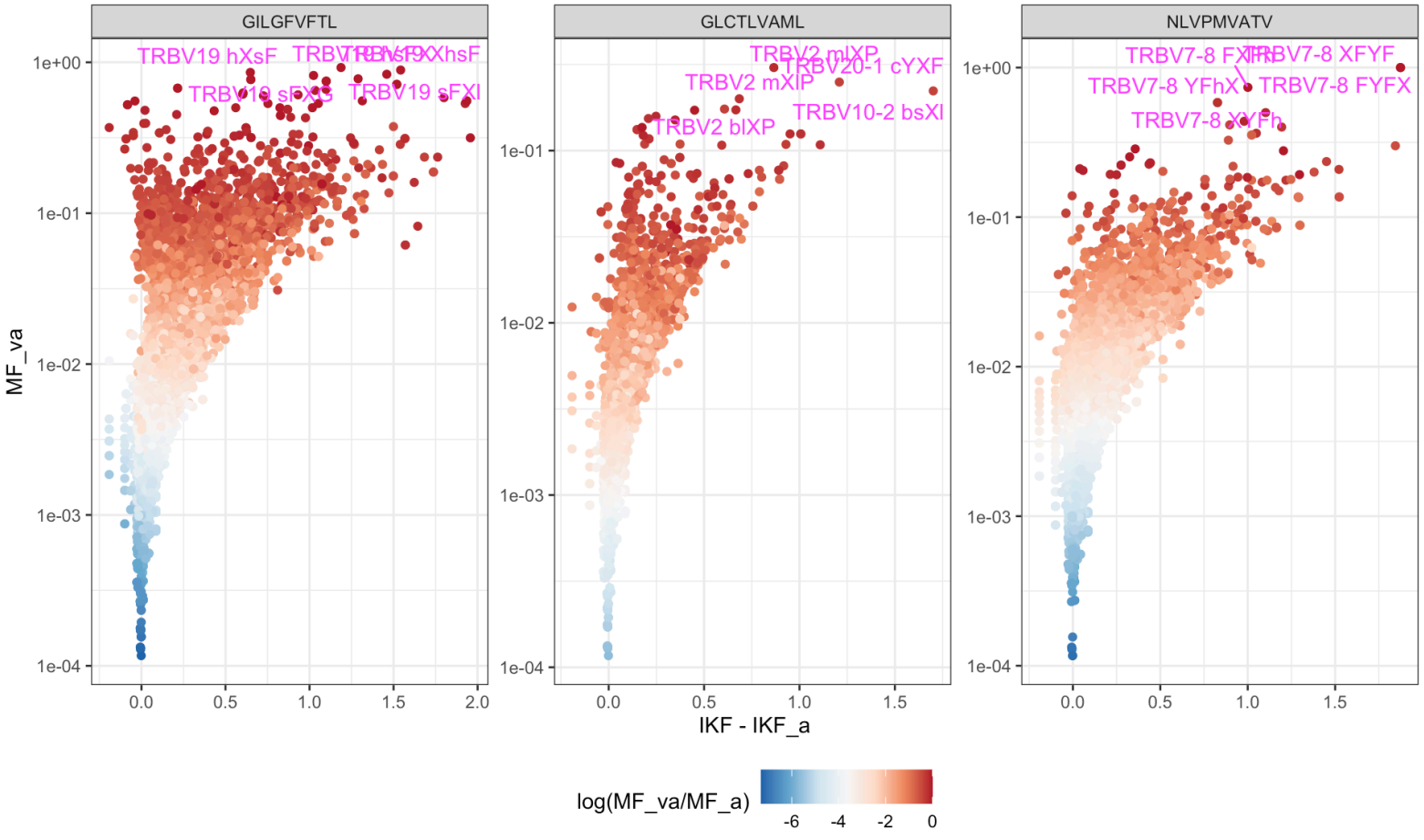
```
data.vdjdb.kmer
```

```
## # A tibble: 885,297 × 23
## # Groups:   kmer [4,942]
##    antigen    antigen_host v_call  junction_aa_len     k  mask kmer  x_lva  x_va
##    <chr>      <chr>        <chr>             <int> <dbl> <int> <chr> <int> <int>
##  1 ALAGIGILTV HomoSapiens  TRBV10…              11     4     1 XGlc      1     1
##  2 ALAGIGILTV HomoSapiens  TRBV10…              11     4     1 XcGl      1     1
##  3 ALAGIGILTV HomoSapiens  TRBV10…              11     4     1 XclF      1     1
##  4 ALAGIGILTV HomoSapiens  TRBV10…              11     4     1 XhcG      1     1
##  5 ALAGIGILTV HomoSapiens  TRBV10…              11     4     1 Xhhc      1     1
##  6 ALAGIGILTV HomoSapiens  TRBV10…              11     4     1 XlFF      1     1
##  7 ALAGIGILTV HomoSapiens  TRBV10…              11     4     1 Xlcl      1     1
##  8 ALAGIGILTV HomoSapiens  TRBV10…              11     4     1 Xlhh      1     1
##  9 ALAGIGILTV HomoSapiens  TRBV10…              11     4     2 GXcl      1     1
## 10 ALAGIGILTV HomoSapiens  TRBV10…              11     4     2 cXFF      1     1
## # ℹ 885,287 more rows
## # ℹ 14 more variables: x_a <int>, x <int>, X_lva <int>, X_va <int>, X_a <int>,
## #   X <int>, MF_lva <dbl>, MF_va <dbl>, MF_a <dbl>, IKF_lva <dbl>,
## #   IKF_va <dbl>, IKF_a <dbl>, IKF <dbl>, confidence <lgl>
```

Check K-mer behavior for well-known antigens

```
data.vdjdb.kmer3 <- data.vdjdb.kmer |>
  filter(confidence,
         substr(antigen, 1, 3) %in% c("NLV", "GIL", "GLC")) |>
  select(antigen, IKF, IKF_a, MF_va, MF_a, v_call, kmer) |>
  unique()

data.vdjdb.kmer3 |>
  ggplot(aes(x = IKF - IKF_a, y = MF_va, color = log(MF_va / MF_a))) +
  geom_point() +
  geom_text_repel(data = data.vdjdb.kmer3 |>
                    group_by(antigen) |>
                    filter(rank(-MF_va, ties.method = "first") <= 5),
                  aes(label = paste(v_call, kmer)),
                  color = "magenta") +
  scale_y_log10() +
  facet_wrap(~antigen, scales = "free") +
  scale_color_distiller(palette = "RdBu", direction = -1) +
  theme_bw() +
  theme(legend.position = "bottom")
```



Match well known specific sequences versus VDJdb. First we do it K-mer-wise

```
match_kmer_db <- function(seqs,
                          v_call,
                          db,
                          conf_filter = T,
                          k = K,
                          mask = 1:k,
                          imgt_aa = T) {
  get_kmers(seqs, k, mask, imgt_aa) |>
    left_join(tibble(seq = seqs,
                     v_call = v_call)) |>
    left_join(db |>
                filter(!conf_filter | confidence))
}

match_vdjdb_kmer <- function(seqs,
                             v_call,
                             conf_filter = T,
                             k = K,
                             mask = 1:k,
                             imgt_aa = T) {
  match_kmer_db(seqs, v_call, data.vdjdb.kmer, conf_filter, k, mask, imgt_aa) |>
    group_by(seq, v_call, kmer, antigen, antigen_host) |>
    mutate(mfikf = sum(MF_va * IKF)) |>
    arrange(-mfikf)
}

test_qq <- tibble(seq = c("CASSLAPGATNEKLFF",
                          "CASSIRSSYEQYF",
                          "CSARDRTGNGYTF"),
                  v_call = c("TRBV7-6",
                             "TRBV19",
                             "TRBV20-1"),
                  query_antigen = c("NLVPMVATV",
                                    "GILGFVFTL",
                                    "GLCTLVAML"))

# No length filter
with(test_qq,
     match_vdjdb_kmer(seq, v_call)) |>
  left_join(test_qq) |>
  group_by(seq) |>
  slice_max(order_by = mfikf, n = 5) |>
  mutate(match = query_antigen == antigen) |>
  select(kmer, mfikf, seq, pos, v_call, antigen, match) |>
  knitr::kable()
```

```
## Joining with `by = join_by(seq)`
## Joining with `by = join_by(mask, k, kmer, v_call)`
## Joining with `by = join_by(seq, v_call)`
```

kmer	mfikf	seq	pos	v_call	antigen	match
hXbh	5.4123346	CASSIRSSYEQYF	4	TRBV19	GILGFVFTL	TRUE
hXbh	5.4123346	CASSIRSSYEQYF	4	TRBV19	GILGFVFTL	TRUE

kmer	mfikf	seq	pos	v_call	antigen	match
hXbh	5.4123346	CASSIRSSYEQYF	4	TRBV19	GILGFVFTL	TRUE
hXbh	5.4123346	CASSIRSSYEQYF	4	TRBV19	GILGFVFTL	TRUE
hXbh	5.4123346	CASSIRSSYEQYF	4	TRBV19	GILGFVFTL	TRUE
XlIP	0.5137936	CASSLAPGATNEKLFF	4	TRBV7-6	NLVPMVATV	TRUE
XlIP	0.5137936	CASSLAPGATNEKLFF	4	TRBV7-6	NLVPMVATV	TRUE
lIPX	0.5137936	CASSLAPGATNEKLFF	5	TRBV7-6	NLVPMVATV	TRUE
lIPX	0.5137936	CASSLAPGATNEKLFF	5	TRBV7-6	NLVPMVATV	TRUE
PXlh	0.4268591	CASSLAPGATNEKLFF	7	TRBV7-6	NLVPMVATV	TRUE
shXb	7.9672533	CSARDRTGNGYTF	1	TRBV20-1	NLVPMVATV	FALSE
shXb	7.9672533	CSARDRTGNGYTF	1	TRBV20-1	NLVPMVATV	FALSE
shXb	7.9672533	CSARDRTGNGYTF	1	TRBV20-1	NLVPMVATV	FALSE
shXb	7.9672533	CSARDRTGNGYTF	1	TRBV20-1	NLVPMVATV	FALSE
shXb	7.9672533	CSARDRTGNGYTF	1	TRBV20-1	NLVPMVATV	FALSE
shXb	7.9672533	CSARDRTGNGYTF	1	TRBV20-1	NLVPMVATV	FALSE
shXb	7.9672533	CSARDRTGNGYTF	1	TRBV20-1	NLVPMVATV	FALSE
shXb	7.9672533	CSARDRTGNGYTF	1	TRBV20-1	NLVPMVATV	FALSE
shXb	7.9672533	CSARDRTGNGYTF	1	TRBV20-1	NLVPMVATV	FALSE
shXb	7.9672533	CSARDRTGNGYTF	1	TRBV20-1	NLVPMVATV	FALSE
shXb	7.9672533	CSARDRTGNGYTF	1	TRBV20-1	NLVPMVATV	FALSE
shXb	7.9672533	CSARDRTGNGYTF	1	TRBV20-1	NLVPMVATV	FALSE

```
# Length filter
with(test_qq,
     match_vdjdb_kmer(seq, v_call)) |>
  mutate(seq_len = nchar(seq)) |>
  left_join(test_qq) |>
  filter(abs(seq_len - junction_aa_len) <= 3) |>
  group_by(seq) |>
  slice_max(order_by = mfikf, n = 5) |>
  mutate(match = query_antigen == antigen) |>
  select(kmer, mfikf, seq, pos, v_call, antigen, match) |>
  knitr::kable()
```

```
## Joining with `by = join_by(seq)`
## Joining with `by = join_by(mask, k, kmer, v_call)`
## Joining with `by = join_by(seq, v_call)`
```

kmer	mfikf	seq	pos	v_call	antigen	match
hXbh	5.4123346	CASSIRSSYEQYF	4	TRBV19	GILGFVFTL	TRUE

kmer	mfikf	seq	pos	v_call	antigen	match
hXbh	5.4123346	CASSIRSSYEQYF	4	TRBV19	GILGFVFTL	TRUE
hXbh	5.4123346	CASSIRSSYEQYF	4	TRBV19	GILGFVFTL	TRUE
hXbh	5.4123346	CASSIRSSYEQYF	4	TRBV19	GILGFVFTL	TRUE
hXbh	5.4123346	CASSIRSSYEQYF	4	TRBV19	GILGFVFTL	TRUE
XlIP	0.5137936	CASSLAPGATNEKLFF	4	TRBV7-6	NLVPMVATV	TRUE
XlIP	0.5137936	CASSLAPGATNEKLFF	4	TRBV7-6	NLVPMVATV	TRUE
lIPX	0.5137936	CASSLAPGATNEKLFF	5	TRBV7-6	NLVPMVATV	TRUE
lIPX	0.5137936	CASSLAPGATNEKLFF	5	TRBV7-6	NLVPMVATV	TRUE
PXlh	0.4268591	CASSLAPGATNEKLFF	7	TRBV7-6	NLVPMVATV	TRUE
shXb	7.9672533	CSARDRTGNGYTF	1	TRBV20-1	NLVPMVATV	FALSE
shXb	7.9672533	CSARDRTGNGYTF	1	TRBV20-1	NLVPMVATV	FALSE
shXb	7.9672533	CSARDRTGNGYTF	1	TRBV20-1	NLVPMVATV	FALSE
shXb	7.9672533	CSARDRTGNGYTF	1	TRBV20-1	NLVPMVATV	FALSE
shXb	7.9672533	CSARDRTGNGYTF	1	TRBV20-1	NLVPMVATV	FALSE
shXb	7.9672533	CSARDRTGNGYTF	1	TRBV20-1	NLVPMVATV	FALSE
shXb	7.9672533	CSARDRTGNGYTF	1	TRBV20-1	NLVPMVATV	FALSE

```
# No length filter
with(test_qq,
     match_vdjdb_kmer(seq, v_call)) |>
  left_join(test_qq) |>
  group_by(seq, v_call, antigen, query_antigen) |>
  summarise(mfikf = sum(mfikf)) |>
  mutate(match = query_antigen == antigen) |>
  group_by(seq) |>
  slice_max(order_by = mfikf, n = 5) |>
  knitr::kable()
```

```
## Joining with `by = join_by(seq)`
## Joining with `by = join_by(mask, k, kmer, v_call)`
## Joining with `by = join_by(seq, v_call)`
## `summarise()` has grouped output by 'seq', 'v_call', 'antigen'. You can
## override using the `.groups` argument.
```

seq	v_call	antigen	query_antigen	mfikf	match
CASSIRSSYEQYF	TRBV19	GILGFVFTL	GILGFVFTL	365.3175309	TRUE
CASSIRSSYEQYF	TRBV19	NLVPMVATV	GILGFVFTL	8.4540987	FALSE
CASSIRSSYEQYF	TRBV19	AVFDRKSDAK	GILGFVFTL	2.2268397	FALSE
CASSIRSSYEQYF	TRBV19	IVTDFSVIK	GILGFVFTL	1.1652848	FALSE

seq	v_call	antigen	query_antigen	mfikf	match
CASSIRSSYEQYF	TRBV19	RAKFKQLL	GILGFVFTL	0.7657442	FALSE
CASSLAPGATNEKLFF	TRBV7-6	NLVPMVATV	NLVPMVATV	13.8570195	TRUE
CASSLAPGATNEKLFF	TRBV7-6	GILGFVFTL	NLVPMVATV	0.3263157	FALSE
CASSLAPGATNEKLFF	TRBV7-6	RAKFKQLL	NLVPMVATV	0.2183970	FALSE
CASSLAPGATNEKLFF	TRBV7-6	GLCTLVAML	NLVPMVATV	0.1667241	FALSE
CASSLAPGATNEKLFF	TRBV7-6	AVFDRKSDAK	NLVPMVATV	0.1260994	FALSE
CSARDRTGNGYTF	TRBV20-1	NLVPMVATV	GLCTLVAML	465.8287417	FALSE
CSARDRTGNGYTF	TRBV20-1	GILGFVFTL	GLCTLVAML	135.3621035	FALSE
CSARDRTGNGYTF	TRBV20-1	GLCTLVAML	GLCTLVAML	65.5864369	TRUE
CSARDRTGNGYTF	TRBV20-1	YLQPRTFLL	GLCTLVAML	36.4919322	FALSE
CSARDRTGNGYTF	TRBV20-1	AVFDRKSDAK	GLCTLVAML	33.9740948	FALSE

```
# Length filter
with(test_qq,
     match_vdjdb_kmer(seq, v_call)) |>
  left_join(test_qq) |>
  mutate(seq_len = nchar(seq)) |>
  filter(abs(seq_len - junction_aa_len) <= 3) |>
  group_by(seq, v_call, antigen, query_antigen) |>
  summarise(mfikf = sum(mfikf)) |>
  mutate(match = query_antigen == antigen) |>
  group_by(seq) |>
  slice_max(order_by = mfikf, n = 5) |>
  knitr::kable()
```

```
## Joining with `by = join_by(seq)`
## Joining with `by = join_by(mask, k, kmer, v_call)`
## Joining with `by = join_by(seq, v_call)`
## `summarise()` has grouped output by 'seq', 'v_call', 'antigen'. You can
## override using the `.groups` argument.
```

seq	v_call	antigen	query_antigen	mfikf	match
CASSIRSSYEQYF	TRBV19	GILGFVFTL	GILGFVFTL	335.3922415	TRUE
CASSIRSSYEQYF	TRBV19	NLVPMVATV	GILGFVFTL	7.3140699	FALSE
CASSIRSSYEQYF	TRBV19	AVFDRKSDAK	GILGFVFTL	1.8967472	FALSE
CASSIRSSYEQYF	TRBV19	IVTDFSVIK	GILGFVFTL	1.1652848	FALSE
CASSIRSSYEQYF	TRBV19	RAKFKQLL	GILGFVFTL	0.7657442	FALSE
CASSLAPGATNEKLFF	TRBV7-6	NLVPMVATV	NLVPMVATV	13.3468877	TRUE
CASSLAPGATNEKLFF	TRBV7-6	GILGFVFTL	NLVPMVATV	0.3125806	FALSE
CASSLAPGATNEKLFF	TRBV7-6	RAKFKQLL	NLVPMVATV	0.2183970	FALSE

seq	v_call	antigen	query_antigen	mfikf	match
CASSLAPGATNEKLFF	TRBV7-6	GLCTLVAML	NLVPMVATV	0.1304446	FALSE
CASSLAPGATNEKLFF	TRBV7-6	AVFDRKSDAK	NLVPMVATV	0.1260994	FALSE
CSARDRTGNGYTF	TRBV20-1	NLVPMVATV	GLCTLVAML	288.6056787	FALSE
CSARDRTGNGYTF	TRBV20-1	GILGFVFTL	GLCTLVAML	97.2336423	FALSE
CSARDRTGNGYTF	TRBV20-1	GLCTLVAML	GLCTLVAML	55.9234579	TRUE
CSARDRTGNGYTF	TRBV20-1	YLQPRTFLL	GLCTLVAML	31.8877060	FALSE
CSARDRTGNGYTF	TRBV20-1	AVFDRKSDAK	GLCTLVAML	26.7486546	FALSE

```
# END todo: think how to fix GLC, need V usage adjustment
```