

R Notebook

Analysis of data from mutation annotated tree

```
red <- "#e31919"  
blue1 <- "#5450f2"  
threshold_branch_length <- 10  
library(Biostrings)
```

Loading required package: BiocGenerics

Attaching package: 'BiocGenerics'

The following objects are masked from 'package:stats':

IQR, mad, sd, var, xtabs

The following objects are masked from 'package:base':

anyDuplicated, aperm, append, as.data.frame, basename, cbind,
colnames, dirname, do.call, duplicated, eval, evalq, Filter, Find,
get, grep, grepl, intersect, is.unsorted, lapply, Map, mapply,
match, mget, order, paste, pmax, pmax.int, pmin, pmin.int,
Position, rank, rbind, Reduce, rownames, sapply, setdiff, sort,
table, tapply, union, unique, unsplit, which.max, which.min

Loading required package: S4Vectors

Loading required package: stats4

```
Attaching package: 'S4Vectors'
```

```
The following objects are masked from 'package:base':
```

```
    expand.grid, I, unname
```

```
Loading required package: IRanges
```

```
Loading required package: XVector
```

```
Loading required package: GenomeInfoDb
```

```
Attaching package: 'Biostrings'
```

```
The following object is masked from 'package:base':
```

```
    strsplit
```

```
library(tidyverse)
```

```
-- Attaching packages ----- tidyverse 1.3.2 --
```

```
v ggplot2 3.4.2      v purrr  1.0.1
v tibble  3.2.1      v dplyr  1.1.2
v tidyr   1.3.0      v stringr 1.5.0
v readr   2.1.4      v forcats 1.0.0
```

```
-- Conflicts ----- tidyverse_conflicts() --
```

```
x dplyr::collapse() masks Biostrings::collapse(), IRanges::collapse()
x dplyr::combine()   masks BiocGenerics::combine()
x purrr::compact()   masks XVector::compact()
x dplyr::desc()       masks IRanges::desc()
x tidyr::expand()     masks S4Vectors::expand()
x dplyr::filter()     masks stats::filter()
x dplyr::first()      masks S4Vectors::first()
x dplyr::lag()        masks stats::lag()
x ggplot2::Position() masks BiocGenerics::Position(), base::Position()
x purrr::reduce()     masks IRanges::reduce()
x dplyr::rename()     masks S4Vectors::rename()
x dplyr::slice()      masks XVector::slice(), IRanges::slice()
```

```
data_nodes <- read_tsv("~/Dropbox/new_mov2/all_nodes.tsv.gz")
```

```
Rows: 17849624 Columns: 19
```

```
-- Column specification -----
```

```
Delimiter: "\t"
```

```
chr (4): node_id, consensus_country, consensus_year, age
```

```
dbl (14): num_descendants, date_length, A>C, A>G, A>T, C>A, C>G, C>T, G>A, ...
```

```
date (1): date
```

```
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_nodes <- data_nodes %>% mutate(total_muts = `A>C` + `A>G` + `A>T` + `C>A` + `C>G` + `C>T` + `G>A` + `G>C` + `G>T` + `T>A` + `T>C` + `T>G` + `T>T`)
```

```
data_muts <- read_tsv("~/Dropbox/new_mov2/all_node_muts.tsv.gz")
```

```
Rows: 17193547 Columns: 11
```

```
-- Column specification -----
```

```
Delimiter: "\t"
```

```
chr (8): node_id, original_nt, alternative_nt, gene, original_aa, alternativ...
```

```
dbl (2): nt_index, aa_index
```

```
lgl (1): is_synonymous
```

```
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.
```

```
parenthood <- read_tsv("~/Dropbox/new_mov2/parenthood.tsv.gz")
```

```
Rows: 17849623 Columns: 2
```

```
-- Column specification -----
```

```
Delimiter: "\t"
```

```
chr (2): child, parent
```

```
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.
```

```

find_children <- function(parenthood, parent) {
  # Find the immediate children of the parent
  children <- parenthood$child[parenthood$parent == parent]

  # Initialize a vector to store all descendants
  all_descendants <- c()

  # Loop through each child and find their descendants
  for (child in children) {
    # Add the child to the list of descendants
    all_descendants <- c(all_descendants, child)

    # Recursively find the descendants of the child
    child_descendants <- find_children(parenthood, child)

    # Add the descendants of the child to the list of all descendants
    all_descendants <- c(all_descendants, child_descendants)
  }

  return(all_descendants)
}

get_parent <- function(parenthood, node) {
  # Find the parent of the node
  parent <- parenthood$parent[parenthood$child == node]

  # If there is no parent (i.e., the node is the root), return NULL
  if (length(parent) == 0) {
    return(NULL)
  }

  return(parent)
}

data_muts <- data_muts %>% filter(gene != "ORF1a")

library(tidyverse)
library(cowplot)
data2 <- read_tsv("~/Dropbox/metadata_2023-04-29_01-16.tsv.gz", col_select = c("date", "co

```

Rows: 15198803 Columns: 2

```
-- Column specification -----
Delimiter: "\t"
chr (2): date, country

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.
```

```
data3 <- data2 %>%
  select(date, country) %>%
  extract(date, "(\\d{4})", into = "year")
countries_totals <- data3 %>%
  group_by(year, country) %>%
  tally() %>%
  mutate(total_genomes = n)

countries_totals
```

```
# A tibble: 724 x 4
# Groups:   year [10]
  year country      n total_genomes
  <chr> <chr>    <int>         <int>
1 2010 Cambodia      2           2
2 2013 China         1           1
3 2017 China         7           7
4 2018 China         1           1
5 2019 China        34          34
6 2020 Afghanistan   9           9
7 2020 Albania        7           7
8 2020 Algeria       95          95
9 2020 Andorra        1           1
10 2020 Angola       151         151
# i 714 more rows
```

```
library(ggrepel)

tallied_big <- data_nodes %>%
  dplyr::rename(country = consensus_country, year = consensus_year) %>%
  filter(flagged, total_muts >= threshold_branch_length) %>%
  group_by(country, year) %>%
  tally() %>%
```

```
dplyr::rename(ga_branches = n) %>%
full_join(countries_totals) %>%
replace_na(list("ga_branches" = 0))
```

Joining with `by = join_by(country, year)`

```
tallied <- tallied_big %>% filter(year == "2022")

# Define approved and not_approved countries
approved <- c(
  "USA", "United Kingdom", "Germany", "Denmark", "Japan", "India", "Australia", "Israel",
  "Russia", "South Korea", "New Zealand", "Belgium", "Mauritius", "Vietnam", "Thailand", "
)
not_approved <- c(
  "France", "Canada", "Sweden", "Netherlands", "Finland", "Switzerland", "Norway", "Ireland", "
)

# Define usage
usage <- c(
  "Australia" = "\n(100 per 10k)",
  "United Kingdom" = "\n(5 per 10k)",
  "Japan" = "(50 per 10k)",
  "Italy" = "\n(10 per 10k)"
)

# List of years
years <- c("2020", "2021", "2022", "2023")
lightpurple <- "#c39ecd"
darkpurple <- "#77488c"
darkorange <- "#fe670a"
lightorange <- "#f1ae85"
midorange <- "#ff883c"
year_pal <- c(lightpurple, darkpurple, darkorange, lightorange)
names(year_pal) <- years

# Loop through each year
for (i in 0:length(years)) {
```

```

# Subset data
data_subset <- data_nodes %>%
  filter(total_muts > 20, consensus_year %in% years[0:i])

# Define plot
scatter <- ggplot(data_subset, aes(x = `G>A` / total_muts, y = transitions / total_muts,
  geom_point() +
  theme_bw() +
  labs(x = "G\u00adto\u00adA proportion", y = "Transition proportion", color = "Year") +
  scale_color_manual(values = year_pal) +
  theme(legend.position = "bottom") +
  scale_x_continuous(label = scales::percent) +
  scale_y_continuous(label = scales::percent) +
  coord_cartesian(xlim = c(0, 0.65), ylim = c(0, 1))

# Save plot
ggsave(paste0("big_scatter_big_", paste(years[0:i], collapse = "_"), ".pdf"), plot = scatter)
ggsave(paste0("scatter_big_", paste(years[0:i], collapse = "_"), ".pdf"), plot = scatter)
}

scatter <- scatter +
  annotate("rect", xmin = 0.25, xmax = 0.6, ymin = 0.6, ymax = 1.05, fill = NA, color = "#

tallied$approved <- case_when(
  tallied$country %in% approved ~ "Available",
  tallied$country %in% not_approved ~ "Not available",
  TRUE ~ "Not identified"
)

country_plot_data = tallied %>% filter(country != "?", total_genomes > 500, year == "2022")

library(knitr)

library(knitr)
library(kableExtra)

```

Attaching package: 'kableExtra'

The following object is masked from 'package:dplyr':

group_rows

```
forlatex = country_plot_data %>% select(country, ga_branches, total_genomes) %>% arrange(-total_genomes)

country_plot_data
```

A tibble: 103 x 7

Groups: country [103]

	country	year	ga_branches	n	total_genomes	approved	usage
	<chr>	<chr>	<int>	<int>	<int>	<fct>	<chr>
1	Australia	2022	149	121602	121602	Available	"\n(100 ~
2	Austria	2022	8	46962	46962	Available	""
3	Belgium	2022	2	84600	84600	Available	""
4	Cambodia	2022	1	1833	1833	<NA>	""
5	Canada	2022	1	217040	217040	Not available	""
6	Czech Republic	2022	4	32124	32124	Available	""
7	Denmark	2022	10	332006	332006	Available	""
8	Egypt	2022	1	1621	1621	<NA>	""
9	France	2022	4	328527	328527	Not available	""
10	Georgia	2022	2	1805	1805	<NA>	""

i 93 more rows

```
names(forlatex) <- c("Country", "High G-to-A branches in 2022", "Total genomes in 2022")
```

```
latex_table <- kable(forlatex, "latex", booktabs = TRUE, linesep = "" ,
                      col.names = names(forlatex),
                      align = c('l', 'r', 'r'))
writeLines(latex_table, "~/movmanuscript2/Figures2/countrytable.tex")
```

```
country_comp <- ggplot(
  country_plot_data,
  aes( # color = approved,
    x = total_genomes, y = ifelse(ga_branches == 0, 0.8, ga_branches), label = country
  )
)
```



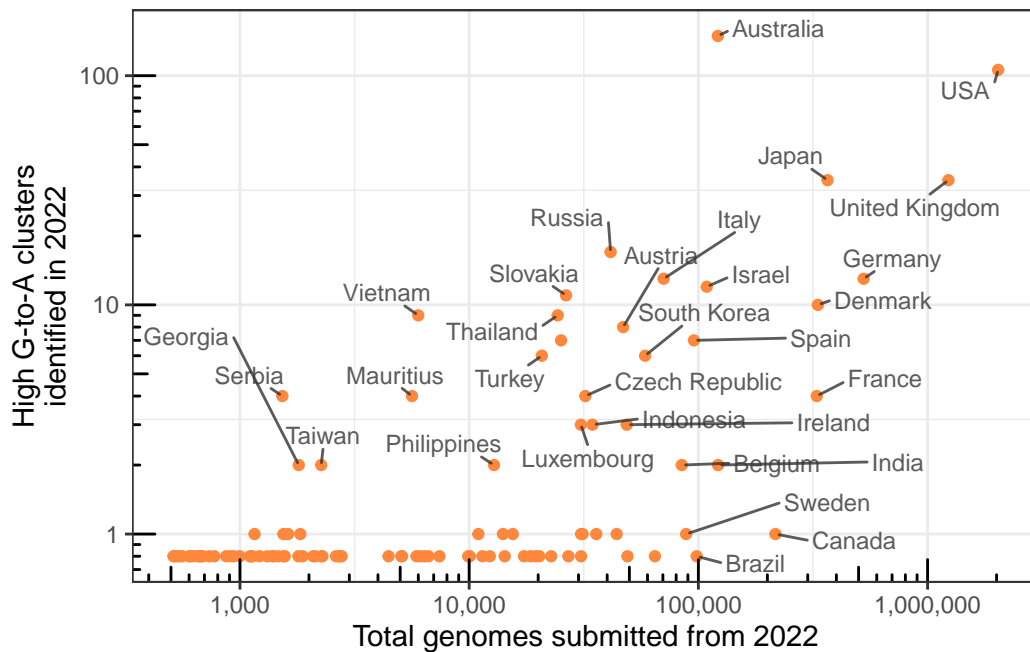
```

) +
  geom_point(alpha = 1, color = midorange) +
  scale_x_log10(labels = scales::comma) +
  scale_y_log10() +
  geom_text_repel(alpha = 0.8, max.overlaps = 7, force = 50, min.segment.length = 0, lineheight = 1.2) +
  theme_bw() +
  labs(x = "Total genomes submitted from 2022", y = "High G-to-A clusters identified in 2022") +
  theme(legend.position = "none") +
  annotation_logticks()

```

country_comp

Warning: ggrepel: 72 unlabeled data points (too many overlaps). Consider increasing max.overlaps



```

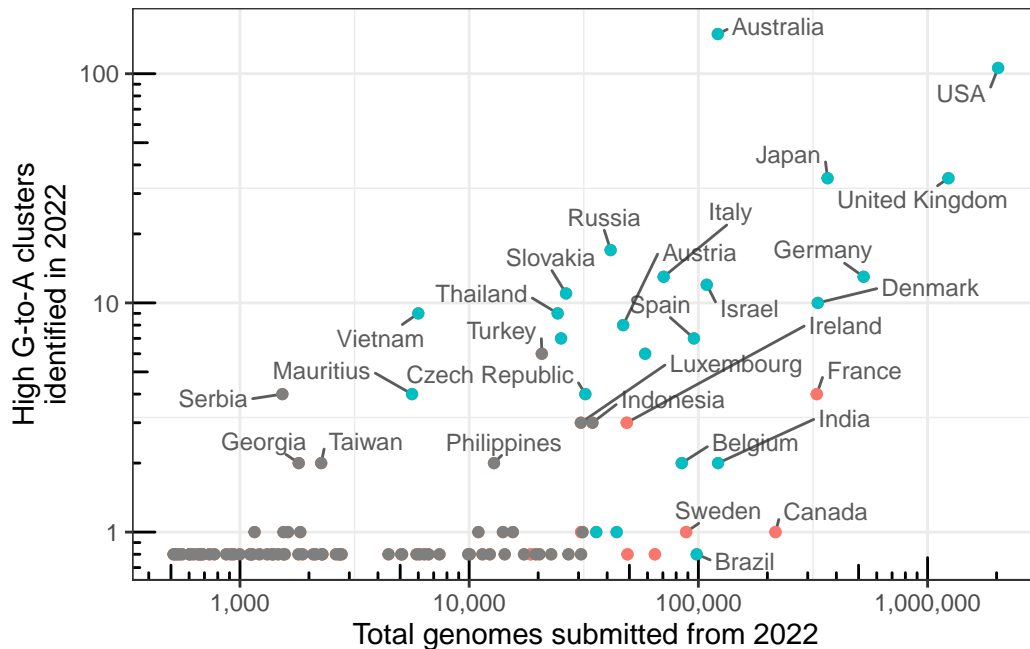
ggsave("country_scatter_big.pdf", width = 4, height = 3.5)

```

Warning: ggrepel: 76 unlabeled data points (too many overlaps). Consider increasing max.overlaps

```
country_comp + geom_point(aes(color = approved))
```

Warning: ggrepel: 73 unlabeled data points (too many overlaps). Consider increasing max.overlaps



```
recents <- data_nodes %>% filter(total_muts >= threshold_branch_length, consensus_year ==

recents$branch_type <- ifelse(recents$flagged, "High\nG\u00adto\u00adA", "Other")
recents$branch_type <- fct_relevel(recents$branch_type, "Other")

age_dataset <- recents %>% filter(consensus_country %in% c("USA"), total_muts > 0)
age <- ggplot(
  age_dataset,
  aes(x = branch_type, y = as.numeric(age), fill = branch_type)
) +
  geom_violin(alpha = 0.7) +
  # geom_jitter(height=0) +
  theme_bw() +
  geom_boxplot(alpha = 0.8, width = 0.15, fill = "white") +
  labs(x = "Branch type", y = "Age") +
```

```

scale_fill_manual(values = c("High\nG\u00adto\u00adA" = "#e31919", "Other" = "#5450f2"))
theme(legend.position = "none")

t.test(as.numeric(age) ~ branch_type, data = age_dataset)

```

Warning in eval(predvars, data, env): NAs introduced by coercion

Welch Two Sample t-test

```

data:  as.numeric(age) by branch_type
t = -7.6281, df = 74.023, p-value = 6.512e-11
alternative hypothesis: true difference in means between group Other and group High
GtoA is not equal to 0
95 percent confidence interval:
 -21.69284 -12.70724
sample estimates:
      mean in group Other mean in group High\nGtoA
          43.07774              60.27778

```

```

ggsave("age_violins.pdf", width = 3, height = 3)

```

Warning in FUN(X[[i]], ...): NAs introduced by coercion

Warning in FUN(X[[i]], ...): NAs introduced by coercion

Warning in FUN(X[[i]], ...): NAs introduced by coercion

Warning: Removed 1504 rows containing non-finite values (`stat_ydensity()`).

Warning: Removed 1504 rows containing non-finite values (`stat_boxplot()`).

```

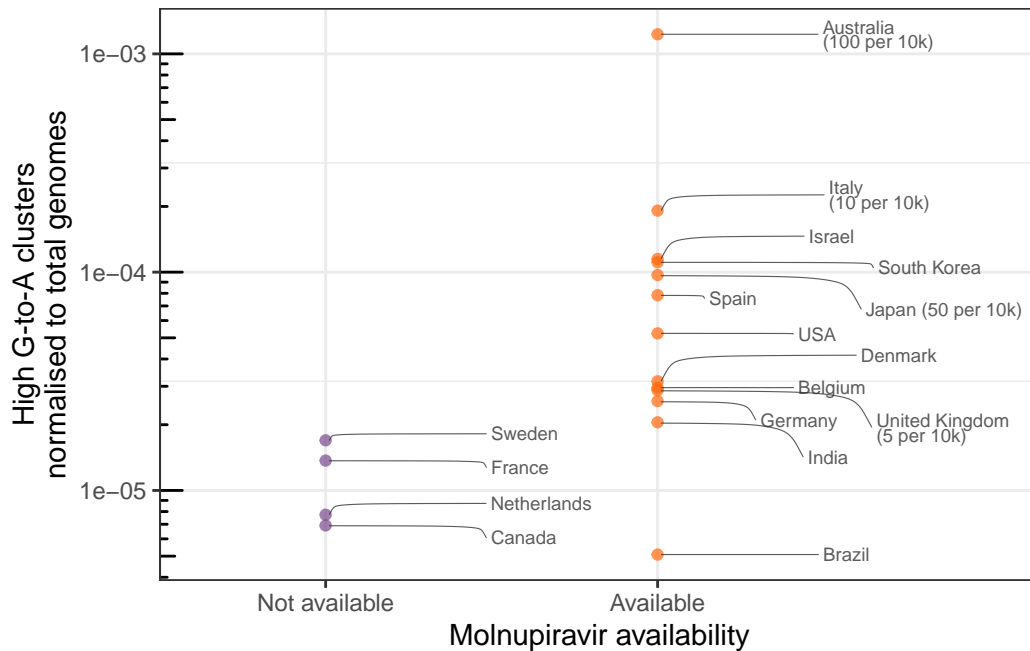
set.seed(339)
availability_dataset <- tallied %>%
  filter(country != "?", total_genomes > 50000) %>%
  mutate(usage = usage[country]) %>%
  mutate(usage = ifelse(is.na(usage), "", usage)) %>%

```

```

mutate(approved = factor(as.character(approved), levels = c("Not available", "Available"))
availability_plot <- ggplot(availability_dataset, aes(color = approved, x = approved, y =
  geom_point(alpha = 0.7) +
  scale_y_log10() +
  geom_text_repel(
    alpha = 0.8, force = 10, min.segment.length = 0, lineheight = .65, size = 2.5, color =
    # do not pull text toward the point at (0,0)
    max.time = 3,
    segment.square = TRUE,
    segment.size = 0.2,
    segment.curvature = 0.3,
    max.iter = 1e7, nudge_x = 0.5,
    max.overlaps = Inf,
    hjust = 0
  ) +
  theme_bw() +
  labs(x = "Molnupiravir availability", color = "Molnupiravir", y = "High G\u00adto\u00adA
  scale_color_manual(values = c("Not identified" = "gray", "Available" = darkorange, "Not
  theme(legend.position = "none") +
  annotation_logticks(sides = "l") +
  scale_x_discrete(
    expand = expansion(mult = c(0.5, 1.15))
  )
availability_plot

```



```
t.test(log10(ga_branches + 0.5) / total_genomes ~ approved, data = availability_dataset)
```

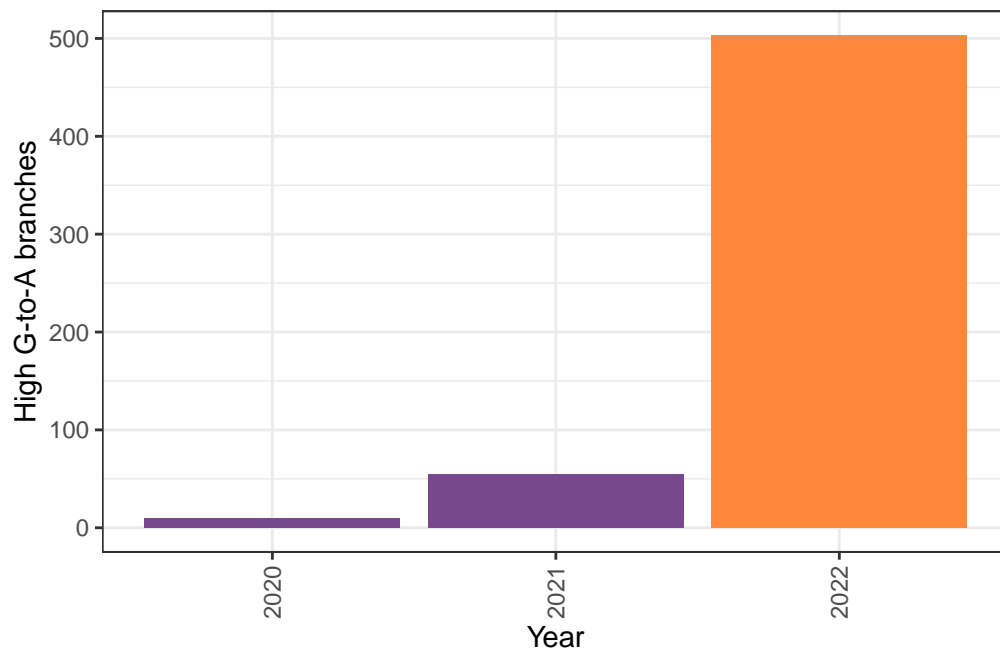
Welch Two Sample t-test

```
data: log10(ga_branches + 0.5)/total_genomes by approved
t = -2.6847, df = 10.935, p-value = 0.02133
alternative hypothesis: true difference in means between group Not available and group Available
95 percent confidence interval:
 -1.165237e-05 -1.149542e-06
sample estimates:
mean in group Not available    mean in group Available
      3.307937e-08              6.434033e-06
```

```
ggsave("availability.pdf", width = 3.5, height = 3.5)
```

```
by_year <- data_nodes %>%
  filter(flagged, total_muts >= threshold_branch_length) %>%
  group_by(consensus_year) %>%
  tally()
```

```
by_year_plot <- ggplot(by_year %>% filter(consensus_year %in% c("2021", "2022", "2020")),
  geom_col() +
  theme_bw() +
  labs(x = "Year", y = "\nHigh G-to-A branches") +
  theme(axis.text.x = element_text(angle = 90, vjust = 0.5, hjust = 1)) +
  scale_fill_manual(values = c(darkpurple, darkpurple, midorange)) +
  theme(legend.position = "none")
by_year_plot
```



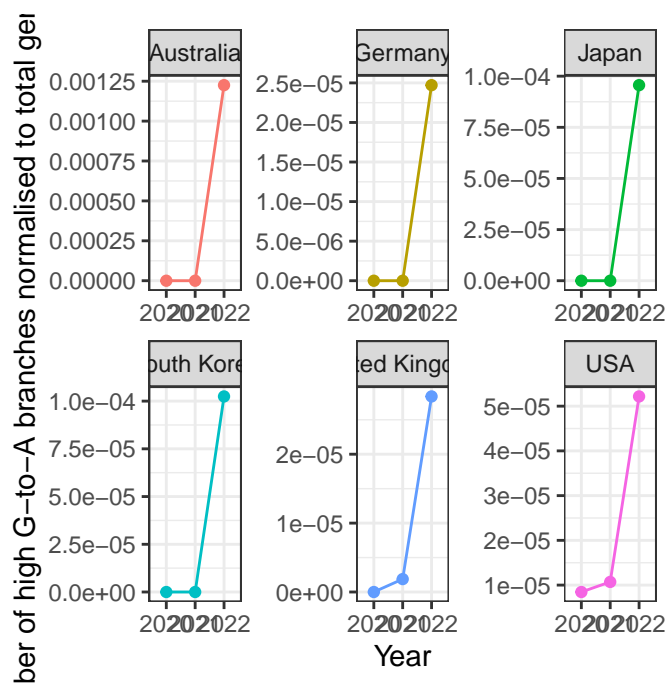
```
ggsave("byyearplot.pdf", width = 2, height = 3)
```

We also display data on timecourse where we normalise for total genome numbers, use a non log axis, and include data for 2023.

```
tallied_big <- tallied_big %>% mutate(p = (ga_branches) / total_genomes)

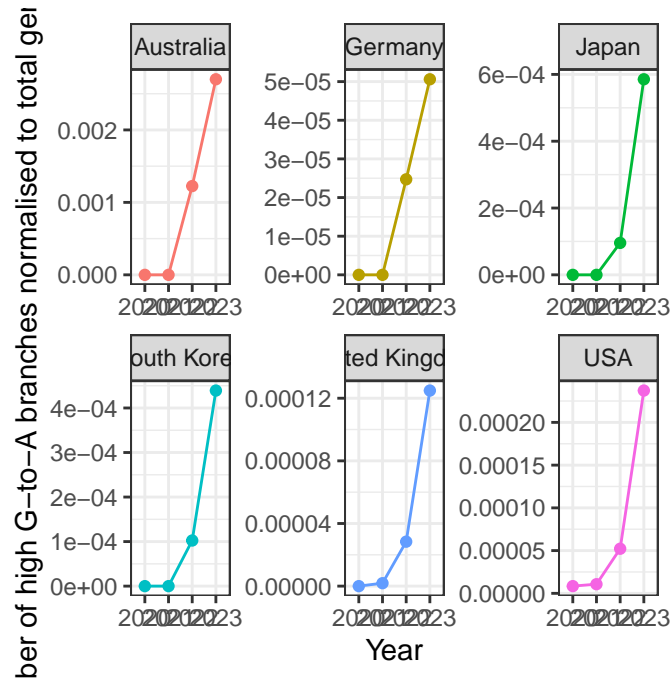
ggplot(tallied_big %>% filter(country %in% c("Australia", "United Kingdom", "USA", "Japan")
  geom_line() +
  geom_point() +
  theme_bw() +
  facet_wrap(~country, scales = "free") +
```

```
theme(legend.position = "none") +
labs(y = "Number of high G-to-A branches normalised to total genomes", x = "Year")
```



```
ggsave("~/movmanuscript2/Figures2/supp-countries_timeline.pdf", width = 7.5, height = 4.5)
```

```
ggplot(tallied_big %>% filter(country %in% c("Australia", "United Kingdom", "USA", "Japan"))
  geom_line() +
  geom_point() +
  theme_bw() +
  facet_wrap(~country, scales = "free") +
  theme(legend.position = "none") +
  labs(y = "Number of high G-to-A branches normalised to total genomes", x = "Year")
```



Processing and analysis of existing genomic datasets

```
library(tidyverse)
tidyverse_conflicts()
```

```
-- Conflicts ----- tidyverse_conflicts() --
x dplyr::collapse()      masks Biostrings::collapse(), IRanges::collapse()
x dplyr::combine()       masks BiocGenerics::combine()
x purrr::compact()       masks XVector::compact()
x dplyr::desc()          masks IRanges::desc()
x tidyr::expand()        masks S4Vectors::expand()
x dplyr::filter()        masks stats::filter()
x dplyr::first()         masks S4Vectors::first()
x kableExtra::group_rows() masks dplyr::group_rows()
x dplyr::lag()           masks stats::lag()
x ggplot2::Position()    masks BiocGenerics::Position(), base::Position()
x purrr::reduce()        masks IRanges::reduce()
x dplyr::rename()        masks S4Vectors::rename()
x dplyr::slice()         masks XVector::slice(), IRanges::slice()
```



```
nuc_genome_counts <- read_csv("./context_count.csv") %>% dplyr::rename(
  par = residue, context_before = residue_before, context_after = residue_after,
  genome_count = count
)
```

Rows: 64 Columns: 4

```
-- Column specification -----
Delimiter: ","
chr (3): residue_before, residue, residue_after
dbl (1): 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.

```
a <- read_csv("./molnupiravir_rescaled_samples.csv") %>% mutate(trial = "2", treat = "mov"
```

Rows: 192 Columns: 2

```
-- Column specification -----
Delimiter: ","
chr (1): Substitution
dbl (1): Number_of_mutations
```

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.

```
b <- read_csv("./MOV_rescaled_contexts_only.csv") %>% mutate(trial = "2", treat = "mov", c
```

Rows: 192 Columns: 2

```
-- Column specification -----
Delimiter: ","
chr (1): Substitution
dbl (1): Number_of_mutations
```

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.

```
c <- read_csv("./naive_rescaled_contexts_only.csv") %>% mutate(trial = "2", treat = "naive", com
```

```
Rows: 192 Columns: 2
```

```
-- Column specification -----
```

```
Delimiter: ","
```

```
chr (1): Substitution
```

```
dbl (1): Number_of_mutations
```

```
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.
```

```
d <- read_csv("./naive_rescaled_samples.csv") %>% mutate(trial = "2", treat = "naive", com
```

```
Rows: 192 Columns: 2
```

```
-- Column specification -----
```

```
Delimiter: ","
```

```
chr (1): Substitution
```

```
dbl (1): Number_of_mutations
```

```
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.
```

```
e <- read_csv("./agile_placebo_spectrum.csv") %>% mutate(trial = "1", treat = "naive", com
```

```
Rows: 192 Columns: 2
```

```
-- Column specification -----
```

```
Delimiter: ","
```

```
chr (1): Substitution
```

```
dbl (1): Number_of_mutations
```

```
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.
```

```
f <- read_csv("./agile_molnupiravir_spectrum.csv") %>% mutate(trial = "1", treat = "mov",
```

```

Rows: 192 Columns: 2
-- Column specification -----
Delimiter: ","
chr (1): Substitution
dbl (1): Number_of_mutations

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.

```

```
g <- read_csv("./BA.1_SBS_spectrum_Ruis.csv") %>% mutate(trial = "3", treat = "normal", co
```

```

Rows: 192 Columns: 2
-- Column specification -----
Delimiter: ","
chr (1): Substitution
dbl (1): Number_of_mutations

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.

```

```
long <- read_csv("./long_branch_spectrum_rescaled.csv") %>% mutate(trial = "4", treat = "1
```

```

Rows: 192 Columns: 2
-- Column specification -----
Delimiter: ","
chr (1): Substitution
dbl (1): Number_of_mutations

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.

```

```
specific <- read_csv("./molnupiravir_spectrum_specific.csv") %>% mutate(trial = "5", treat
```

```

Rows: 192 Columns: 2
-- Column specification -----
Delimiter: ","
chr (1): Substitution

```

```
dbl (1): Number_of_mutations
```

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.

```
colors <- c("#3055a2", "#221f20", "#da4837", "#939598", "#3f8347", "#edb9c0", "#4a68af", "
my_levels <- c("C\u00adto\u00adA", "C\u00adto\u00adG", "C\u00adto\u00adT", "T\u00adto\u00adA", "T\u00adto\u00adG", "T\u00adto\u00adT")

combo <- bind_rows(a, b, c, d, e, f, g, long, specific) %>%
  filter(!contexts_only) %>%
  separate(Substitution, into = c("context_before", "par", "mut", "context_after"), sep = " ")

data <- combo %>% mutate(mutation_type = factor(paste0(par, "\u00adto\u00ad", mut),
  levels = my_levels
))
```

For convenience to get the total number of each type of mutation we reverse MutTui's normalisations of context numbers.

```
totals <- data %>%
  group_by(trial) %>%
  summarise(total = sum(Number_of_mutations))

normed <- data %>%
  inner_join(totals) %>%
  mutate(Number_of_mutations = Number_of_mutations / total)
```

Joining with ``by = join_by(trial)``

```
multiplied <- normed %>%
  inner_join(nuc_genome_counts) %>%
  mutate(Number_of_mutations = Number_of_mutations * genome_count)
```

Joining with ``by = join_by(context_before, par, context_after)``

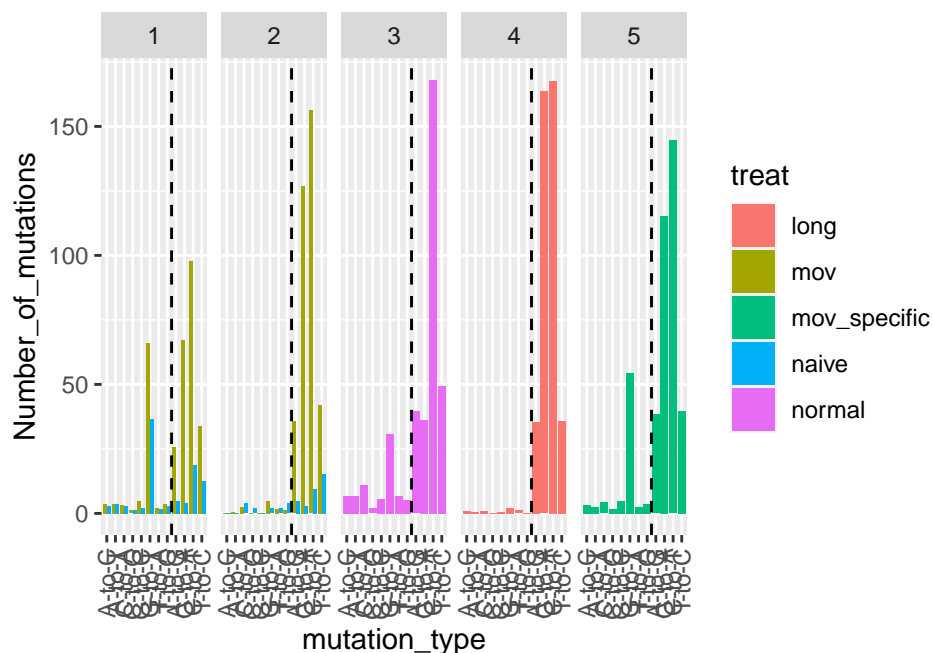
```
just_class <- multiplied %>%
  group_by(mutation_type, treat, trial) %>%
  summarise(Number_of_mutations = sum(Number_of_mutations))
```

``summarise()`` has grouped output by 'mutation_type', 'treat'. You can override using the ``groups`` argument.

```
transversions <- c("A\u00adto\u00adC", "A\u00adto\u00adT", "C\u00adto\u00adA", "C\u00adto\u00adG",
                  "G\u00adto\u00adA", "G\u00adto\u00adC", "T\u00adto\u00adA", "T\u00adto\u00adC")
transitions <- c(
  "A\u00adto\u00adG", "G\u00adto\u00adA",
  "C\u00adto\u00adT",
  "T\u00adto\u00adC"
)
```

```
just_class <- just_class %>%
  mutate(mutation_type = fct_relevel(mutation_type,
    c(transversions, transitions),
    after = Inf
  ))
```

```
ggplot(just_class %>% filter() %>% arrange(mutation_type), aes(y = Number_of_mutations, x = trial)) +
  geom_col(position = "dodge") +
  facet_grid(. ~ trial) +
  theme(axis.text.x = element_text(angle = 90, vjust = 0.5, hjust = 1)) +
  geom_vline(xintercept = 8.5, linetype = "dashed", color = "black")
```



```
# Directory where your TSV files are
dir <- "./tsv_files"

# List all .tsv files in the directory
files <- list.files(path = dir, pattern = "\\\\.tsv$", full.names = TRUE)

# Read all files into a list of tibbles, adding the file name as a new column
big_df <- map_dfr(files, ~ read_tsv(.x, col_names = c("index", "par", "A", "C", "G", "T")))
```

Rows: 29812 Columns: 6

```
-- Column specification -----
Delimiter: "\t"
chr (1): par
dbl (5): index, A, C, G, T
```

```
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.
Rows: 29694 Columns: 6
-- Column specification -----
Delimiter: "\t"
chr (1): par
dbl (5): index, A, C, G, T
```

```

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.
Rows: 29617 Columns: 6
-- Column specification -----
Delimiter: "\t"
chr (1): par
dbl (5): index, A, C, G, T

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.
Rows: 29624 Columns: 6
-- Column specification -----
Delimiter: "\t"
chr (1): par
dbl (5): index, A, C, G, T

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.
Rows: 28827 Columns: 6
-- Column specification -----
Delimiter: "\t"
chr (1): par
dbl (5): index, A, C, G, T

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.
Rows: 25577 Columns: 6
-- Column specification -----
Delimiter: "\t"
chr (1): par
dbl (5): index, A, C, G, T

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.
Rows: 28243 Columns: 6
-- Column specification -----
Delimiter: "\t"
chr (1): par
dbl (5): index, A, C, G, T

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.

```

```

Rows: 28934 Columns: 6
-- Column specification -----
Delimiter: "\t"
chr (1): par
dbl (5): index, A, C, G, T

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.
Rows: 28601 Columns: 6
-- Column specification -----
Delimiter: "\t"
chr (1): par
dbl (5): index, A, C, G, T

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.
Rows: 27536 Columns: 6
-- Column specification -----
Delimiter: "\t"
chr (1): par
dbl (5): index, A, C, G, T

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.
Rows: 29625 Columns: 6
-- Column specification -----
Delimiter: "\t"
chr (1): par
dbl (5): index, A, C, G, T

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.
Rows: 29398 Columns: 6
-- Column specification -----
Delimiter: "\t"
chr (1): par
dbl (5): index, A, C, G, T

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.
Rows: 28785 Columns: 6
-- Column specification -----
Delimiter: "\t"

```



```
chr (1): par
dbl (5): index, A, C, G, T
```

```
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.
```

```
Rows: 18869 Columns: 6
```

```
-- Column specification -----
```

```
Delimiter: "\t"
```

```
chr (1): par
dbl (5): index, A, C, G, T
```

```
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.
```

```
Rows: 29494 Columns: 6
```

```
-- Column specification -----
```

```
Delimiter: "\t"
```

```
chr (1): par
dbl (5): index, A, C, G, T
```

```
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.
```

```
Rows: 29322 Columns: 6
```

```
-- Column specification -----
```

```
Delimiter: "\t"
```

```
chr (1): par
dbl (5): index, A, C, G, T
```

```
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.
```

```
Rows: 27603 Columns: 6
```

```
-- Column specification -----
```

```
Delimiter: "\t"
```

```
chr (1): par
dbl (5): index, A, C, G, T
```

```
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.
```

```
Rows: 29686 Columns: 6
```

```
-- Column specification -----
```

```
Delimiter: "\t"
```

```
chr (1): par
dbl (5): index, A, C, G, T
```

```

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.
Rows: 29849 Columns: 6
-- Column specification -----
Delimiter: "\t"
chr (1): par
dbl (5): index, A, C, G, T

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.
Rows: 29664 Columns: 6
-- Column specification -----
Delimiter: "\t"
chr (1): par
dbl (5): index, A, C, G, T

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.
Rows: 29348 Columns: 6
-- Column specification -----
Delimiter: "\t"
chr (1): par
dbl (5): index, A, C, G, T

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.
Rows: 29836 Columns: 6
-- Column specification -----
Delimiter: "\t"
chr (1): par
dbl (5): index, A, C, G, T

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.
Rows: 29796 Columns: 6
-- Column specification -----
Delimiter: "\t"
chr (1): par
dbl (5): index, A, C, G, T

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.
Rows: 29638 Columns: 6

```

```

-- Column specification -----
Delimiter: "\t"
chr (1): par
dbl (5): index, A, C, G, T

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.
Rows: 29668 Columns: 6
-- Column specification -----
Delimiter: "\t"
chr (1): par
dbl (5): index, A, C, G, T

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.
Rows: 29635 Columns: 6
-- Column specification -----
Delimiter: "\t"
chr (1): par
dbl (5): index, A, C, G, T

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.
Rows: 29691 Columns: 6
-- Column specification -----
Delimiter: "\t"
chr (1): par
dbl (5): index, A, C, G, T

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.
Rows: 29662 Columns: 6
-- Column specification -----
Delimiter: "\t"
chr (1): par
dbl (5): index, A, C, G, T

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.
Rows: 28896 Columns: 6
-- Column specification -----
Delimiter: "\t"
chr (1): par

```

```
dbl (5): index, A, C, G, T
```

```
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.
```

```
Rows: 29625 Columns: 6
```

```
-- Column specification -----
```

```
Delimiter: "\t"
```

```
chr (1): par
```

```
dbl (5): index, A, C, G, T
```

```
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.
```

```
Rows: 29761 Columns: 6
```

```
-- Column specification -----
```

```
Delimiter: "\t"
```

```
chr (1): par
```

```
dbl (5): index, A, C, G, T
```

```
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.
```

```
Rows: 29656 Columns: 6
```

```
-- Column specification -----
```

```
Delimiter: "\t"
```

```
chr (1): par
```

```
dbl (5): index, A, C, G, T
```

```
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.
```

```
Rows: 28572 Columns: 6
```

```
-- Column specification -----
```

```
Delimiter: "\t"
```

```
chr (1): par
```

```
dbl (5): index, A, C, G, T
```

```
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.
```

```
Rows: 29602 Columns: 6
```

```
-- Column specification -----
```

```
Delimiter: "\t"
```

```
chr (1): par
```

```
dbl (5): index, A, C, G, T
```

```
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.
Rows: 29651 Columns: 6
-- Column specification -----
Delimiter: "\t"
chr (1): par
dbl (5): index, A, C, G, T

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.
Rows: 29507 Columns: 6
-- Column specification -----
Delimiter: "\t"
chr (1): par
dbl (5): index, A, C, G, T

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.
Rows: 28393 Columns: 6
-- Column specification -----
Delimiter: "\t"
chr (1): par
dbl (5): index, A, C, G, T

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.
Rows: 24314 Columns: 6
-- Column specification -----
Delimiter: "\t"
chr (1): par
dbl (5): index, A, C, G, T

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.
Rows: 29243 Columns: 6
-- Column specification -----
Delimiter: "\t"
chr (1): par
dbl (5): index, A, C, G, T

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.
Rows: 28345 Columns: 6
-- Column specification -----

```

```

Delimiter: "\t"
chr (1): par
dbl (5): index, A, C, G, T

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.
Rows: 29482 Columns: 6
-- Column specification -----
Delimiter: "\t"
chr (1): par
dbl (5): index, A, C, G, T

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.
Rows: 29651 Columns: 6
-- Column specification -----
Delimiter: "\t"
chr (1): par
dbl (5): index, A, C, G, T

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.
Rows: 29624 Columns: 6
-- Column specification -----
Delimiter: "\t"
chr (1): par
dbl (5): index, A, C, G, T

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.
Rows: 29663 Columns: 6
-- Column specification -----
Delimiter: "\t"
chr (1): par
dbl (5): index, A, C, G, T

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.
Rows: 27976 Columns: 6
-- Column specification -----
Delimiter: "\t"
chr (1): par
dbl (5): index, A, C, G, T

```

- 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.

```
big_df <- big_df %>% mutate(total_depth = A + C + G + T)

big_df <- big_df %>% separate(file_name, into = c("treat", "patient", "timepoint"), sep =

long_df <- big_df %>%
  pivot_longer(
    cols = c(A, C, G, T),
    names_to = "base",
    values_to = "count"
  ) %>%
  filter(par != base, count > 0) %>%
  filter(count >= total_depth * 0.05, total_depth >= 100) %>%
  mutate(mutation_type = as.factor(paste0(par, "\u00adto\u00ad", base))) %>%
  filter(par != "N") %>%
  group_by(patient, index, par, base) %>%
  filter(row_number() == 1) # ensures we only count each mutation once

burdens <- long_df %>%
  filter(treat != "PAXLOVID") %>%
  group_by(treat, patient) %>%
  tally()

# Split mutation counts into two vectors based on treatment
naive_burden <- burdens %>%
  filter(treat == "NAIVE") %>%
  pull(n)
mov_burden <- burdens %>%
  filter(treat == "MOLNUPIRAVIR") %>%
  pull(n)

length(naive_burden)
```

[1] 5

```
sd(naive_burden)
```

```
[1] 3.714835
```

```
mean(naive_burden)
```

```
[1] 9.6
```

```
length(mov_burden)
```

```
[1] 8
```

```
sd(mov_burden)
```

```
[1] 63.19118
```

```
mean(mov_burden)
```

```
[1] 78.375
```

```
n_patients_naive <- 5  
n_patients_mov <- 8
```

```
ba1_basic <- just_class %>% filter(trial == 3)  
ba1_normed <- ba1_basic %>% mutate(Number_of_mutations = Number_of_mutations * sum(naive_b
```

```
lookup <- c("MOLNUPIRAVIR" = "mov", "NAIVE" = "normal")
```

```
mov_dataset <- long_df %>%  
  group_by(mutation_type, treat) %>%  
  tally() %>%  
  filter(treat == "MOLNUPIRAVIR") %>%  
  mutate(treat = "mov") %>%
```



```

mutate(Number_of_mutations = n) %>%
mutate(mutation_type = fct_relevel(mutation_type, c(transversions, transitions))) %>%
mutate(Number_of_mutations = Number_of_mutations / n_patients_mov)
naive_dataset <- bal_normed %>%
mutate(treat = "normal") %>%
mutate(mutation_type = fct_relevel(mutation_type, c(transversions, transitions))) %>%
mutate(Number_of_mutations = Number_of_mutations / n_patients_naive)

relevant_dataset <- bind_rows(mov_dataset, naive_dataset)

relevant_dataset

# A tibble: 21 x 5
# Groups:   mutation_type [12]
  mutation_type treat      n Number_of_mutations trial
  <fct>          <chr> <int>          <dbl> <chr>
1 AtoG          mov      60             7.5 <NA>
2 AtoT          mov       1             0.125 <NA>
3 CtoA          mov       4             0.5 <NA>
4 CtoT          mov     263            32.9 <NA>
5 GtoA          mov     215            26.9 <NA>
6 GtoT          mov       8              1 <NA>
7 TtoA          mov       3             0.375 <NA>
8 TtoC          mov      71             8.88 <NA>
9 TtoG          mov       2             0.25 <NA>
10 CtoA         normal    NA             0.285 3
# i 11 more rows

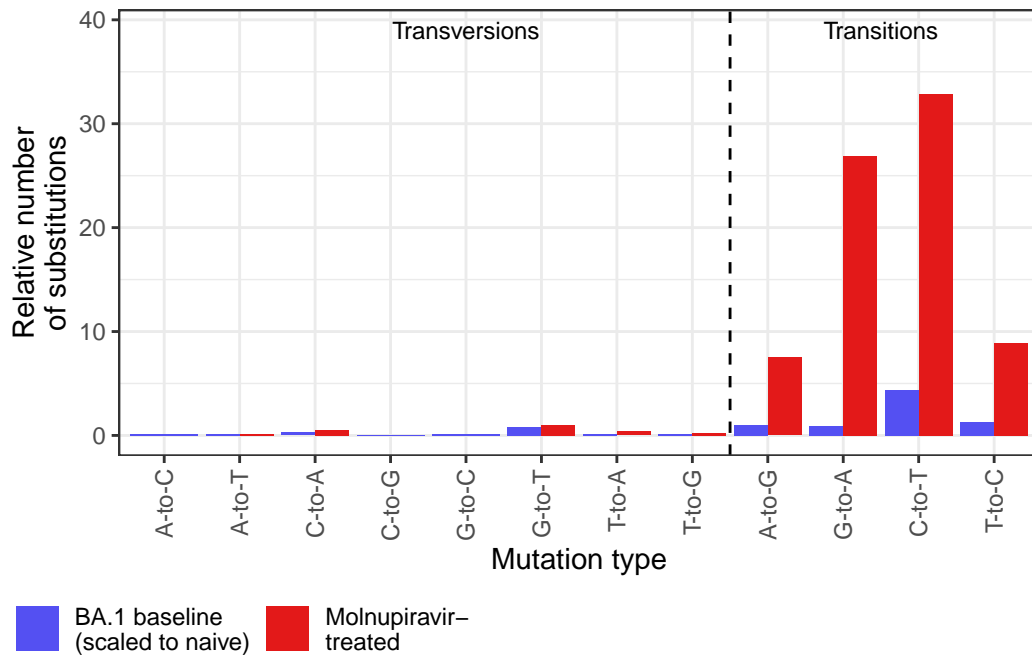
a <- ggplot(relevant_dataset, aes(y = Number_of_mutations, x = mutation_type, fill = treat)) +
  geom_col(position = "dodge") +
  geom_vline(xintercept = 8.5, linetype = "dashed", color = "black") +
  scale_fill_manual(values = c(blue1, red), labels = c("BA.1 baseline\n(scaled to naive)",
  theme_bw() +
  theme(axis.text.x = element_text(angle = 90, vjust = 0.5, hjust = 1)) +
  labs(fill = "") +
  annotate("text", x = 5, y = 39, label = "Transversions", size = 3) +
  labs(x = "Mutation type", y = "Relative number\nof substitutions") +
  annotate("text", x = 10.5, y = 39, label = "Transitions", size = 3) +
  theme(

```

```

legend.position = "bottom",
legend.justification = c(0, 1),
legend.margin = margin(t = 0, r = 0, b = 0, l = -45, unit = "pt")
)
a

```



```

naive_props <- naive_dataset %>%
  ungroup() %>%
  mutate(p = Number_of_mutations / sum(Number_of_mutations))
# The BA.1 spectrum props is based on so many mutations (hundreds of thousands) that we ca

naive_props

```

```

# A tibble: 12 x 5
  mutation_type treat trial Number_of_mutations p
  <fct>          <chr> <chr>          <dbl> <dbl>
1 CtoA          normal 3              0.285 0.0297
2 CtoG          normal 3              0.0489 0.00509
3 CtoT          normal 3              4.39 0.457
4 TtoA          normal 3              0.178 0.0185
5 TtoC          normal 3              1.29 0.134

```

6	TtoG	normal	3	0.135	0.0141
7	GtoT	normal	3	0.806	0.0840
8	GtoC	normal	3	0.143	0.0149
9	GtoA	normal	3	0.946	0.0985
10	AtoT	normal	3	0.176	0.0183
11	AtoG	normal	3	1.03	0.107
12	AtoC	normal	3	0.172	0.0180

```

mov_for_props <- long_df %>%
  filter(treat == "MOLNUPIRAVIR") %>%
  ungroup()

resample_and_calc_ratios <- function(long_df) {
  resampled <- sample_n(mov_for_props, size = nrow(mov_for_props), replace = TRUE)
  props <- resampled %>%
    group_by(mutation_type) %>%
    tally() %>%
    mutate(p = n / sum(n))
  together <- inner_join(props, naive_props, by = "mutation_type") %>% mutate(ratio = p.x)
  return(together %>% select(mutation_type, ratio))
}

bootstrap_count <- 100
bootstrap_ratios <- list()

for (i in 1:bootstrap_count) {
  bootstrap_ratios[[i]] <- resample_and_calc_ratios(long_df)
}

# Convert list to data frame
bootstrap_ratios_df <- bind_rows(bootstrap_ratios)
bootstrap_ratios_df

# A tibble: 841 x 2
  mutation_type ratio
  <fct>         <dbl>
1 AtoG          0.891
2 AtoT          0.174
3 CtoA          0.107
4 CtoT          0.816
5 GtoA          3.84

```

```

6 GtoT      0.285
7 TtoA      0.259
8 TtoC      0.843
9 TtoG      0.340
10 AtoG     0.935
# i 831 more rows

```

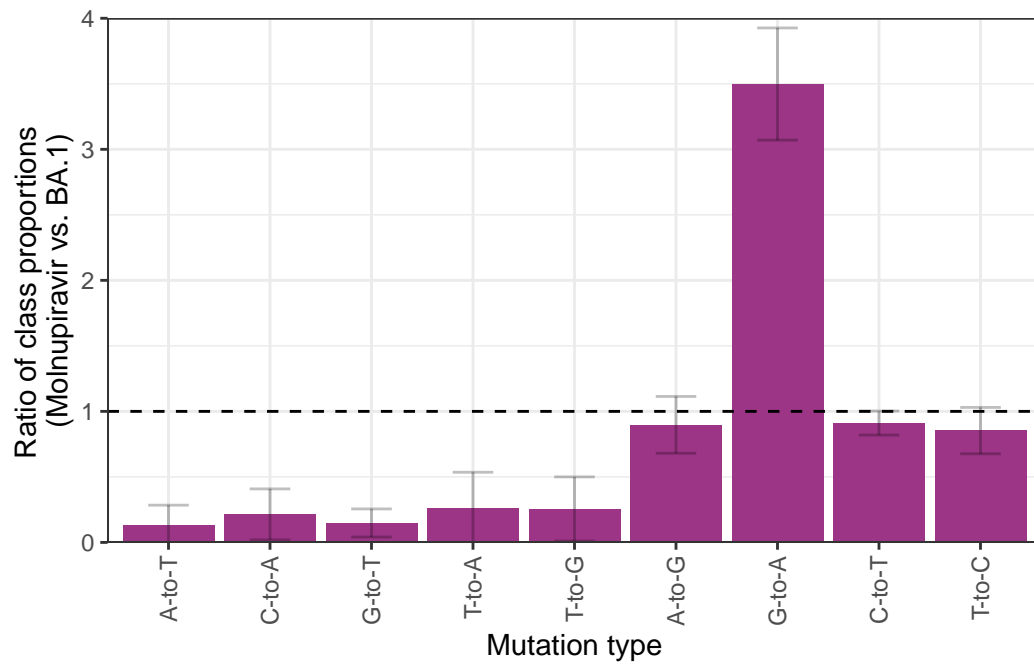
```

proportions_wider <- bootstrap_ratios_df %>%
  group_by(mutation_type) %>%
  summarise(sd = sd(ratio), ratio = mean(ratio))

b <- ggplot(proportions_wider %>% mutate(mutation_type = fct_relevel(mutation_type, c("tran
  geom_col(position = "dodge", fill = "#9C3586") +
  scale_y_continuous(expand = c(0, 0)) +
  theme_bw() +
  theme(axis.text.x = element_text(angle = 90, vjust = 0.5, hjust = 1)) +
  labs(x = "Mutation type", y = "Ratio of class proportions \n(Molnupiravir vs. BA.1)")
  geom_hline(yintercept = 1, linetype = "dashed", color = "black") +
  geom_errorbar(alpha = 0.25, width = 0.4) +
  coord_cartesian(ylim = c(0, 4))

```

b



```
proportions
```

```
function (x, margin = NULL)
{
  if (length(margin))
    sweep(x, margin, marginSums(x, margin), `/\`, check.margin = FALSE)
  else x/sum(x)
}
<bytecode: 0x142d11a80>
<environment: namespace:base>
```

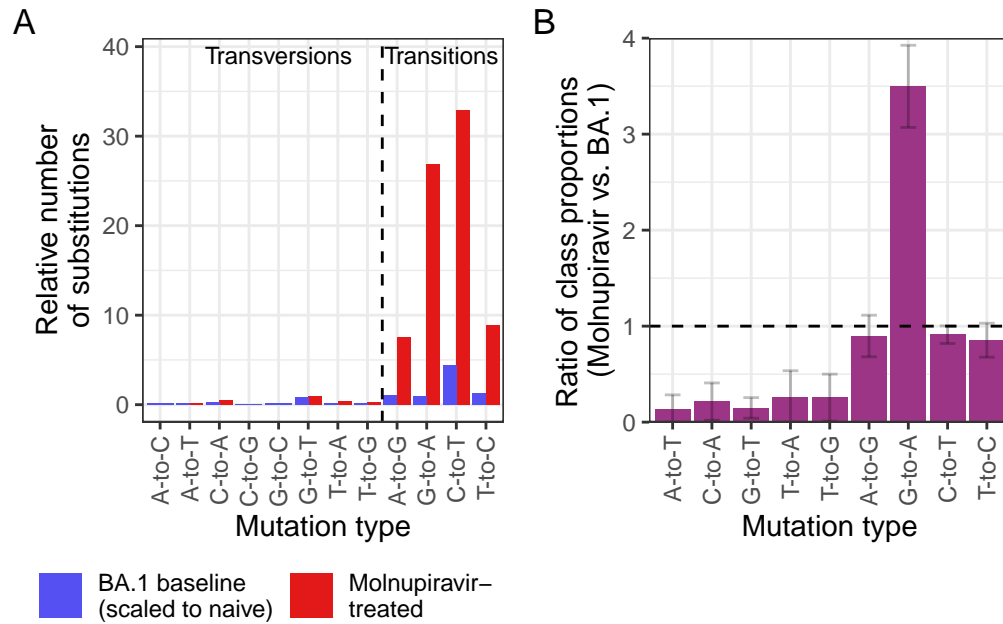
```
library(patchwork)
```

Attaching package: 'patchwork'

The following object is masked from 'package:cowplot':

```
align_plots
```

```
ab <- a + b + plot_annotation(tag_levels = "A")
ab
```



```
ggsave("a.pdf", a, width = 3, height = 3.5)
ggsave("b.pdf", b, width = 3, height = 3.5)
```

```
mov_props <- mov_for_props %>%
  group_by(mutation_type) %>%
  tally() %>%
  mutate(p = n / sum(n))
```

```
perform_sim <- function(n_sample, relevant_props) {
  # Set the number of iterations and the sample size
  n_iterations <- 10000
  15

  # Initialize a vector to hold the result of each iteration
  result <- vector(mode = "logical", length = n_iterations)

  # Run the simulation
```

```

for (i in 1:n_iterations) {
  # Sample mutation types according to their probabilities
  sample_mutation <- sample(relevant_props$mutation_type, size = n_sample, replace = TRUE)

  # Calculate the proportions of each mutation type in the sample
  sample_prop <- table(sample_mutation) / n_sample

  # Calculate the transition proportion
  transition_prop <- sum(sample_prop[c("C\u00adto\u00adT", "G\u00adto\u00adA", "T\u00adto\u00adC", "A\u00adto\u00adG")], na.rm = TRUE)

  # Check whether the proportions meet the thresholds
  result[i] <- (sample_prop["C\u00adto\u00adT"] > 0.2 & sample_prop["G\u00adto\u00adA"] > 0.2 &
    sample_prop["T\u00adto\u00adC"] > 0.2 & sample_prop["A\u00adto\u00adG"] > 0.2)

}

# Calculate the proportion of iterations that meet the condition
proportion <- sum(result) / n_iterations
proportion
return(proportion)
}

# Define the mutation counts to consider
mutations <- c(10, 15, 20)

# Initialize vectors to hold results
sensitivity <- numeric(length(mutations))
specificity <- numeric(length(mutations))

# Loop over each mutation count
for (i in seq_along(mutations)) {
  # Compute sensitivity and specificity
  sensitivity[i] <- perform_sim(mutations[i], mov_props)
  specificity[i] <- 1 - perform_sim(mutations[i], naive_props)
}

# Create a data frame with the results
results <- data.frame(
  Mutations = mutations,
  Sensitivity = sensitivity,
  Specificity = specificity
)

# Print the results

```

```
print(results)
```

	Mutations	Sensitivity	Specificity
1	10	0.4754	0.9884
2	15	0.5025	0.9975
3	20	0.5423	0.9988

```
library(ggpmisc)
```

Loading required package: ggpp

Attaching package: 'ggpp'

The following object is masked from 'package:ggplot2':

annotate

```
library(ggtext)
```

normed

A tibble: 1,344 x 10

	context_before <chr>	par <chr>	mut <chr>	context_after <chr>	Number_of_mutations <dbl>	trial <chr>	treat <chr>
1	A	C	A	A	0.00147	2	mov
2	A	C	A	C	0	2	mov
3	A	C	A	G	0	2	mov
4	A	C	A	T	0	2	mov
5	C	C	A	A	0	2	mov
6	C	C	A	C	0	2	mov
7	C	C	A	G	0	2	mov
8	C	C	A	T	0	2	mov
9	G	C	A	A	0	2	mov
10	G	C	A	C	0.00317	2	mov

i 1,334 more rows

i 3 more variables: contexts_only <lgl>, mutation_type <fct>, total <dbl>


```

trial2 <- normed %>%
  filter((treat == "mov" & trial == "2")) %>%
  group_by(mutation_type) %>%
  mutate(Number_of_mutations = Number_of_mutations / sum(Number_of_mutations))
trial1 <- normed %>%
  filter((treat == "mov" & trial == "1")) %>%
  group_by(mutation_type) %>%
  mutate(Number_of_mutations = Number_of_mutations / sum(Number_of_mutations))
long <- normed %>%
  filter((trial == "4")) %>%
  select(-treat, -total, -contexts_only, -trial) %>%
  group_by(mutation_type) %>%
  mutate(Number_of_mutations = Number_of_mutations / sum(Number_of_mutations))
normal <- normed %>%
  filter((trial == "3")) %>%
  select(-treat, -total, -contexts_only, -trial) %>%
  group_by(mutation_type) %>%
  mutate(Number_of_mutations = Number_of_mutations / sum(Number_of_mutations))
normal

```

A tibble: 192 x 6

Groups: mutation_type [12]

	context_before	par	mut	context_after	Number_of_mutations	mutation_type
	<chr>	<chr>	<chr>	<chr>	<dbl>	<fct>
1	A	C	A	A	0.0423	CtoA
2	A	C	A	C	0.0618	CtoA
3	A	C	A	G	0.0655	CtoA
4	A	C	A	T	0.0737	CtoA
5	C	C	A	A	0.0922	CtoA
6	C	C	A	C	0.0506	CtoA
7	C	C	A	G	0.125	CtoA
8	C	C	A	T	0.0994	CtoA
9	G	C	A	A	0.0500	CtoA
10	G	C	A	C	0.0386	CtoA

i 182 more rows

```

merged <- normed %>%
  group_by(context_before, context_after, par, mut, treat, mutation_type) %>%
  summarise(Number_of_mutations = mean(Number_of_mutations)) %>%
  filter(treat == "mov")

```

``summarise()`` has grouped output by 'context_before', 'context_after', 'par', 'mut', 'treat'. You can override using the ``.groups`` argument.

```
long_v_merged <- inner_join(long %>% rename(v1 = Number_of_mutations), merged %>% rename(v1 = Number_of_mutations))
```

Joining with ``by = join_by(context_before, par, mut, context_after, mutation_type)``

```
t1_v_merged <- inner_join(long %>% rename(v1 = Number_of_mutations), trial1 %>% rename(v1 = Number_of_mutations))
```

Joining with ``by = join_by(context_before, par, mut, context_after, mutation_type)``

```
t2_v_merged <- inner_join(long %>% rename(v1 = Number_of_mutations), trial2 %>% rename(v1 = Number_of_mutations))
```

Joining with ``by = join_by(context_before, par, mut, context_after, mutation_type)``

```
long_v_normal <- inner_join(long %>% rename(v1 = Number_of_mutations), normal %>% rename(v1 = Number_of_mutations))
```

Joining with ``by = join_by(context_before, par, mut, context_after, mutation_type)``

```
oneset <- unique((t2_v_merged %>% filter(mutation_type %in% c("G\u00adto\u00adA")))$context_after)
```

```
library(pals)
```

Attaching package: 'pals'

The following object is masked from 'package:Biostrings':

alphabet

```

colors_16 <- unname(c(alphabet()[26:26], alphabet()[9], alphabet()[2:7], alphabet()[11:15])

reverse_complement <- function(context) {
  rev_nucleotide <- function(x) {
    switch(x,
      "A" = "T",
      "T" = "A",
      "C" = "G",
      "G" = "C",
      x
    )
  }
  rev_context <- sapply(strsplit(context, "")[[1]], rev_nucleotide)
  paste(rev(rev_context), collapse = "")
}

context_colors <- c()
for (i in 1:length(oneset)) {
  context <- oneset[i]
  reverse_context <- reverse_complement(context)

  if (!context %in% names(context_colors)) {
    context_colors[context] <- colors_16[i]
  }

  if (!reverse_context %in% names(context_colors)) {
    context_colors[reverse_context] <- colors_16[i]
  }
}

scatters <- ggplot(t2_v_merged %>% filter(mutation_type %in% c("G\u00adto\u00adA", "C\u00ad
  geom_point() +
  labs(x = "Alteri et al. molnupiravir proportion", y = "Long branch proportion") +
  facet_wrap(~mutation_type, ncol = 2) +
  theme_bw() +
  stat_correlation(method = "pearson", aes(group = 1, label = ..r.label..), digits = 2, r.
  coord_fixed(xlim = c(0, NA), ylim = c(0, NA)) +
  geom_abline(
    intercept = 0, slope = 1, # linetype = "black",

```

```

    color = "darkgray"
  ) +
  geom_text_repel(alpha = 0.5, size = 2, max.overlaps = Inf, force = 10) +
  scale_x_continuous(labels = scales::percent) +
  scale_y_continuous(labels = scales::percent) +
  scale_color_manual(values = context_colors) +
  theme(legend.position = "none")

```

Warning in stat_correlation(method = "pearson", aes(group = 1, label
= ..r.label..), : Ignoring unknown parameters: `digits` and `r.accuracy`

```

start <- trial2 %>%
  mutate(context_full = paste0(context_before, par, context_after)) %>%
  mutate(rc_context = sapply(context_full, reverse_complement))
GtoA <- start %>% filter(mutation_type == "G\u00adto\u00adA")
CtoT <- start %>% filter(mutation_type == "C\u00adto\u00adT")
joint <- inner_join(GtoA, CtoT, by = c("context_full" = "rc_context"))

comp <- ggplot(joint, aes(x = Number_of_mutations.x, y = Number_of_mutations.y, label = pa
  geom_point(color = "darkblue") +
  theme_bw() +
  geom_abline(
    intercept = 0, slope = 1, # linetype = "black",
    color = "darkgray"
  ) +
  stat_correlation(color = "black", method = "pearson", aes(group = 1, label = ..r.label..
  stat_correlation(color = "black", method = "pearson", aes(group = 1, label = ..p.value.l
  geom_text_repel(color = "darkgray") +
  scale_x_continuous(labels = scales::percent) +
  scale_y_continuous(labels = scales::percent) +
  labs(x = "Context proportion for G-to-A", y = "Context proportion for C-to-T")

```

```
names(colors) <- my_levels
```

```

other_colors <- c("A" = "#111111", "C" = "#555555", "G" = "#999999", "T" = "#cccccc")
all_colors <- c(colors, other_colors)

```

```

colors_new <- all_colors
colors_new["A\u00adto\u00adG"] <- "#5c4987"
colors_new["T\u00adto\u00adC"] <- "#5377ad"

```

```

create_scatter_plot <- function(df, x_label, file_name) {
  plot <- ggplot(df %>%
    filter(mutation_type %in% c("G\u00adto\u00adA", "C\u00adto\u00adT", "A\u00adto\u00adG")
    mutate(label = context_full), aes(x = v2, y = v1, label = label, color = mutation_type)
    geom_point() +
    labs(x = x_label, y = "Long branch proportion") +
    facet_wrap(~mutation_type, ncol = 2) +
    theme_bw() +
    stat_correlation(color = "black", method = "pearson", aes(group = 1, label = ..r.label)
    coord_fixed(xlim = c(0, NA), ylim = c(0, NA)) +
    # geom_abline(intercept = 0, slope = 1, color = "darkgray")+
    geom_text_repel(alpha = 0.5, size = 2, max.overlaps = Inf, force = 10) +
    scale_x_continuous(labels = scales::percent) +
    scale_y_continuous(labels = scales::percent) +
    scale_color_manual(values = colors_new) +
    theme(legend.position = "none") +
    stat_correlation(color = "black", method = "pearson", aes(group = 1, label = ..p.value)
    geom_smooth(method = "lm", se = FALSE, color = "darkgray", fullrange = F, size = 1)

  return(plot)
}

# Call the function three times with different dataframes and labels
scatters_supplemental <- create_scatter_plot(t2_v_merged, "Alteri et al. molnupiravir prop

```

Warning: Using `size` aesthetic for lines was deprecated in ggplot2 3.4.0.
 i Please use `linewidth` instead.

```

scatters_normal <- create_scatter_plot(long_v_normal, "Ruis et al. BA.1 proportion", "scat
scatters_supplemental2 <- create_scatter_plot(t1_v_merged, "Donovan-Banfield et al. molnup

scatters_supplemental + scatters_supplemental2 + scatters_normal + comp + plot_annotation(

```

Warning: The dot-dot notation (`..r.label..`) was deprecated in ggplot2 3.4.0.
 i Please use `after_stat(r.label)` instead.

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

Warning: The following aesthetics were dropped during statistical transformation: label
i This can happen when ggplot fails to infer the correct grouping structure in the data.

i Did you forget to specify a `group` aesthetic or to convert a numerical variable into a factor?

Warning: The following aesthetics were dropped during statistical transformation: label
i This can happen when ggplot fails to infer the correct grouping structure in the data.

i Did you forget to specify a `group` aesthetic or to convert a numerical variable into a factor?

The following aesthetics were dropped during statistical transformation: label
i This can happen when ggplot fails to infer the correct grouping structure in the data.

i Did you forget to specify a `group` aesthetic or to convert a numerical variable into a factor?

The following aesthetics were dropped during statistical transformation: label
i This can happen when ggplot fails to infer the correct grouping structure in the data.

i Did you forget to specify a `group` aesthetic or to convert a numerical variable into a factor?

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

Warning: The following aesthetics were dropped during statistical transformation: label
i This can happen when ggplot fails to infer the correct grouping structure in the data.

i Did you forget to specify a `group` aesthetic or to convert a numerical variable into a factor?

The following aesthetics were dropped during statistical transformation: label
i This can happen when ggplot fails to infer the correct grouping structure in the data.

i Did you forget to specify a `group` aesthetic or to convert a numerical variable into a factor?

The following aesthetics were dropped during statistical transformation: label
i This can happen when ggplot fails to infer the correct grouping structure in the data.

i Did you forget to specify a `group` aesthetic or to convert a numerical variable into a factor?

The following aesthetics were dropped during statistical transformation: label
i This can happen when ggplot fails to infer the correct grouping structure in the data.

i Did you forget to specify a `group` aesthetic or to convert a numerical variable into a factor?

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

Warning: The following aesthetics were dropped during statistical transformation: label

i This can happen when ggplot fails to infer the correct grouping structure in the data.

i Did you forget to specify a `group` aesthetic or to convert a numerical variable into a factor?

The following aesthetics were dropped during statistical transformation: label

i This can happen when ggplot fails to infer the correct grouping structure in the data.

i Did you forget to specify a `group` aesthetic or to convert a numerical variable into a factor?

The following aesthetics were dropped during statistical transformation: label

i This can happen when ggplot fails to infer the correct grouping structure in the data.

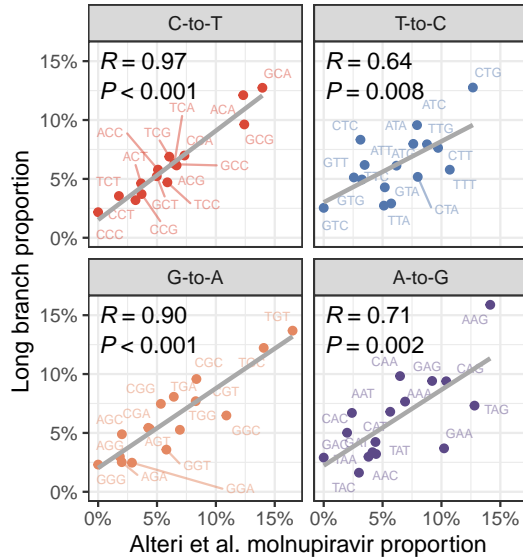
i Did you forget to specify a `group` aesthetic or to convert a numerical variable into a factor?

The following aesthetics were dropped during statistical transformation: label

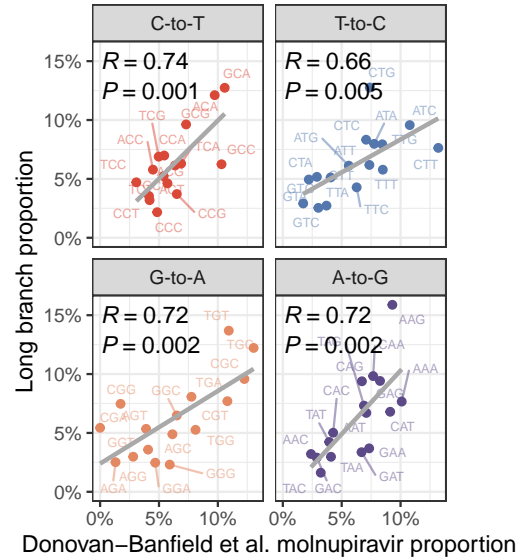
i This can happen when ggplot fails to infer the correct grouping structure in the data.

i Did you forget to specify a `group` aesthetic or to convert a numerical variable into a factor?

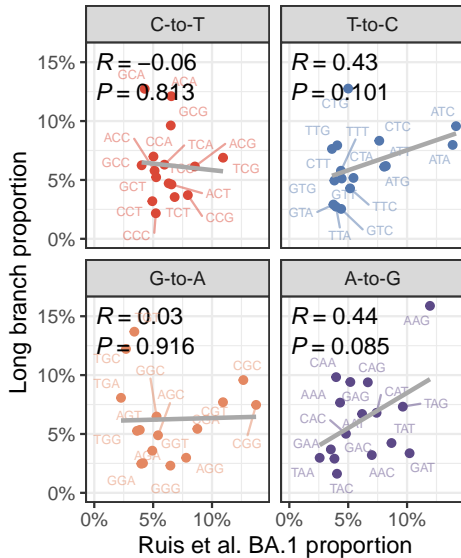
A



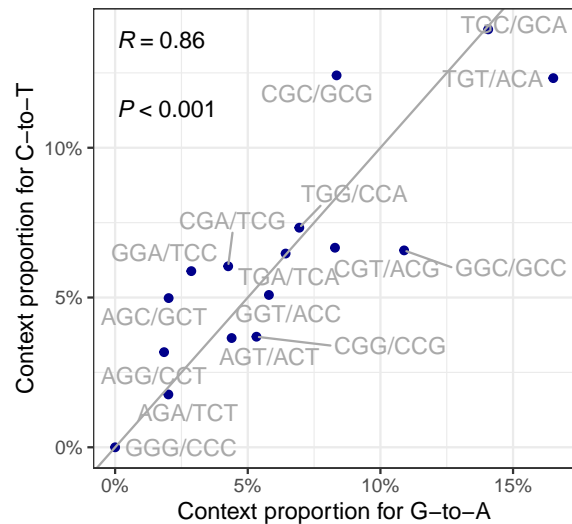
B



C



D



```
ggsave("supplemental_scatters.pdf")
```

Saving 8 x 8 in image

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

Warning: The following aesthetics were dropped during statistical transformation: label
i This can happen when ggplot fails to infer the correct grouping structure in


```

the data.
i Did you forget to specify a `group` aesthetic or to convert a numerical
  variable into a factor?
The following aesthetics were dropped during statistical transformation: label
i This can happen when ggplot fails to infer the correct grouping structure in
  the data.
i Did you forget to specify a `group` aesthetic or to convert a numerical
  variable into a factor?
The following aesthetics were dropped during statistical transformation: label
i This can happen when ggplot fails to infer the correct grouping structure in
  the data.
i Did you forget to specify a `group` aesthetic or to convert a numerical
  variable into a factor?
The following aesthetics were dropped during statistical transformation: label
i This can happen when ggplot fails to infer the correct grouping structure in
  the data.
i Did you forget to specify a `group` aesthetic or to convert a numerical
  variable into a factor?

```

```

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

```

```

Warning: The following aesthetics were dropped during statistical transformation: label
i This can happen when ggplot fails to infer the correct grouping structure in
  the data.
i Did you forget to specify a `group` aesthetic or to convert a numerical
  variable into a factor?
The following aesthetics were dropped during statistical transformation: label
i This can happen when ggplot fails to infer the correct grouping structure in
  the data.
i Did you forget to specify a `group` aesthetic or to convert a numerical
  variable into a factor?
The following aesthetics were dropped during statistical transformation: label
i This can happen when ggplot fails to infer the correct grouping structure in
  the data.
i Did you forget to specify a `group` aesthetic or to convert a numerical
  variable into a factor?
The following aesthetics were dropped during statistical transformation: label
i This can happen when ggplot fails to infer the correct grouping structure in
  the data.
i Did you forget to specify a `group` aesthetic or to convert a numerical
  variable into a factor?

```

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

Warning: The following aesthetics were dropped during statistical transformation: label
i This can happen when ggplot fails to infer the correct grouping structure in the data.

i Did you forget to specify a `group` aesthetic or to convert a numerical variable into a factor?

The following aesthetics were dropped during statistical transformation: label
i This can happen when ggplot fails to infer the correct grouping structure in the data.

i Did you forget to specify a `group` aesthetic or to convert a numerical variable into a factor?

The following aesthetics were dropped during statistical transformation: label
i This can happen when ggplot fails to infer the correct grouping structure in the data.

i Did you forget to specify a `group` aesthetic or to convert a numerical variable into a factor?

The following aesthetics were dropped during statistical transformation: label
i This can happen when ggplot fails to infer the correct grouping structure in the data.

i Did you forget to specify a `group` aesthetic or to convert a numerical variable into a factor?

```
plot_spectrum <- function(data, globalmax = 0, limit = 0.1, extra_axis = FALSE, title = "")  
  if (!globalmax) {  
    globalmax <- max(data$Number_of_mutations)  
  }  
  my_levels <- sort(unique(paste0(data$context_before, data$context_after)))  
  
  data$level <- factor(paste0(data$context_before, data$context_after), levels = my_levels)  
  data$levelno <- as.numeric(data$level)  
  
  precedings <- data %>%  
    group_by(mutation_type, context_before) %>%  
    summarise(levelno = mean(levelno))  
  
  offset <- 0.05  
  
  facet_style_labels <- data %>%  
    group_by(mutation_type) %>%  
    tally() %>%  
    mutate(x = mean(data$levelno), y = -0.13 * globalmax - offset * globalmax)
```

```

p <- ggplot(data, aes(x = levelno, y = `Number_of_mutations`, fill = mutation_type)) +
  facet_wrap(~mutation_type, nrow = 1, strip.position = "top") +
  theme_bw() +
  geom_col() +
  theme(panel.spacing = unit(0, "lines"), panel.border = element_blank()) +
  geom_bar(stat = "identity") +
  theme( # remove the vertical grid lines
    panel.grid.major.x = element_blank(),
    panel.grid.minor.x = element_blank()
    # explicitly set the horizontal lines (or they will disappear too)
    # panel.grid.major.y = element_line( size=.2, color="black" )
  ) +
  theme(legend.position = "none") +
  theme(
    axis.title.x = element_blank(),
    axis.text.x = element_blank(),
    axis.ticks.x = element_blank()
  ) +
  scale_x_continuous(expand = c(0, 0)) +
  theme(
    strip.background = element_blank(),
    strip.text.x = element_blank()
  ) +
  scale_fill_manual(values = all_colors) +
  scale_y_continuous(labels = scales::percent, breaks = c(0, 0.02, 0.04), limits = c(NA,
  labs(y = " ", title = title) +
  theme(plot.title = element_text(margin = margin(t = 0, b = -10), size = 10, hjust = 1))
  geom_hline(yintercept = 0, color = "#222222")

if (extra_axis) {
  p <- p + geom_rect(data = data, aes(xmin = levelno - 0.5, xmax = levelno + 0.5, ymin =

    geom_tile(data = precedings, aes(x = levelno, y = -.09 * .7 * globalmax - globalmax
    geom_text(data = precedings, aes(x = levelno, y = -.09 * .7 * globalmax - globalmax
    geom_tile(data = facet_style_labels, aes(label = mutation_type, fill = mutation_type
    geom_text(data = facet_style_labels, aes(label = mutation_type, label = mutation_type
  }
  print(p)
  return(p)
}

```

```

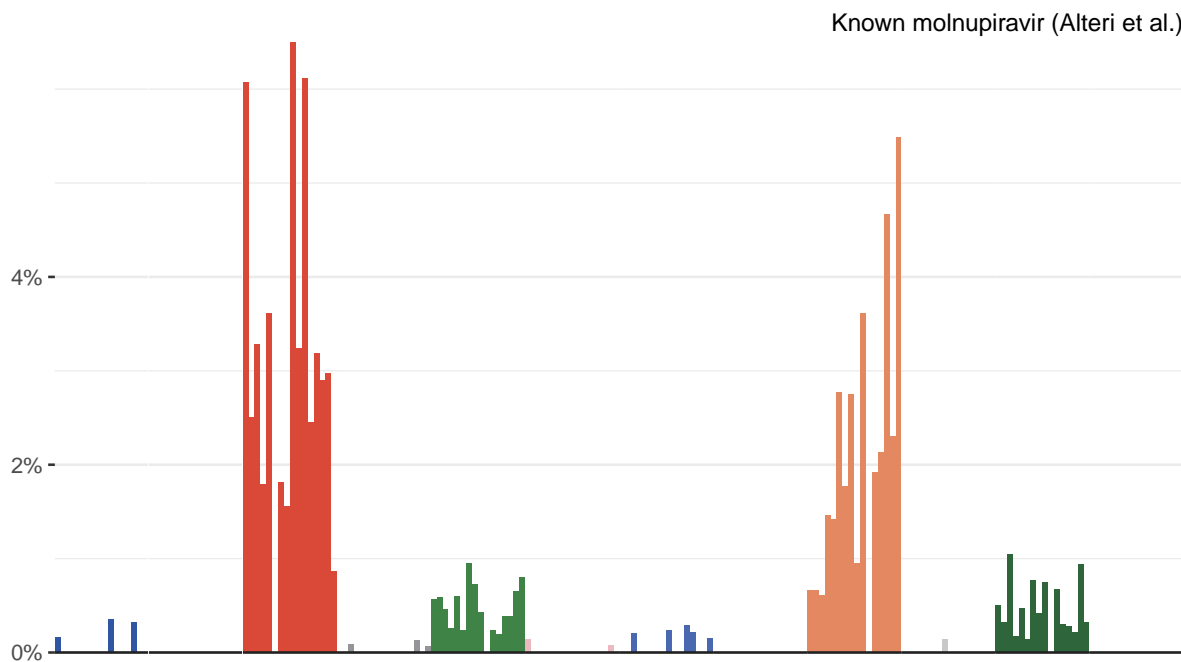
trial2 <- normed %>%
  filter((treat == "mov" & trial == "2")) %>%
  mutate(Number_of_mutations = Number_of_mutations / sum(`Number_of_mutations`))
ba1 <- normed %>%
  filter((trial == "3")) %>%
  mutate(Number_of_mutations = Number_of_mutations / sum(`Number_of_mutations`))

long <- normed %>%
  filter((trial == "4")) %>%
  mutate(Number_of_mutations = Number_of_mutations / sum(`Number_of_mutations`))

p_t2 <- plot_spectrum(trial2, 0.1, 0.065, FALSE, "Known molnupiravir (Alteri et al.)")

```

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



```

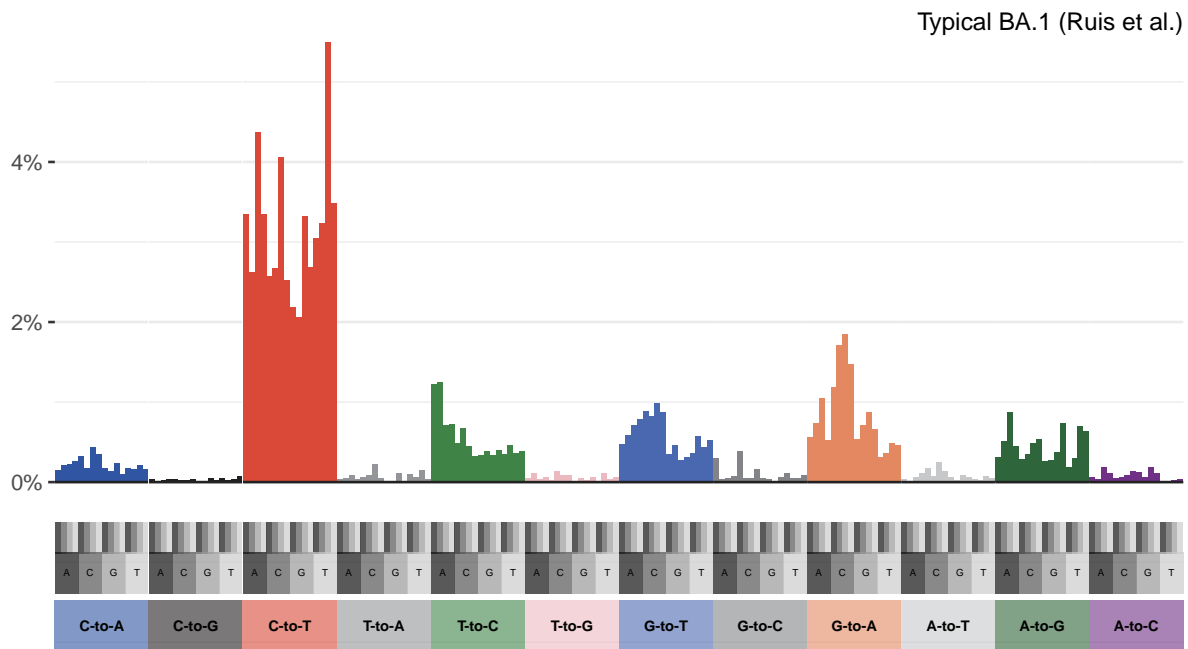
p_ba1 <- plot_spectrum(ba1, 0.1, 0.055, TRUE, "Typical BA.1 (Ruis et al.)")

```

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

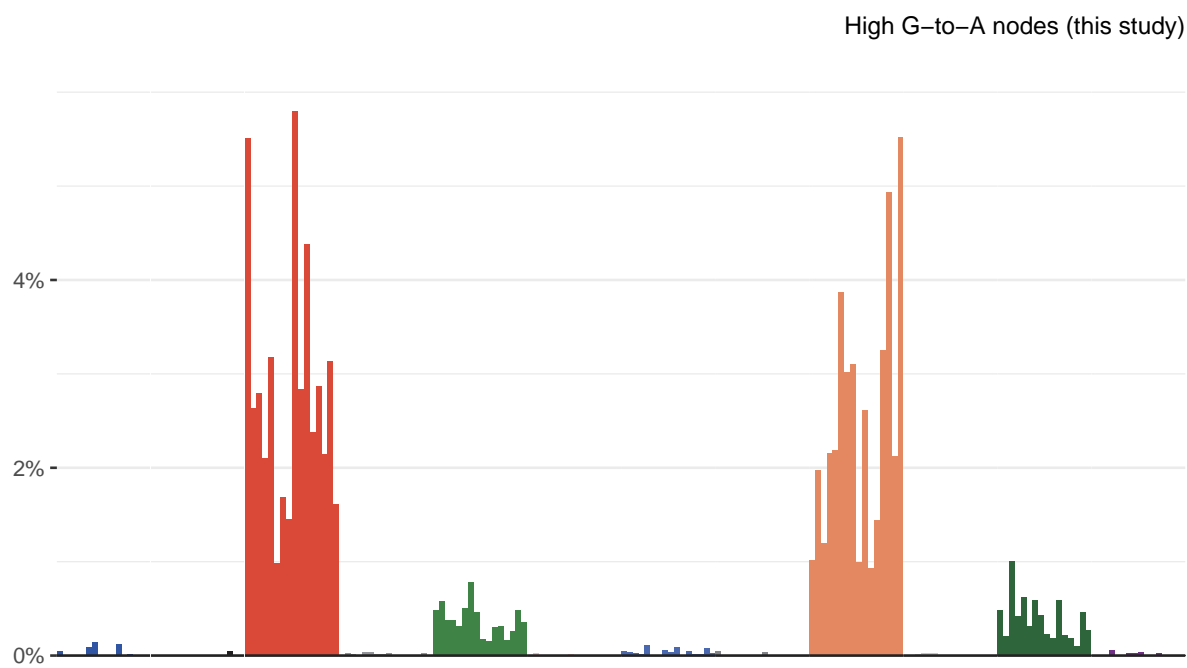
Warning in `geom_tile(data = facet_style_labels, aes(label = mutation_type, : Ignoring unknown aesthetics: label`

Warning: Duplicated aesthetics after name standardisation: label
Duplicated aesthetics after name standardisation: label

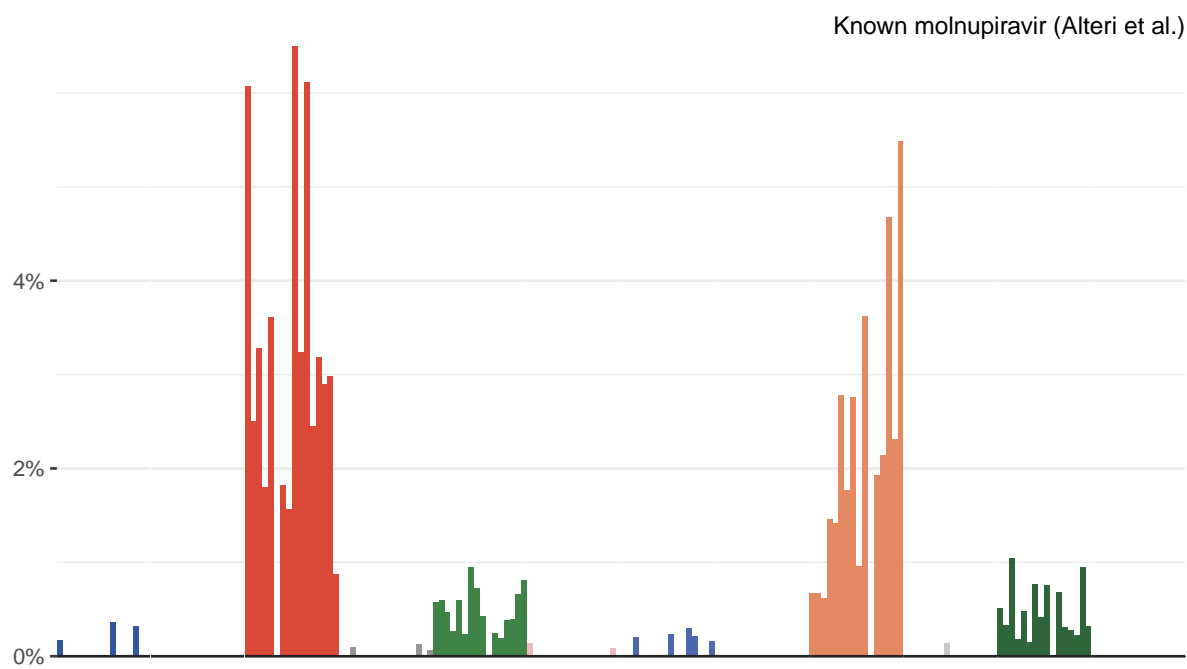


```
p_long <- plot_spectrum(long, 0.1, 0.065, FALSE, "High G-to-A nodes (this study)")
```

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



p_t2



```
stacked <- (p_ba1 / p_t2 / p_long)
```

```
plot_grid(p_long + labs(y = "Norm. proportion"), p_t2, p_ba1, (scatters), labels = c("A",
```

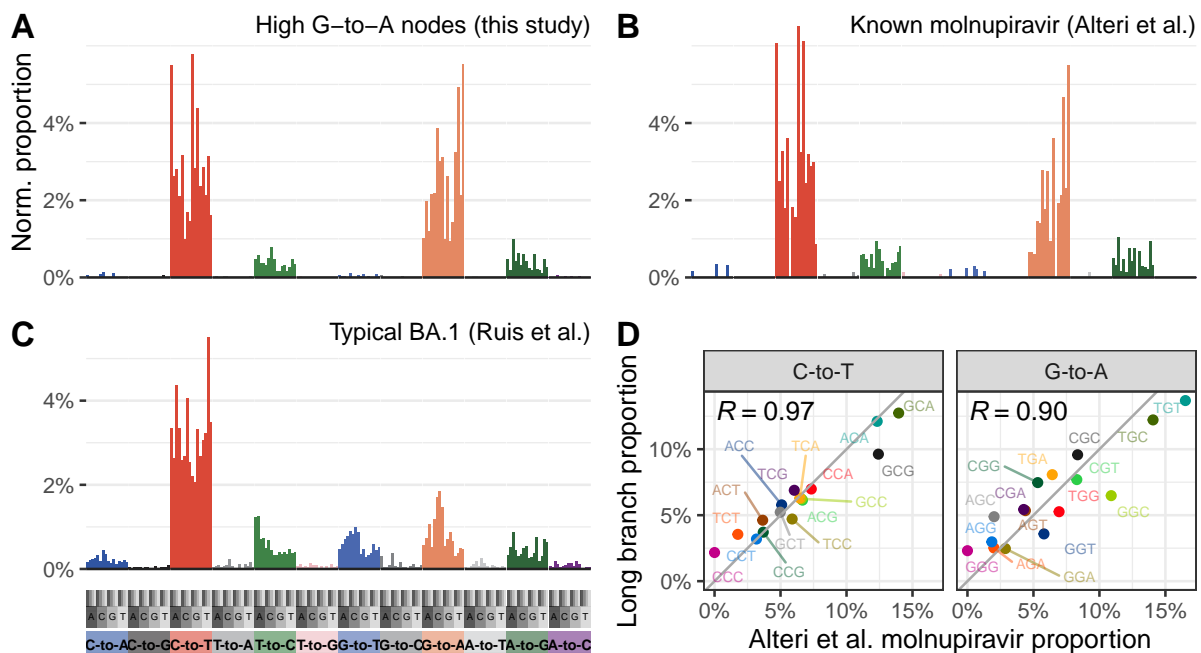
Warning: Duplicated aesthetics after name standardisation: label

Warning: The following aesthetics were dropped during statistical transformation: colour
 i This can happen when ggplot fails to infer the correct grouping structure in the data.

i Did you forget to specify a `group` aesthetic or to convert a numerical variable into a factor?

The following aesthetics were dropped during statistical transformation: colour
 i This can happen when ggplot fails to infer the correct grouping structure in the data.

i Did you forget to specify a `group` aesthetic or to convert a numerical variable into a factor?



```
ggsave("t2vlong.pdf", width = 8, height = 4)
```

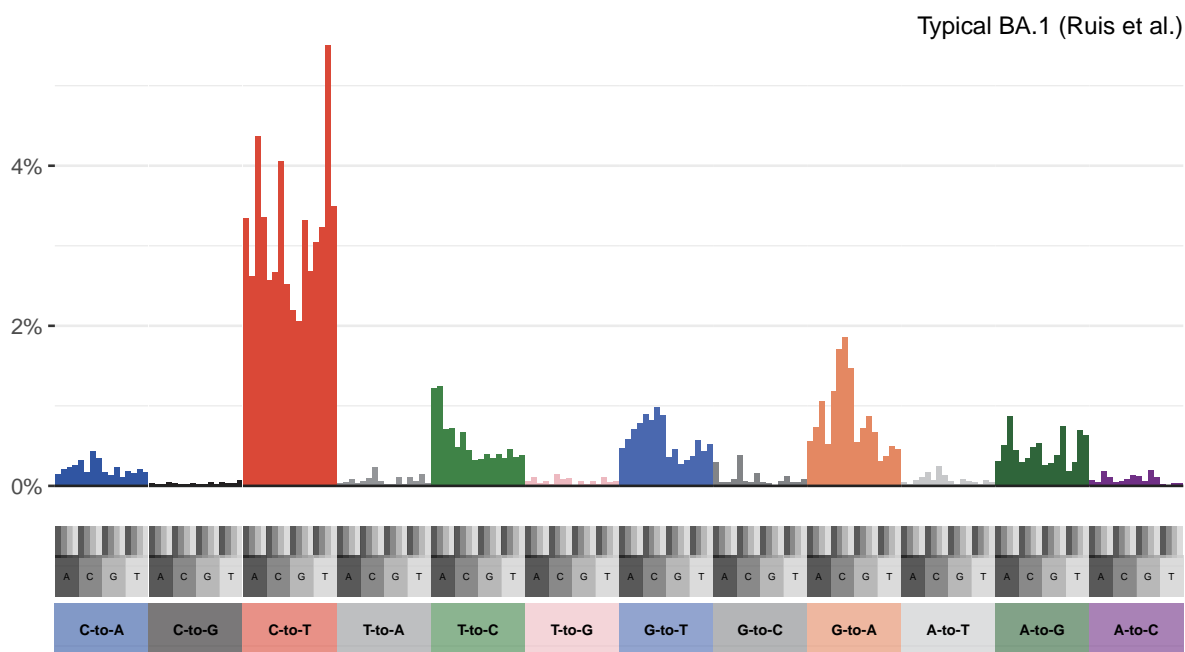
```
# plot_grid(p_long +labs(y="Norm. proportion") , p_t2 , p_ba1 , (scatters) , rel_heights

ggsave("t2vlong-present.pdf", width = 8, height = 4)

ggsave("~/movmanuscript2/Figures2/spectra.pdf", width = 8, height = 4)

p_ba1
```

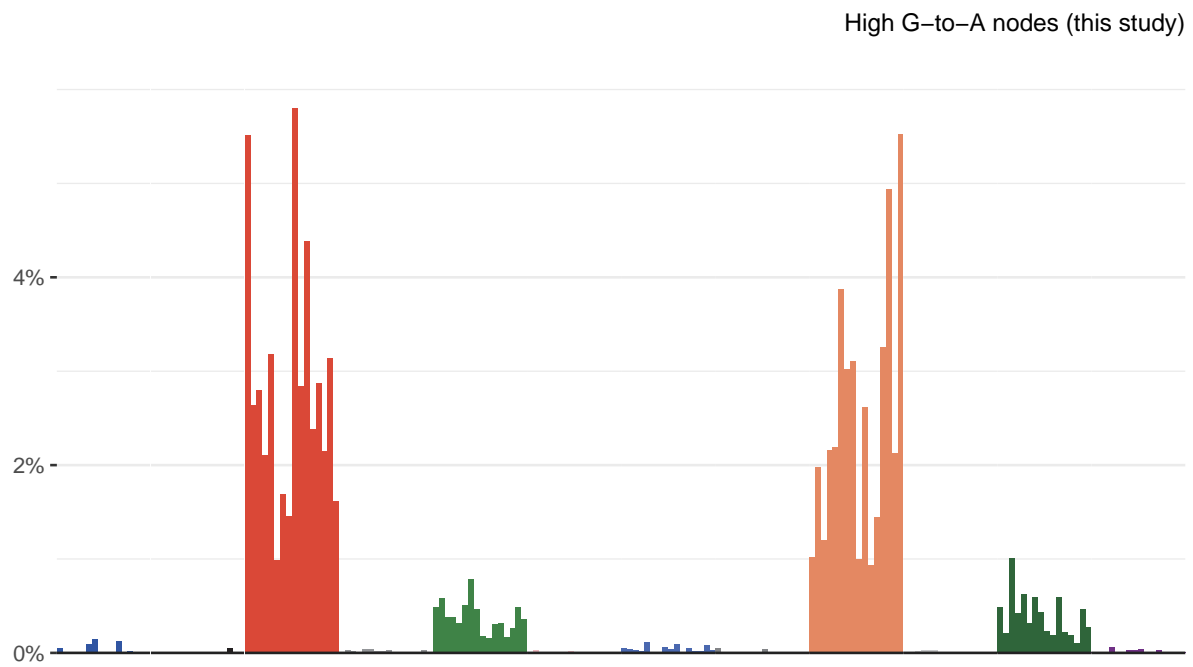
Warning: Duplicated aesthetics after name standardisation: label



```
ggsave("p_ba1.pdf", width = 0.5 * 10, height = 0.5 * 4.5)
```

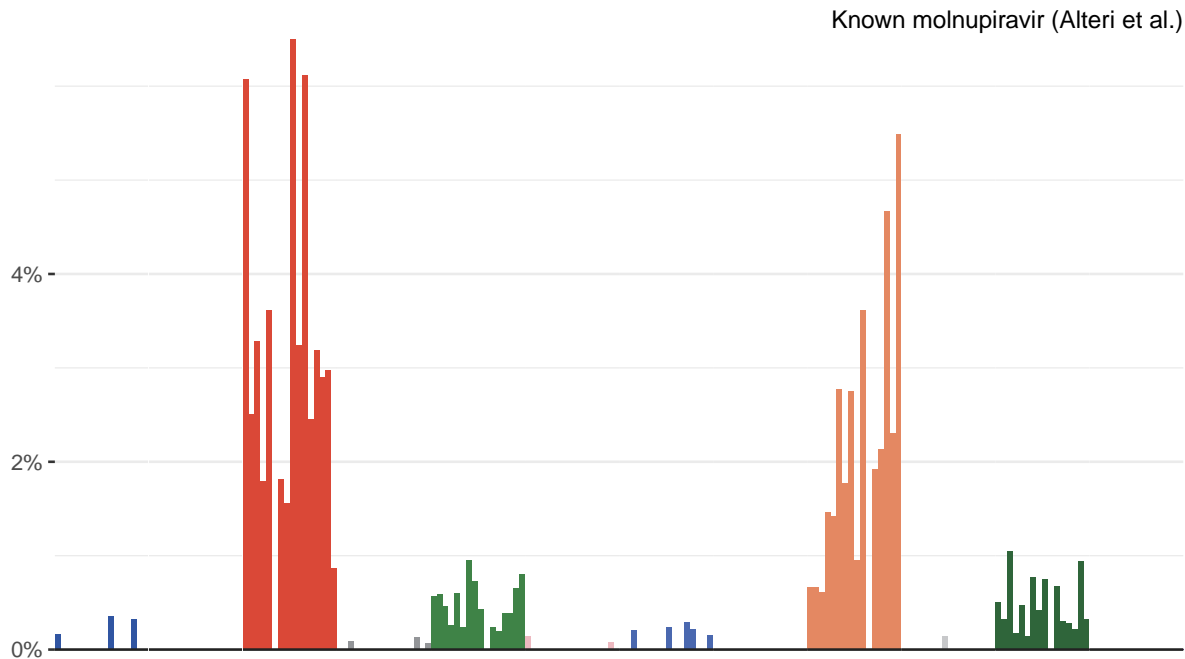
Warning: Duplicated aesthetics after name standardisation: label


```
p_long
```



```
ggsave("p_long.pdf", width = 0.5 * 10, height = 0.5 * 4.5)
```

```
p_t2
```



```
ggsave("p_t2.pdf", width = 0.5 * 10, height = 0.5 * 4.5)
```

```
scatters
```

Warning: The following aesthetics were dropped during statistical transformation: colour
 i This can happen when ggplot fails to infer the correct grouping structure in the data.

i Did you forget to specify a `group` aesthetic or to convert a numerical variable into a factor?

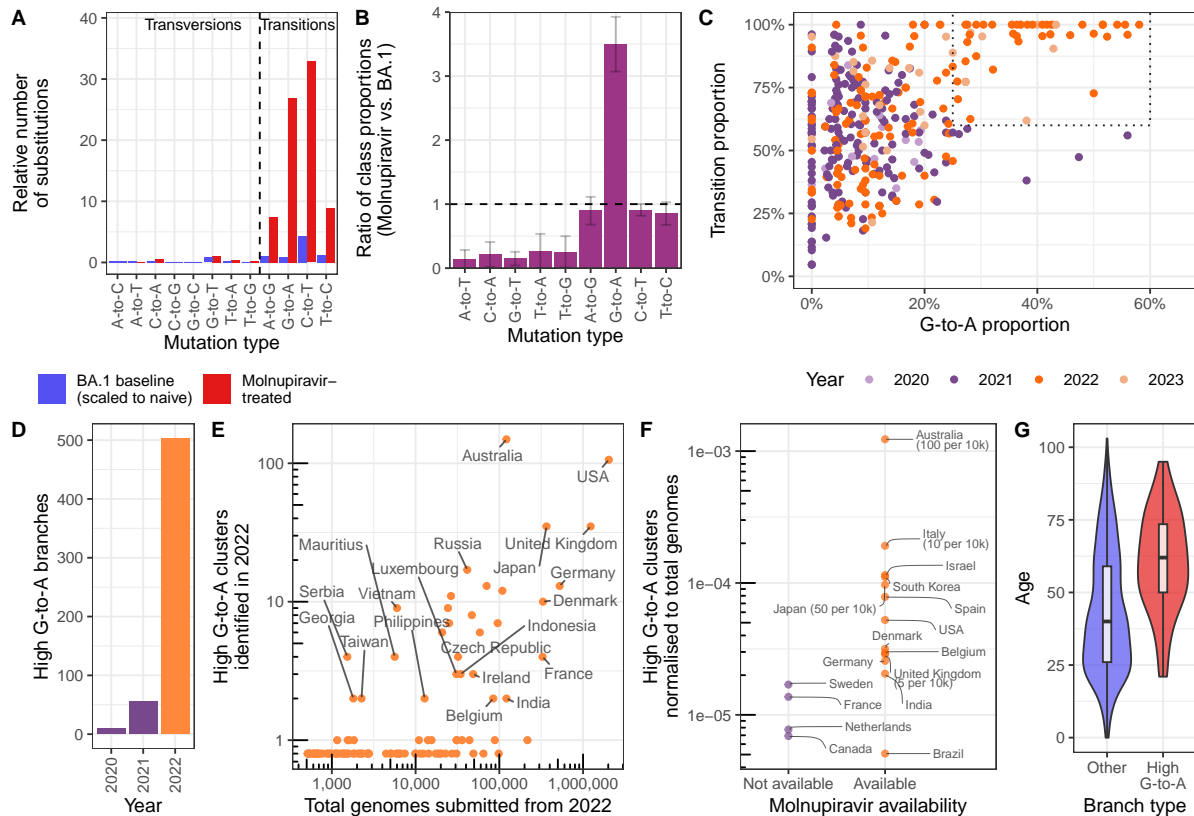
The following aesthetics were dropped during statistical transformation: colour
 i This can happen when ggplot fails to infer the correct grouping structure in the data.

i Did you forget to specify a `group` aesthetic or to convert a numerical variable into a factor?

Warning: Removed 1504 rows containing non-finite values (`stat_boxplot()`).

```
plot_grid(toprow, bottomrow, ncol = 1)
```

Warning: ggrepel: 83 unlabeled data points (too many overlaps). Consider increasing max.overlaps



```
ggsave("plot.pdf", width = 9.5, height = 6.5)
```

Warning: ggrepel: 83 unlabeled data points (too many overlaps). Consider increasing max.overlaps

```
library(patchwork)
```

```
layout <- "  
AAABBBBCCCCCCC  
DDEEEEEFFFFFGG  
"
```

```
a + b + scatter + by_year_plot + country_comp + availability_plot + age + plot_layout(desi
```

Warning in FUN(X[[i]], ...): NAs introduced by coercion

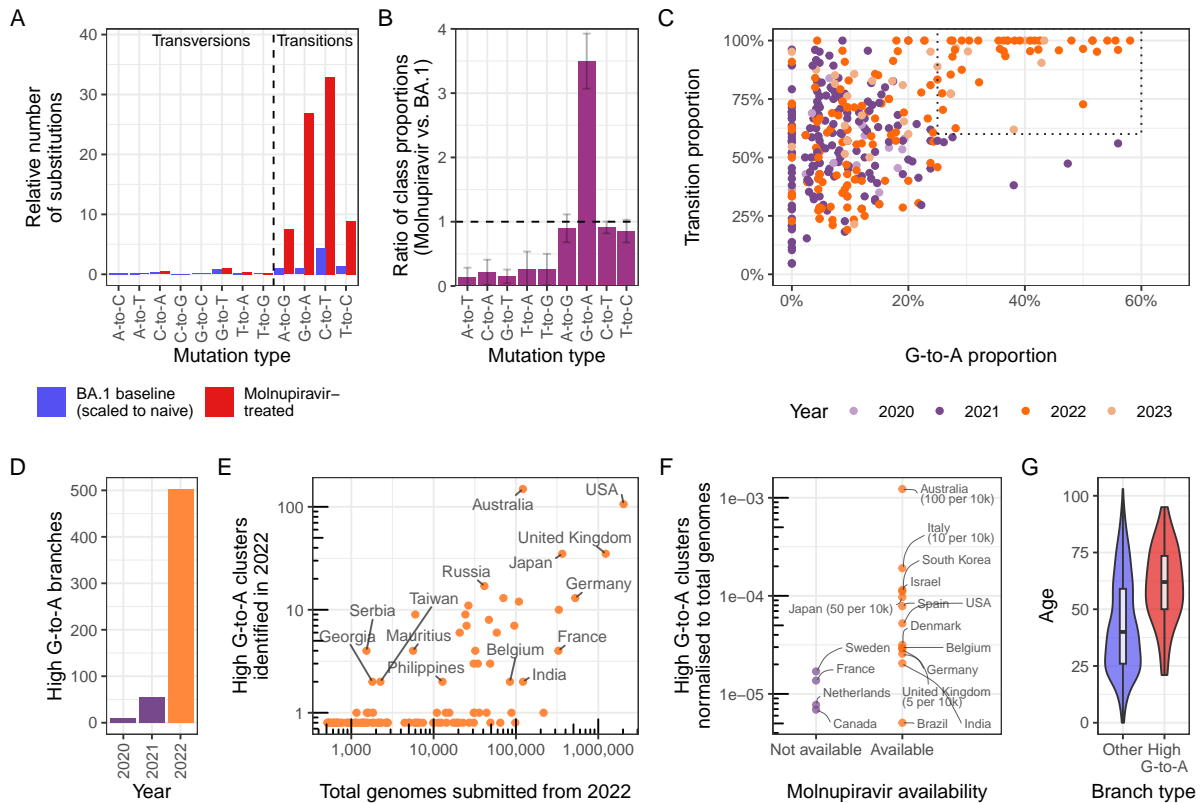
Warning in FUN(X[[i]], ...): NAs introduced by coercion

Warning in FUN(X[[i]], ...): NAs introduced by coercion

Warning: Removed 1504 rows containing non-finite values (`stat_ydensity()`).

Warning: Removed 1504 rows containing non-finite values (`stat_boxplot()`).

Warning: ggrepel: 89 unlabeled data points (too many overlaps). Consider increasing max.overlaps



```
ggsave("patchwork_version.pdf", width = 9.5, height = 6.5)
```

Warning in FUN(X[[i]], ...): NAs introduced by coercion

Warning in FUN(X[[i]], ...): NAs introduced by coercion

Warning in FUN(X[[i]], ...): NAs introduced by coercion

Warning: Removed 1504 rows containing non-finite values (`stat_ydensity()`).

Warning: Removed 1504 rows containing non-finite values (`stat_boxplot()`).

Warning: ggrepel: 89 unlabeled data points (too many overlaps). Consider increasing max.overlaps

```
ggsave("~/movmanuscript2/Figures2/patchworkcombo.pdf", width = 9.5, height = 6.5)
```

Warning in FUN(X[[i]], ...): NAs introduced by coercion

Warning in FUN(X[[i]], ...): NAs introduced by coercion

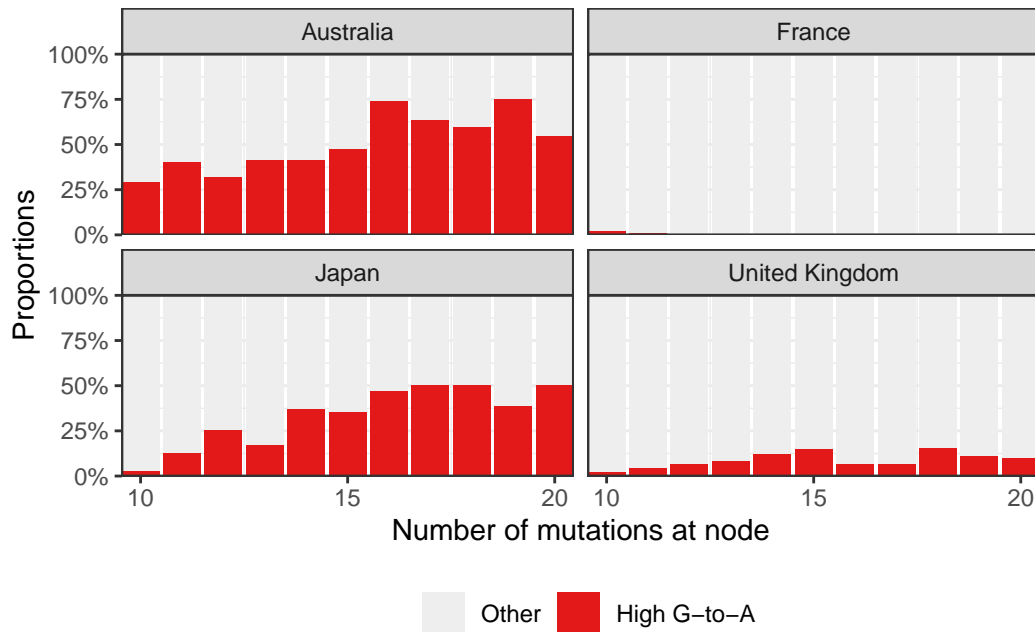
Warning in FUN(X[[i]], ...): NAs introduced by coercion

Warning: Removed 1504 rows containing non-finite values (``stat_ydensity()``).

Warning: Removed 1504 rows containing non-finite values (``stat_boxplot()``).

Warning: ggrepel: 89 unlabeled data points (too many overlaps). Consider increasing `max.overlaps`

```
countries <- c("Australia", "United Kingdom", "Japan", "France")
proportions_of_long_branches <- ggplot(data_nodes %>% filter(total_muts > 9, total_muts <
  geom_bar(position = "fill") +
  facet_wrap(~consensus_country) +
  theme_bw() +
  scale_y_continuous(labels = scales::percent, expand = c(0, 0)) +
  labs(x = "Number of mutations at node", y = "Proportions") +
  scale_fill_manual(labels = c("Other", "High G-to-A"), values = c("#eeeeee", red)) +
  scale_x_continuous(expand = c(0, 0), breaks = c(10, 15, 20)) +
  labs(fill = "") +
  theme(legend.position = "bottom")
proportions_of_long_branches
```



```
ggsave("plotter.pdf", width = 5, height = 4)
```

```
library(Biostrings)
data("BLOSUM62")
bl62 <- as.data.frame(as.table(BLOSUM62))

colnames(bl62) <- c("original_aa", "alternative_aa", "bl62_score")
```

```
all <- inner_join(data_nodes, data_muts %>% right_join(bl62), by = "node_id")
```

Joining with `by = join_by(original_aa, alternative_aa)`

```
adjustment_factor <- 3.24

dnds_stuff <- all %>%
  mutate(branch_type = case_when(
    total_muts >= threshold_branch_length & flagged ~ "Long branch, high G-to-A",
    total_muts >= threshold_branch_length & !flagged ~ "Long branch, not high G-to-A",
    TRUE ~ "Short branch"
  )) %>%
```



```

group_by(branch_type, is_synonymous) %>%
tally() %>%
group_by(branch_type) %>%
mutate(p = n / sum(n), ratio = n / (sum(n) - n), dnds = ratio / adjustment_factor)

```

```
dnds_stuff
```

```
# A tibble: 6 x 6
```

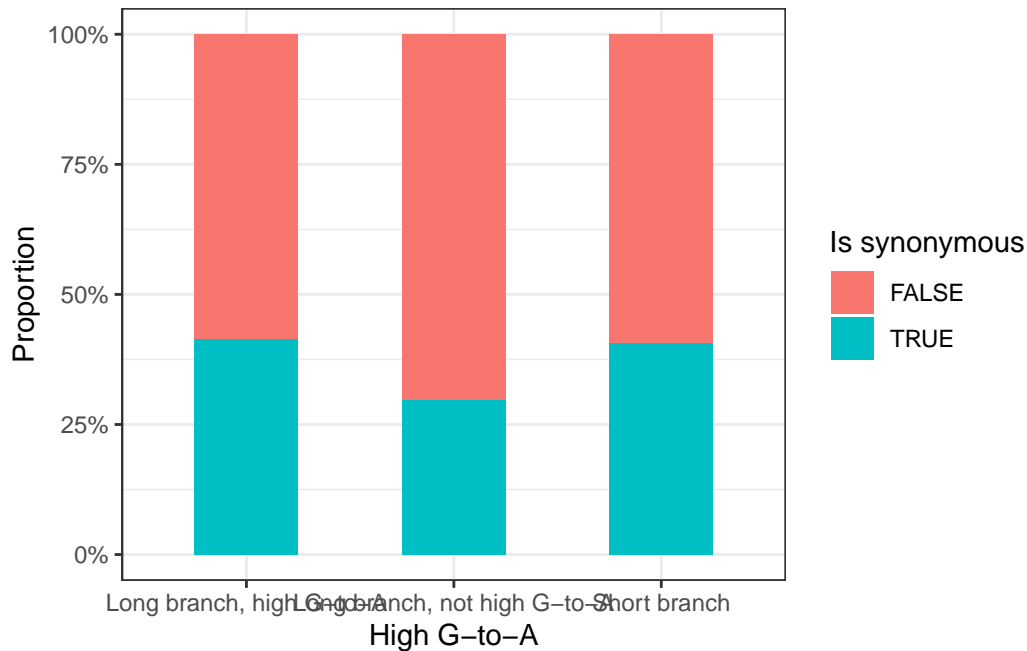
```
# Groups:   branch_type [3]
```

	branch_type	is_synonymous	n	p	ratio	dnds
	<chr>	<lgl>	<int>	<dbl>	<dbl>	<dbl>
1	Long branch, high G-to-A	FALSE	6284	0.586	1.41	0.436
2	Long branch, high G-to-A	TRUE	4447	0.414	0.708	0.218
3	Long branch, not high G-to-A	FALSE	242743	0.704	2.38	0.734
4	Long branch, not high G-to-A	TRUE	102123	0.296	0.421	0.130
5	Short branch	FALSE	6947450	0.593	1.46	0.450
6	Short branch	TRUE	4769323	0.407	0.686	0.212

```

dnds_stuff %>% ggplot(aes(y = p, fill = is_synonymous, x = branch_type)) +
  geom_col(width = 0.5) +
  scale_y_continuous(label = scales::percent) +
  theme_bw() +
  labs(fill = "Is synonymous", x = "High G-to-A", y = "Proportion")

```



```
ggsave("plot.png", width = 4, height = 3)

library(gt)
dnds_stuff %>% dplyr::filter(!is_synonymous)
```

```
# A tibble: 3 x 6
# Groups:   branch_type [3]
  branch_type      is_synonymous      n    p ratio dnds
  <chr>           <lgl>      <int> <dbl> <dbl> <dbl>
1 Long branch, high G-to-A FALSE      6284 0.586  1.41 0.436
2 Long branch, not high G-to-A FALSE    242743 0.704  2.38 0.734
3 Short branch      FALSE    6947450 0.593  1.46 0.450
```

```
# Assuming dnds_stuff is your dataframe
gt_table <- gt(dnds_stuff %>% dplyr::filter(!is_synonymous)) %>%
  cols_label(
    branch_type = "Branch Type",
    is_synonymous = "Is Synonymous",
    n = "Count",
    p = "Proportion",
    ratio = "Ratio",
```

```

    dnds = "dN/dS"
  ) %>%
  fmt_number(
    columns = vars(p, ratio, dnds),
    decimals = 2 # Adjust as needed
  )

```

Warning: Since gt v0.3.0, `columns = vars(...)` has been deprecated.

* Please use `columns = c(...)` instead.

Since gt v0.3.0, `columns = vars(...)` has been deprecated.

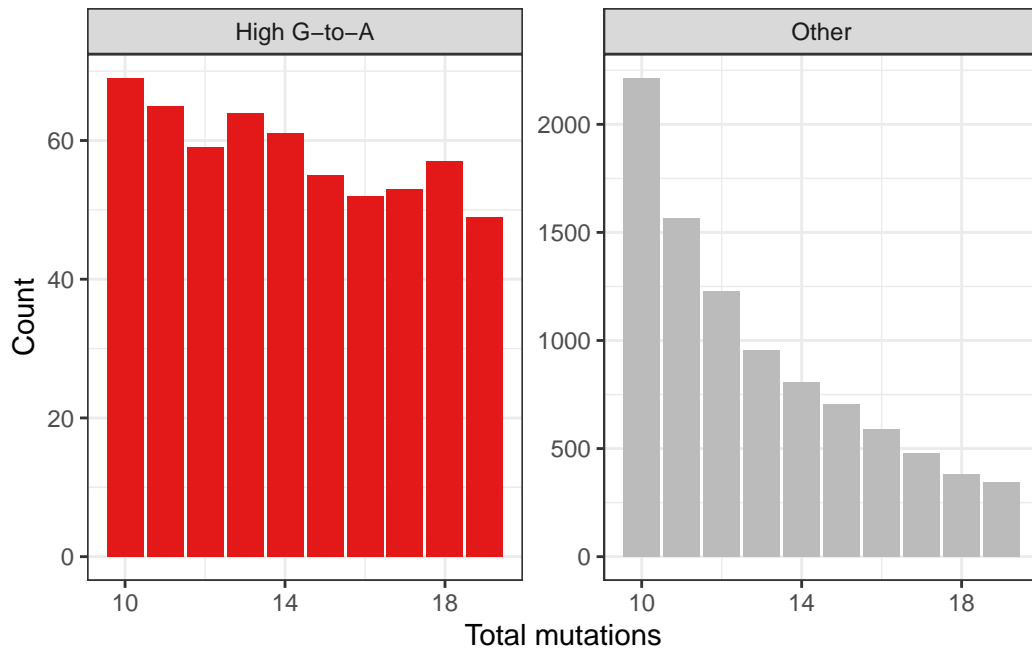
* Please use `columns = c(...)` instead.

```

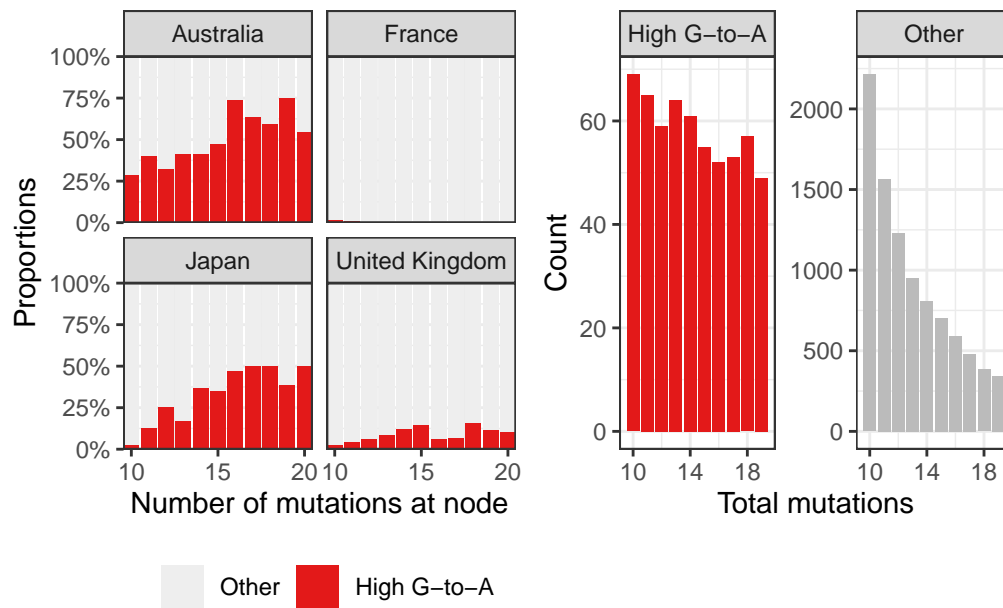
for_plot <- data_nodes %>%
  filter(consensus_year %in% c("2022", "2023"), total_muts > 9, total_muts < 20) %>%
  mutate(new = ifelse(flagged, "High G-to-A", "Other"))

distributions <- ggplot(for_plot, aes(x = total_muts, fill = flagged)) +
  geom_bar() +
  facet_wrap(~new, scales = "free_y") +
  theme_bw() +
  scale_x_continuous(breaks = c(10, 14, 18)) +
  scale_fill_manual(values = c("#bbbbbb", red)) +
  labs(x = "Total mutations", y = "Count") +
  theme(legend.position = "none")
distributions

```



proportions_of_long_branches + distributions

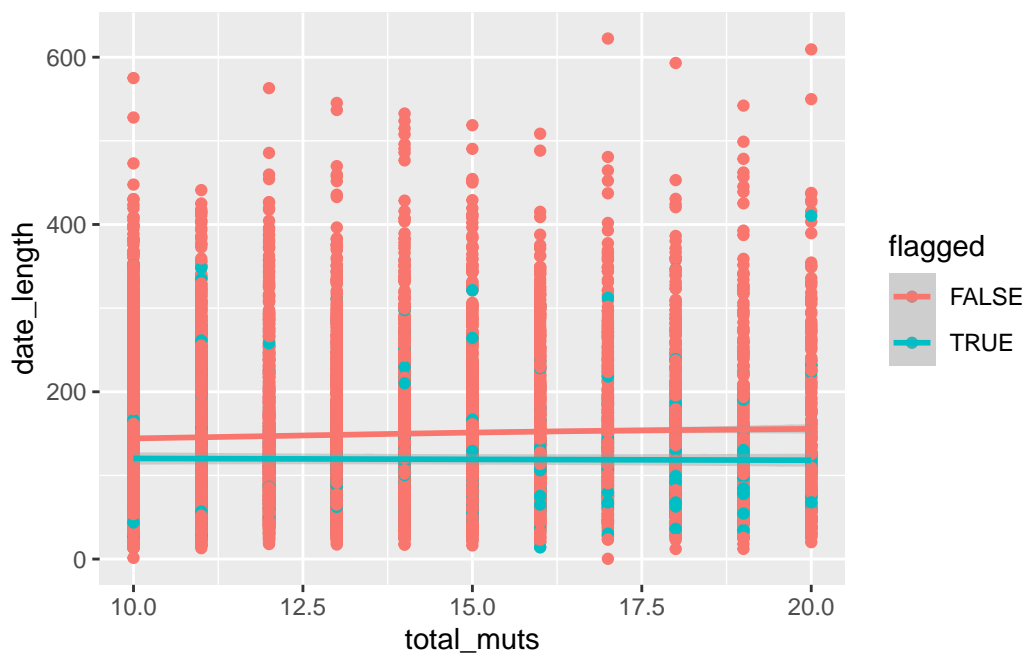


```
library(broom)
countries <- c("Australia", "Japan", "United Kingdom", "USA")

dataset <- data_nodes %>% filter(consensus_year == "2022", total_muts >= threshold_branch_

ggplot(dataset, aes(x = total_muts, y = date_length, color = flagged)) +
  geom_point() +
  geom_smooth()
```

`geom_smooth()` using method = 'gam' and formula = 'y ~ s(x, bs = "cs")'



dataset

A tibble: 8,388 x 23

	node_id	num_descendants	consensus_country	consensus_year	date
	<chr>	<dbl>	<chr>	<chr>	<date>
1	USA/TN-ASC-21050~	1	USA	2022	2022-01-04
2	India/MH-Kasturb~	1	India	2022	2022-01-31
3	Indonesia/JI-GS~	1	Indonesia	2022	2022-01-07
4	USA/NC-CDC-MMB13~	1	USA	2022	2022-01-19

```

5 Philippines/PH-R~          1 Philippines      2022          2022-04-11
6 Philippines/PH-R~          1 Philippines      2022          2022-04-24
7 Russia/SAR-RII-M~         1 Russia        2022          2022-01-13
8 node_39341                 2 Indonesia     2022          2021-12-21
9 node_39384                 2 Indonesia     2022          2021-12-23
10 Indonesia/KI-GS--         1 Indonesia     2022          2022-01-31
# i 8,378 more rows
# i 18 more variables: date_length <dbl>, age <chr>, `A>C` <dbl>, `A>G` <dbl>,
#   `A>T` <dbl>, `C>A` <dbl>, `C>G` <dbl>, `C>T` <dbl>, `G>A` <dbl>,
#   `G>C` <dbl>, `G>T` <dbl>, `T>A` <dbl>, `T>C` <dbl>, `T>G` <dbl>,
#   total_muts <dbl>, transitions <dbl>, transversions <dbl>, flagged <lgl>

```

```
t.test(data = dataset, date_length ~ flagged)
```

Welch Two Sample t-test

```

data:  date_length by flagged
t = 8.7949, df = 552.31, p-value < 2.2e-16
alternative hypothesis: true difference in means between group FALSE and group TRUE is not equal to 0
95 percent confidence interval:
 22.57814 35.56353
sample estimates:
mean in group FALSE  mean in group TRUE
      148.3975         119.3266

```

```

dataset2 <- dataset %>% filter(consensus_country %in% countries)

dataset3 <- dataset2 %>%
  group_by(total_muts, flagged, consensus_country) %>%
  summarize(
    mean_date_length = mean(date_length),
    se_date_length = sd(date_length) / sqrt(n()),
    conf.low = mean_date_length - qt(0.975, df = n() - 1) * se_date_length,
    conf.high = mean_date_length + qt(0.975, df = n() - 1) * se_date_length
  )

```

Warning: There were 8 warnings in `summarize()`.
The first warning was:

```
i In argument: `conf.low = mean_date_length - qt(0.975, df = n() - 1) *  
  se_date_length`.
```

```
i In group 6: `total_muts = 10`, `flagged = TRUE`, `consensus_country =  
  "Japan"`.
```

Caused by warning in `qt()`:

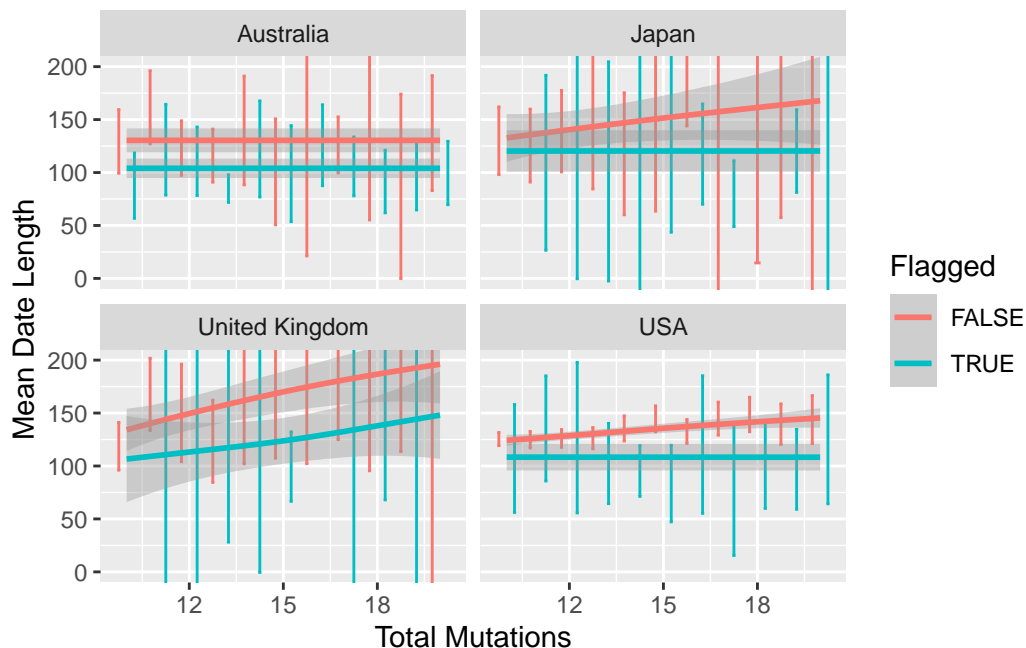
! NaNs produced

```
i Run `dplyr::last_dplyr_warnings()` to see the 7 remaining warnings.
```

`summarise()` has grouped output by 'total_muts', 'flagged'. You can override using the `.groups` argument.

```
ggplot(dataset3, aes(x = total_muts, y = mean_date_length, color = flagged)) +  
  geom_errorbar(aes(ymin = conf.low, ymax = conf.high), width = 0.2, position = position_d  
  facet_wrap(~flagged) +  
  labs(x = "Total Mutations", y = "Mean Date Length", color = "Flagged") +  
  facet_wrap(~consensus_country) +  
  coord_cartesian(ylim = c(0, 200)) +  
  geom_smooth(data = dataset2, aes(y = date_length))
```

`geom_smooth()` using method = 'gam' and formula = 'y ~ s(x, bs = "cs")'



```

final_df <- tibble()

many_descendants <- data_nodes %>%
  filter(total_muts > 9, flagged, num_descendants > 1)

# Loop through every single_node in many_descendants
for (i in 1:nrow(many_descendants)) {
  single_node <- many_descendants$node_id[i]
  children <- find_children(parenthood, single_node)
  children <- children[!grepl("^node_", children)]

  # Create a temporary tibble for the current node
  cluster_df <- tibble(node_id = children, cluster = single_node)

  # bind the current tibble with the final one
  final_df <- bind_rows(final_df, cluster_df)
}

single_descendants <- data_nodes %>%
  filter(total_muts > 9, flagged, num_descendants == 1)

single_df <- tibble(node_id = single_descendants$node_id, cluster = single_descendants$node_id)

final_df <- bind_rows(final_df, single_df)

final_df2 <- final_df %>%
  separate_wider_delim(node_id, delim = "|", names = c("name", "epi", "date"), cols_remove = 1)
  separate_wider_delim(name, delim = "/", names = c("country", "name2", "year"), cols_remove = 1)

final_df2 %>% filter(country == "England")

```

A tibble: 52 x 8

	country	name2	year	name	epi	date	node_id	cluster
	<chr>	<chr>	<chr>	<chr>	<chr>	<chr>	<chr>	<chr>
1	England	HSL-1AF5265	2021	England/HSL-1AF5265/~	EPI_~	2021~	Englan~	node_1~
2	England	HSL-1BBA08F	2021	England/HSL-1BBA08F/~	EPI_~	2021~	Englan~	node_1~
3	England	PHEP-YYFYNOA	2022	England/PHEP-YYFYNOA/~	EPI_~	2022~	Englan~	node_1~
4	England	PHEP-YYR9UXC	2022	England/PHEP-YYR9UXC/~	EPI_~	2022~	Englan~	node_1~


```

5 England HSL-383EB43 2022 England/HSL-383EB43/~ EPI_~ 2022~ Englan~ node_1~
6 England MILK-384AB80 2022 England/MILK-384AB80/~ EPI_~ 2022~ Englan~ node_1~
7 England PHEC-YYDRDI8 2022 England/PHEC-YYDRDI8/~ EPI_~ 2022~ Englan~ node_1~
8 England PHEC-YYDEK4Q 2022 England/PHEC-YYDEK4Q/~ EPI_~ 2022~ Englan~ node_1~
9 England DHSC-CYNN473 2022 England/DHSC-CYNN473/~ EPI_~ 2022~ Englan~ node_1~
10 England DHSC-CYD46UZ 2022 England/DHSC-CYD46UZ/~ EPI_~ 2022~ Englan~ node_1~
# i 42 more rows

```

```
write_csv(final_df2, "associated.csv")
```

```
library(ggtree)
```

```

format_mutation_counts <- function(node_data) {
  # Extract mutation count columns
  mutation_cols <- c("A>C", "A>G", "A>T", "C>A", "C>G", "C>T", "G>A", "G>C", "G>T", "T>A",
                    "T>C", "T>G", "T>T")

  # Create a named vector of mutation counts
  mutation_counts <- sapply(mutation_cols, function(x) node_data[[x]])
  names(mutation_counts) <- mutation_cols

  # Remove zeros
  mutation_counts <- mutation_counts[mutation_counts > 0]

  # Sort in descending order
  mutation_counts <- sort(mutation_counts, decreasing = TRUE)

  # Format as a string
  mutation_str <- paste(names(mutation_counts), mutation_counts, sep = ": ", collapse = ", ")
  mutation_str <- gsub(">", "\u00ad", mutation_str)

  return(mutation_str)
}

prune_and_plot <- function(node_id, parent, node_data) {
  mutation_title <- format_mutation_counts(node_data)
  print(node_id)

  # Create directories if they do not exist
  if (!dir.exists("data")) {
    dir.create("data")
  }
}

```

```

if (!dir.exists("trees")) {
  dir.create("trees")
}

gotree_command <- paste0("~/Dropbox/new_mov_data/gotree_arm64_darwin subtree -i ~/Dropbox

print(gotree_command)

# Execute the system call
system(gotree_command)

# Read the newick file
tree <- read.tree(paste0("data/pruned_", node_id, ".nwk"))

get_node_index <- function(tree, node_name) {
  for (i in 1:length(tree$node.label)) {
    if (tree$node.label[i] == node_name) {
      return(i + ape::Ntip(tree)) # Return the index of the node
    }
  }
  return(NULL) # Return NULL if no node with that name was found
}

node_index <- get_node_index(tree, node_id)

# Plot the tree using ggtree
ggtree_plot <- ggtree(tree, aes( # color=node==node_index
)) +
  geom_tiplab(size = 3, aes(label = label)) + # Add tip labels
  geom_point2(aes(subset = !is.na(num_tips)), color = "#4561de") + # Add points to visual
  coord_cartesian(clip = "off") +
  theme_tree2(plot.margin = margin(6, 290, 6, 6)) +
  theme(legend.position = "none") + #+scale_color_manual(values = c("TRUE" = "darkblue",
  geom_text(aes(x = branch, label = ifelse(node == node_index, mutation_title, "")),
    size = 3,
    vjust = -.4, color = "firebrick"
  ) #+ggtitle(node_id)

```

```

# Calculate the number of tips
num_tips <- ape::Ntip(tree)

# Calculate a reasonable height for the plot
# Adjust this calculation as needed
plot_height <- max(1.5, num_tips / 5)

# Save the plot to a pdf
# ggsave(filename = paste0("trees/node_", node_id, ".pdf"), plot = ggtree_plot, height =

return(list(ggtree_plot, plot_height))
}

filtered_nodes <- data_nodes %>%
  filter(total_muts > 9, flagged, num_descendants > 2) %>%
  arrange(desc(num_descendants))

filtered_nodes

library(patchwork)

# Step 1
plots_list <- list()
heights_list <- c()
total_height <- 0
# Introduce plot_number
plot_number <- 1

# In your loop

pdf("trees/combined_plots.pdf", height = 11.7, width = 8.3) # Create a PDF file

for (i in 1:nrow(filtered_nodes)) {
  print(i)

  listed <- prune_and_plot(filtered_nodes$node_id[i], get_parent(parenthood, filtered_node

  ggtree_plot <- listed[[1]]
  plot_height <- listed[[2]]

  # Check if adding the new plot will exceed the page size

```

```

if ((total_height + plot_height) >= 16) { # A4 height in inches
  # Save the existing plots
  if (length(plots_list) > 0) {
    combined_plot <- wrap_plots(plots_list) +
      plot_layout(heights = heights_list / total_height) # Normalize to make it relative
    print(combined_plot)
    ggsave(filename = paste0("trees/combined_", plot_number, ".pdf"), plot = combined_pl

    plot_number <- plot_number + 1 # Increment plot_number
  }
  # Reset the list and total height
  plots_list <- list()
  heights_list <- c()
  total_height <- 0
}

if (plot_height < 15 * 5) {
  # Add the new plot
  plots_list[[length(plots_list) + 1]] <- ggtree_plot
  heights_list <- c(heights_list, plot_height)
  total_height <- total_height + plot_height
}
}

# After the loop, save any remaining plots
if (length(plots_list) > 0) {
  combined_plot <- wrap_plots(plots_list) +
    plot_layout(heights = heights_list / total_height) # Normalize to make it relative
  ggsave(filename = paste0("trees/combined_", plot_number, ".pdf"), plot = combined_plot,
}

dev.off() # Close the PDF device

library(gggenes)
library(tidyverse)

# Read data
hu1 <- read_tsv("./hu1.tsv")

```

Rows: 38 Columns: 4

```
-- Column specification -----
Delimiter: "\t"
chr (2): feature_name, feature_type
dbl (2): start, end

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.
```

```
# Define unique end_points
end_points <- unique(hu1$end)

# Define a function to generate the vertical line
generate_vline <- function(end_points) {
  geom_vline(
    xintercept = end_points # , linetype = "dashed"
    , color = "lightgray", size = .2
  )
}

# Define common theme
common_theme <- theme(
  axis.ticks = element_line(color = "black"),
  panel.grid.major = element_blank(),
  panel.grid.minor = element_blank()
)

# Define filtered hu1
filtered_hu1 <- hu1 %>% filter(feature_type %in% c("CDS", "mat_peptide"))

# hu1_plot
hu1_plot <- ggplot(filtered_hu1, aes(xmin = start, xmax = end, y = feature_type, fill = fe
  generate_vline(end_points) +
  scale_fill_manual(values = c("#fbe4bc", "#dff3f8")) +
  labs(x = "Nucleotide position", y = "Feature", fill = "Type") +
  theme_minimal() +
  geom_gene_arrow() +
  geom_gene_label() +
  common_theme +
  labs(y = "") +
  theme(axis.text.y = element_blank(), axis.ticks.y = element_blank()) +
  theme(plot.margin = margin(t = 0, r = 5, l = 5, b = 0)) +
```

```

xlim(c(0, NA))

# Define myset
myset <- all %>%
  mutate(blcut = cut(bl62_score, 3)) %>%
  filter(total_muts > 10, flagged) %>%
  mutate(mut_type = case_when(
    (alternative_aa == "*") & (original_aa != "*") ~ "STOP",
    # bl62_score < -0 ~ "Negative BLOSUM",
    is_synonymous ~ "Synonymous",
    TRUE ~ "Non-synonymous (all)"
  )) %>%
  filter(mut_type != "STOP")

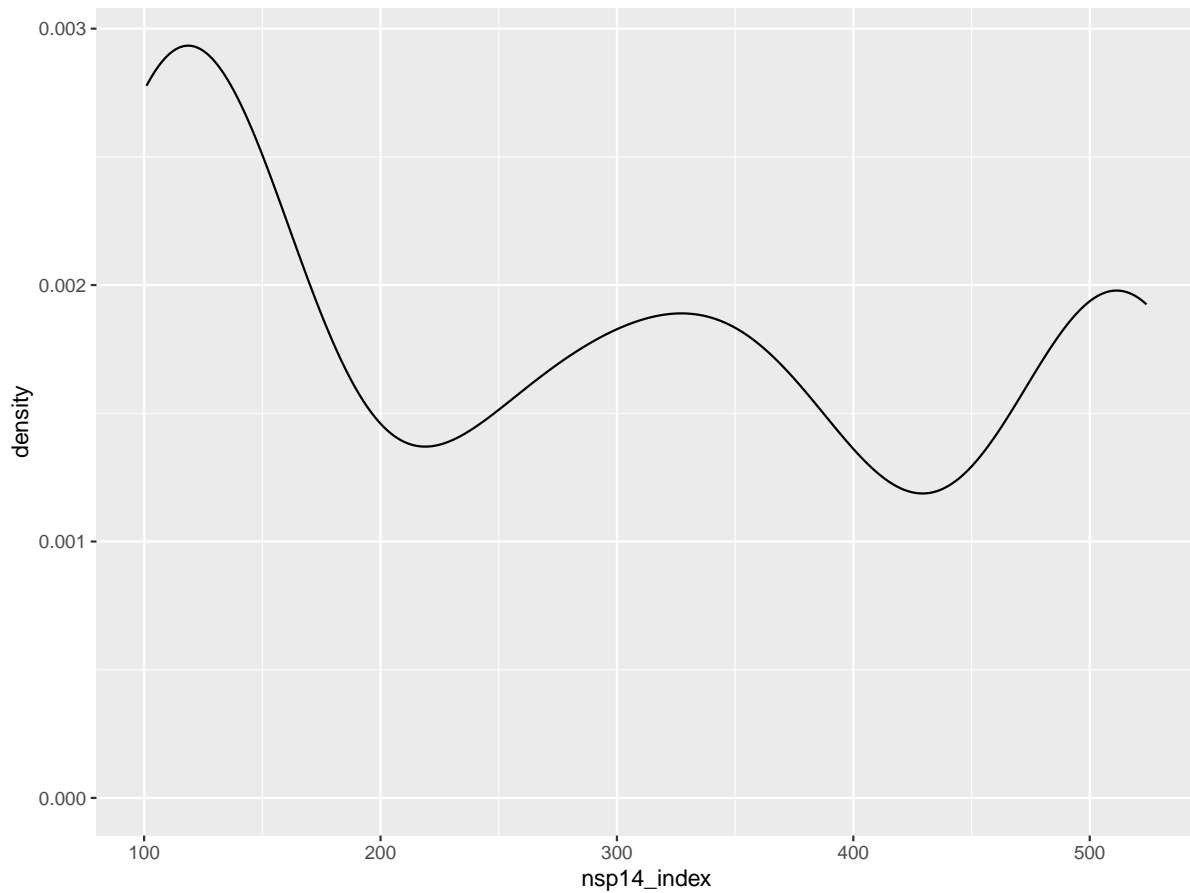
start_nsp14_codon <- 5926
end_nsp14_codon <- 6452

myset <- myset %>%
  mutate(
    is_nsp14 = ifelse(gene == "ORF1ab" & aa_index >= start_nsp14_codon & aa_index <= end_nsp14_codon, 1, 0),
    nsp14_index = ifelse(is_nsp14, aa_index - start_nsp14_codon + 1, NA)
  )

nsp14_muts <- myset %>%
  filter(is_nsp14, !is_synonymous) %>%
  group_by(aa_string, nsp14_index) %>%
  tally() %>%
  arrange(-n)

ggplot(nsp14_muts %>% filter(n > 4), aes(x = nsp14_index)) +
  geom_density(bw = 50)

```



```
multiple <- myset %>%  
  filter(!is_synonymous) %>%  
  group_by(nt_index) %>%  
  tally() %>%  
  filter(n > 3) %>%  
  mutate(mut_type = "Non-synonymous (site recurrent 4+ times)")  
  
fullmyset <- bind_rows(myset, multiple)  
  
my_colors <- c(  
  "STOP" = "#D55E00",  
  "Synonymous" = "#008837",  
  "Non-synonymous (all)" = "#c2a5cf",  
  "Non-synonymous (site recurrent 4+ times)" = "#7b3294"  
)
```

```

density_plot <- ggplot(fullmyset, aes(x = nt_index, color = mut_type, group = mut_type)) +
  generate_vline(end_points) +
  geom_density(bw = 500) +
  theme_minimal() +
  common_theme +
  theme(
    axis.title.x = element_blank(),
    axis.text.x = element_blank(),
    axis.ticks.x = element_blank(),
    legend.position = "bottom", # change position to top, bottom, left, right or c(x, y) f
    legend.direction = "horizontal"
  ) +
  geom_hline(yintercept = 0, color = "#444444", size = 0.4) +
  geom_vline(xintercept = 0, color = "#444444", size = 0.4) +
  scale_color_manual(values = my_colors) +
  labs(y = "Density", color = "") +
  theme(plot.margin = margin(t = 5, r = 5, l = 5, b = 0))

```

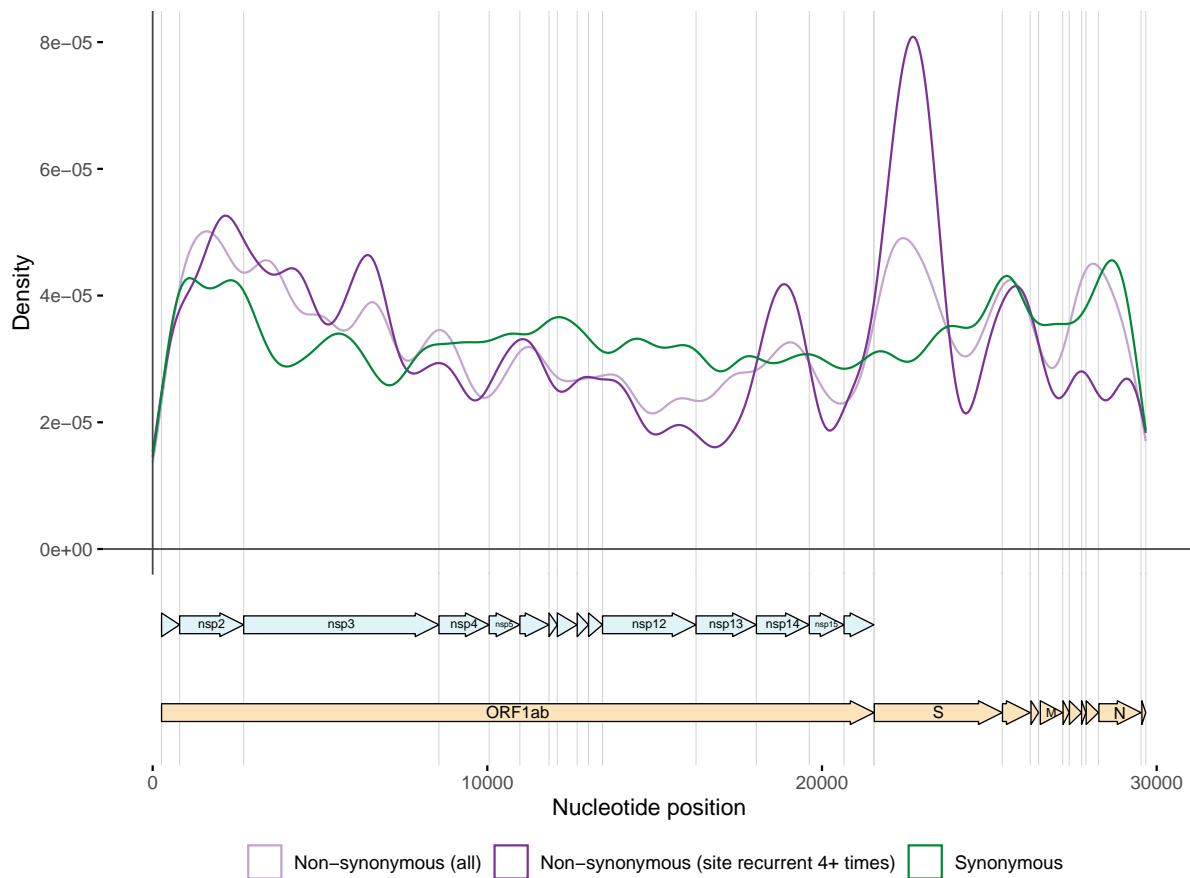
density_plot



```
# Legends
legends <- plot_grid(get_legend(density_plot))
legends
```

Non-synonymous (all) Non-synonymous (site recurrent 4+ times) Synonymous

```
# Final plot
final_plot <- plot_grid(density_plot + theme(legend.position = "none"), NULL, hu1_plot + t
final_plot
```



```
ggsave("genome.pdf", width = 10, height = 4)
```

```
fortable <- myset %>%
  filter(!is_synonymous) %>%
  group_by(original_aa, alternative_aa, gene, aa_index, mutation_type) %>%
  tally() %>%
  mutate(mut_types = paste0(mutation_type, ":", n)) %>%
  group_by(original_aa, alternative_aa, gene, aa_index) %>%
  summarise(n = sum(n), types = paste(mut_types, collapse = ", ")) %>%
  arrange(-n) %>%
  filter(gene == "S") %>%
  mutate(mut_format = paste0("S:", original_aa, aa_index, alternative_aa)) %>%
  mutate(type = substr(types, 1, 3)) %>%
  ungroup()
```

`summarise()` has grouped output by 'original_aa', 'alternative_aa', 'gene'.

You can override using the ``.groups`` argument.

```
library(gridExtra)
```

Attaching package: 'gridExtra'

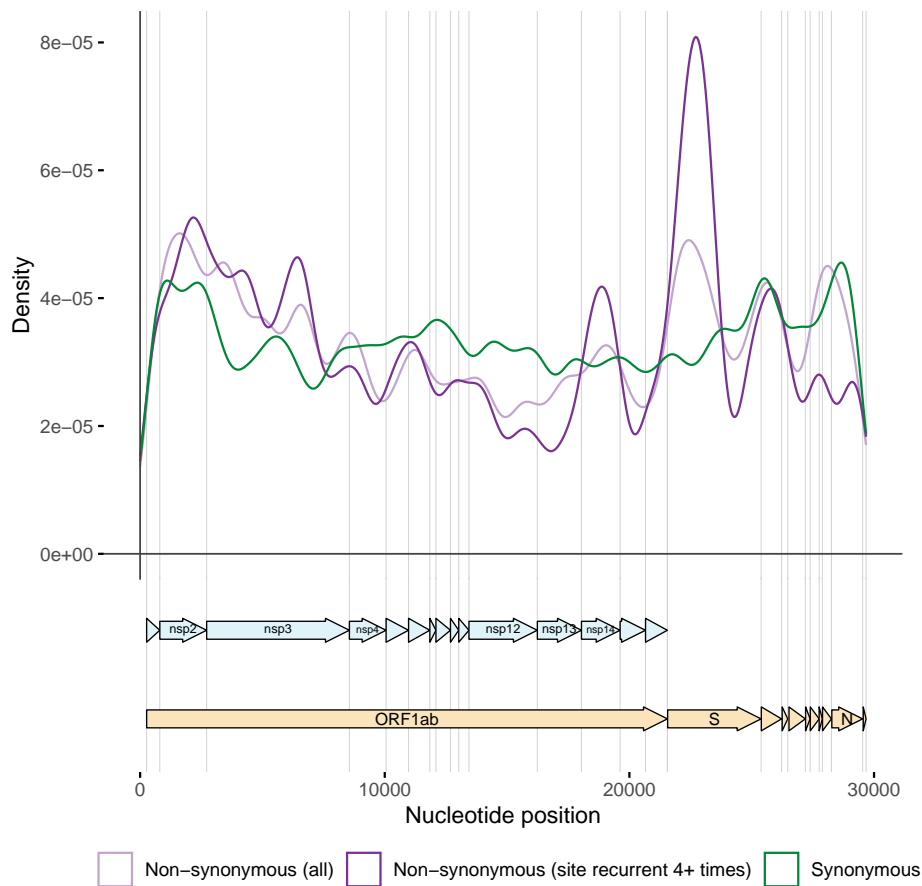
The following object is masked from 'package:dplyr':

`combine`

The following object is masked from 'package:BiocGenerics':

`combine`

```
table_theme <- ttheme_default(  
  core = list(fg_params = list(cex = 0.6)), # font size for table body  
  colhead = list(fg_params = list(cex = 0.6)), # font size for column headers  
  rowhead = list(fg_params = list(cex = 0.6)) # font size for row headers  
)  
  
# Convert the fortable data frame to a table grob  
table_grob <- tableGrob(fortable %>% filter(n > 4) %>% arrange(-n, aa_index) %>% select(mu  
  
grid.arrange(final_plot, table_grob, ncol = 2, widths = c(3, 1))
```



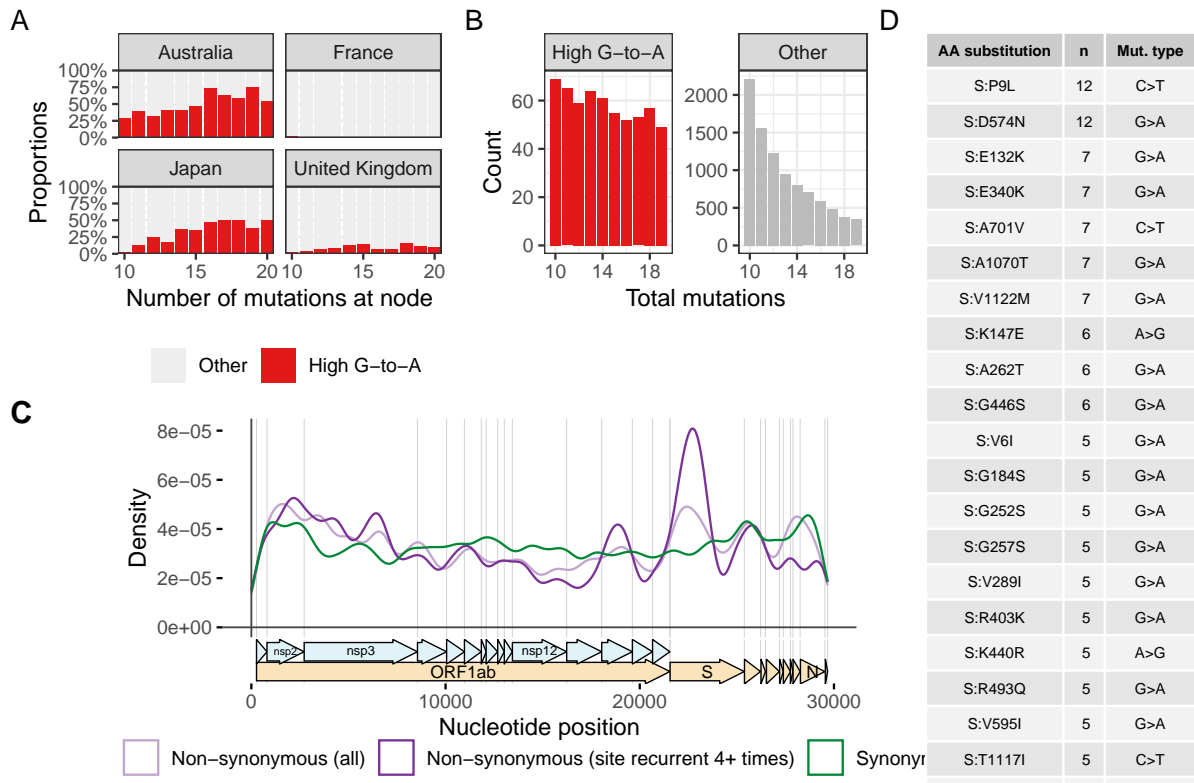
AA substitution	n	Mut. type
S:P9L	12	C>T
S:D574N	12	G>A
S:E132K	7	G>A
S:E340K	7	G>A
S:A701V	7	C>T
S:A1070T	7	G>A
S:V1122M	7	G>A
S:K147E	6	A>G
S:A262T	6	G>A
S:G446S	6	G>A
S:V6I	5	G>A
S:G184S	5	G>A
S:G252S	5	G>A
S:G257S	5	G>A
S:V289I	5	G>A
S:R403K	5	G>A
S:K440R	5	A>G
S:R493Q	5	G>A
S:V595I	5	G>A
S:T1117I	5	C>T
S:V1128I	5	G>A

```
library(ggplotify)
table_plot <- as.ggplot(table_grob)

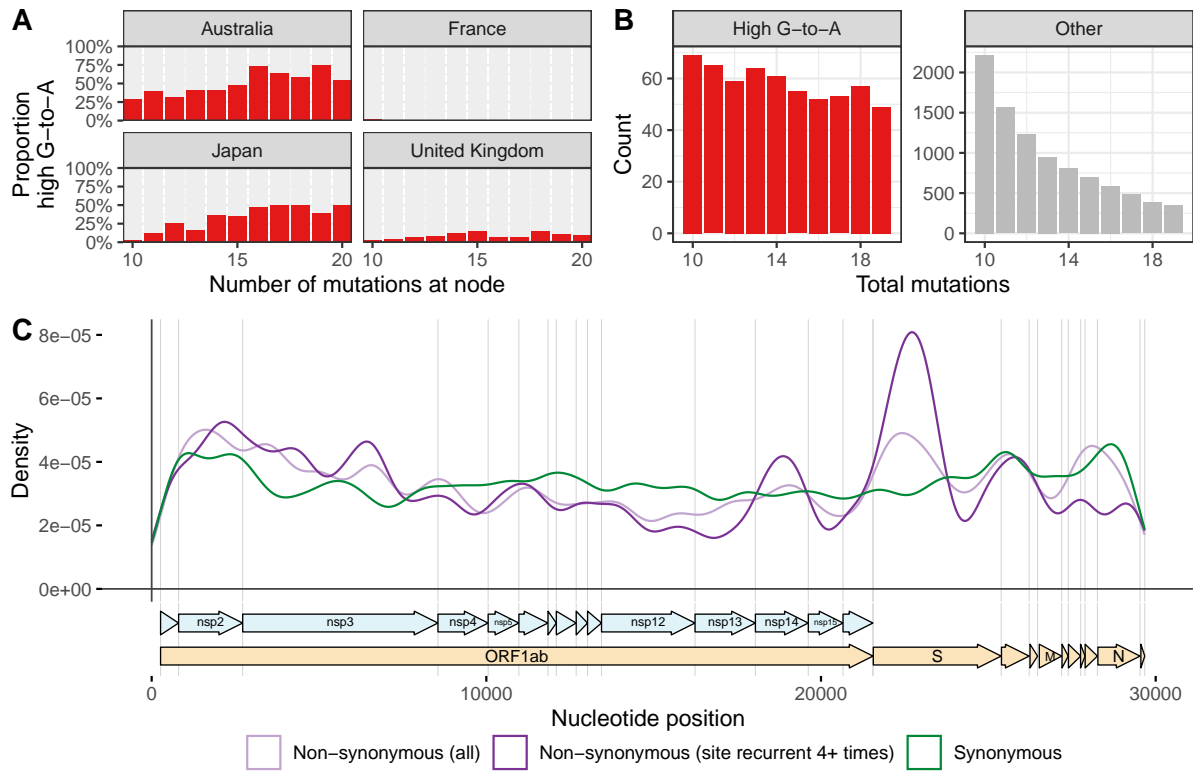
# Arrange the plot and table using patchwork
final_figure <-
  proportions_of_long_branches + distributions +
  final_plot + table_plot +
  plot_layout(ncol = 2, widths = c(3, 1))

layout <- "
AABBDD
CCCCDD
CCCCDD
"
proportions_of_long_branches + distributions +
```

```
final_plot + table_plot +
plot_layout(design = layout) + plot_annotation(tag_levels = "A")
```

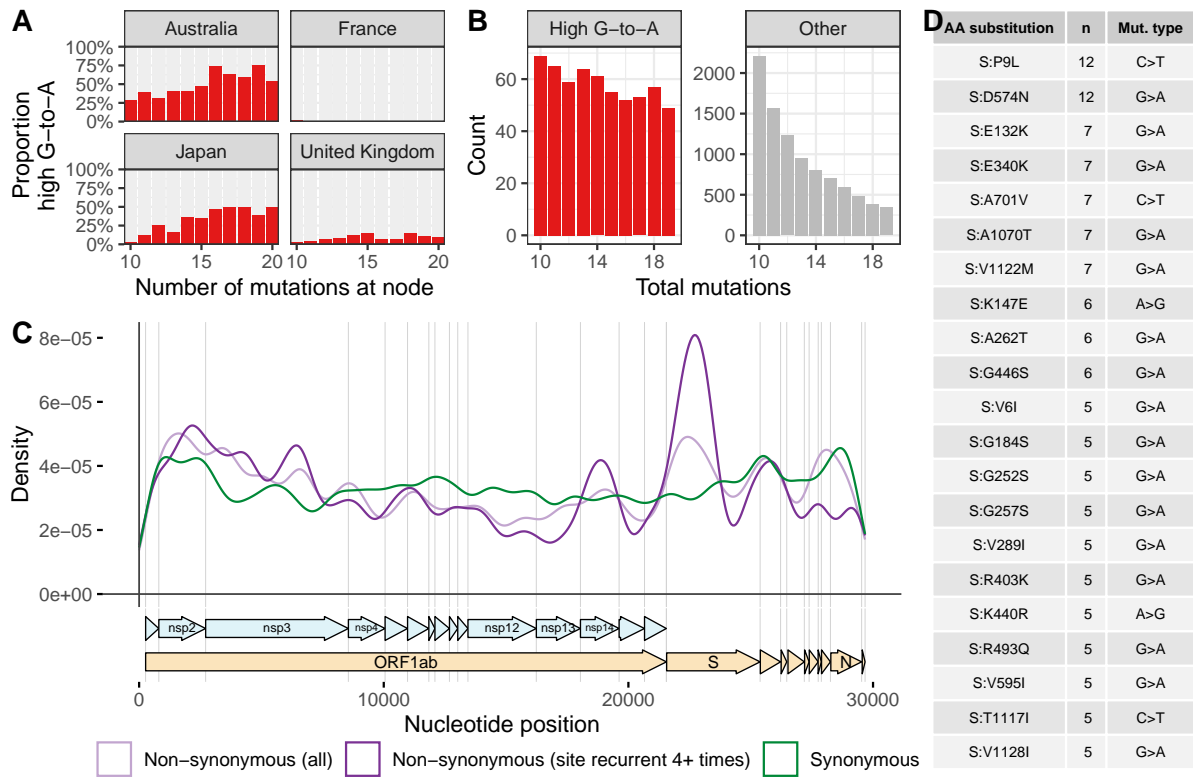


```
a <- plot_grid(proportions_of_long_branches + theme(legend.position = "none") + labs(y = "
b <- plot_grid(a, final_plot, ncol = 1, labels = c("", "C"), rel_heights = c(0.4, 0.6))
b
```



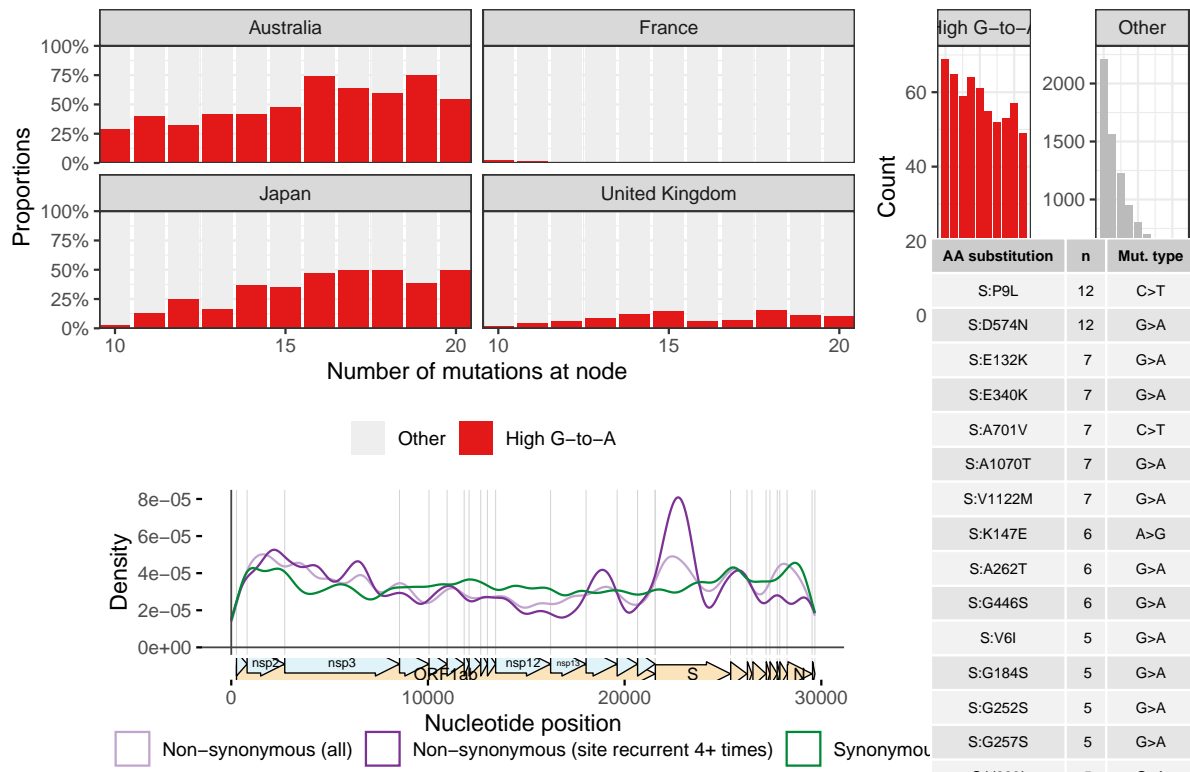
```
c <- plot_grid(b, table_plot, labels = c("", "D"), rel_widths = c(0.75, 0.25))
```

c



```
ggsave("~/movmanuscript2/Figures2/fig_last.pdf", width = 8, height = 5.15)
```

```
# Print the final figure
print(final_figure)
```



nsp14_muts

```
# A tibble: 177 x 3
# Groups:   aa_string [177]
  aa_string      nsp14_index    n
  <chr>          <dbl> <int>
1 ORF1ab:S6428L      503     6
2 ORF1ab:A6044T      119     5
3 ORF1ab:A6245V      320     5
4 ORF1ab:T6056I      131     5
5 ORF1ab:T6175I      250     5
6 ORF1ab:T6303I      378     5
7 ORF1ab:T6449I      524     5
8 ORF1ab:V6026I      101     5
9 ORF1ab:A6296V      371     4
10 ORF1ab:A6319T      394     4
# i 167 more rows
```



```
nsp14_muts$nsp14_index[1:30]
```

```
[1] 503 119 320 131 250 378 524 101 371 394 228 113    4 120 317 437 308 471 432
[20] 427 315 461   31 372 287 346 119 187 307 344
```

```
data_nodes %>%
  filter(flagged) %>%
  filter(total_muts >= 10) %>%
  arrange(-num_descendants)
```

```
# A tibble: 732 x 23
```

	node_id	num_descendants	consensus_country	consensus_year	date
	<chr>	<dbl>	<chr>	<chr>	<date>
1	node_2589094	20	Australia	2022	2022-07-12
2	node_2669086	13	New Zealand	2023	2023-01-11
3	node_2661929	11	Australia	2022	2022-11-27
4	node_2324333	9	Australia	2022	2022-09-01
5	node_1676393	7	Italy	2022	2022-03-30
6	node_3155949	7	Australia	2022	2022-09-22
7	node_388695	6	India	2021	2021-10-29
8	node_2325826	6	Japan	2023	2023-01-31
9	node_1994127	5	Japan	2022	2022-03-05
10	node_2359282	5	Australia	2022	2022-05-31

```
# i 722 more rows
```

```
# i 18 more variables: date_length <dbl>, age <chr>, `A>C` <dbl>, `A>G` <dbl>,
#   `A>T` <dbl>, `C>A` <dbl>, `C>G` <dbl>, `C>T` <dbl>, `G>A` <dbl>,
#   `G>C` <dbl>, `G>T` <dbl>, `T>A` <dbl>, `T>C` <dbl>, `T>G` <dbl>,
#   total_muts <dbl>, transitions <dbl>, transversions <dbl>, flagged <lgl>
```

```
# Function to read FASTA file and convert to a tibble
```

```
read_fasta_to_tibble <- function(file_path) {
```

```
  # Load the fasta file
```

```
  fasta_data <- readDNASTringSet(file_path)
```

```
  # Get sequence from the first (and possibly only) sequence in the fasta file
```

```
  sequence <- as.character(fasta_data[[1]])
```

```
  residues <- strsplit(sequence, "")[[1]]
```

```
  # Create a tibble with residue and index
```

```
  tibble(
```

```

      index = seq_along(residues),
      residue = residues
    )
  }

ref_tib <- read_fasta_to_tibble("ref.fa.fasta") %>% mutate(context_before = lag(residue),
ref_tib

# A tibble: 29,903 x 4
  index residue context_before context_after
  <int> <chr>      <chr>          <chr>
1     1  A      <NA>          T
2     2  T      A            T
3     3  T      T            A
4     4  A      T            A
5     5  A      A            A
6     6  A      A            G
7     7  G      A            G
8     8  G      G            T
9     9  T      G            T
10    10  T      T            T
# i 29,893 more rows

mutations <- c("G405A", "A761G", "C912A", "G1820A", "C1931A", "C2455T", "T2954C", "C5284T")

mutations_df <- all %>%
  filter(total_muts > 10, flagged) %>%
  dplyr::select(original_nt, alternative_nt, nt_index) %>%
  dplyr::rename(par = original_nt, mut = alternative_nt, index = nt_index) %>%
  inner_join(ref_tib, by = "index") %>%
  mutate(type = paste0(par, mut))

mutations_df <- all %>%
  filter(consensus_country == "United Kingdom", total_muts > 10, flagged) %>%
  dplyr::select(original_nt, alternative_nt, nt_index) %>%
  dplyr::rename(par = original_nt, mut = alternative_nt, index = nt_index) %>%
  inner_join(ref_tib, by = "index") %>%

```

```

mutate(type = paste0(par, mut))

mutations_df

# A tibble: 681 x 7
  par    mut    index residue context_before context_after type
  <chr> <chr> <dbl> <chr>    <chr>          <chr>        <chr>
1 G     A     2635 G      T             C            GA
2 T     C     6409 T      C             G            TC
3 C     T     6633 C      G             T            CT
4 G     A    10549 G      T             G            GA
5 C     T    13326 C      A             A            CT
6 G     A    15202 G      A             T            GA
7 G     A    18145 G      T             A            GA
8 A     G    18202 A      C             T            AG
9 G     A    20433 G      T             G            GA
10 G    A     2840 G      T             C            GA
# i 671 more rows

t2_copy <- trial2 %>%
  group_by(par, mut) %>%
  arrange(Number_of_mutations) %>%
  mutate(my_rank = Number_of_mutations / sum(Number_of_mutations))

norm_copy <- normal %>%
  group_by(par, mut) %>%
  arrange(Number_of_mutations) %>%
  mutate(my_rank = Number_of_mutations / sum(Number_of_mutations))

# Create new columns par, index and mut using str_sub()
mutations_df <- tibble(mutations = mutations) %>%
  mutate(
    original_nt = str_sub(mutations, start = 1, end = 1),
    nt_index = as.numeric(str_sub(mutations, start = 2, end = -2)), # convert index to num
    alternative_nt = str_sub(mutations, start = -1, end = -1)
  )

```

```

get_ratio <- function(mutations_df2) {
  # print(mutations_df2$node_id)

  mutations_df2 <- mutations_df2 %>%
    dplyr::select(original_nt, alternative_nt, nt_index) %>%
    dplyr::rename(par = original_nt, mut = alternative_nt, index = nt_index) %>%
    inner_join(ref_tib, by = "index") %>%
    mutate(type = paste0(par, mut))

  # print(mutations_df2)

  a <- ((mutations_df2 %>% filter(type %in% c("GA", "CT"))) %>% inner_join(t2_copy))$my_ratio

  b <- ((mutations_df2 %>% filter(type %in% c("GA", "CT"))) %>% inner_join(norm_copy))$my_ratio
  return(mean(a) / mean(b))
}

mutations_df <- data_muts %>% inner_join(data_nodes %>% filter(consensus_country == "United States"))

```

Joining with `by = join_by(node_id)`

```

smallerset <- mutations_df %>%
  filter(total_muts >= 10) %>%
  group_by(node_id) %>%
  group_split()
# Initialize a data frame to store the results

results <- data.frame()

# Loop over each data frame in the list
for (i in seq_along(smallerset)) {
  # Extract the current dataframe
  subset_df <- smallerset[[i]]

  # Extract the current node_id (which is the same for all rows in subset_df)
  current_node_id <- subset_df$node_id[1]
}

```


[illegible]

```

Joining with `by` = join_by(par, mut, context_before, context_after)`
Joining with `by` = join_by(par, mut, context_before, context_after)`
Joining with `by` = join_by(par, mut, context_before, context_after)`
Joining with `by` = join_by(par, mut, context_before, context_after)`
Joining with `by` = join_by(par, mut, context_before, context_after)`
Joining with `by` = join_by(par, mut, context_before, context_after)`
Joining with `by` = join_by(par, mut, context_before, context_after)`
Joining with `by` = join_by(par, mut, context_before, context_after)`
Joining with `by` = join_by(par, mut, context_before, context_after)`
Joining with `by` = join_by(par, mut, context_before, context_after)`
Joining with `by` = join_by(par, mut, context_before, context_after)`
Joining with `by` = join_by(par, mut, context_before, context_after)`
Joining with `by` = join_by(par, mut, context_before, context_after)`
Joining with `by` = join_by(par, mut, context_before, context_after)`
Joining with `by` = join_by(par, mut, context_before, context_after)`
Joining with `by` = join_by(par, mut, context_before, context_after)`

```

results

		node_id	ratio
1	England/DHSC-CYBN7RK/2022 EPI_ISL_10441882 2022-01-16	1.9375502	
2	England/LSPA-36832B5/2022 EPI_ISL_10059724 2022-02-09	2.2344241	
3	England/LSPA-3E01AEC/2022 EPI_ISL_12666642 2022-05-06	1.5552579	
4	England/LSPA-3E3AA80/2022 EPI_ISL_13106533 2022-05-31	1.3502548	
5	England/LSPA-3E5E7F8/2022 EPI_ISL_13376289 2022-06-14	1.5513246	
6	England/LSPA-3E6B9C5/2022 EPI_ISL_13465017 2022-06-17	1.1675430	
7	England/LSPA-3E7B151/2022 EPI_ISL_13498905 2022-06-22	1.9725280	
8	England/LSPA-3F76CEB/2022 EPI_ISL_14804388 2022-08-23	1.5287211	
9	England/MILK-1D4166B/2021 EPI_ISL_4012065 2021-09-01	2.0099512	
10	England/MILK-29AF22C/2021 EPI_ISL_6298830 2021-11-07	1.8581043	
11	England/MILK-341735B/2022 EPI_ISL_9382677 2022-01-23	2.1424364	
12	England/NORT-YNNFUWW/2022 EPI_ISL_10999056 2022	2.1295458	
13	England/NORT-YNNU759/2022 EPI_ISL_11782428 2022	1.7030183	
14	England/PHEC-YYDSFCU/2022 EPI_ISL_11746075 2022-03-19	0.8779639	
15	England/PHEC-YYEKP64/2023 EPI_ISL_17245777 2023-03-05	2.1888708	
16	England/PHEC-YYEWOQD/2023 EPI_ISL_17548893 2023-03-31	1.1507352	
17	England/PHEC-YYFBK WG/2022 EPI_ISL_12582161 2022-04-22	1.3256094	
18	England/PHEC-YYFCTB0/2022 EPI_ISL_13108276 2022-05-18	1.4444910	
19	England/PHEC-YYFYAPJ/2022 EPI_ISL_12583464 2022-04-24	1.2445100	
20	England/PHEC-YYN6ZRN/2022 EPI_ISL_10766361 2022-02-16	1.5469134	
21	England/PHEC-YYRB796/2022 EPI_ISL_12043591 2022-03-23	1.8045944	
22	England/PHEC-YYRSCKX/2022 EPI_ISL_12260199 2022-04-06	1.5413067	

```

23 England/PHEP-YYG1QR9/2023|EPI_ISL_17542736|2023-04-03 1.3408797
24 England/PLYM-342FAF1/2022|EPI_ISL_9367297|2022-01-24 2.3371710
25 England/PLYM-349E389/2022|EPI_ISL_9560035|2022-01-29 1.4988714
26 England/PLYM-3B80DC1/2022|EPI_ISL_11416915|2022-03-20 1.7334451
27 England/QEUH-33ABAA3/2022|EPI_ISL_9292221|2022-01-21 2.0341530
28 England/QEUH-3DEF99D/2022|EPI_ISL_12630743|2022-05-02 2.4198509
29 England/QEUH-3EE35D0/2022|EPI_ISL_13968198|2022-07-12 2.6897023
30 Scotland/QEUH-3AC1624/2022|EPI_ISL_11282729|2022-03-14 1.5048839
31 Scotland/QEUH-3D84068/2022|EPI_ISL_12338023|2022-04-18 1.6604011
32 Scotland/SCOT-14169/2022|EPI_ISL_13694672|2022-06-11 1.8329147
33 Scotland/SCOT-25772/2023|EPI_ISL_17033184|2023-02-02 0.7023873
34 Wales/PHWC-PKP3Z4/2023|EPI_ISL_17326431|2023-03-07 1.6040964
35 Wales/QEUH-32668E85/2022|EPI_ISL_16468471|2022-12-16 2.0046688
36 node_1020547 2.1427499
37 node_1722110 2.3916273
38 node_1795876 1.9993184
39 node_1803541 1.9298921
40 node_1901667 2.9857302
41 node_2070965 2.2587313
42 node_2080633 1.7971382
43 node_2213964 0.9362092
44 node_2915795 1.5017071
45 node_2956486 1.7531576
46 node_3048254 1.1728616

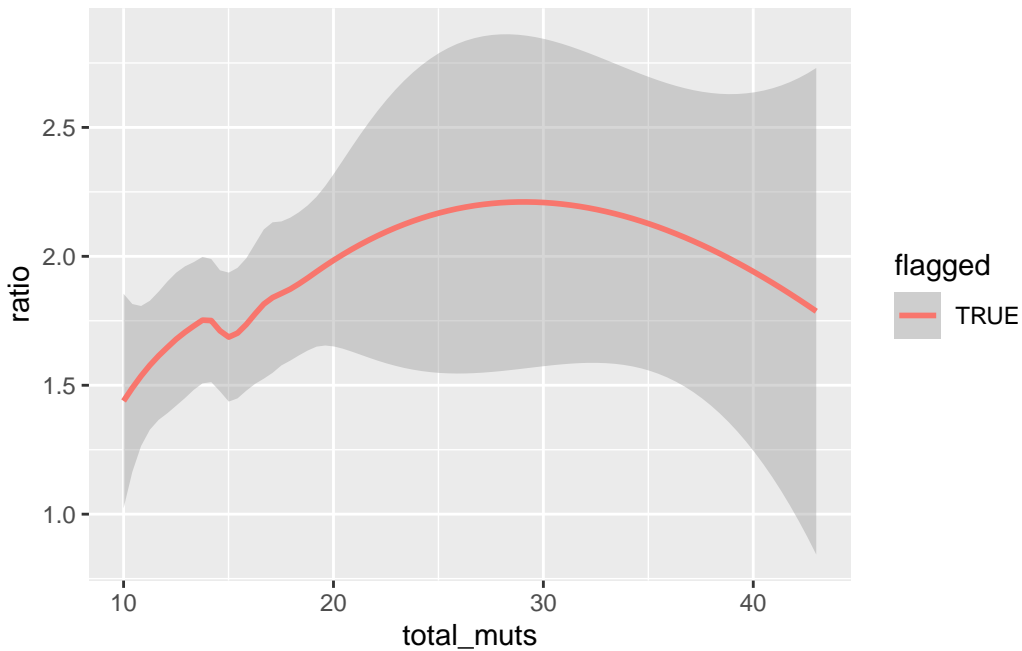
```

```
everything <- inner_join(results, data_nodes)
```

Joining with `by = join_by(node_id)`

```
ggplot(everything, aes(x = total_muts, y = ratio, color = flagged)) +
  geom_smooth()
```

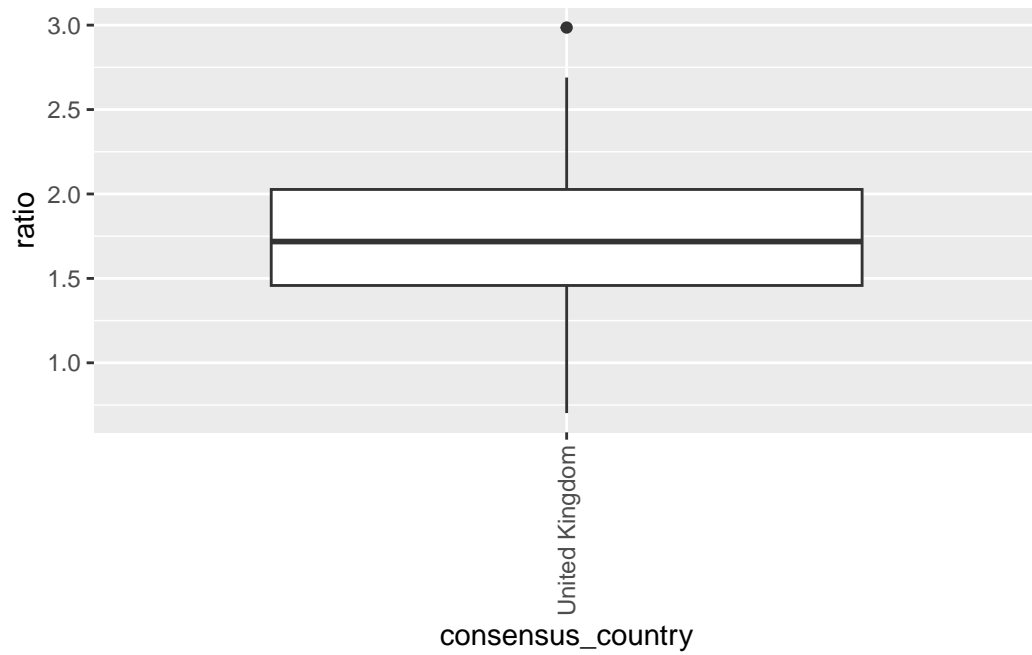
`geom_smooth()` using method = 'loess' and formula = 'y ~ x'



```
everything %>%
  group_by(flagged) %>%
  tally()
```

```
# A tibble: 1 x 2
  flagged     n
  <lgl>   <int>
1 TRUE     46
```

```
common_countries <- everything %>%
  group_by(consensus_country) %>%
  tally() %>%
  filter(n > 10)
ggplot(everything %>% filter(consensus_country %in% common_countries$consensus_country, co
  geom_boxplot() +
  theme(axis.text.x = element_text(angle = 90, vjust = 0.5, hjust = 1))
```



```
ggplot(everything, aes(x = total_muts, fill = ratio > 1.2)) +
  geom_bar(position = "fill")
```

