

Results of PangoVis

Devan Becker

2021-03-16

Load Packages and Data

```
# Packages that Art hates
library(dplyr)

##
## Attaching package: 'dplyr'
## The following objects are masked from 'package:stats':
##
##   filter, lag
## The following objects are masked from 'package:base':
##
##   intersect, setdiff, setequal, union

library(tidyr)
library(ggplot2)
library(stringr)
library(here)

## here() starts at /home/devan/OneDriveUWO/0postdoc/sup
dirich <- params$dirich

# Read in CSV files
csvs <- list.files("../data/pangolineages",
  pattern = ifelse(dirich, "_d.csv", "*.csv"),
  full.names = TRUE)

# Remove any copies
csvs <- csvs[!grepl("-1", csvs)]
# Proper names
#csvs <- paste0("../data/pangolineages/", csvs)

# Bring them into one data frame
lins <- bind_rows(lapply(csvs, read.csv))

# Taxon is encoded as _ACCESSIONNUMBER.ID, split into ACCESSIONNUMBER and ID
lins <- lins %>%
  separate(col = "taxon", sep = "\\.",
    into = c("taxon", "sample")) %>%
  mutate(taxon = str_replace(taxon, "\\_", ""))
```

```
#### Visualize the uncertainty in the base calls ----
taxons <- unique(lins$taxon)
length(taxons)
```

```
## [1] 36
```

Abstract Info

```
summs <- lins %>%
  group_by(taxon) %>%
  summarise(maxperc = mean(lineage == names(sort(table(lineage),
    decreasing = TRUE))[1]),
    uniques = length(unique(lineage)),
    minpango = min(probability),
    maxpango = max(probability),
    menpango = mean(probability),
    max = names(sort(table(lineage), decreasing = TRUE))[1])

## 'summarise()' ungrouping output (override with '.groups' argument)
print("summary info")

## [1] "summary info"
print(summs)
```

```
## # A tibble: 36 x 7
##   taxon      maxperc uniques minpango maxpango menpango max
##   <chr>      <dbl>   <int>   <dbl>   <dbl>   <dbl> <chr>
## 1 ERR4085809 0.349     78      1      1      1 A
## 2 ERR4204823 0.632     11      1      1      1 A
## 3 ERR4363387 0.921     14      1      1      1 B.1.222
## 4 ERR4364007 0.753     47      1      1      1 B.1.1.29
## 5 ERR4664555 0.965     12      1      1      1 B.1.1.253
## 6 ERR4667618 0.994      5      1      1      1 B.1.1.315
## 7 ERR4692364 0.922     26      1      1      1 B.1
## 8 ERR4693034 0.886     28      1      1      1 B.1.1.310
## 9 ERR4693061 0.955      7      1      1      1 B.23
## 10 ERR4693079 0.865     36      1      1      1 B.1.1.310
## # ... with 26 more rows
```

```
1 - mean(summs$maxperc); 1 - mean(summs$menpango)
```

```
## [1] 0.1488491
```

```
## [1] 0.02777778
```

As a Pareto plot

```
par(mfrow = c(6, 6))
for(i in 1:length(unique(lins$taxon))){
  thistab <- table(lins$lineage[lins$taxon == taxons[i]])
  called <- lins$lineage[lins$taxon == taxons[i] &
    lins$sample == 0]
```

```

print(called)

if(length(called) == 1) {
  colours <- rep(1, length(thistab))
  colours[names(thistab) == called] <- 2
} else {
  colours <- "lightgrey"
}

barplot(sort(thistab, decreasing = TRUE),
        las = 2, col = colours, border = NA)
}

```

```

## [1] "B.1"
## [1] "B.40"
## [1] "B.1.222"
## [1] "B.1.1.29"
## [1] "B.1.1.253"
## [1] "B.1.1.315"
## [1] "B.1"
## [1] "B.1.1.310"
## [1] "B.23"
## [1] "B.1.1.310"
## [1] "B.1.177.7"
## [1] "B.1.177.3"
## [1] "B.1.1.7"
## [1] "B.1.1.7"
## [1] "B.1"
## [1] "B.1.1.7"
## [1] "B.1.177"
## [1] "B.1.160"
## [1] "B.1.177.19"
## [1] "B.1.177.19"
## [1] "B.1.177"
## [1] "B.1.1.7"
## [1] "B.1.1.7"
## [1] "B.1.1.315"
## [1] "A"
## [1] "B.1.1.273"
## [1] "None"

```

[1] "B.1.1.241"

[1] "A.2.2"

[1] "B.1"

[1] "A.2.2"

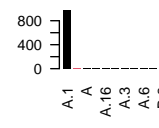
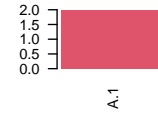
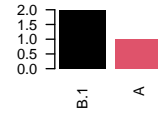
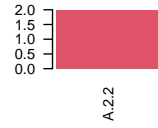
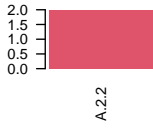
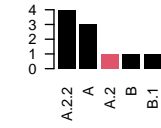
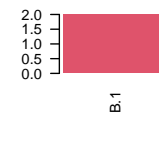
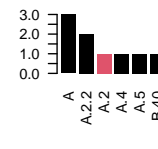
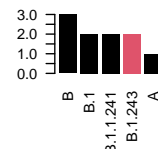
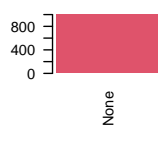
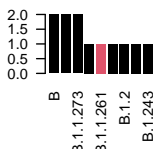
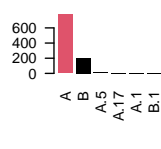
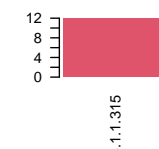
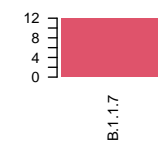
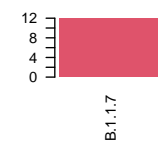
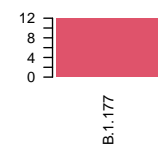
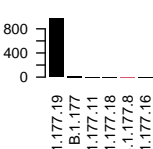
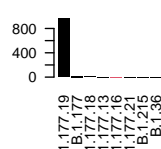
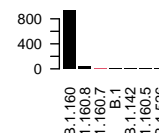
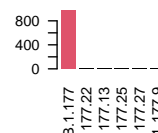
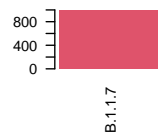
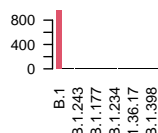
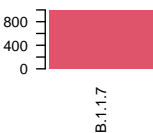
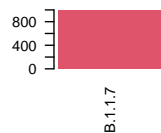
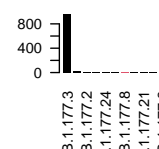
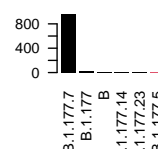
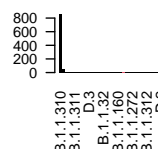
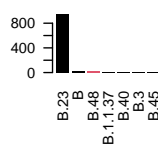
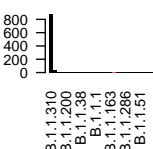
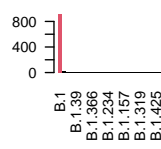
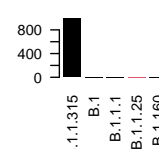
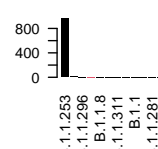
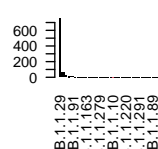
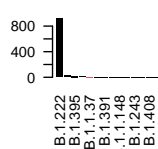
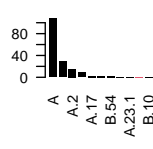
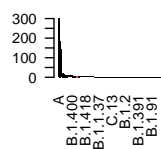
[1] "A.2.2"

[1] "A.2.2"

[1] "B.1"

[1] "A.1"

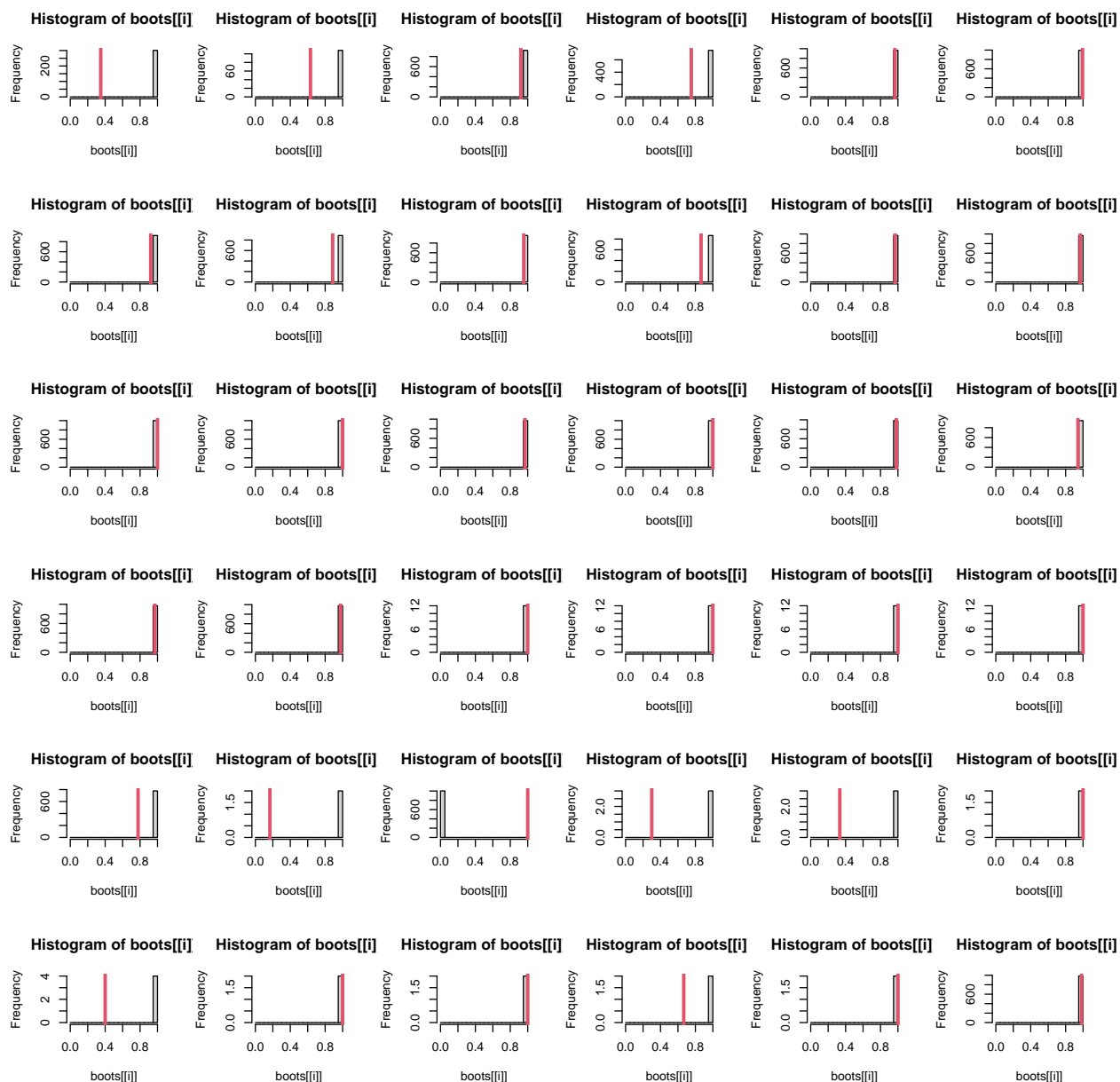
[1] "A.1"



Probability Bars

```
boots <- vector(mode = "list", length = length(taxons))
par(mfrow = c(6,6))
for(i in 1:length(taxons)){
  # Find modal lineage
  thistab <- table(lins$lineage[lins$taxon == taxons[i]])
  maxtab <- names(thistab)[which.max(thistab)]
  # Record "probability" for all sequences labelled with this lineage
  boots[[i]] <- lins$probability[lins$taxon == taxons[i] &
    lins$lineage == maxtab]

  # histogram of probabilities of that lineage
  hist(boots[[i]], breaks = seq(0, 1, 0.05),
    xlim = c(0, 1))
  # Add a line for the number of genomes that actually had that lineage
  abline(v = mean(lins$lineage[lins$taxon == taxons[i]] == maxtab),
    col = 2, lwd = 3)
}
```



Scatterplot

```
gglist <- list()
for(i in 1:length(taxons)){
  pang <- lins[lins$taxon == taxons[i], ]
  pang2 <- lapply(unique(pang$lineage), function(x) {
    data.frame(lineage = x,
               prop = mean(pang$lineage == x))
  }) %>%
  bind_rows() %>%
  right_join(pang, by = "lineage")
  #pang2
  ggplot(pang2) +
    aes(x = prop, y = probability,
```

```

    colour = lineage, label = lineage) +
  geom_point() +
  #geom_text_repel() +
  theme(legend.position = "none")

pangtab <- pang2 %>%
  group_by(prop, lineage) %>%
  summarise(y = 1, count = n(), .groups = "drop") %>%
  filter(prop > 0.025)

gglist[[i]] <- pang2 %>%
  # round to nearest 0.5
  #mutate(prop = round(prop*2, 1)/2,
  #  probability = round(probability*2, 1)/2) %>%
  group_by(prop, probability, lineage) %>%
  summarise(count = n(), .groups = "drop") %>%
  ggplot() + theme_bw() +
  aes(x = prop, y = probability, colour = lineage,
      label = count) +
  geom_text() +
  theme(legend.position = "none") +
  annotate("text", x = pangtab$prop, y = 1,
    label = pangtab$lineage,
    hjust = 0.5, vjust = -1) +
  labs(x = "Proportion of Lineage",
    y = "Bootstrap Probability",
    title = NULL) +
  scale_x_continuous(breaks = seq(0,1,0.1)) +
  scale_y_continuous(breaks = seq(0,1.1,0.1)) +
  coord_cartesian(ylim = c(0, 1.1)) +
  geom_abline(slope = 1, intercept = 0)
}
cowplot::plot_grid(plotlist = gglist)

```

