

Results Section: Public Metadata

```
library(staphopia)
library(ggplot2)
library(reshape2)
```

Aggregating Data For Public Samples

First we'll get all publicly available *S. aureus* samples.

```
ps <- get_public_samples()
```

Variation From *S. aureus* N315

In Staphopia all samples had variants (SNPs and InDels) called using *S. aureus* N315 as the reference genome. In this section we'll visualize the total number of variants each sample has. This will give us an idea of the sequenced genetic diversity with respect to N315.

Gather Variant Counts

We will use `get_variant_counts()` to get the variant counts for each sample. We will also order the counts by the total.

```
variant_counts <- get_variant_counts(ps$sample_id)
variant_counts <- variant_counts[order(total),]
```

Summary of Variant Counts

Total Variants (SNPs and InDels)

```
summary(variant_counts$total)
```

##	Min.	1st Qu.	Median	Mean	3rd Qu.	Max.
##	10	19457	23891	26505	37343	146962

SNPs

```
summary(variant_counts$snp_count)
```

##	Min.	1st Qu.	Median	Mean	3rd Qu.	Max.
##	6	18712	23162	25560	36062	141893

InDels

```
summary(variant_counts$indel_count)
```

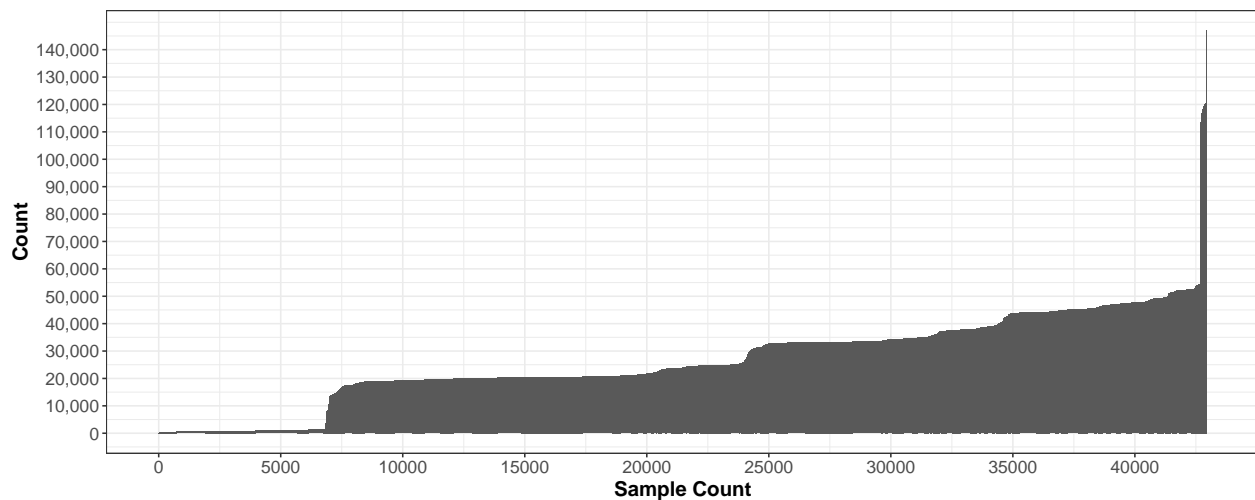
##	Min.	1st Qu.	Median	Mean	3rd Qu.	Max.
##	1.0	709.0	901.0	944.4	1293.0	5125.0

Visualizing Variant Counts

Total Variants (SNPs and InDels)

```
p <- ggplot(data=variant_counts, aes(x=seq(1,nrow(variant_counts)), y=total)) +  
  xlab("Sample Count") +  
  ylab("Count") +  
  geom_bar(stat='identity') +  
  scale_x_continuous(breaks = seq(0, nrow(variant_counts), by = 5000)) +  
  scale_y_continuous(breaks = seq(0, max(variant_counts$total), by=10000),  
    labels = scales::comma) +  
  theme_bw() +  
  theme(axis.text=element_text(size=12),  
    axis.title=element_text(size=14,face="bold"))
```

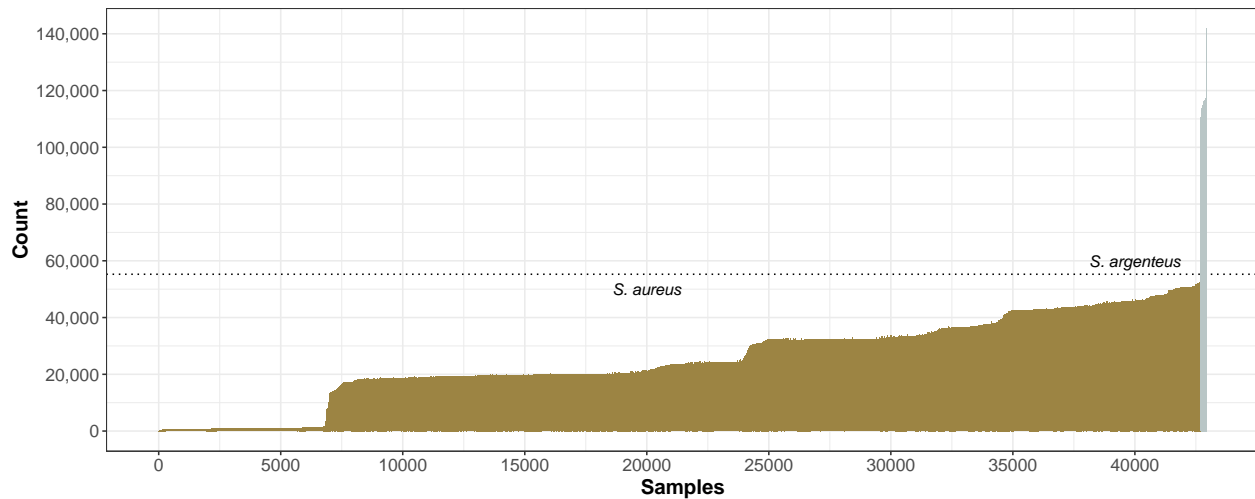
p



SNPs Only

```
cutoff <- max(variant_counts[variant_counts$snp_count < 60000,]$snp_count)  
variant_counts$fill <- ifelse(variant_counts$snp_count > cutoff, TRUE, FALSE)  
p <- ggplot(data=variant_counts, aes(x=seq(1,nrow(variant_counts)), y=snp_count,  
  fill=fill)) +  
  xlab("Samples") +  
  ylab("Count") +  
  geom_hline(yintercept = cutoff, linetype="dotted") +  
  geom_bar(stat='identity') +  
  annotate("text", x = 40000, y = 60000, label = "S. argenteus", fontface=3) +  
  annotate("text", x = 20000, y = 50000, label = "S. aureus", fontface=3) +  
  scale_x_continuous(breaks = seq(0, nrow(variant_counts), by = 5000)) +  
  scale_y_continuous(breaks = seq(0, max(variant_counts$snp_count), by=20000),  
    labels = scales::comma) +  
  scale_fill_manual(values=c("#9C8443", "#B9C6C6")) +  
  theme_bw() +  
  theme(axis.text=element_text(size=12),  
    axis.title=element_text(size=14,face="bold"),  
    legend.position="none")
```

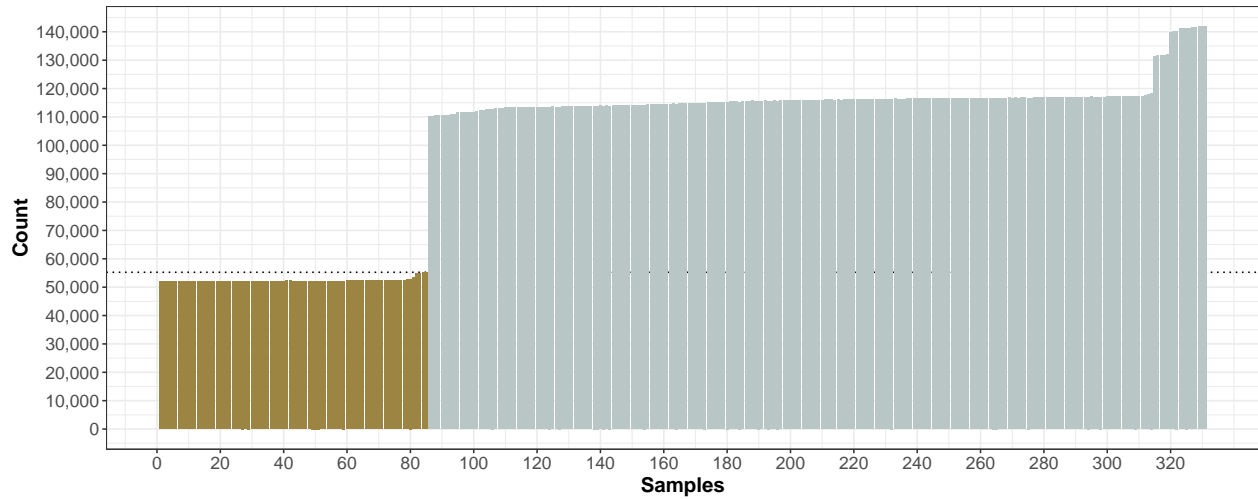
p



```
# Output plot to PDF and PNG
staphopia::write_plot(p, paste0(getwd(), '/../figures/figure-05-snp-accumulation'))
```

```
cutoff <- max(variant_counts[variant_counts$snp_count < 60000,]$snp_count)
variant_counts$fill <- ifelse(variant_counts$snp_count > cutoff, TRUE, FALSE)
p <- ggplot(data=variant_counts[variant_counts$snp_count > 52000,], aes(
  x=seq(1,nrow(variant_counts[variant_counts$snp_count > 52000,])),
  y=snp_count,
  fill=fill)
) +
  xlab("Samples") +
  ylab("Count") +
  geom_hline(yintercept = cutoff, linetype="dotted") +
  geom_bar(stat='identity') +
  scale_x_continuous(breaks = seq(
    0, nrow(variant_counts[variant_counts$snp_count > 52000,]), by = 20)
) +
  scale_y_continuous(breaks = seq(0, max(variant_counts$snp_count), by=10000),
    labels = scales::comma) +
  scale_fill_manual(values=c("#9C8443", "#B9C6C6")) +
  theme_bw() +
  theme(axis.text=element_text(size=12),
    axis.title=element_text(size=14,face="bold"),
    legend.position="none")
```

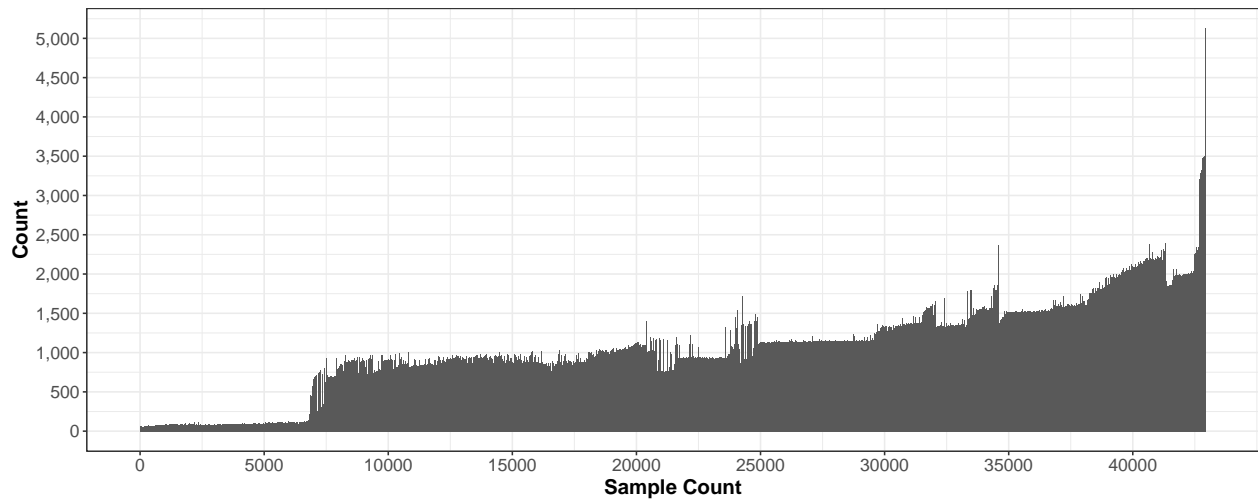
p



InDels Only

```
p <- ggplot(data=variant_counts, aes(x=seq(1,nrow(variant_counts)), y=indel_count)) +
  xlab("Sample Count") +
  ylab("Count") +
  geom_bar(stat='identity') +
  scale_x_continuous(breaks = seq(0, nrow(variant_counts), by = 5000)) +
  scale_y_continuous(breaks = seq(0, max(variant_counts$indel_count), by=500),
    labels = scales::comma) +
  theme_bw() +
  theme(axis.text=element_text(size=12),
    axis.title=element_text(size=14,face="bold"))
```

p



Non-Redundant Diversity Dataset

Below are the set of commands used to generate the NRD dataset. These were executed on a local server using Staphopia functions not available through the API due to execution times.

```

# Create NRD dataset (non-redundant-set.txt)
~/staphopia-web/manage.py create_nonredundant_set nrd-dataset --published --rank 3 \
--singletons

# Extract all genes without InDels and validate SNPs via 31-mers
NRD="nrd-annotation-summary.txt"
STAPHOPIA="/home/rpetit/cgc-staphopia/staphopia"
mkdir extract-genome
awk '{print $8}' non-redundant-set.txt | \
grep -v sample_id | \
xargs -I {} ~/staphopia-web/manage.py extract_genome {} $NRD $STAPHOPIA

# Identify genes with less than 3 samples with missing 31-mers
cat extract-genome/*-annotation-details.txt | \
awk '{if ($3=="False"){print $2}}' | \
sort | \
uniq -c | \
awk '{if ($1 >= 377){print $2}}' | \
sort -n > validated-genes.txt

# Extract validated genes (extract-core.fasta, extract-core.phylip)
~/staphopia-web/manage.py extract_core validated-genes.txt extract-genome/

# Build Phylogeny
# Start Tree
iqtree -s extract-core.phylip -m GTR -nt AUTO -fast -pre start-tree | \
tee iqtree-start.stdout.txt

# Identify Recombination
ClonalFrameML start-tree.treefile extract-core.fasta clonalframe -emsim 100 | \
tee clonalframe.stdout.txt

# Remove Recombination
maskrc-svg.py clonalframe --aln extract-core.fasta \
--out extract-core-cfmasked.fasta --symbol '-' | \
tee maskrc-svg.stdout.txt

# Final Tree
iqtree -s extract-core-cfmasked.fasta -m GTR -nt 11 -pre final-tree \
-bb 1000 -alrt 1000 -wbt -wbtl -alninfo | \
tee iqtree-final.stdout.txt

```

Session Info

```
sessionInfo()
```

```

## R version 3.4.3 (2017-11-30)
## Platform: x86_64-pc-linux-gnu (64-bit)
## Running under: Ubuntu 16.04.2 LTS
##
## Matrix products: default

```

```

## BLAS: /usr/lib/libblas/libblas.so.3.6.0
## LAPACK: /usr/lib/lapack/liblapack.so.3.6.0
##
## locale:
## [1] LC_CTYPE=en_US.UTF-8      LC_NUMERIC=C
## [3] LC_TIME=en_US.UTF-8      LC_COLLATE=en_US.UTF-8
## [5] LC_MONETARY=en_US.UTF-8  LC_MESSAGES=en_US.UTF-8
## [7] LC_PAPER=en_US.UTF-8     LC_NAME=C
## [9] LC_ADDRESS=C             LC_TELEPHONE=C
## [11] LC_MEASUREMENT=en_US.UTF-8 LC_IDENTIFICATION=C
##
## attached base packages:
## [1] stats      graphics  grDevices  utils      datasets  methods    base
##
## other attached packages:
## [1] reshape2_1.4.3  ggplot2_2.2.1  staphopia_0.1.9
##
## loaded via a namespace (and not attached):
## [1] Rcpp_0.12.15      knitr_1.20       magrittr_1.5
## [4] munsell_0.4.3     colorspace_1.3-2 R6_2.2.2
## [7] rlang_0.1.6       stringr_1.2.0    httr_1.3.1
## [10] plyr_1.8.4        tools_3.4.3      grid_3.4.3
## [13] data.table_1.10.4-3 gtable_0.2.0     htmltools_0.3.6
## [16] yaml_2.1.18       lazyeval_0.2.1   rprojroot_1.3-2
## [19] digest_0.6.15     tibble_1.4.2     curl_3.1
## [22] evaluate_0.10.1   rmarkdown_1.9    stringi_1.1.6
## [25] compiler_3.4.3    pillar_1.1.0     scales_0.5.0
## [28] backports_1.1.2   jsonlite_1.5

```