Results Section: Public Genetic Diversity

```r
library(staphopia)
library(ggplot2)
library(reshape2)
library(scales)
USE_DEV = TRUE
```

In this section we will look into genetic diversity that has been sequenced in *Staphylococcus aureus*. In order to do so, we'll use variant counts, cgMLST and MLST as measures of diversity.

Aggregating Data For Public Samples

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

```r
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 genitic 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.

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

Summary of Variant Counts

Total Variants (SNPs and InDels)

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

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

SNPs

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

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

InDels

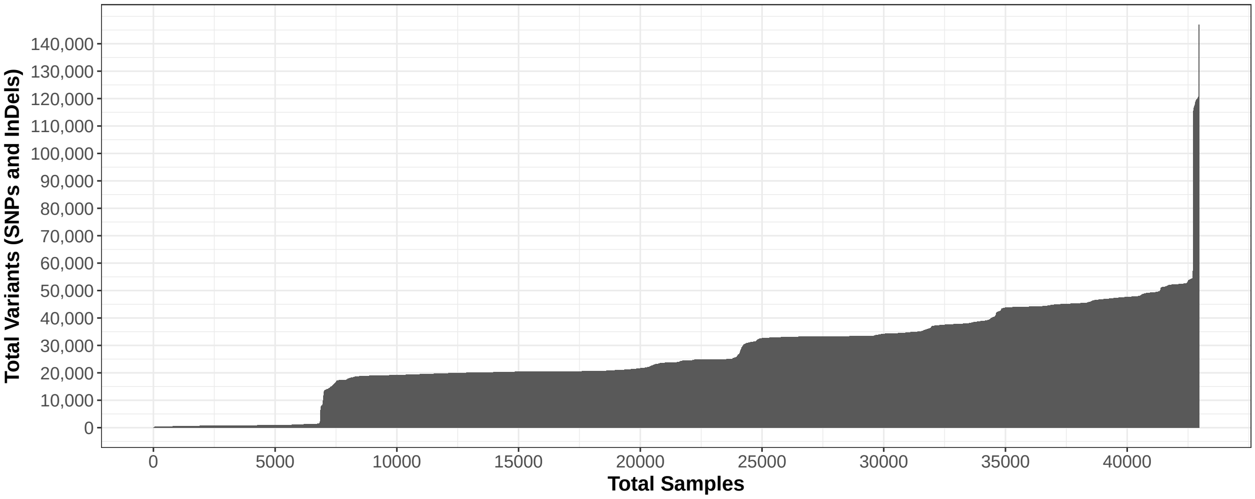
```
summary(variant_counts$indel_count)
```

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

Visualizing Variant Counts

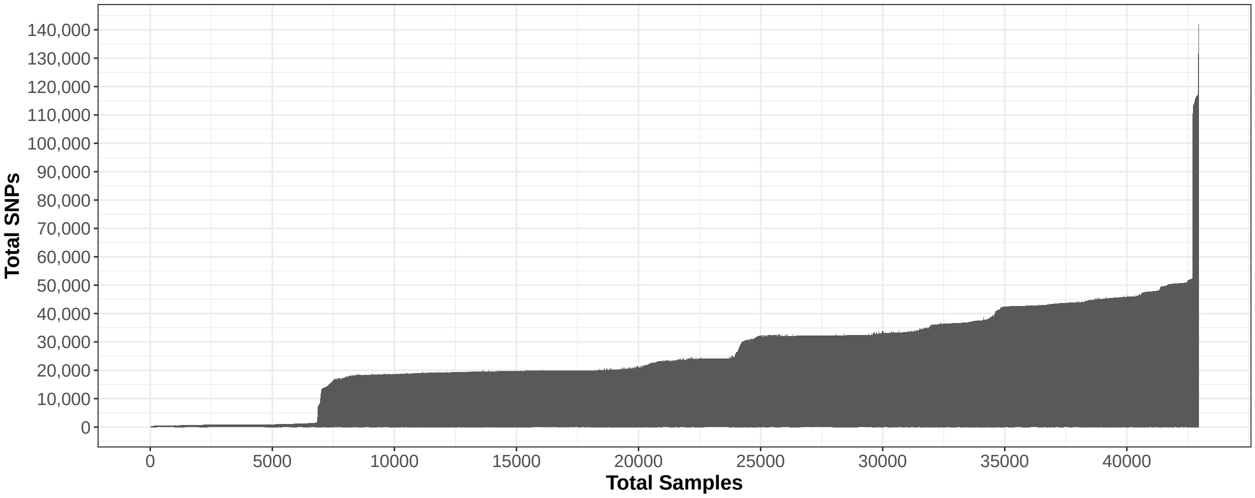
Total Variants (SNPs and InDels)

```
p <- ggplot(data=variant_counts, aes(x=seq(1,nrow(variant_counts)), y=total)) +
    xlab("Total Samples") +
    ylab("Total Variants (SNPs and InDels)") +
    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
```



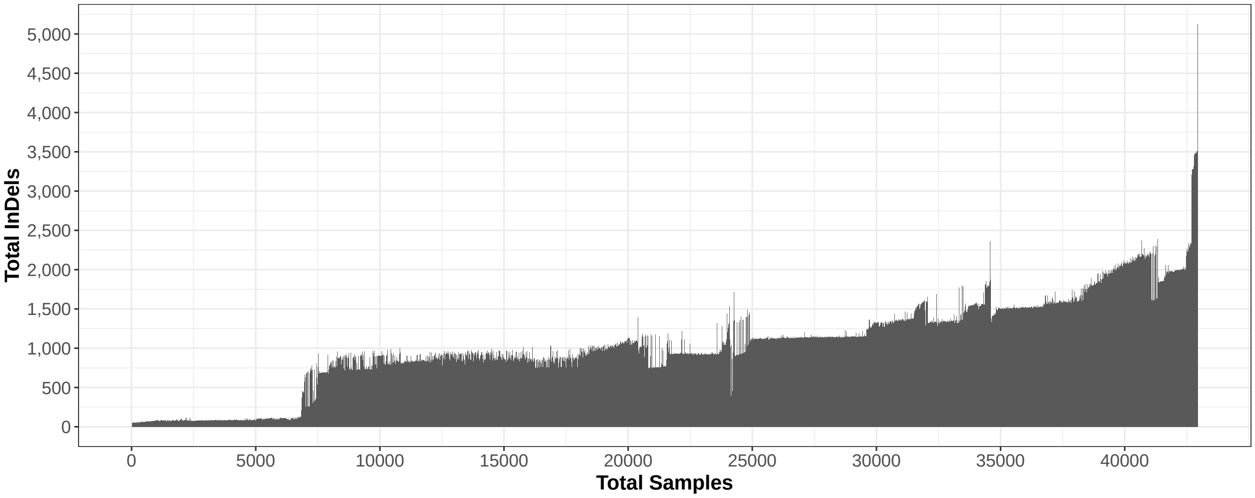
SNPs Only

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



InDels Only

```
p <- ggplot(data=variant_counts, aes(x=seq(1,nrow(variant_counts)), y=indel_count)) +
    xlab("Total Samples") +
    ylab("Total InDels") +
    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
```



MLST

Next we will will use the MLST information has a measure of genitic diversity. In this case we are interested in the total number of unique sequence types sequenced. We'll use *get_st_by_year()* to get some basic stats about how many STs have been sequenced. We will also use *get_top_sequence_types()* to get each ST represented in the database and the total number of samples with each ST. (*Note: 5000 is just an arbitrarly large number to retreive all STs*)

```r
sequence_types <- get_st_by_year()
top_st <- get_top_sequence_types(5000)
colnames(sequence_types)
```

```
##  [1] "year"                        "unique"                     
##  [3] "novel"                       "assigned"                   
##  [5] "assigned_agree"              "assigned_disagree"          
##  [7] "unassigned"                  "unassigned_agree"           
##  [9] "unassigned_disagree"         "predicted_novel"            
## [11] "all"                         "partial"                    
## [13] "ariba_blast"                 "mentalist_blast"            
## [15] "mentalist_ariba"             "single"                     
## [17] "ariba"                       "mentalist"                  
## [19] "blast"                       "count"                      
## [21] "overall_novel"               "overall_assigned"           
## [23] "overall_assigned_agree"      "overall_assigned_disagree"  
## [25] "overall_unassigned"          "overall_unassigned_agree"   
## [27] "overall_unassigned_disagree" "overall_predicted_novel"    
## [29] "overall_all"                 "overall_partial"            
## [31] "overall_ariba_blast"         "overall_mentalist_blast"    
## [33] "overall_mentalist_ariba"     "overall_single"             
## [35] "overall_ariba"               "overall_mentalist"          
## [37] "overall_blast"               "overall"
```

This gives us 38 columns for each year. These columns are:

1. year: The year.
2. unique: The Number of unique STs for a given year.
3. novel: Number of STs not sequenced previously.
4. assigned: Samples which a ST was determined.
5. assigned_agree: Samples in which each program that called an ST agreed in ST.
6. assigned_disagree: Samples in which programs did not each call the same ST.
7. unassigned: Samples which a ST was not determined.
8. unassigned_agree: Each program was unable to assign an ST.
9. unassigned_disagree: Samples in which no ST was determined, but each program does not agree
10. predicted_novel: Samples with a match to each Loci, but allele pattern does not exist.
11. all: Samples with an ST determined with agreement between each program.
12. partial: Samples with an ST determined with agreement between two programs. 13: ariba_blast: Samples with an ST determined with agreement between Ariba and BLAST.
13. mentalist_blast: Samples with an ST determined with agreement between MentaLiST and BLAST.
14. mentalist_ariba: Samples with an ST determined with agreement between MentaLiST and Ariba.
15. single: Samples with an ST determined by only a single program.
16. ariba: Samples with an ST determined by only Ariba.
17. mentalist: Samples with an ST determined by only MentaLiST.
18. blast: Samples with an ST determined by only BLAST.
19. count: Total number of samples in a given year. 21-38: overall_X: The cumulative totals of previous years for column x

Summary of MLST Diversity

Assignment Breakdown

```r
t(sequence_types[sequence_types$year == max(sequence_types$year),21:38])
```

```
##                                  8
## overall_novel                  1098
## overall_assigned              42337
## overall_assigned_agree        42243
## overall_assigned_disagree        94
## overall_unassigned              612
## overall_unassigned_agree        612
## overall_unassigned_disagree       0
## overall_predicted_novel         306
## overall_all                   41226
## overall_partial                 922
## overall_ariba_blast              81
## overall_mentalist_blast         669
## overall_mentalist_ariba         172
## overall_single                  189
## overall_ariba                    29
## overall_mentalist               111
## overall_blast                    49
## overall                       42949
```

Top STs

```
top_st[1:10,]
```

```
##     st count percent overall
## 1   22  7189   16.74   16.74
## 2    8  6184   14.40   31.14
## 3    5  4664   10.86   42.00
## 4  239  3123    7.27   49.27
## 5  398  2326    5.42   54.68
## 6   30  1872    4.36   59.04
## 7   45  1663    3.87   62.91
## 8   15  1172    2.73   65.64
## 9   36   857    2.00   67.64
## 10 105   857    2.00   69.63
```

This gives us 4 columns for each ST, in descending order based on the *count* column. In other words the most represented STs are seen first. These columns are:

1. st: The sequence type.
2. count: The number of samples with given ST.
3. percent: The percent of samples represented by given ST.
4. overall: The percent of samples represented by given ST and previous STs.

How many unique STs represented?

```
nrow(top_st[top_st$st > 0,])
```

```
## [1] 1098
```

How many STs represented by a single sample?

```
nrow(top_st[top_st$count == 1, ])
```

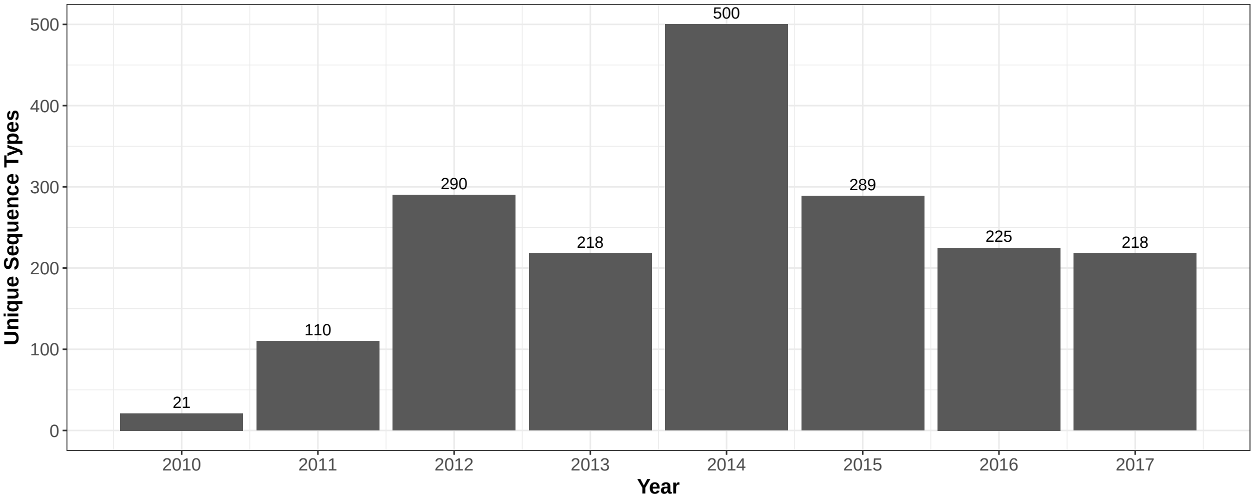
```
## [1] 588
```

Visualizing MLST Diversity

The following sections will be plots to visualize relationships in the data.

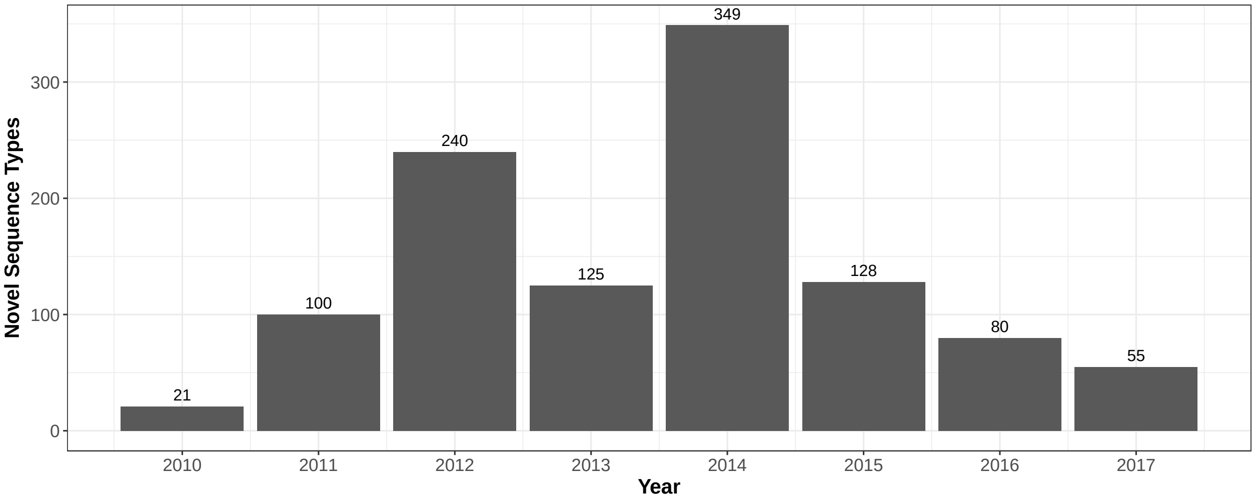
Unique Sequence Types By Year

```
p <- ggplot(data=sequence_types, aes(x=year, y=unique)) +
    xlab("Year") +
    ylab("Unique Sequence Types") +
    geom_bar(stat='identity') +
    geom_text(aes(label=unique), vjust = -0.5) +
    scale_x_continuous(breaks = round(seq(min(sequence_types$year), max(sequence_types$year), by = 1),1
    theme_bw() +
    theme(axis.text=element_text(size=12),
          axis.title=element_text(size=14,face="bold"))
p
```



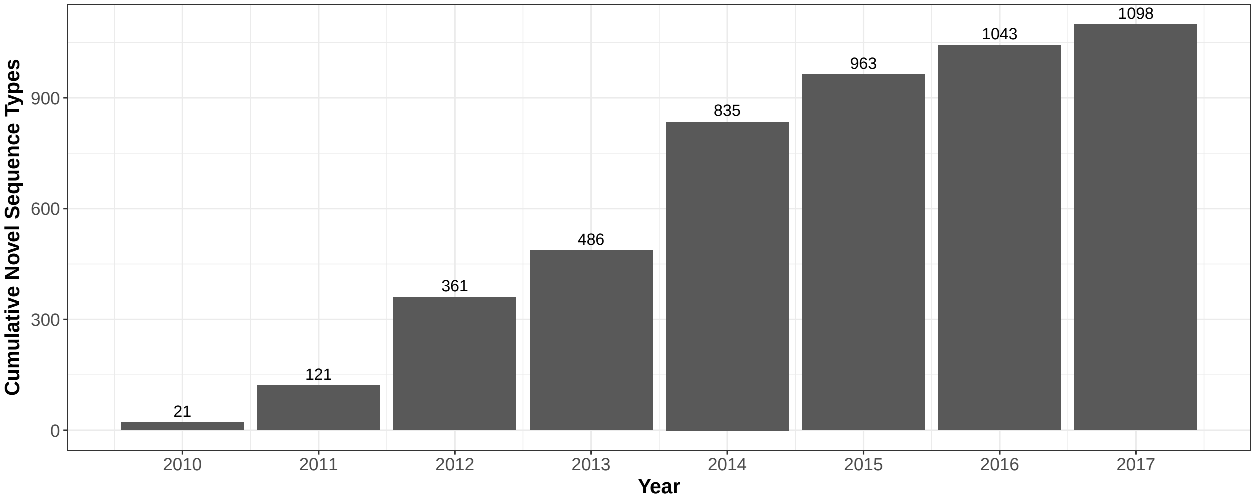
Novel Sequence Types By Year

```
p <- ggplot(data=sequence_types, aes(x=year, y=novel)) +
    xlab("Year") +
    ylab("Novel Sequence Types") +
    geom_bar(stat='identity') +
    geom_text(aes(label=novel), vjust = -0.5) +
    scale_x_continuous(breaks = round(seq(min(sequence_types$year), max(sequence_types$year), by = 1),1
    theme_bw() +
    theme(axis.text=element_text(size=12),
          axis.title=element_text(size=14,face="bold"))
p
```



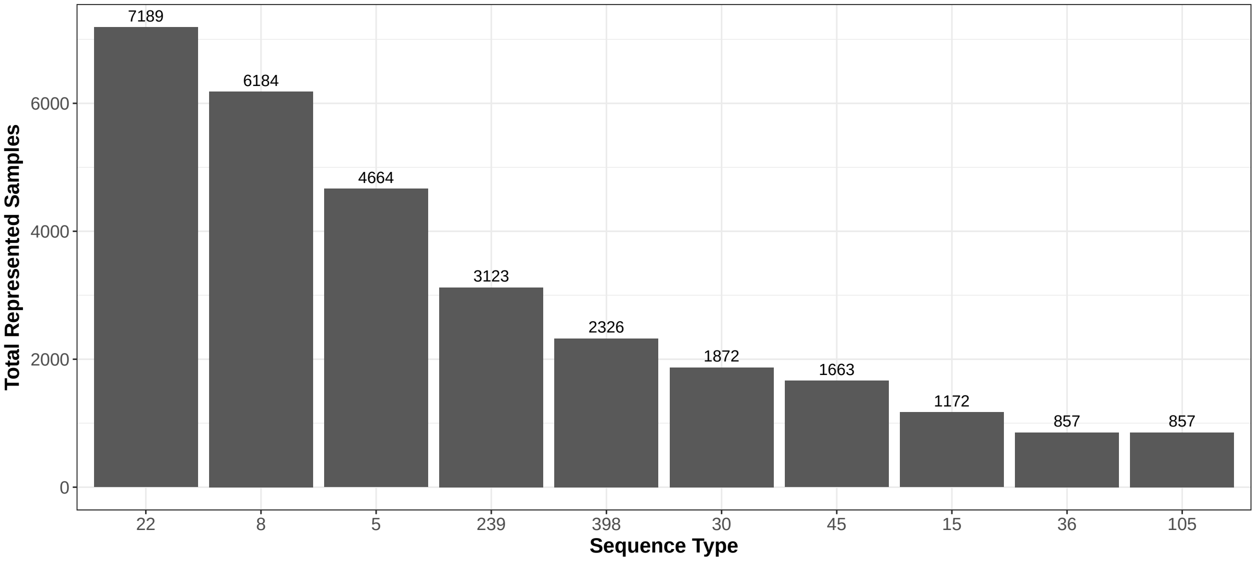
Overall Novel Sequence Types By Year

```
p <- ggplot(data=sequence_types, aes(x=year, y=overall_novel)) +
    xlab("Year") +
    ylab("Cumulative Novel Sequence Types") +
    geom_bar(stat='identity') +
    geom_text(aes(label=overall_novel), vjust = -0.5) +
    scale_x_continuous(breaks = round(seq(min(sequence_types$year), max(sequence_types$year), by = 1),1
    theme_bw() +
    theme(axis.text=element_text(size=12),
          axis.title=element_text(size=14,face="bold"))
p
```



Top 10 Sequence Types

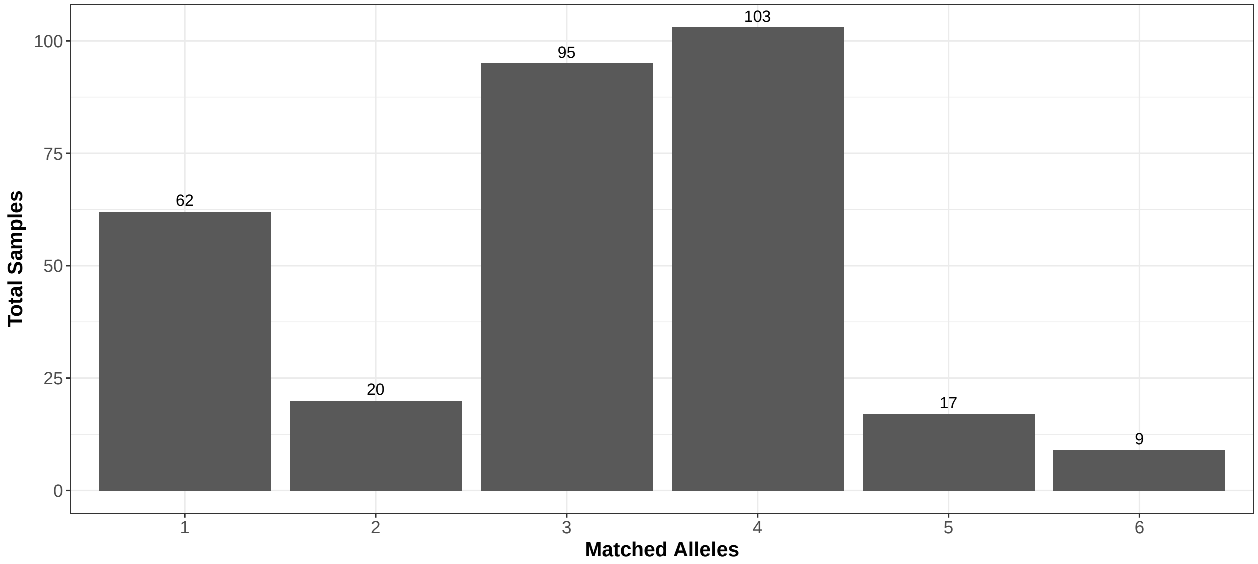
```
p <- ggplot(data=top_st[1:10,], aes(x=reorder(st, -count), y=count)) +
    xlab("Sequence Type") +
    ylab("Total Represented Samples") +
    geom_bar(stat="identity") +
    geom_text(aes(label=count), vjust = -0.5) +
    theme_bw() +
    theme(axis.text=element_text(size=12),
          axis.title=element_text(size=14,face="bold"))
p
```



Total Allele Matches For Unassigned Samples

```
allele_matches <- get_mlst_allele_matches(ps[ps$st == 0,]$sample_id)
df <- as.data.frame(table(allele_matches[allele_matches$matches < 7,]$matches))
colnames(df) <- c("matches", "count")

p <- ggplot(data=df, aes(x=matches, y=count)) +
    xlab("Matched Alleles") +
    ylab("Total Samples") +
    geom_bar(stat="identity") +
    geom_text(aes(label=count), vjust = -0.5) +
    theme_bw() +
    theme(axis.text=element_text(size=12),
          axis.title=element_text(size=14,face="bold"))
p
```



cgMLST Patterns

Finally, we'll look at cgMLST as a measure of genetic diversity. We will use the *get_cgmlst()* function to get the cgMLST results for each Sample. This function might take a little while to retrieve all teh results.

```
cgmlst <- get_publis_cgmlst_patterns()
cgmlst$percent <- cgmlst$count / sum(cgmlst$total_samples)
cgmlst
```

```
##    samples_in_pattern count total_samples      percent
## 1                 170     1           170 2.328343e-05
## 2                 133     1           133 2.328343e-05
## 3                  99     1            99 2.328343e-05
## 4                  83     1            83 2.328343e-05
## 5                  79     1            79 2.328343e-05
## 6                  61     1            61 2.328343e-05
## 7                  59     1            59 2.328343e-05
## 8                  52     1            52 2.328343e-05
## 9                  39     1            39 2.328343e-05
## 10                 36     1            36 2.328343e-05
## 11                 34     1            34 2.328343e-05
## 12                 33     1            33 2.328343e-05
## 13                 30     3            90 6.985029e-05
## 14                 29     1            29 2.328343e-05
## 15                 28     1            28 2.328343e-05
## 16                 26     1            26 2.328343e-05
## 17                 24     3            72 6.985029e-05
## 18                 22     1            22 2.328343e-05
## 19                 21     4            84 9.313372e-05
## 20                 19     2            38 4.656686e-05
## 21                 18     2            36 4.656686e-05
## 22                 15     3            45 6.985029e-05
## 23                 14     4            56 9.313372e-05
## 24                 13     3            39 6.985029e-05
## 25                 12     4            48 9.313372e-05
## 26                 11     8            88 1.862674e-04
## 27                 10     5            50 1.164171e-04
## 28                  9     5            45 1.164171e-04
## 29                  8    16           128 3.725349e-04
## 30                  7    28           196 6.519360e-04
## 31                  6    25           150 5.820857e-04
## 32                  5    47           235 1.094321e-03
## 33                  4    86           344 2.002375e-03
## 34                  3   223           669 5.192205e-03
## 35                  2  1363          2726 3.173531e-02
## 36                  1 36827         36827 8.574588e-01
```

This gives us two columns:

1. samples_in_pattern: The number of samples with a given cgMLST pattern.
2. count: The number patterns with a given number of samples.
3. total_samples: Number of samples represented by a row (samples_in_pattern * count)
4. percent: Percent of samples represented

For example, if samples_in_pattern is 100 and the count is 2. That means there are **2** (count=2) cgMLST patterns that are shared by **100 samples** (samples_in_count=100) each, representing a total of **200 samples** (count * samples_in_count).

Total Number of Distinct cgMLST Patterns

```r
sum(cgmlst$count)
```

```
## [1] 38677
```

How many shared cgMLST patterns?

```r
sum(cgmlst[cgmlst$samples_in_pattern > 1, ]$count)
```

```
## [1] 1850
```

How many samples share a cgMLST pattern?

```r
sum(cgmlst[cgmlst$samples_in_pattern > 1, ]$total_samples)
```

```
## [1] 6122
```

How many samples have a unique cgMLST pattern?

```r
cgmlst$percent <- cgmlst$count / sum(cgmlst$total_samples)
cgmlst[cgmlst$samples_in_pattern == 1, ]
```

```
##    samples_in_pattern count total_samples   percent
## 36                  1 36827         36827 0.8574588
```