all_strain_genome_stats

2023-07-12

Assembly stats for 13 Emiliania huxleyi genomes

```
genstats <- list()
gc <- list()
contam_stats <- list()
strains <- as.character(read.csv('data/strains.csv',header=FALSE))
for (strain in strains){
  genstats[[strain]] <- read.table(paste0("data/2023-genome-stats/stats_no_pilon_contaminated/"
                                          ,strain,".genstats.txt"),sep="\t",header=F)
  colnames(genstats[[strain]]) <- c(    'Contig_name','Avg_fold',   'Length',   'Ref_GC',
                                    'Covered_percent', 'Covered_bases','Plus_reads',
                                    'Minus_reads', 'Read_GC',  'Median_fold', 'Std_Dev')

  gc[[strain]] <- read.table(paste0("data/2023-genome-stats/stats_no_pilon_contaminated/"
                                    ,strain,".gcscaffold.txt"),sep="\t",header=F)
  colnames(gc[[strain]]) <- c('Contig_name',    'Length',   'A',    'C',    'G',    'T',    'N',
                              'IUPAC','Other',  'GC')

    #genstats[[strain]]$Contig_name <- str_split(genstats[[strain]]$Contig_name,regex("_1$"),simplify =

  contam_stats[[strain]] <- inner_join(genstats[[strain]],gc[[strain]],by="Contig_name")


}
```

Read in stats for pre-decontamination assemblies

```
genstats <- list()
gc <- list()
dup_stats <- list()
for (strain in strains){
  genstats[[strain]] <- read.table(paste0("data/2023-genome-stats/stats_after_pilon_round_2_decontam/"
                                          ,strain,".genstats.txt"),sep="\t",header=F)
  colnames(genstats[[strain]]) <- c(    'Contig_name','Avg_fold',   'Length',   'Ref_GC',
                                    'Covered_percent', 'Covered_bases','Plus_reads',
                                    'Minus_reads', 'Read_GC',  'Median_fold', 'Std_Dev')

  gc[[strain]] <- read.table(paste0("data/2023-genome-stats/stats_after_pilon_round_2_decontam/"
                                    ,strain,".gcscaffold.txt"),sep="\t",header=F)
```

```
  colnames(gc[[strain]]) <- c('Contig_name',    'Length',   'A',    'C',    'G',    'T',    'N',
                             'IUPAC','Other',  'GC')

    #genstats[[strain]]$Contig_name <- str_split(genstats[[strain]]$Contig_name,regex("_1$"),simplify =

  dup_stats[[strain]] <- inner_join(genstats[[strain]],gc[[strain]],by="Contig_name")


}
```

Read in assembly stats data for non-dedupped assemblies

```
genstats <- list()
gc <- list()
rmdup_stats <- list()
for (strain in strains){
  genstats[[strain]] <- read.table(paste0("data/2023-genome-stats/stats_after_pilon_round_2_decontam_rm
                                          ,strain,".genstats.txt"),sep="\t",header=F)

  colnames(genstats[[strain]]) <- c(    'Contig_name','Avg_fold',   'Length',   'Ref_GC',
                                    'Covered_percent', 'Covered_bases','Plus_reads',
                                    'Minus_reads', 'Read_GC',  'Median_fold', 'Std_Dev')

    gc[[strain]] <- read.table(paste0("data/2023-genome-stats/stats_after_pilon_round_2_decontam_rmdup/
                                     ,strain,".gcscaffold.txt"),sep="\t",header=F)
  colnames(gc[[strain]]) <- c('Contig_name',    'Length',   'A',    'C',    'G',    'T',    'N',
                             'IUPAC','Other',  'GC')

    #genstats[[strain]]$Contig_name <- str_split(genstats[[strain]]$Contig_name,regex("_1$"),simplify =

  rmdup_stats[[strain]] <- inner_join(genstats[[strain]],gc[[strain]],by="Contig_name")


}
```

Read in stats for dedup assemblies

```
calculate_stats <- function(data,prefix){
  n50s <- c()
lengths <- c()
contig_counts <- c()
mins <- c()
maxs <- c()
l50s <- c()
for (strain in strains){
contig_lengths <- data[[strain]]$Length.x
contig_counts <- c(contig_counts,length(contig_lengths))
```

```
total_assembled_length=sum(contig_lengths)
contig_lengths <- sort(contig_lengths,decreasing=TRUE)

sum <- 0
count <- 0
for (length in contig_lengths){
  sum <- sum+length
  count <- count+1
  if (sum>=total_assembled_length/2){
    l50s <- c(l50s,count)
    n50s <- c(n50s,as.numeric(length))
    lengths <- c(lengths,total_assembled_length)
    maxs <- c(maxs,contig_lengths[1])
    mins <- c(mins,contig_lengths[length(contig_lengths)])
    break
  }
}

}
stats <- data.frame(n50s,l50s,lengths,contig_counts,mins,maxs)
rownames(stats) <- strains
colnames(stats) <- c(paste0(prefix,'_N50'),paste0(prefix,'_L50'),  paste0(prefix,'_Total assembled leng
                           paste0(prefix,'_Min contig length'),paste0(prefix,'_Max contig length'))
return(stats)
}

global_stats <- calculate_stats(dup_stats,"dup")
```

Calculate globals assembly stats including N50 and total assembled length

```
global_stats <- cbind(global_stats,calculate_stats(rmdup_stats,"rmdup"))
```

Add in rmdup stats

Add in contam stats

```
global_stats <- cbind(global_stats,calculate_stats(contam_stats,"contam"))
```

```
name_translation <- read.table('data/genomescope/illumina-run-conversions.txt',sep='')
temp <- name_translation$V2
names(temp) <- name_translation$V1
name_translation <- temp
```

```
global_stats$genome_haploid_length <- seq(1,nrow(global_stats))
global_stats$genome_unique_length <- seq(1,nrow(global_stats))
for (folder in list.dirs('data/genomescope/')){
  if (grepl('HA',folder)){
  key <- str_split(folder,'_',simplify=TRUE)[,1]
  key <- str_split(key,'/',simplify=TRUE)[,4]
  if (sum(grepl(name_translation[key],rownames(global_stats)))==1){
  temp <- read.csv(paste0(folder,"/summary.txt_fixed.csv"))
  global_stats[name_translation[key],'genome_haploid_length'] <-
    mean(temp[2,'min'],temp[2,'max'])
  global_stats[name_translation[key],'genome_unique_length'] <-
    mean(temp[4,'min'],temp[4,'max'])
  }
  }
}
#print(temp)
print(global_stats)
```

Add in estimated genome size stats, calculated using Genomescope

```
##          dup_N50 dup_L50 dup_Total assembled length dup_Contig count
## CCMP371   167293     442                  299709025             5373
## CCMP375    82247     791                  247806427             8312
## CCMP377    70514    1125                  297357770             9460
## CCMP1280  104319     573                  241936265             7168
## RCC874   3015506      16                  152968590              396
## RCC914     63139     769                  210245484             9395
## RCC1222    55476     873                  179814701            10077
## RCC1239   364457     140                  171886557             2195
## RCC1256    43379    1463                  236745271            10971
## RCC3492    70125     586                  211559349             9734
## RCC3963   146371     484                  281844499             5463
## RCC6071   511805     108                  200137934             1929
## RCC6856    78893    1210                  333916182             9838
##          dup_Min contig length dup_Max contig length rmdup_N50 rmdup_L50
## CCMP371                    671                1476961    175654       255
## CCMP375                    505                1947962     87072       535
## CCMP377                    493                 606376     78015       871
## CCMP1280                   510                4348365    113435       361
## RCC874                     522                7853977   3369442        15
## RCC914                     508                 653764     77641       462
## RCC1222                    160                1451258     55027       677
## RCC1239                    506                1376033    370740       138
## RCC1256                    393                 579311     44965      1147
## RCC3492                    496                1245722    106475       366
## RCC3963                    544                1638904    158060       295
## RCC6071                    545                2885505    546189       101
## RCC6856                    528                 509900     83052       795
##          rmdup_Total assembled length rmdup_Contig count
## CCMP371                     178311892              3749
## CCMP375                     184716260              7179
## CCMP377                     252888127              8513
```

```
## CCMP1280                   176639739               6028
## RCC874                     152306371                383
## RCC914                     147848163               7788
## RCC1222                    145640905               9374
## RCC1239                    170164584               2174
## RCC1256                    198119043               9915
## RCC3492                    171317621               8474
## RCC3963                    183446030               4042
## RCC6071                    192603649               1770
## RCC6856                    238298182               8184
##          rmdup_Min contig length rmdup_Max contig length contam_N50 contam_L50
## CCMP371                      671                 1476961     165430        455
## CCMP375                      505                 1947962      84373        813
## CCMP377                      493                  606376      72757       1172
## CCMP1280                     510                 4348365     103752        602
## RCC874                       522                 7853977    3436699         19
## RCC914                       508                  653764      71786        675
## RCC1222                      160                 1451258      62076       1112
## RCC1239                      506                 1376033     349891        150
## RCC1256                      393                  579311      41276       1648
## RCC3492                      496                 1245722      77326        542
## RCC3963                      544                 1638904     152703        470
## RCC6071                      545                 2885505     502076        116
## RCC6856                      528                  509900      80031       1238
##          contam_Total assembled length contam_Contig count
## CCMP371                     308351352                6334
## CCMP375                     265468378               10285
## CCMP377                     330382234               13113
## CCMP1280                    251827621                9074
## RCC874                      187857755                 646
## RCC914                      230137985               10566
## RCC1222                     271680264               17279
## RCC1239                     187058884                3497
## RCC1256                     257423911               13985
## RCC3492                     226943767               11053
## RCC3963                     302124558                6207
## RCC6071                     211278552                2687
## RCC6856                     355030595               12938
##          contam_Min contig length contam_Max contig length
## CCMP371                      455                 1477328
## CCMP375                       23                 1943894
## CCMP377                        9                  853175
## CCMP1280                      29                 4342429
## RCC874                        36                 7837603
## RCC914                        18                 4586128
## RCC1222                       10                 4200644
## RCC1239                       96                 3735226
## RCC1256                       26                  577965
## RCC3492                       13                 4907647
## RCC3963                      495                 3141847
## RCC6071                       12                 2882936
## RCC6856                        9                  745925
##          genome_haploid_length genome_unique_length
## CCMP371               98806191             72440020
```

```
## CCMP375              103218139             69423426
## CCMP377              138388017             76045285
## CCMP1280             111401730             64523269
## RCC874               118376237             91959391
## RCC914                86242315             53701350
## RCC1222              107508536             66175986
## RCC1239              116873978             83660661
## RCC1256               93127971             69271484
## RCC3492               77328042             62355972
## RCC3963              107212419             77275117
## RCC6071              131960871             81986586
## RCC6856              123410186             76317769
```

```
global_stats$Strain <- rownames(global_stats)
colors <- c("Dup" = "black", "Rmdup" = "blue","Contam"="red")

ggplot(global_stats,aes(x=dup_N50,y=genome_haploid_length,label=Strain,color="Dup"))+
  geom_point()+
  geom_point(aes(x=rmdup_N50,y=genome_haploid_length, color = "Rmdup"))+
  geom_point(aes(x=contam_N50,y=genome_haploid_length, color = "Contam"))+
  theme_bw()+
  scale_x_log10()+
  geom_smooth(method=lm,colour="black")+
  ylab("Haploid genome length")+
  xlab("Contig N50")+
  labs(caption=str_wrap("Figure 1.  Relationship between predicted haploid genome
                        length and contig N50 for 13 Emiliania huxleyi
                        genomes.",75))+
    geom_text_repel(size=3)+
  theme(plot.caption = element_text(hjust = 0,size=12))+
  labs(color = "Legend")+   scale_color_manual(values = colors)
```

Genomescope predicted haploid genome length vs N50

```
## `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?
```

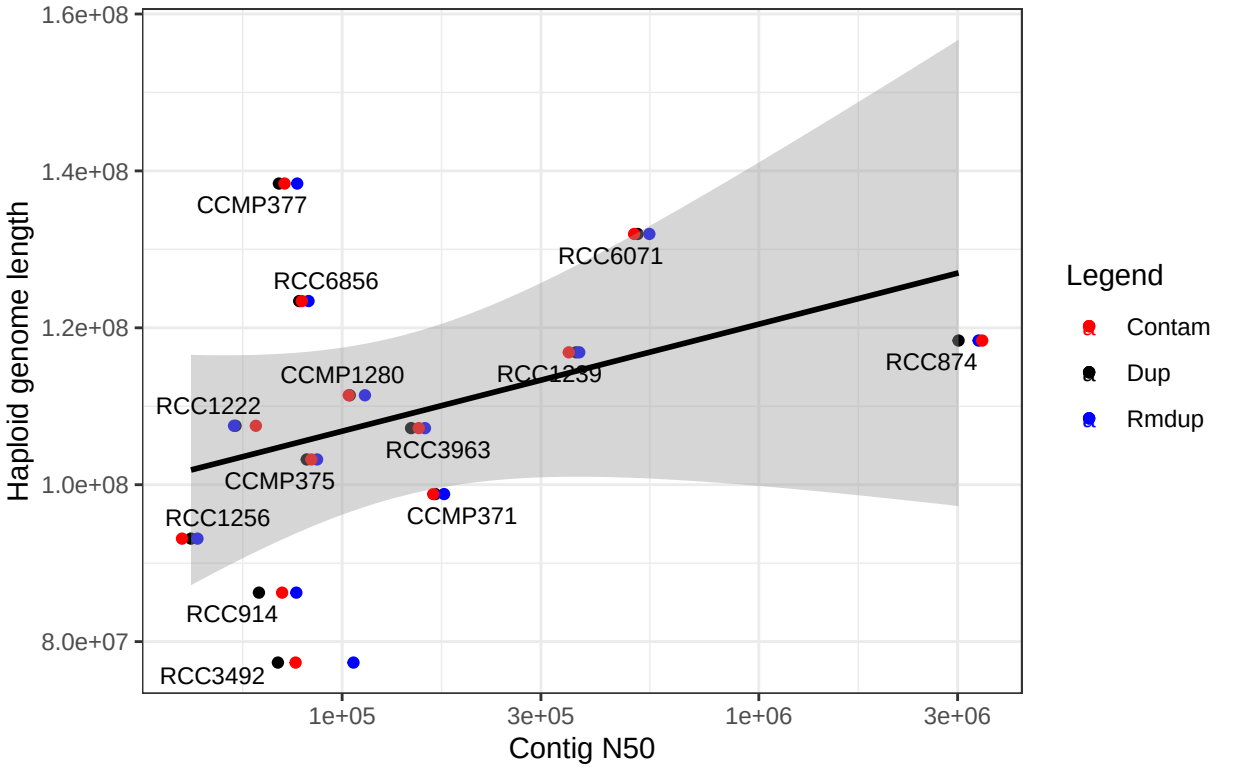

Figure 1. Relationship between predicted haploid genome length and contig N50 for 13 Emiliania huxleyi genomes.

```
ggsave("plots/haploid_genome_length_vs_N50.png")
```

```
## Saving 6.5 x 4.5 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?
```

```
global_stats$Strain <- rownames(global_stats)
ggplot(global_stats,aes(x=dup_N50,y=genome_unique_length,label=Strain,color="Dup"))+
  geom_point()+
  geom_point(aes(x=rmdup_N50,y=genome_unique_length, color = "Rmdup"))+
  geom_point(aes(x=contam_N50,y=genome_unique_length, color = "Contam"))+
  theme_bw()+
  scale_x_log10()+
  geom_smooth(method=lm,colour="black")+
  ylab("Unique genome length")+
  xlab("Contig N50")+
  labs(caption=str_wrap("Figure 2.  Relationship between predicted unique genome
```

```
                        length and contig N50 for 13 Emiliania huxleyi
                        genomes.",75))+
    geom_text_repel(size=3)+
  theme(plot.caption = element_text(hjust = 0,size=12))+
  labs(color = "Legend")+
  scale_color_manual(values = colors)
```

Genomescope predicted unique genome length vs N50

```
## `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?
```

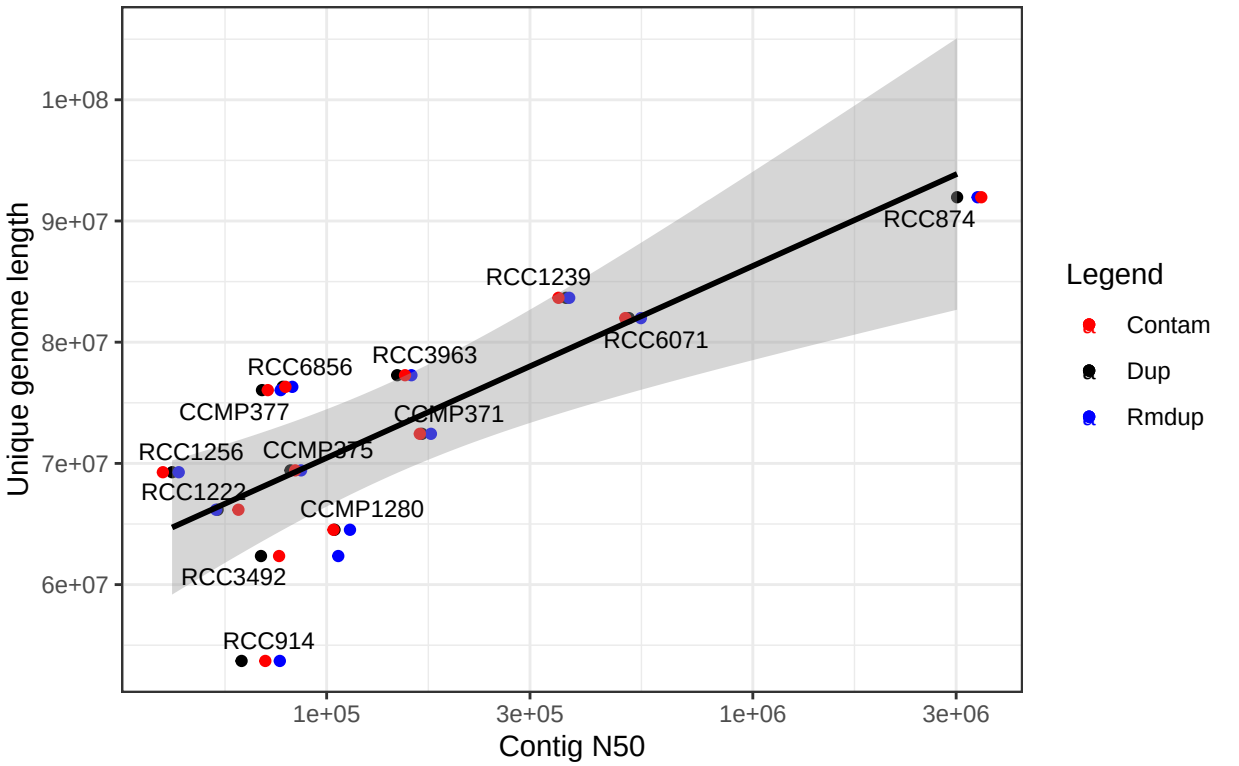

Figure 2. Relationship between predicted unique genome length and contig N50 for 13 Emiliania huxleyi genomes.

```
ggsave("plots/unique_genome_length_vs_N50.png")
```

```
## Saving 6.5 x 4.5 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?
```

```
stats_joined <- ldply(contam_stats,rbind)
contam <- ggplot(stats_joined,aes(x=Length.x))+geom_histogram()+scale_x_log10()+
           theme_bw()+
  xlab("Contig size")+
  ylab("Count")+facet_wrap(vars(.id))+
  ylim(0,2500)+
  theme(plot.caption = element_text(hjust = 0,size=12),
        axis.text.x = element_text(angle = 90))

stats_joined <- ldply(dup_stats,rbind)
dup <- ggplot(stats_joined,aes(x=Length.x))+geom_histogram()+scale_x_log10()+
           theme_bw()+
  xlab("Contig size")+
  ylab("Count")+facet_wrap(vars(.id))+
  ylim(0,2500)+
  theme(plot.caption = element_text(hjust = 0,size=12), axis.text.x = element_text(angle = 90))

stats_joined <- ldply(rmdup_stats,rbind)
rmdup <- ggplot(stats_joined,aes(x=Length.x))+geom_histogram()+scale_x_log10()+
           theme_bw()+
  xlab("Contig size")+
  ylab("Count")+facet_wrap(vars(.id))+
  ylim(0,2500)+
  theme(plot.caption = element_text(hjust = 0,size=12),
        axis.text.x = element_text(angle = 90))

combined <- plot_grid(contam,dup,rmdup, labels = c('Contam','Dup', 'Rmdup'), label_size = 12,ncol=3,vjus
```

Contig length distribution for each strain

```
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```

```
## Warning: Removed 1 rows containing missing values (`geom_bar()`).
```

```
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```

```
title <- ggdraw() + draw_label("Figure 3. Contig length distribution for 13 Emiliania huxleyi genomes",s
plot_grid(combined,title, ncol=1, rel_heights=c(1, 0.1))
```

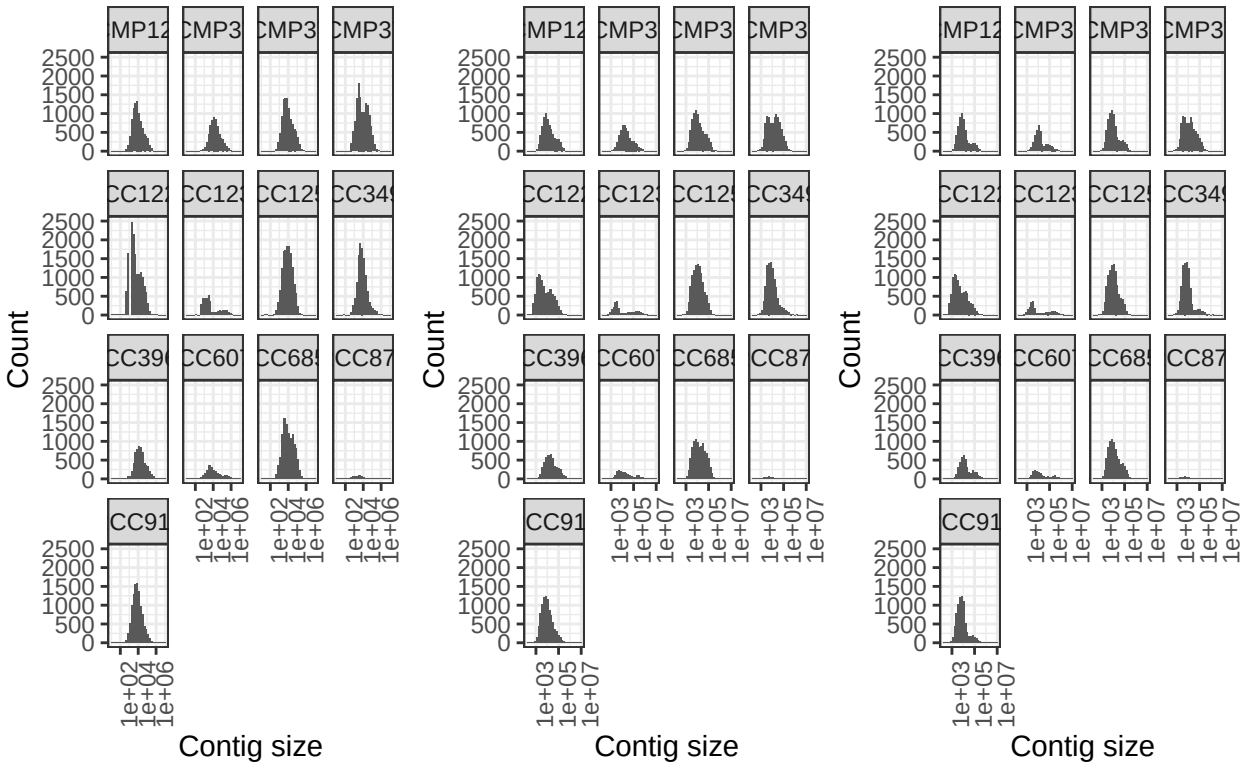

Figure 3. Contig length distribution for 13 Emiliania huxleyi genomes

```
ggsave("plots/contig_length_distributions.png",width=20,height=10)
```

```
stats_joined <- ldply(contam_stats,rbind)
contam <- ggplot(data = stats_joined, aes(x = GC,y=Avg_fold)) +
  geom_point(size=0.5)+
  ylim(0,300)+
  xlim(0.25,0.9)+
  #scale_y_log10()+
  facet_wrap(vars(.id))+
             theme_bw()+
  xlab("Contig GC%")+
  ylab("Average Fold Coverage")+
  theme(plot.caption = element_text(hjust = 0,size=12), axis.text.x = element_text(angle = 90))
nrow(stats_joined)
```

Scatterplots of fold coverage vs GC percentage (each point represents a contig)

```
## [1] 117664
```

```
stats_joined <- ldply(dup_stats,rbind)
dup <- ggplot(data = stats_joined, aes(x = GC,y=Avg_fold)) +
  geom_point(size=0.5)+
```

```
  ylim(0,300)+
  xlim(0.25,0.9)+
  #scale_y_log10()+
  facet_wrap(vars(.id))+
            theme_bw()+
  xlab("Contig GC%")+
  ylab("Average Fold Coverage")+
  theme(plot.caption = element_text(hjust = 0,size=12), axis.text.x = element_text(angle = 90))
nrow(stats_joined)
```

```
## [1] 90311
```

```
stats_joined <- ldply(rmdup_stats,rbind)
rmdup <- ggplot(data = stats_joined, aes(x = GC,y=Avg_fold)) +
  geom_point(size=0.5)+
  ylim(0,300)+
  xlim(0.25,0.9)+
  #scale_y_log10()+
  facet_wrap(vars(.id))+
            theme_bw()+
  xlab("Contig GC%")+
  ylab("Average Fold Coverage")+
  theme(plot.caption = element_text(hjust = 0,size=12),
        axis.text.x = element_text(angle = 90))
nrow(stats_joined)
```

```
## [1] 77573
```

```
combined <- plot_grid(contam,dup,rmdup, labels = c('Contam','Dup', 'Rmdup'), label_size = 12,ncol=3,vjus
```

```
## Warning: Removed 112 rows containing missing values (`geom_point()`).
```

```
## Warning: Removed 150 rows containing missing values (`geom_point()`).
```

```
## Warning: Removed 174 rows containing missing values (`geom_point()`).
```

```
title <- ggdraw() + draw_label("Figure 4. Contig fold coverage vs GC percentage for 13 Emiliania huxley
plot_grid(combined,title, ncol=1, rel_heights=c(1, 0.1))
```

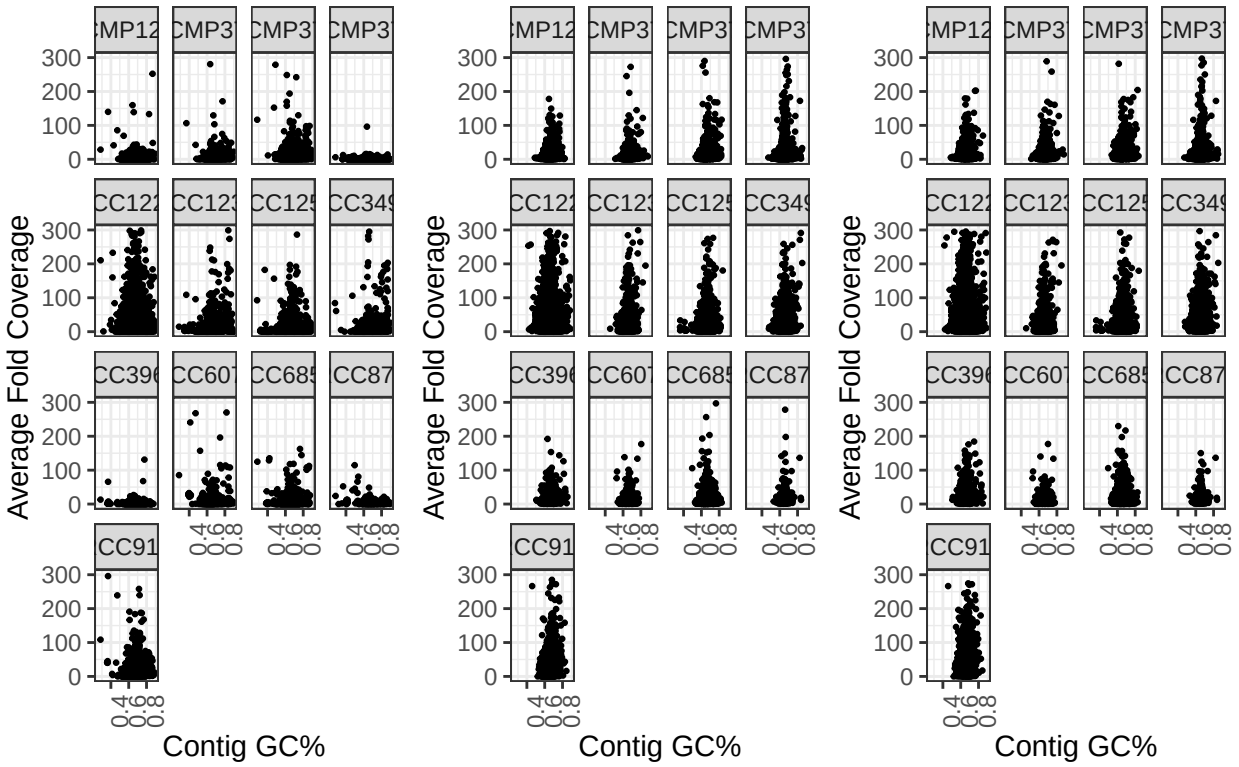

Figure 4. Contig fold coverage vs GC percentage for 13 Emiliania huxleyi genomes

```
ggsave("plots/coverage_vs_GC.png",height=10,width=20)
```

```
global_stats$rmdup_total_assembled_length <- global_stats[,"rmdup_Total assembled length"]
global_stats$dup_total_assembled_length <- global_stats[,"dup_Total assembled length"]
global_stats$contam_total_assembled_length <- global_stats[,"contam_Total assembled length"]

ggplot(global_stats,aes(x=rmdup_total_assembled_length,y=genome_unique_length,label=Strain,color="Rmdup
  geom_point()+
  geom_point(aes(x=dup_total_assembled_length,y=genome_unique_length, color = "Dup"))+
  geom_point(aes(x=contam_total_assembled_length,y=genome_unique_length, color = "Contam"))+
  theme_bw()+
  geom_abline(color="black",slope=1,linetype = "dashed")+
    geom_abline(color="black",slope=0.5,linetype = "dashed")+
  xlim(0,4e+8)+
  ylim(0,1.5e+8)+
  ylab("Unique genome length")+
  xlab("Total assembled length")+
  labs(caption=str_wrap("Figure 5.  Relationship between predicted unique genome
                        length and total assembled length for 13 Emiliania huxleyi
                        genomes.",75))+
    geom_text_repel(size=3,nudge_x=10,min.segment.length=0)+
    annotate("text",x=1e8,y=1e8,label="y=x",color="red",size=5)+
      annotate("text",x=2.3e8,y=1e8,label="y=0.5x",color="red",size=5)+
```

```
  theme(plot.caption = element_text(hjust = 0,size=12))+
  labs(color = "Legend")+
  scale_color_manual(values = colors)
```

Genomescope predicted unique genome length vs assembled length

```
## Warning: ggrepel: 2 unlabeled data points (too many overlaps). Consider
## increasing max.overlaps
```

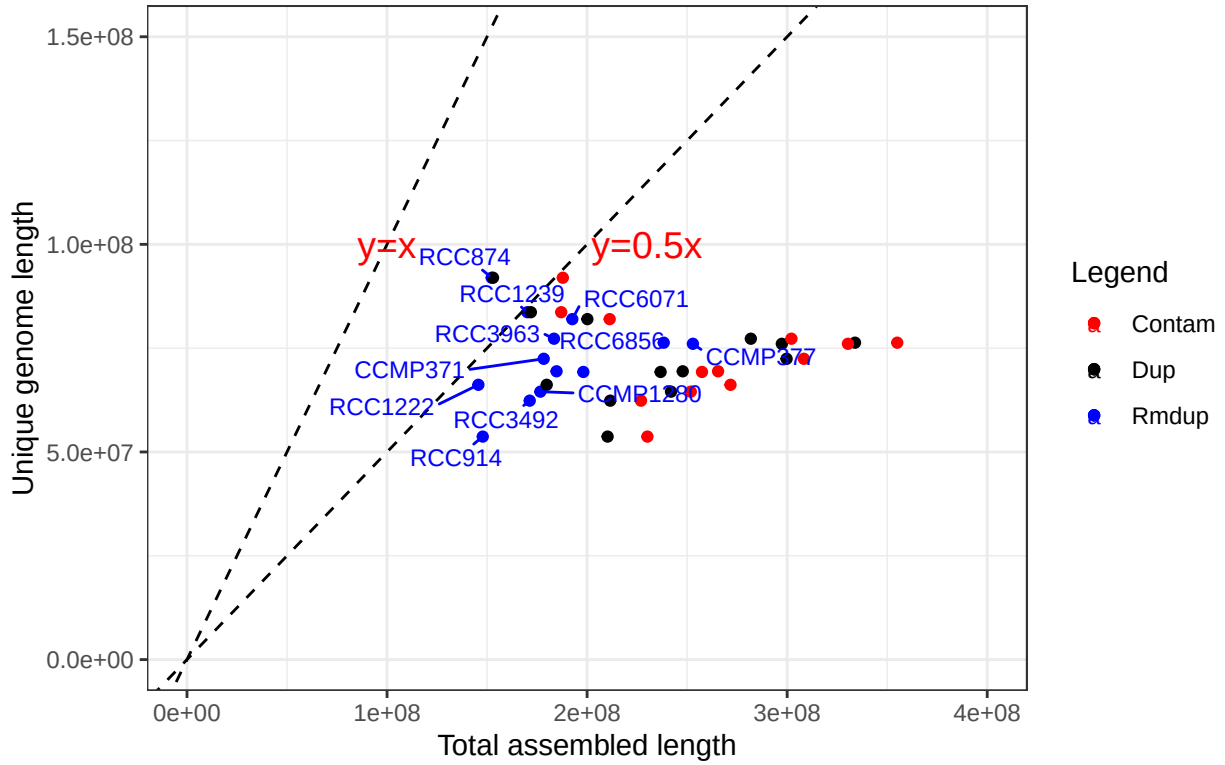

Figure 5. Relationship between predicted unique genome length and total assembled length for 13 Emiliania huxleyi genomes.

```
ggsave("plots/unique_genome_length_vs_total_assembled_length.png")
```

```
## Saving 6.5 x 4.5 in image
```

```
## Warning: ggrepel: 1 unlabeled data points (too many overlaps). Consider
## increasing max.overlaps
```

```
ggplot(global_stats,aes(x=rmdup_total_assembled_length,y=genome_haploid_length,label=Strain,color="Rmdup
  geom_point()+
  geom_point(aes(x=dup_total_assembled_length,y=genome_haploid_length, color = "Dup"))+
  geom_point(aes(x=contam_total_assembled_length,y=genome_haploid_length, color = "Contam"))+
  theme_bw()+
  geom_abline(color="black",slope=1,linetype = "dashed")+
    geom_abline(color="black",slope=0.5,linetype = "dashed")+
```

```
  xlim(0,4e+8)+
  ylim(0,1.5e+8)+
  ylab("Haploid genome length")+
  xlab("Total assembled length")+
  labs(caption=str_wrap("Figure 6.  Relationship between predicted haploid genome
                        length and total assembled length for 13 Emiliania huxleyi
                        genomes.",75))+
    geom_text_repel(size=3,nudge_x=10,min.segment.length=0)+
    annotate("text",x=1e8,y=1e8,label="y=x",color="red",size=5)+
      annotate("text",x=1.3e8,y=5e7,label="y=0.5x",color="red",size=5)+
  theme(plot.caption = element_text(hjust = 0,size=12))+
  labs(color = "Legend")+
  scale_color_manual(values = colors)
```

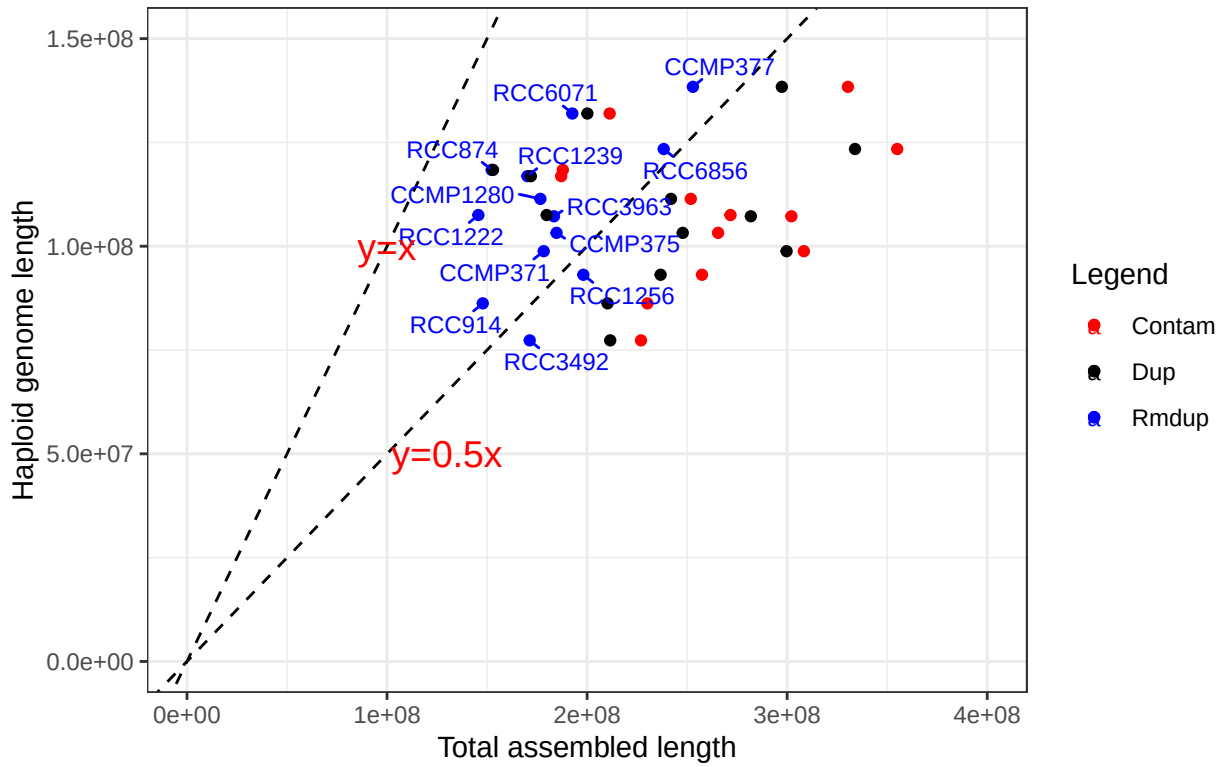

Figure 6. Relationship between predicted haploid genome length and total assembled length for 13 Emiliania huxleyi genomes.

```
ggsave("plots/haploid_genome_length_vs_total_assembled_length.png")
```

```
## Saving 6.5 x 4.5 in image
```