Script R1: Virome Contig and Sequencing Statistics

HANNIGAN GD, GRICE EA, ET AL.

Abstract

This protocol outlines the analysis used to plot contig coverage statistics, as well as sequence count and length stats. We begin with visualizing contig length vs coverage. We then visualize the distributions of sequence counts per sample as a probability density plot (similar idea as a histogram), and then do the same for median sequence length. Based on the methods from the following publication:

Hannigan, Geoffrey D., et al. "The Human Skin Double-Stranded DNA Virome: Topographical and Temporal Diversity, Genetic Enrichment, and Dynamic Associations with the Host Microbiome." *mBio* 6.5 (2015): e01578-15.

Citation: HANNIGAN GD, GRICE EA, ET AL. Script R1: Virome Contig and Sequencing Statistics. **protocols.io** dx.doi.org/10.17504/protocols.io.eh5bb86
Published: 10 Mar 2016

Guidelines

sessionInfo()

```
## R version 3.2.0 (2015-04-16)
## Platform: x86_64-apple-darwin13.4.0 (64-bit)
## Running under: OS X 10.10.4 (Yosemite)
## ## locale:
## [1] en_US.UTF-8/en_US.UTF-8/en_US.UTF-8/C/en_US.UTF-8/en_US.UTF-8
##
## attached base packages:
## [1] stats graphics grDevices utils datasets methods base
##
## loaded via a namespace (and not attached):
## [1] magrittr_1.5    formatR_1.2    tools_3.2.0    htmltools_0.2.6
## [5] yaml_2.1.13    stringi_0.4-1    rmarkdown_0.7    knitr_1.10.5
## [9] stringr_1.0.0    digest_0.6.8    evaluate_0.7
```

Before start

Supplemental information available at:

https://figshare.com/articles/The_Human_Skin_dsDNA_Virome_Topographical_and_Temporal_Diversity_Genetic_Enrichment_and_Dynamic_Associations_with_the_Host_Microbiome/1281248

Protocol

Step 1.

Load the required R packages.

cmd COMMAND

```
library("plyr")
packageVersion("plyr")

library("ggplot2")
packageVersion("ggplot2")

library("hexbin")
packageVersion("hexbin")

library("reshape2")
packageVersion("reshape2")
```

EXPECTED RESULTS

```
## [1] '1.8.2'
## [1] '1.0.1'
## [1] '1.27.0'
## [1] '1.4.1'
```

Step 2.

We will start by processing the virome samples. Import the tab delimited file containing the virome contig statistics.

cmd COMMAND

```
contig_stats <-
 read.delim("../../IntermediateOutput/Virome_Sequence_Counts/contig_length_with_coverage_fo
r_graphing.tsv", header=TRUE, sep='\t')
head(contig_stats)
```

EXPECTED RESULTS

```
##    contig  count  contig_length
## 1 34408_     40           570
## 2   578_     26           648
## 3 45993_     44           668
## 4  9954_    360          2674
## 5 66003_     18           830
## 6 28197_     17           570
```

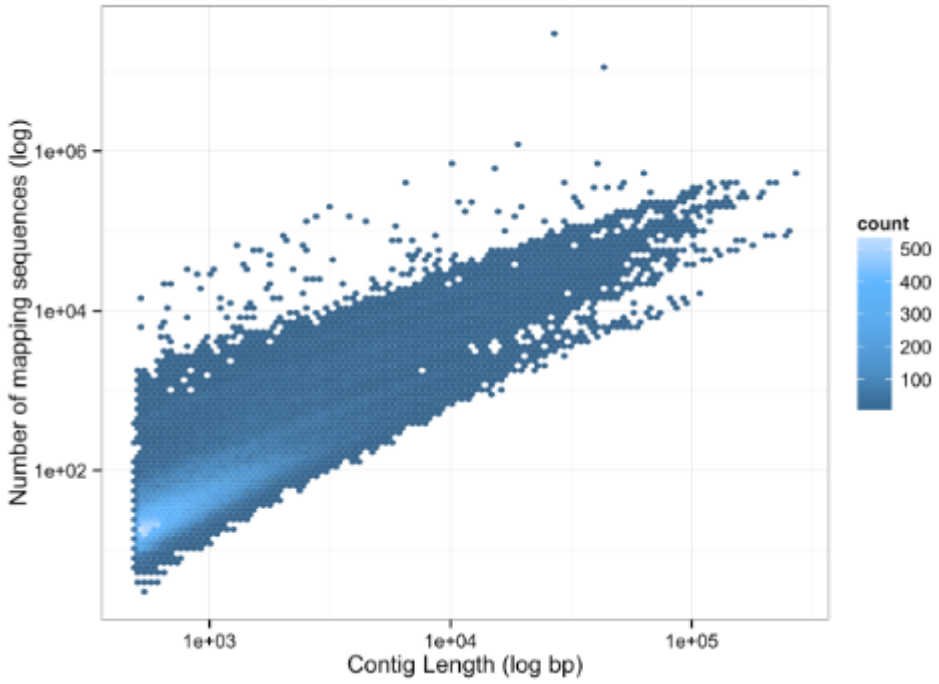
Step 3.

Generate a scatter plot comparing the number of mapped sequences to each contig, compared to the contig length.

cmd COMMAND

```
ggplot(contig_stats, aes(x=contig_length, y=count)) + theme_bw() + stat_binhex(bins=100)+ s
cale_fill_gradientn(colours=c("steelblue4", "steelblue3", "steelblue2","steelblue1","slateg
ray1")) + scale_y_log10() + scale_x_log10() + xlab("Contig Length (log bp)") + ylab("Number
 of mapping sequences (log)")
```

EXPECTED RESULTS



Step 4.

Next we will estimate contig coverage, using a sequence length value of 150 bp.

cmd COMMAND

```
contig_stats$coverage <- (contig_stats$count * 150) / contig_stats$contig_length
head(contig_stats)
```

EXPECTED RESULTS

##	contig	count	contig_length	coverage
## 1	34408_	40	570	10.526316
## 2	578_	26	648	6.018519
## 3	45993_	44	668	9.880240
## 4	9954_	360	2674	20.194465
## 5	66003_	18	830	3.253012
## 6	28197_	17	570	4.473684

Step 5.

Like above, we will plot the contig coverage against the contig length.

cmd COMMAND

```
ggplot(contig_stats, aes(x=contig_length, y=coverage)) + theme_bw() + stat_binhex(bins=100)
+ scale_fill_gradientn(colours=c("steelblue4", "steelblue3", "steelblue2","steelblue1","sla
tegray1")) + scale_y_log10(breaks=c(1e+01, 1e+02, 1e+03, 1e+04, 1e+05)) + scale_x_log10() +
 xlab("Contig Length (log bp)") + ylab("Contig Coverage (log)")
```

Step 6.

Next we can calculate the sequence statistics by anatomical site. First import the sequence summary statistics table. Be sure to change the column names of the input data frames, and to merge the data frames together for downstream processing. Here we will also import and format the mapping file.

cmd COMMAND

```
#Import the sequence counts before quality control
```

```
INPUT_RAW <-
 read.delim("../../IntermediateOutput/Virome_Sequence_Counts/raw_sequence_counts.tsv", head
er=FALSE, sep="\t")
colnames(INPUT_RAW) <- c("SampleID","Raw")
```

Step 7.

Import the sequence counts after sequence quality control.

cmd COMMAND

```
INPUT_TRIM <-
 read.delim("../../IntermediateOutput/Virome_Sequence_Counts/trimmed_sequence_counts.tsv",
header=FALSE, sep="\t")
colnames(INPUT_TRIM) <- c("SampleID","Trim")
```

Step 8.

Import the sequence counts after human sequence decontamination.

cmd COMMAND

```
INPUT_HUMAN <-
 read.delim("../../IntermediateOutput/Virome_Sequence_Counts/human_deconseq_sequence_counts
.tsv", header=FALSE, sep="\t")
colnames(INPUT_HUMAN) <- c("SampleID","Human")
```

Step 9.

Import the sequence counts after environmental background has been removed.

cmd COMMAND

```
INPUT_NEG <-
 read.delim("../../IntermediateOutput/Virome_Sequence_Counts/negative_clean_sequence_counts
.tsv", header=FALSE, sep="\t")
colnames(INPUT_NEG) <- c("SampleID","Neg")
```

Step 10.

Merge the input file so they are easier to deal with downstream.

cmd COMMAND

```
INPUT_MERGE <-
 merge(merge(merge(INPUT_RAW, INPUT_TRIM, by="SampleID"), INPUT_HUMAN, by="SampleID"), INPU
T_NEG, by="SampleID", all=TRUE)

INPUT_MAP <-
 read.delim("../../IntermediateOutput/Mapping_files/SkinMet_and_Virome_001_metadata.tsv", h
eader=TRUE)
```

Step 11.

Merge the mapping file and merged data frame.

cmd COMMAND

```
MERGE_MAP <-
 merge(INPUT_MERGE, INPUT_MAP, by.x="SampleID", by.y="NexteraXT_Virome_SampleID")
head(MERGE_MAP)[,c(1:5)]
```

EXPECTED RESULTS

```
##     SampleID    Raw   Trim  Human   Neg
## 1 MG100098 188574 185953 169740 90875
## 2 MG100099 238653   3263   3193  2094
## 3 MG100100 211300 155254 148954 57638
## 4 MG100101  55803  21080  20964 14366
## 5 MG100102 169504  77972  74997 34095
## 6 MG100103 105538  38790  35722 18480
```

Step 12.

Now we can caluclate the sequence statistics by site and plot the sequence count information after

some more formatting and merging.

cmd COMMAND

```
MERGE_SUBSET <- MERGE_MAP[,c(2:5,1,10,11)]
MERGE_SUB_RAW <- MERGE_SUBSET[,c("Raw","SampleID","Site_Symbol")]
colnames(MERGE_SUB_RAW) <- c("Count","SampleID","Site_Symbol")
MERGE_SUB_RAW$GROUP <- "Raw"
MERGE_SUB_TRIM <- MERGE_SUBSET[,c("Trim","SampleID","Site_Symbol")]
colnames(MERGE_SUB_TRIM) <- c("Count","SampleID","Site_Symbol")
MERGE_SUB_TRIM$GROUP <- "Trim"
MERGE_SUB_HUMAN <- MERGE_SUBSET[,c("Human","SampleID","Site_Symbol")]
colnames(MERGE_SUB_HUMAN) <- c("Count","SampleID","Site_Symbol")
MERGE_SUB_HUMAN$GROUP <- "Human"
MERGE_SUB_NEG <- MERGE_SUBSET[,c("Neg","SampleID","Site_Symbol")]
colnames(MERGE_SUB_NEG) <- c("Count","SampleID","Site_Symbol")
MERGE_SUB_NEG$GROUP <- "Neg"
CAT_MERGE <- rbind(MERGE_SUB_RAW, MERGE_SUB_TRIM, MERGE_SUB_HUMAN, MERGE_SUB_NEG)
```

Step 13.

Remove the incomplete sites.

cmd COMMAND

```
CAT_MERGE_SUB <- CAT_MERGE[-which(CAT_MERGE$Site_Symbol %in% c("Ba","Ph","Vf","No")), ]
```

Step 14.

Check the head of this data frame.

cmd COMMAND

```
head(CAT_MERGE_SUB)
```

EXPECTED RESULTS

##	Count	SampleID	Site_Symbol	Group
## 1	188574	MG100098	Fh	Raw
## 2	238653	MG100099	Ra	Raw
## 3	211300	MG100100	Oc	Raw
## 4	55803	MG100101	Ax	Raw
## 5	169504	MG100102	Ac	Raw
## 6	363647	MG100104	Pa	Raw

Step 15.

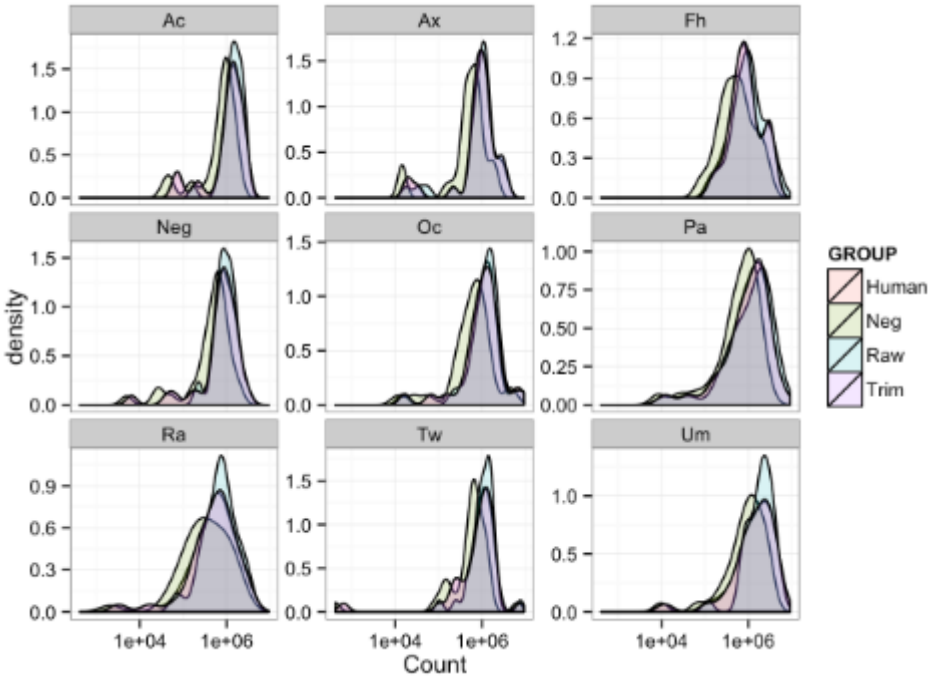
Now plot the sequence count information.

cmd COMMAND

```
ggplot(CAT_MERGE_SUB, aes(x=Count, fill=GROUP)) + theme_bw() + geom_density(alpha=0.2) + sc
ale_x_log10() + facet_wrap(~Site_Symbol, scale="free_y")
```

EXPECTED RESULTS

Warning in loop_apply(n, do.ply): Removed 4 rows containing non-finite values (stat_density).



Step 16.

Now we are going to do the same thing as the above sequence count stats, except now we are using the sequence length stats.

cmd COMMAND

```
LENGTH_RAW <-
 read.delim("../../IntermediateOutput/Virome_Sequence_Counts/raw_sequence_length_medians.ts
v", header=FALSE, sep="\t")
colnames(LENGTH_RAW) <- c("Raw","SampleID")
LENGTH_TRIM <-
 read.delim("../../IntermediateOutput/Virome_Sequence_Counts/trimmed_sequence_length_median
s.tsv", header=FALSE, sep="\t")
colnames(LENGTH_TRIM) <- c("Trim","SampleID")
LENGTH_HUMAN <-
 read.delim("../../IntermediateOutput/Virome_Sequence_Counts/clean_sequence_length_medians.
tsv", header=FALSE, sep="\t")
colnames(LENGTH_HUMAN) <- c("Human","SampleID")
LENGTH_NEG <-
 read.delim("../../IntermediateOutput/Virome_Sequence_Counts/negative_clean_sequence_length
_medians.tsv", header=FALSE, sep="\t")
colnames(LENGTH_NEG) <- c("Neg","SampleID")

LENGTH_MERGE <-
 merge(merge(merge(LENGTH_RAW, LENGTH_TRIM, by="SampleID"), LENGTH_HUMAN, by="SampleID"), L
ENGTH_NEG, by="SampleID", all=TRUE)

MERGE_MAP <-
 merge(LENGTH_MERGE, INPUT_MAP, by.x="SampleID", by.y="NexteraXT_Virome_SampleID")
```

Step 17.

Prepare format to plot sequence length.

cmd COMMAND

```
MERGE_SUBSET <- MERGE_MAP[,c(2:5,1,10,11)]
```

```
MERGE_SUB_RAW <- MERGE_SUBSET[,c("Raw","SampleID","Site_Symbol")]
colnames(MERGE_SUB_RAW) <- c("Count","SampleID","Site_Symbol")
MERGE_SUB_RAW$GROUP <- "Raw"
MERGE_SUB_TRIM <- MERGE_SUBSET[,c("Trim","SampleID","Site_Symbol")]
colnames(MERGE_SUB_TRIM) <- c("Count","SampleID","Site_Symbol")
MERGE_SUB_TRIM$GROUP <- "Trim"
MERGE_SUB_HUMAN <- MERGE_SUBSET[,c("Human","SampleID","Site_Symbol")]
colnames(MERGE_SUB_HUMAN) <- c("Count","SampleID","Site_Symbol")
MERGE_SUB_HUMAN$GROUP <- "Human"
MERGE_SUB_NEG <- MERGE_SUBSET[,c("Neg","SampleID","Site_Symbol")]
colnames(MERGE_SUB_NEG) <- c("Count","SampleID","Site_Symbol")
MERGE_SUB_NEG$GROUP <- "Neg"
CAT_MERGE <- rbind(MERGE_SUB_RAW, MERGE_SUB_TRIM, MERGE_SUB_HUMAN, MERGE_SUB_NEG)
CAT_MERGE_SUB <- CAT_MERGE[-which(CAT_MERGE$Site_Symbol %in% c("Ba","Ph","Vf","No")), ]
```

Step 18.

Plot the sequence count information.

cmd COMMAND

```
ggplot(CAT_MERGE_SUB, aes(x=Count, fill=GROUP)) + theme_bw() + geom_density(alpha=0.2) + fa
cet_wrap(~Site_Symbol, scale="free_y")
```

EXPECTED RESULTS

Warning in loop_apply(n, do.ply): Removed 4 rows containing non-finite values (stat_density).