Script R3: Quality Control Statistics

HANNIGAN GD, GRICE EA, ET AL.

Abstract

This protocol outlines the quality control analysis measuring the percent relative abundance of 16S rRNA levels in the virome and whole metagenome, as well as the number of virome sequences mapping to the whole metagenome, and vice versa. These measures are used to validate the quality of the sequence datasets. 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 R3: Quality Control Statistics. **protocols.io**
dx.doi.org/10.17504/protocols.io.eh9bb96
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 proper R libraries.

cmd COMMAND

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

library(plyr)
packageVersion("plyr")

library(reshape2)
packageVersion("reshape2")

library(RColorBrewer)
packageVersion("RColorBrewer")
```

EXPECTED RESULTS

```
## [1] '1.0.1'
```

```
## [1] '1.8.2'
```

```
## [1] '1.4.1'
```

```
## [1] '1.1.2'
```

Step 2.

First read in the expected values for the mock community sample.

cmd COMMAND

```
mock_community <-
 read.delim("../../IntermediateOutput/Mock_Community/mock_community_samples.txt")
mock_community<-
mock_community[,c("Phylum","Class", "Order", "Family", "Genus","Species", "Relative_Abundan
ce")]

mock_community_genus<-mock_community[,c("Genus", "Relative_Abundance")]
mock_community_genus$Genus<-
paste("p__", mock_community$Phylum, ";g__", mock_community$Genus, sep="")
mock_community_genus<-
aggregate(mock_community_genus$Relative_Abundance, list(mock_community_genus$Genus), FUN=su
m)
colnames(mock_community_genus)<-c("Genus", "MockCommunity")
```

Step 3.

Then read in the MetaPhlAn output for the mock community.

cmd COMMAND

```
WMS_all<-
```

```
read.delim("../../IntermediateOutput/Mock_Community/MG100410_R1_trimmed_metaphlan_all.txt",
 sep="\t", header=FALSE)
```

Step 4.

Format the data for plotting, looking at the genus level.

cmd COMMAND

```
format_WMS_data_metaphlan<-function(skinmet_data){
  colnames(skinmet_data)<-c("Taxon", "MG100410_met")
  skinmet_data$Taxon<-
gsub(skinmet_data$Taxon, pattern="^k__Bacteria$", replacement="REMOVE")
  skinmet_data$Taxon<-gsub(skinmet_data$Taxon, pattern="^k__Bacteria\\|p__[A-
z]*$", replacement="REMOVE")
  skinmet_data$Taxon<-gsub(skinmet_data$Taxon, pattern="^k__Bacteria.*c__[A-
z]*$", replacement="REMOVE")
  skinmet_data$Taxon<-gsub(skinmet_data$Taxon, pattern="^k__Bacteria.*o__[A-
z]*$", replacement="REMOVE")
  skinmet_data$Taxon<-gsub(skinmet_data$Taxon, pattern="^k__Bacteria.*f__[A-
z]*$", replacement="REMOVE")
  skinmet_data$Taxon<-
gsub(skinmet_data$Taxon, pattern="^k__.*\\|s__.*", replacement="REMOVE")
  skinmet_data<-subset(skinmet_data, skinmet_data$Taxon != "REMOVE")

  skinmet_data$Taxon<-gsub(skinmet_data$Taxon, pattern="\\|", replacement=";")
```

Step 5.

Convert to relative abundances.

cmd COMMAND

```
skinmet_data[,2]<-skinmet_data[,2]/sum(skinmet_data[,2])
  return(skinmet_data)
}

WMS_updated_genus<-format_WMS_data_metaphlan(WMS_all)

WMS_updated_genus$Taxon<-
gsub(WMS_updated_genus$Taxon, pattern="Thermi", replacement="Deinococcus-Thermus")
row.names(WMS_updated_genus)<-WMS_updated_genus$Taxon
WMS_updated_genus$Taxon<-NULL
row.names(WMS_updated_genus)<-
gsub(row.names(WMS_updated_genus), pattern="k__Bacteria;p__", replacement="p__")
row.names(WMS_updated_genus)<-
gsub(row.names(WMS_updated_genus), pattern="c__.*;g__", replacement="g__")

genus<-
merge(mock_community_genus, WMS_updated_genus, by.x="Genus", by.y="row.names", all.x=TRUE)
genus[is.na(genus)]<-0

genus_m<-melt(genus, id.vars=c("Genus"))
colnames(genus_m)<-c("Genus","SampleID","RelativeAbundance")

genus_m$Genus<-gsub(genus_m$Genus, pattern="^.*g__", replacement="", perl=TRUE)
genus_m<-merge(unique(mock_community[,c("Phylum","Genus")]), genus_m, by="Genus")
genus_m$Taxa<-paste("p__",genus_m$Phylum,";g__",genus_m$Genus, sep="")
```

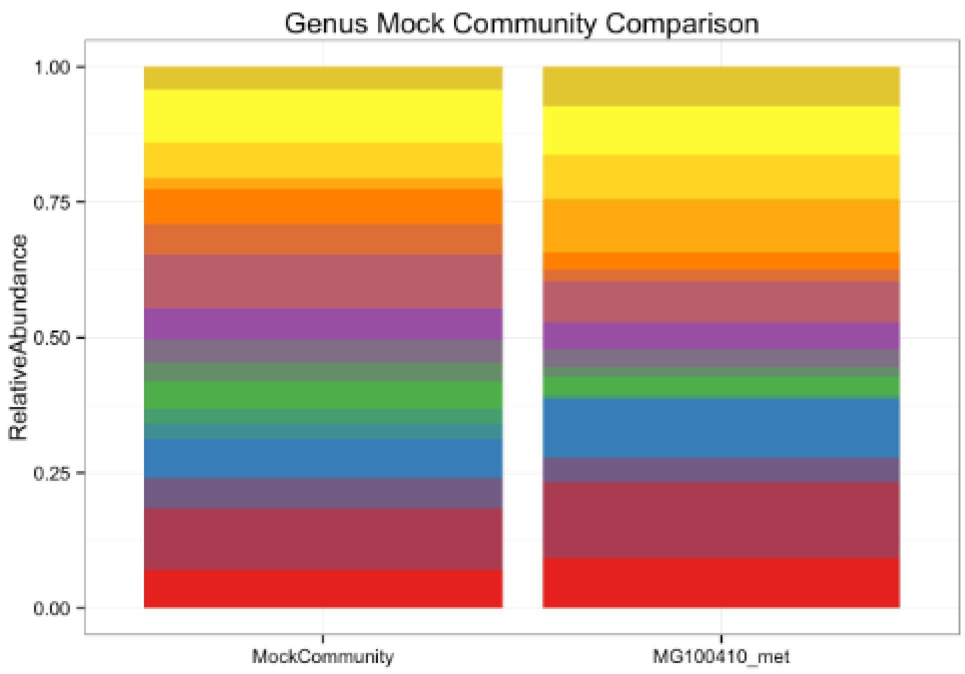
Step 6.

Plot genus level comparisons.

cmd COMMAND

```
ggplot(genus_m[order(genus_m$Phylum,genus_m$Genus),], aes(x=SampleID, y=RelativeAbundance,
fill=Taxa))+theme_bw()+geom_bar(stat = "identity")+theme(axis.title.x = element_blank(),leg
end.position='none')+ guides(fill = guide_legend(keywidth = 1,keyheight = 0.8)) +scale_fill
_manual(values = colorRampPalette(brewer.pal(8,"Set1"))(22)) + ggtitle("Genus Mock Communit
y Comparison")
```

EXPECTED RESULTS



Step 7.
Then, we calculated the 16S rRNA levels in each dataset. Import data frame.

COMMAND
```
INPUT <-
 read.delim("../../IntermediateOutput/Quality_Control_Stats/total_contamination_info_for_R.
tsv", sep="\t", header=FALSE)
```

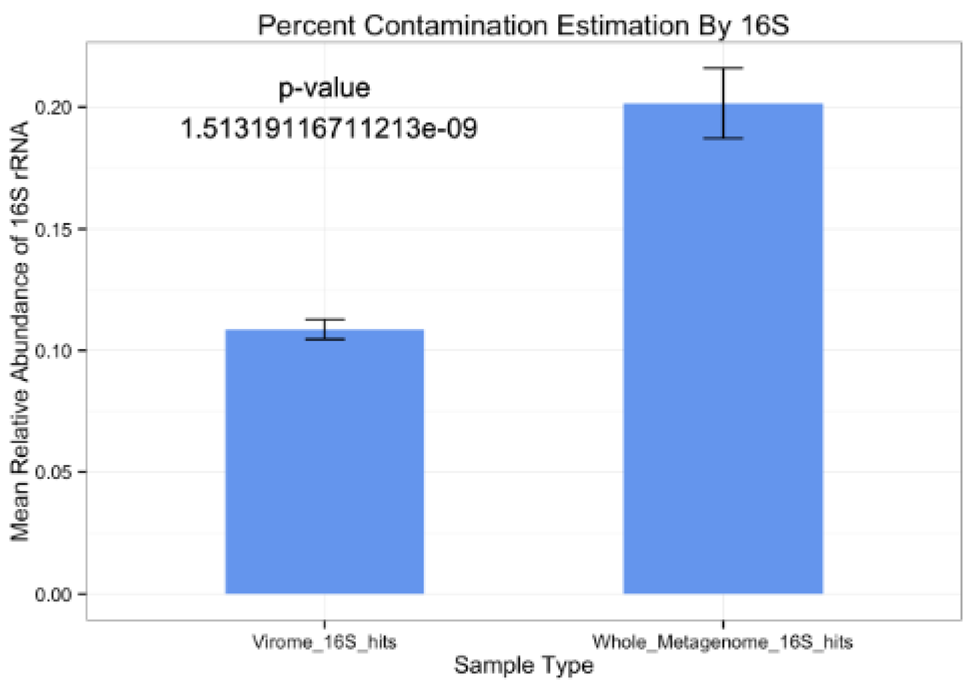
Step 8.
Add column for level percent contamination.

COMMAND
```
INPUT$Contamination <- 100 * INPUT$V1 / INPUT$V4
```

Step 9.
Generate bar graph with StdErr bars.

COMMAND
```
INPUT_STATS <-
 ddply(INPUT, c("V3"), summarise, N = length(Contamination), mean = mean(Contamination), sd
 = sd(Contamination), se   = sd / sqrt(N) )

T_TEST_P_VALUE <-
 t.test(INPUT$Contamination[INPUT$V3=="Virome_16S_hits"], INPUT$Contamination[INPUT$V3=="Wh
ole_Metagenome_16S_hits"])$p.value

ggplot(INPUT_STATS, aes(x=V3, y=mean)) + theme_bw()  + geom_bar(fill="cornflowerblue", widt
h=0.5, stat="identity") + geom_errorbar(aes(ymin=mean-
se, ymax=mean+se), width=0.1) + geom_text(x=1, y=0.20, label=paste("p-
value\n", T_TEST_P_VALUE)) + ggtitle("Percent Contamination Estimation By 16S") + ylab("Mea
n Relative Abundance of 16S rRNA") + xlab("Sample Type")
```

EXPECTED RESULTS



Step 10.

We next calculated the median numbers of virome sequences that map to the whole metagenome, and vice versa. Import data files and mapping file.

cmd COMMAND

```
VIR_TO_MET <-
 read.delim("../../IntermediateOutput/Quality_Control_Stats/virome_to_metagenome_shared_res
ults.tsv", sep="\t", header=FALSE)
MET_TO_VIR <-
 read.delim("../../IntermediateOutput/Quality_Control_Stats/metagenome_to_virome_shared_res
ults.tsv", sep="\t", header=FALSE)
MAP <-
 read.delim("../../IntermediateOutput/Mapping_files/SkinMet_and_Virome_001_metadata.tsv", h
eader=TRUE, sep="\t")
MAP_CUT <- MAP[-which(MAP$NexteraXT_SampleID %in% c(NA)), ]
MAP_CUT <- MAP_CUT[-which(MAP_CUT$NexteraXT_Virome_SampleID %in% c(NA)), ]

MET_TO_VIR_MERGE <- merge(MET_TO_VIR, MAP_CUT, by.x="V1", by.y="NexteraXT_SampleID")
MET_TO_VIR_MERGE$Percent_match_met_to_vir <-
 100 * MET_TO_VIR_MERGE$V3 / MET_TO_VIR_MERGE$V2

VIR_TO_MET_MERGE <- merge(VIR_TO_MET, MAP_CUT, by.x="V1", by.y="NexteraXT_Virome_SampleID")
VIR_TO_MET_MERGE$Percent_match_vir_to_met <-
 100 * VIR_TO_MET_MERGE$V3 / VIR_TO_MET_MERGE$V2
```

Step 11.

Get the range of percent matches.

cmd COMMAND

```
V_TO_M_RANGE <- range(VIR_TO_MET_MERGE$Percent_match_vir_to_met)
M_TO_V_RANGE <- range(MET_TO_VIR_MERGE$Percent_match_met_to_vir)

MET_TO_VIR_MEAN <- mean(MET_TO_VIR_MERGE$Percent_match_met_to_vir)
VIR_TO_MET_MEAN <- mean(VIR_TO_MET_MERGE$Percent_match_vir_to_met)
MET_TO_VIR_SEM <-
 sd(MET_TO_VIR_MERGE$Percent_match_met_to_vir)/sqrt(length(MET_TO_VIR_MERGE$Percent_match_m
et_to_vir))
VIR_TO_MET_SEM <-
```

```
 sd(VIR_TO_MET_MERGE$Percent_match_vir_to_met)/sqrt(length(VIR_TO_MET_MERGE$Percent_match_v
ir_to_met))
MERGED_MEAN <- as.data.frame(c(MET_TO_VIR_MEAN,VIR_TO_MET_MEAN))
MERGED_SEM <- as.data.frame(c(MET_TO_VIR_SEM,VIR_TO_MET_SEM))
MERGED_NAMES <- as.data.frame(c("met_to_vir","vir_to_met"))
MERGED <- cbind(MERGED_MEAN,MERGED_SEM,MERGED_NAMES)
colnames(MERGED) <- c("mean","SEM","category")
```

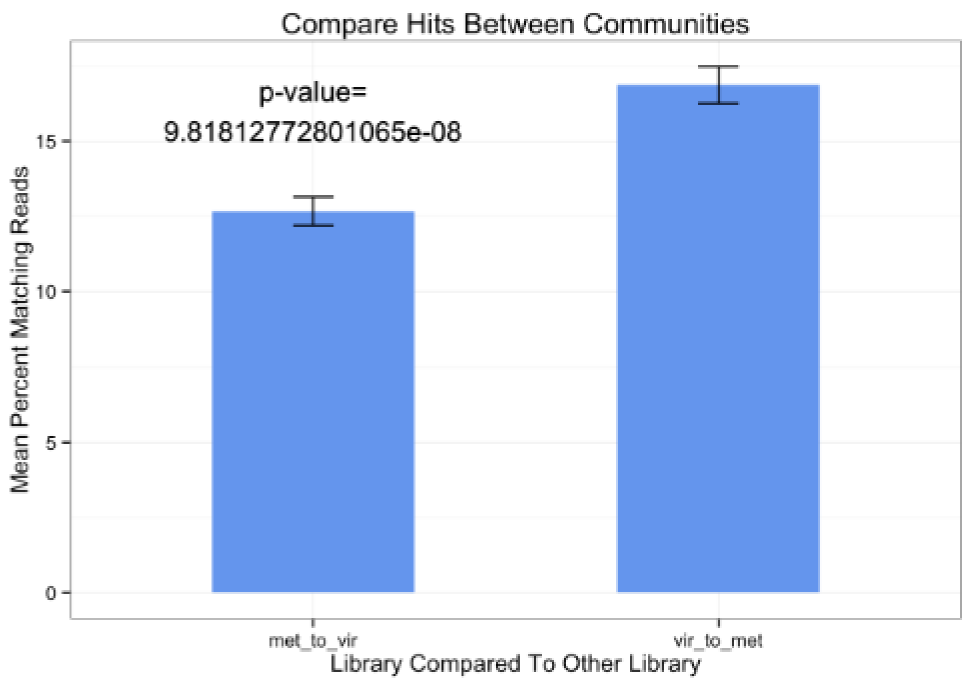
Step 12.

Perform statistical analysis and plot.

cmd COMMAND

```
TTEST <-
 t.test(MET_TO_VIR_MERGE$Percent_match_met_to_vir, VIR_TO_MET_MERGE$Percent_match_vir_to_me
t)
PVALUE <- TTEST$p.value

ggplot(MERGED, aes(x=category, y=mean)) + theme_bw() + geom_bar(position=position_dodge(),
stat="identity", fill="cornflowerblue", width=0.5) + geom_errorbar(aes(ymin=mean-
SEM, ymax=mean+SEM), width=0.1, position=position_dodge(.9)) + geom_text(label=paste("p-
value=",PVALUE,sep="\n"),y=16,x=1) + ggtitle("Compare Hits Between Communities") + xlab("Li
brary Compared To Other Library") + ylab("Mean Percent Matching Reads")
```

EXPECTED RESULTS



Step 13.

Finally, we calculated the percentage of human contamination in the virome versus the whole metagenome samples.

cmd COMMAND

```
wm_total_seq_counts<-
read.delim("../../IntermediateOutput/Whole_Microbiome_Sequence_Counts/trimmed_sequence_coun
ts.tsv",header=FALSE)
colnames(wm_total_seq_counts)<-c("SampleID","TotalTrimmedSeq")
wm_clean_seq_counts<-
read.delim("../../IntermediateOutput/Whole_Microbiome_Sequence_Counts/human_deconseq_sequen
ce_counts.tsv",header=FALSE)
colnames(wm_clean_seq_counts)<-c("SampleID","TotalCleanSeq")

wm_counts<-merge(wm_total_seq_counts,wm_clean_seq_counts,by="SampleID")
```

```
wm_counts$PercentContamination<-(wm_counts$TotalTrimmedSeq-
wm_counts$TotalCleanSeq)/wm_counts$TotalTrimmedSeq * 100

wm_counts_sub<-wm_counts[,c("SampleID","PercentContamination")]
wm_counts_sub$Type<-"WholeMetagenome"
```

Calculates human contamination percent in WHOLE METAGENOME samples

Step 14.

Calculate human contamination in virome samples.

cmd COMMAND

```
v_total_seq_counts<-
read.delim("../../IntermediateOutput/Virome_Sequence_Counts/trimmed_sequence_counts.tsv",he
ader=FALSE)
colnames(v_total_seq_counts)<-c("SampleID","TotalTrimmedSeq")
v_clean_seq_counts<-
read.delim("../../IntermediateOutput/Virome_Sequence_Counts/human_deconseq_sequence_counts.
tsv",header=FALSE)
colnames(v_clean_seq_counts)<-c("SampleID","TotalCleanSeq")

v_counts<-merge(v_total_seq_counts,v_clean_seq_counts,by="SampleID")
v_counts$PercentContamination<-(v_counts$TotalTrimmedSeq-
v_counts$TotalCleanSeq)/v_counts$TotalTrimmedSeq * 100
```

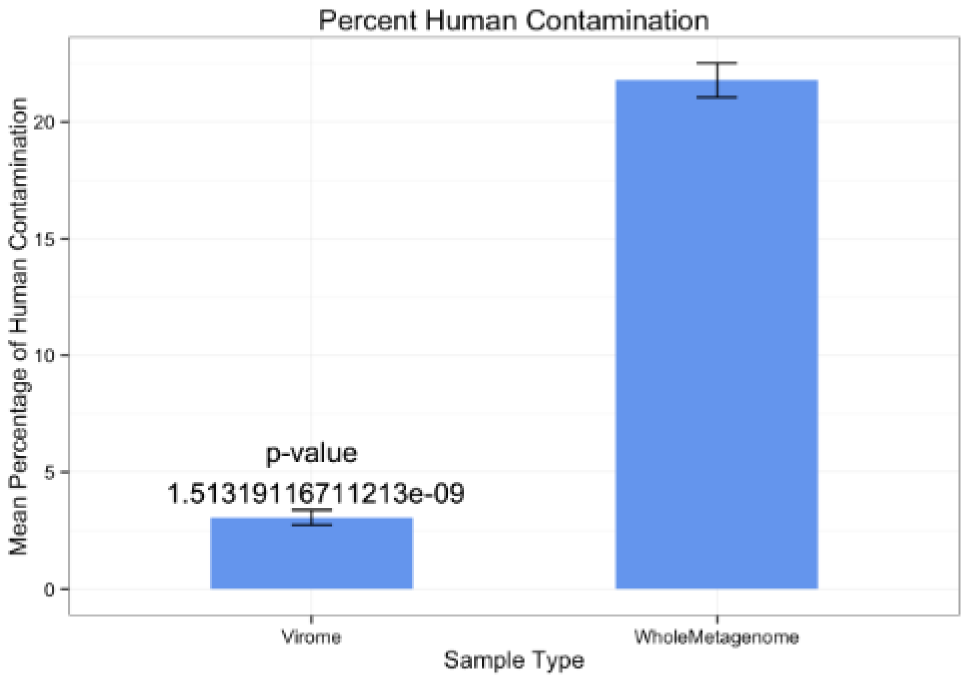
Step 15.

Remove row with total and plot graph.

cmd COMMAND

```
v_counts<-v_counts[-nrow(v_counts),]

v_counts_sub<-v_counts[,c("SampleID","PercentContamination")]
v_counts_sub$Type<-"Virome"

counts_all<-rbind(wm_counts_sub, v_counts_sub)

INPUT_STATS <-
 ddply(counts_all, c("Type"), summarise, N = length(PercentContamination), mean = mean(Perc
entContamination), sd = sd(PercentContamination), se   = sd / sqrt(N) )

ggplot(INPUT_STATS, aes(x=Type, y=mean)) + theme_bw()  + geom_bar(fill="cornflowerblue", wi
dth=0.5, stat="identity") + geom_errorbar(aes(ymin=mean-
se, ymax=mean+se), width=0.1) + geom_text(x=1, y=5, label=paste("p-
value\n", T_TEST_P_VALUE)) + ggtitle("Percent Human Contamination") + ylab("Mean Percentage
 of Human Contamination") + xlab("Sample Type")
```

EXPECTED RESULTS



Step 16.
Perform T test for p-value.

cmd COMMAND

```
T_TEST_P_VALUE <-
 t.test(counts_all$PercentContamination[counts_all$Type=="Virome"], counts_all$PercentConta
mination[counts_all$Type=="WholeMetagenome"])$p.value

T_TEST_P_VALUE
```

EXPECTED RESULTS

```
## [1] 8.07085e-81
```