Script R9: Plotting Microbial Taxonomy from MEGAN

HANNIGAN GD, GRICE EA, ET AL.

Abstract

This protocol outlines the analysis used to plot MEGAN taxonomic assignments. 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 R9: Plotting Microbial Taxonomy from MEGAN. **protocols.io**
dx.doi.org/10.17504/protocols.io.ejdbci6
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.

First load the libraries necessary for analysis.

cmd COMMAND

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

library(reshape2)
packageVersion("reshape2")

library(plyr)
packageVersion("plyr")

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

EXPECTED RESULTS

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

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

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

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

Step 2.

Then read in and format the metadata file.

cmd COMMAND

```
skinmet_metadata<-
read.delim("../../IntermediateOutput/Mapping_files/SkinMet_and_Virome_001_metadata.tsv")
skinmet_metadata<-
skinmet_metadata[,c("NexteraXT_SampleID","SubjectID","Site_Symbol","TimePoint")]
skinmet_metadata$NexteraXT_SampleID<-as.character(skinmet_metadata$NexteraXT_SampleID)
colnames(skinmet_metadata)[1]<-"SampleID"
skinmet_metadata<-subset(skinmet_metadata, skinmet_metadata$SampleID != "NA")
skinmet_metadata<-subset(skinmet_metadata, skinmet_metadata$TimePoint != 1)
skinmet_metadata<-subset(skinmet_metadata, !(skinmet_metadata$SubjectID %in% c(2,3,9,11)))
skinmet_metadata<-
subset(skinmet_metadata, !(skinmet_metadata$Site_Symbol %in% c("Neg", "Vf", "Ba", "Ph")))
skinmet_metadata$SubjectID<-NULL
skinmet_metadata$TimePoint<-NULL
```

Step 3.

Now we are ready to read in the actual data files. In the initial analysis, each sample had its own file. We wrote a function to read in the data for each sample and combine it into one data file. For simplicity, we only put the file with the combined data in the intermediate files, but the function readData shows you how we generated those files.

cmd COMMAND

```
readData <- function(taxa){

    ## Extract vector of empty files' names
    empty <- taxa[file.info(taxa)[["size"]]==0]
    ## Remove empty files
    unlink(empty, recursive=TRUE, force=FALSE)

    for(i in taxa){
        name<-gsub(x=i,pattern="_megan_.*.txt",replacement="",perl=TRUE)
        tmp<-read.delim(i, header=FALSE)
        colnames(tmp)<-c("Taxa", name)
        if(i==taxa[1]) {data<-tmp} else {data<-merge(data,tmp,"Taxa", all=TRUE)}
    }
    data[is.na(data)]=0

    name<-gsub(x=taxa[1], pattern="MG100.*_megan_", replacement="megan_", perl=TRUE)
    name2<-paste("../../IntermediateOutput/MEGAN/", name, sep="")
    write.table(data, name2, row.names=FALSE, quote=FALSE, eol="\r\n", sep="\t")

    return(data)
}
```

Step 4.

To make our results more interpretable, we only want to look at a certain number of taxa. We want to write a function that will keep a certain number of taxa and combine the remaining taxa into the category “Other”.

cmd COMMAND

```
topTaxa<- function(data, numTaxa){
    # check to see if you need to condense the taxa list to the specified number
    if(ncol(data)>numTaxa){
        # the data is organized where the taxa are the columns and the samples are the rows
        # add a row containing the total cumulative frequency of the taxon
        data[nrow(data)+1,]<-colSums(data)
        # order the columns by their frequency
        data<-data[,order(-data[nrow(data),])]
        # remove the last row containing the cumulative frequency
        data<-data[-nrow(data),]
        # remove the least frequent taxa
        tmp<-data[,numTaxa:ncol(data)]
        data<-data[,-c(numTaxa:ncol(data))]
        other<-rowSums(tmp)
        data<-cbind(data,other)
        colnames(data)[ncol(data)]<-"Other"
    }
    return(data)
}
```

Step 5.

Some of the genus level taxa were not correctly classified at the phylum level (they incorrectly labeled unclassified at phylum level). We need to correct this:

cmd COMMAND

```
removeDupes <- function(taxa_list, data){
  for (i in 1:length(taxa_list)){
    taxa<-taxa_list[i]
    dup<-grep(pattern=taxa, x=data$Taxa)
    dup_sum<-colSums(data[grep(pattern=taxa, x=data$Taxa),-1])
    data[dup[1],2:ncol(data)]<-dup_sum
    data<-data[-dup[2],]
  }
  return(data)

}
```

Step 6.

We want to read the data in and generate bar plots to visualize it.

cmd COMMAND

```
plotTaxa<- function(level, data){

    # Remove duplicates that were correctly classified at phylum level but also unclassifie
d at phylum level
    if( level=="bacteria"){
      to_remove<-
c("g__Corynebacterium","g__Mycobacterium", "g__Propionibacterium", "g__Staphylococcus")
      data<-removeDupes(to_remove, data)
    }

    if( level=="eukaryotes"){
      to_remove<-c("g__Malassezia","g__Ustilago", "g__Canis", "g__Homo")
      data<-removeDupes(to_remove, data)
    }

    # format data table
    row.names(data)<-data$Taxa
    data$Taxa<-NULL

    # if the data is for the eukaryotes, we only want to look at the fungal assignments
    if( level=="eukaryotes"){
        fungi_bas<-grep(pattern="p__Basidiomycota", x=row.names(data), fixed=FALSE)
        fungi_asc<-grep(pattern="p__Ascomycota", x=row.names(data), fixed=FALSE)
        fungi<-c(fungi_bas,fungi_asc)
        taxa_level<-data[fungi,]

        # look at fungal vs. nonfungal assignments for use with superkingdom analysis
        # fungi:
        eukaryota_level.t<-t(taxa_level)
        eukaryota_level2<-
merge(eukaryota_level.t,skinmet_metadata,by.x="row.names",by.y="SampleID")

        eukaryota_level_sum<-ddply(eukaryota_level2, c("Site_Symbol"), numcolwise(sum))
        row.names(eukaryota_level_sum)<-eukaryota_level_sum$Site_Symbol
        eukaryota_level_sum$Site_Symbol<-NULL
        eukaryota_level_sum<-eukaryota_level_sum[,order(-colSums(eukaryota_level_sum))]
        eukaryota_fungi<-rowSums(eukaryota_level_sum)

        # non-fungi
        eukaryota_level_other<-data[-fungi,]

        eukaryota_level_other.t<-t(eukaryota_level_other)
        eukaryota_level_other2<-
merge(eukaryota_level_other.t,skinmet_metadata,by.x="row.names",by.y="SampleID")
        eukaryota_level_other_sum<-
ddply(eukaryota_level_other2, c("Site_Symbol"), numcolwise(sum))
        row.names(eukaryota_level_other_sum)<-eukaryota_level_other_sum$Site_Symbol
        eukaryota_level_other_sum$Site_Symbol<-NULL
        eukaryota_non_fungi<-rowSums(eukaryota_level_other_sum)
        eukaryotes<-cbind(eukaryota_non_fungi, eukaryota_fungi)
        colnames(eukaryotes)<-c("sk__Eukaryota:Other","sk__Eukaryota:Fungi")
```

Step 7.

Write out to table.

cmd COMMAND

```
write.table(eukaryotes, "../../IntermediateOutput/MEGAN/eukaryote_counts.txt", row.names=FA
LSE, quote=FALSE, eol="\r\n", sep="\t")
    }
    else{
        taxa_level<-data
    }
```

Step 8.

Merge the data with the mapping file.

cmd COMMAND

```
taxa_level.t<-t(taxa_level)
    taxa_level2<-merge(taxa_level.t,skinmet_metadata,by.x="row.names",by.y="SampleID")
```

Step 9.

Combine samples from the same body sites.

cmd COMMAND

```
taxa_level_sum<-ddply(taxa_level2, c("Site_Symbol"), numcolwise(sum))
    # if doing super kingdom level analysis, break up eukaryotes into fungi vs. non-fungi
    if(level=="superkingdom")
    {
        sk_eukaryotes<-read.delim("../../IntermediateOutput/MEGAN/eukaryote_counts.txt")
        taxa_level_sum<-cbind(taxa_level_sum, sk_eukaryotes)
        taxa_level_sum$sk__Eukaryota<-NULL
        taxa_level_sum$Unassigned<-
rowSums(taxa_level_sum[,c("sk__Not_assigned","sk__other_sequences","sk__Low_complexity", "s
k__No_hits")])
        taxa_level_sum$sk__Not_assigned <-NULL
        taxa_level_sum$sk__other_sequences<-NULL
        taxa_level_sum$sk__Low_complexity<-NULL
        taxa_level_sum$sk__No_hits<-NULL
    }
    row.names(taxa_level_sum)<-taxa_level_sum$Site_Symbol
    taxa_level_sum$Site_Symbol<-NULL
```

Step 10.

Look at the top 10 taxa.

cmd COMMAND

```
taxa_level_sum<-topTaxa(taxa_level_sum,10)
    if(ncol(taxa_level_sum) < 10){
      taxa_level_sum<-taxa_level_sum[,order(-colSums(taxa_level_sum))]
    }
    taxa_order<-as.vector(colnames(taxa_level))
```

Step 11.

Convert counts into relative abundances.

cmd COMMAND

```
taxa_level_rel_abund<-(taxa_level_sum/rowSums(taxa_level_sum))*100
```

Step 12.

Format the data frame for use with ggplot2.

cmd COMMAND

```
taxa_level_rel_abund$Site<-row.names(taxa_level_rel_abund)
    colnames(taxa_level_rel_abund)<-
gsub(x=colnames(taxa_level_rel_abund), pattern=".*s__",replacement="", perl=TRUE)
    colnames(taxa_level_rel_abund)<-
gsub(x=colnames(taxa_level_rel_abund), pattern=".*g__",replacement="", perl=TRUE)
        colnames(taxa_level_rel_abund)<-
```

```
gsub(x=colnames(taxa_level_rel_abund), pattern="_<phylum>",replacement="", perl=TRUE)
    colnames(taxa_level_rel_abund)<-
gsub(x=colnames(taxa_level_rel_abund), pattern=".*sk__",replacement="", perl=TRUE)
    taxa<-melt(taxa_level_rel_abund, id=c("Site"))
    taxa_level_rel_abund$Site<-NULL
    taxa$varaible<-factor(taxa$variable, levels=taxa_order)
```

Step 13.

Plot.

cmd COMMAND

```
ggplot(taxa, aes(x=Site, y=value, fill=variable, order=variable))+theme_bw()+geom_bar(stat=
"identity")+guides(fill = guide_legend(reverse = TRUE))+scale_fill_manual(values = c("#e41a
1c", "#377eb8", "#33CCCC", "#4daf4a", "#984ea3", "#ff7f00", "#ffff33", "#a65628","#f781bf",
 "#999999"))
}
```

Step 14.

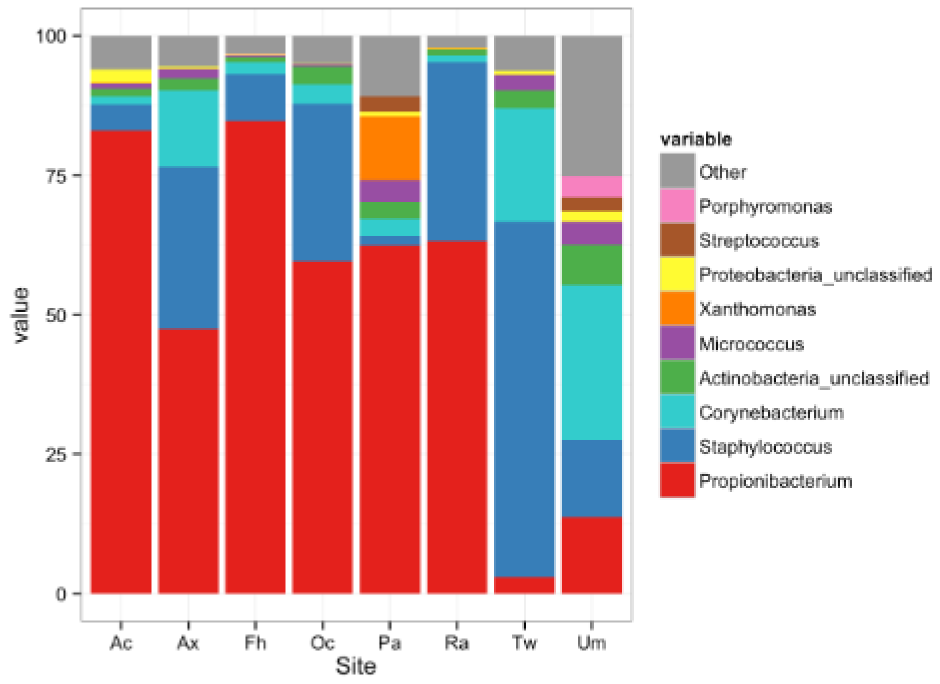
Plot the bacteria data.

cmd COMMAND

```
#setwd("formatted_output/genera")
#bact_genera=list.files("~/Desktop/SkinMet1/Updated_MEGAN_Figure/formatted_output/genera",
pattern="*_bacteria.txt")
#bact_data<-readData(bact_genera)
bact_data<-read.delim("../../IntermediateOutput/MEGAN/megan_genera_bacteria.txt")
plotTaxa("bacteria", bact_data)
```

The commented out lines are how we initially generated the data files that we now just read in.

EXPECTED RESULTS



Step 15.

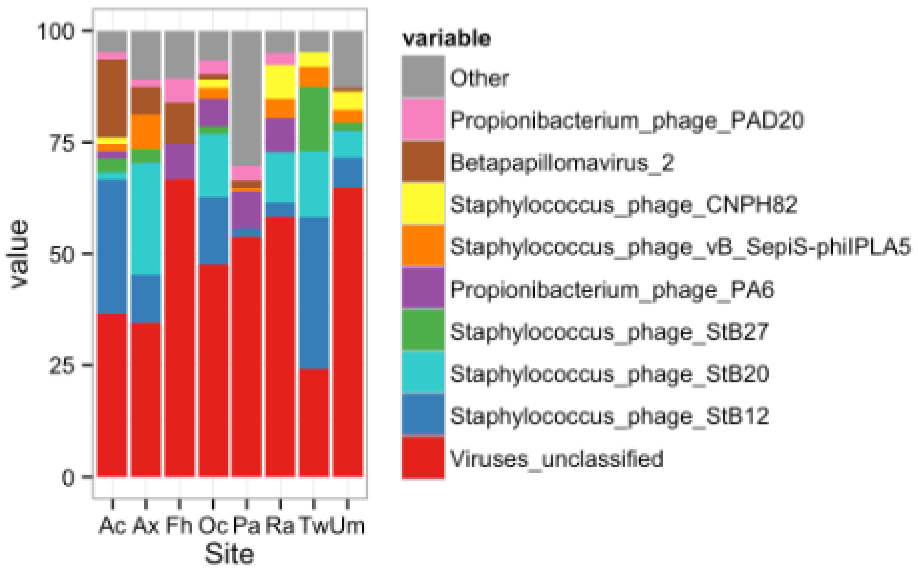
Plot the viruses.

cmd COMMAND

```
#viral_species=list.files("formatted_output/species", pattern="*_viruses.txt")
#viral_data<-readData(viral_species)
viral_data<-read.delim("../../IntermediateOutput/MEGAN/megan_species_viruses.txt")
plotTaxa("viruses", viral_data)
```

The commented out lines are how we initially generated the data files that we now just read in.

EXPECTED RESULTS



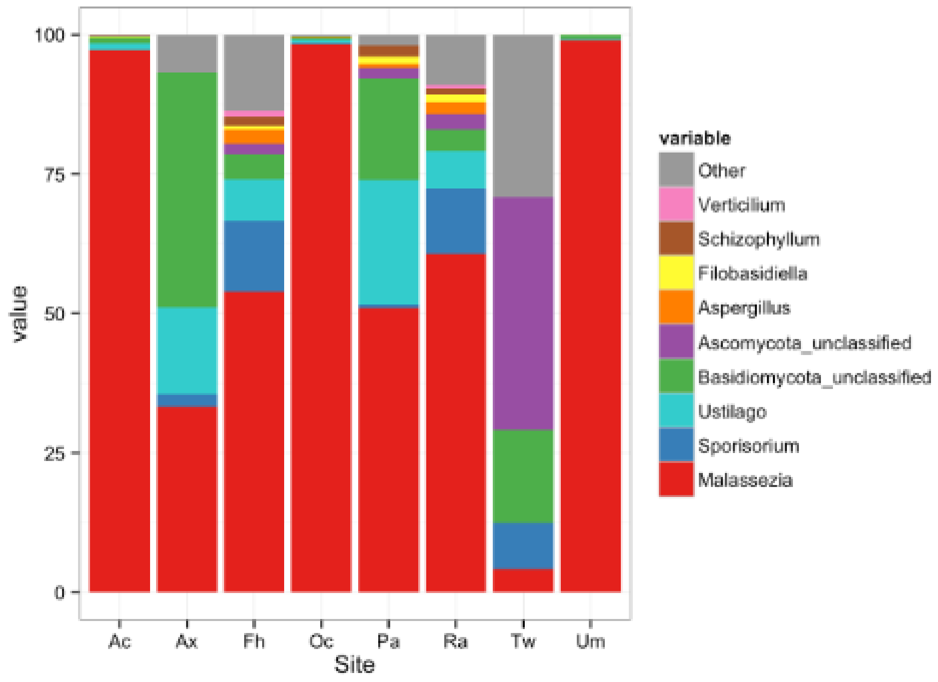
Step 16.
Plot the fungi.

COMMAND

```
#eukaryota_genera=list.files("formatted_output/genera", pattern="*_eukaryotes.txt")
#eukaryota_data<-readData(eukaryota_genera)
eukaryota_data<-read.delim("../../IntermediateOutput/MEGAN/megan_genera_eukaryotes.txt")
plotTaxa("eukaryotes", eukaryota_data)
```

The commented out lines are how we initially generated the data files that we now just read in.

EXPECTED RESULTS



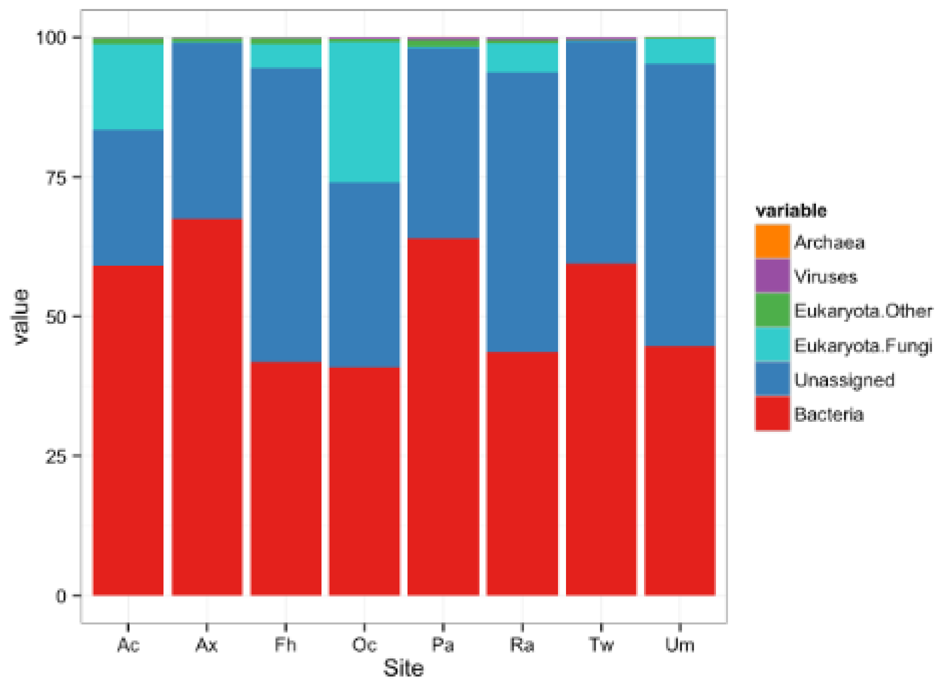
Step 17.
Finally plot an overview of everything.

COMMAND

```
#setwd("formatted_output/super_kingdom/")
#super_kingdom=list.files("formatted_output/super_kingdom/")
```

```
#sk_data<- readData(super_kingdom)
sk_data<-read.delim("../../IntermediateOutput/MEGAN/megan_sk.txt")
plotTaxa("superkingdom",sk_data)
```
The commented out lines are how we initially generated the data files that we now just read in.

EXPECTED RESULTS


value
Ac
Ax
Fh
Oc
Pa
Ra
Tw
Um
Site
variable
Archaea
Viruses
Eukaryota.Other
Eukaryota.Fungi
Unassigned
Bacteria