Script R13: OPF and AMG Analysis

HANNIGAN GD, GRICE EA, ET AL.

Abstract

This protocols outlines our definitions of core operational protein families (OPFs), potential auxiliary metabolic genes (AMGs), sharing of genes across anatomical sites, and Bray-Curtis dissimilarity of the OPFs by anatomical site. Based on 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 R13: OPF and AMG Analysis. **protocols.io** dx.doi.org/10.17504/protocols.io.ejkbckw
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(ggplot2)
packageVersion("ggplot2")

library(vegan)
packageVersion("vegan")

library(ggplot2)
packageVersion("ggplot2")

library(scatterplot3d)
packageVersion("scatterplot3d")
```

EXPECTED RESULTS

```
## [1] '1.0.1'



## [1] '2.3.0'



## [1] '1.0.1'



## [1] '0.3.35'
```

Step 2.

Import the needed data files and quantify the number of core operational protein families. Also write the names of the core OPFs so that I can query them and identify them after subsetting the input files.

cmd COMMAND

```
# Import the protein cluster relative abundance data frame
INPUT <-
 read.delim("../../IntermediateOutput/AuxilaryMetabolicGenes/contig_otu_table.txt", sep="\t
", header=TRUE)
MAP <-
 read.delim("../../IntermediateOutput/Mapping_files/SkinMet_and_Virome_001_metadata.tsv", s
ep="\t", header=TRUE)
MAP_ORDER <- MAP[order(MAP$NexteraXT_Virome_SampleID),]
MAP_SUBSET <- MAP_ORDER[-which(MAP_ORDER$TimePoint %in% 1), ]
MAP_SUBSET <- MAP_SUBSET[-which(MAP_SUBSET$NexteraXT_Virome_SampleID %in% NA), ]
MAP_SUBSET <- MAP_SUBSET[-which(MAP_SUBSET$Site_Symbol %in% c("Ba","Ph","Vf","Neg")), ]
MAP_SUBSET <- MAP_SUBSET[-which(MAP_SUBSET$SubjectID %in% c(2,3,9,11)), ]
```

Step 3.
Keep only those columns with values matching these found in the mapping file subset. This will get rid of the locations we are not evaluating.

cmd COMMAND
```
InputSubset <- INPUT[,which(colnames(INPUT) %in% MAP_SUBSET$NexteraXT_Virome_SampleID)]
InputSubsetWithNames <-
 INPUT[,c(1,which(colnames(INPUT) %in% MAP_SUBSET$NexteraXT_Virome_SampleID))]
rownames(InputSubsetWithNames) <- c(as.character(InputSubsetWithNames[,1]))
InputSubsetWithNames <- InputSubsetWithNames[,-1]
```

Step 4.
From here you can go to the diversity calculations below.

cmd COMMAND
```
TotalOrfs <- length(InputSubsetWithNames[,1])
# Remove the rows with any fields having a relative abundance of zero (gene not present)
# First go through and see what rows have zero values; see http://stackoverflow.com/questio
ns/9977686/how-to-remove-rows-with-a-zero-value-in-r
RowSubset <- apply(InputSubsetWithNames, 1, function(row) all(row !=0 ))
CoreAMG <- InputSubsetWithNames[RowSubset,]
TotalCoreOrfs <- length(CoreAMG[,1])
```

Step 5.
Write the names of the core protein clusters so that I can pull out the fasta sequences and query them against UniProt.

cmd COMMAND
```
write.table(rownames(CoreAMG), file="../../IntermediateOutput/AuxilaryMetabolicGenes/CoreAm
gOrfIds.tsv", quote=FALSE, sep='\t', col.names=FALSE, row.names=FALSE)
```

Step 6.
Determine how many genes are core to sites by location type.

cmd COMMAND
```
UniqSiteTypes <- factor(unique(MAP_SUBSET$Site_Categories))

for(x in UniqSiteTypes)
{
  MapSebaceous <- MAP_SUBSET[which(MAP_SUBSET$Site_Categories %in% x), ]
  InputSebaceous <-
 InputSubset[,which(colnames(InputSubset) %in% MapSebaceous$NexteraXT_Virome_SampleID)]
  RowSubset <- apply(InputSebaceous, 1, function(row) all(row !=0 ))
  SebaceousAMG <- InputSebaceous[RowSubset,]
  print(x)
  print(length(SebaceousAMG[,1]))
}
```

EXPECTED RESULTS
```
## [1] "Sebaceous"
## [1] 25
## [1] "Moist"
## [1] 15
## [1] "Intermittently_Moist"
## [1] 38
```

Step 7.
See number of core protein clusters by site.

cmd COMMAND
```
UniqOcc <- factor(unique(MAP_SUBSET$Occlusion))
for(x in UniqOcc)
{
```

```
  MapSebaceous <- MAP_SUBSET[which(MAP_SUBSET$Occlusion %in% x), ]
  InputSebaceous <-
 InputSubset[,which(colnames(InputSubset) %in% MapSebaceous$NexteraXT_Virome_SampleID)]
  RowSubset <- apply(InputSebaceous, 1, function(row) all(row !=0 ))
  SebaceousAMG <- InputSebaceous[RowSubset,]
  print(x)
  print(length(SebaceousAMG[,1]))
}
```

EXPECTED RESULTS

```
## [1] "Occluded"
## [1] 15
## [1] "Exposed"
## [1] 24
## [1] "Intermittently_Occluded"
## [1] 174
```

Step 8.

Try again for all of the different anatomical groups for plotting.

cmd COMMAND

```
UniqSites <- factor(unique(MAP_SUBSET$Site_Symbol))
anatomicalSiteDF <- c()
for(x in UniqSites) {
  MapSebaceous <- MAP_SUBSET[which(MAP_SUBSET$Site_Symbol %in% x), ]
  InputSebaceous <-
 InputSubset[,which(colnames(InputSubset) %in% MapSebaceous$NexteraXT_Virome_SampleID)]
  RowSubset <- apply(InputSebaceous, 1, function(row) all(row !=0 ))
  SebaceousAMG <- InputSebaceous[RowSubset,]
  anatomicalSiteDF <- rbind(anatomicalSiteDF,c(x,length(SebaceousAMG[,1])))
}
anatomicalSiteDF <- as.data.frame(anatomicalSiteDF)
as.numeric(as.character(anatomicalSiteDF[,2]))
```

EXPECTED RESULTS

```
## [1]  72 114  64 494  41  15  45 174
```

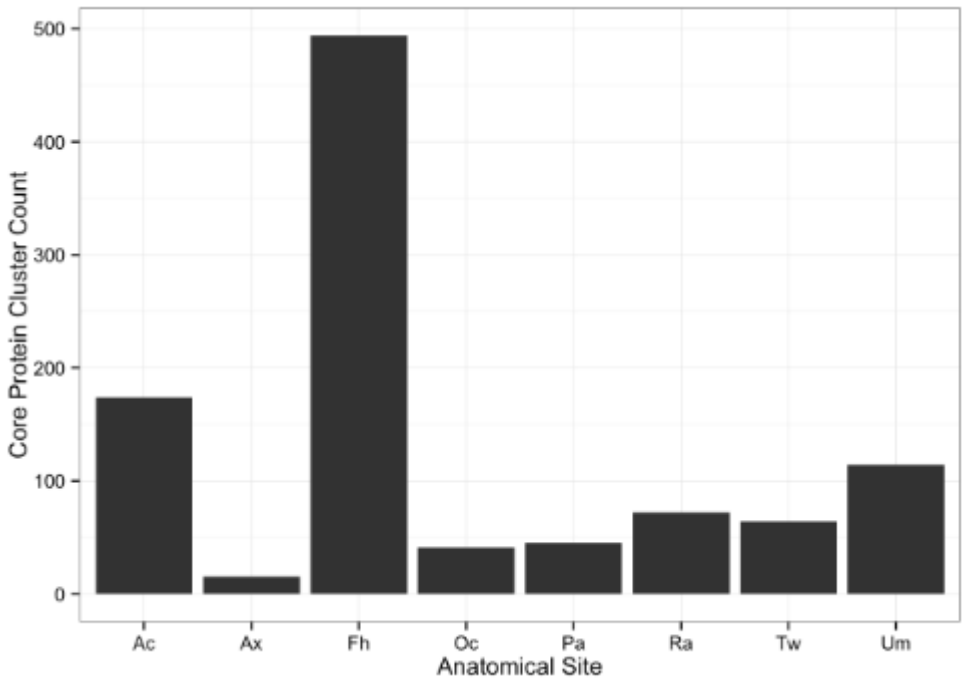
Step 9.

Plot the protein cluster count.

cmd COMMAND

```
AmgSitePlot <-
 ggplot(anatomicalSiteDF, aes(x=V1, y=as.numeric(as.character(V2)))) + theme_bw() + geom_ba
r(stat="identity") + xlab("Anatomical Site") + ylab("Core Protein Cluster Count")
AmgSitePlot
```

EXPECTED RESULTS



Step 10.

Finally we calculated the diversity of ORFs by different anatomical sites. Transpose the input file.

cmd COMMAND

```
InputT <- data.frame(t(InputSubset))
INPUT_SUBSET_DIST_MATRIX <- vegdist(InputT, method = "bray")
```

Step 11.

Visualize the distance matrix using NMDS.

cmd COMMAND

```
BRAY_ORD_NMDS <- metaMDS(INPUT_SUBSET_DIST_MATRIX,k=3)

BRAY_ORD_FIT = data.frame(MDS1 = BRAY_ORD_NMDS$points[,1], MDS2 = BRAY_ORD_NMDS$points[,2],
 MDS3 = BRAY_ORD_NMDS$points[,3])
#Record the stress value
BRAY_ORD_NMDS$stress

BRAY_ORD_FIT$SampleID <- rownames(BRAY_ORD_FIT)
NMDS_AND_MAP <-
 merge(BRAY_ORD_FIT, MAP_SUBSET, by.x="SampleID", by.y="NexteraXT_Virome_SampleID")
```

EXPECTED RESULTS

```
## Run 0 stress 0.142143
## Run 1 stress 0.1460666
## Run 2 stress 0.1438215
## Run 3 stress 0.1435173
## Run 4 stress 0.1408786
## ... New best solution
## ... procrustes: rmse 0.03486057 max resid 0.3573651
## Run 5 stress 0.1429197
## Run 6 stress 0.1436577
## Run 7 stress 0.1439074
## Run 8 stress 0.1473716
## Run 9 stress 0.1443224
## Run 10 stress 0.1431153
```

```
## Run 11 stress 0.147699
## Run 12 stress 0.1447852
## Run 13 stress 0.1427725
## Run 14 stress 0.1455643
## Run 15 stress 0.1470337
## Run 16 stress 0.1446069
## Run 17 stress 0.1431871
## Run 18 stress 0.1442179
## Run 19 stress 0.146792
## Run 20 stress 0.1463412

## [1] 0.1408786
```

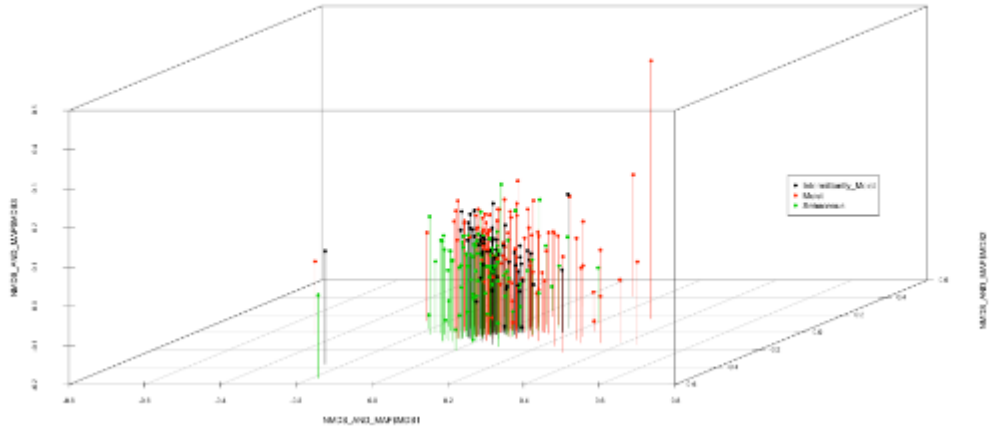
Step 12.
Plot the data.

cmd COMMAND

```
s3d <-
 scatterplot3d(NMDS_AND_MAP$MDS1,NMDS_AND_MAP$MDS2,NMDS_AND_MAP$MDS3, pch=16, color=as.inte
ger(factor(NMDS_AND_MAP$Site_Categories)), type="h")
legend('right', pch = 16,legend = levels(factor(NMDS_AND_MAP$Site_Categories)),  col = seq_
along(levels(NMDS_AND_MAP$Site_Categories)), inset=c(0.1,0))

adonis(INPUT_SUBSET_DIST_MATRIX ~ factor(NMDS_AND_MAP$Site_Categories), perm=999, strata =
factor(NMDS_AND_MAP$SubjectID))
```

EXPECTED RESULTS



Step 13.
Plot by occlusion site status.

cmd COMMAND

```
s3d <-
 scatterplot3d(NMDS_AND_MAP$MDS1,NMDS_AND_MAP$MDS2,NMDS_AND_MAP$MDS3, pch=16, color=as.inte
ger(factor(NMDS_AND_MAP$Occlusion)), type="h")
legend('right', pch = 16,legend = levels(factor(NMDS_AND_MAP$Occlusion)),  col = seq_along(
levels(NMDS_AND_MAP$Occlusion)), inset=c(0.1,0))

adonis(INPUT_SUBSET_DIST_MATRIX ~ factor(NMDS_AND_MAP$Occlusion), perm=999, strata = factor
(NMDS_AND_MAP$SubjectID))
```

EXPECTED RESULTS