Compendium Transfer: Data Accession Tutorial

Daniel Chawla

2022-12-27

Contents

outstanding tasks:

[!] figure 6 github plz

[-] shiny app add citation information, ? button, toupper for symbols

[–>] GEO pipeline metadata

```
Every dataset should have a metadata field called "geo_accession" containing a gsm identifier.
If a dataset does NOT have this field, it was manually re-processed.

  [ ] pull processed files from GEO
  [ ] check gsm column (if NULL, then processed locally), subset data to matching samples
  [ ] check spearman correlation between matched gsms (if == 1, GEO, else, local)

If data processing matters, you can always pull processed GEO, match the geo_accession column with
```

[ ] Chapter —2: Where is everything??? [ ] a: Include links to GitHubs and anywhere where there might be relevant code. a map of resources spread out [ ] b: excel sheets mentioned in chapter 0

[ ] move metaintegrator discussion to section 0 or —1

Chapter —2: Other Code & Where 2 Find It

Manuscript Code

Shiny App

Misc. Pre-Processing / Other

Google Drive with data itself

Chapter —1: Preliminaries and helper functions:

```
library(MetaIntegrator)
library(dplyr)
```

```
##
## Attaching package: 'dplyr'
```

```
## The following objects are masked from 'package:stats':
##
##     filter, lag
```

```
## The following objects are masked from 'package:base':
##
##     intersect, setdiff, setequal, union
```

```
library(openxlsx)
knitr::opts_chunk$set(echo = TRUE)

# This function takes a MetaIntegrator object and logical index vector indicating
# which samples to keep/remove, and returns a MetaIntegrator object containing
# the indicated subset
filterMIobj <- function(MIobj, index){
  tmpMIobj <- MIobj
  tmpMIobj$expr <- tmpMIobj$expr[, index] # filter expression
  tmpMIobj$pheno <- tmpMIobj$pheno[index, ] # filter pheno metadata
  tmpMIobj$class <- tmpMIobj$class[index] # filter class vector

  if(checkDataObject(tmpMIobj, 'Dataset')){ # verify that the resulting object is still a MetaIntegrato
    return(tmpMIobj)
  } else {
    stop()
  }
}
```

Chapter 0: Accessing study and sample metadata

Study-level metadata is available in two tables: 1) darpa_compendium_metadata_v1 contains metadata for all studies used in the manuscript 2) DARPA_originalmetadata_transfer contains metadata from GEO searches, meaning this document contains metadata for studies which may be of interest but ultimately did not meet inclusion criteria

We begin by loading the dataset-level metadata table, available in the Drive transfer folder:

```
study_metadata <- read.xlsx('darpa_compendium_metadata_v1.xlsx', sheet = 1)
printable_df <- study_metadata %>%
  dplyr::select(Accession, Platform, Standardized.Exposure, Classes, Control.Type, Age)
head(printable_df)
```

```
##   Accession Platform
## 1 GSE101702 GPL21185
## 2 GSE103119 GPL10558
## 3 GSE103842 GPL10558
## 4 GSE110106 GPL10558
## 5 GSE110106 GPL10558
## 6   GSE1124    GPL96
##
## 1
## 2 Adenovirus; Bocavirus; Coronavirus; Human metapneumovirus; Human parainfluenza virus; Mycoplasma;
## 3
## 4
## 5
## 6
##                              Classes Control.Type      Age
## 1                     Healthy; Virus      Healthy    Adult
## 2 Bact+Vir; Bacteria; Healthy; Virus      Healthy Children
## 3                     Healthy; Virus      Healthy  Infants
## 4                  Bacteria; Healthy      Healthy    Adult
## 5                  Bacteria; Healthy      Healthy    Adult
## 6          Healthy; Other.Infectious         <NA>     <NA>
```

This resource contains information about exposures, classes (e.g., Virus, Bacteria, etc.)

Note: the originalmetadata resource is the original metadata annotation and pathogen information contained here has NOT been thoroughly cross-checked. For the highest quality metadata mapping, please refer to the metadata_v1 resource or the sample metadata section below.

0.1 Sample Infection/Class Metadata

We begin by reading the data_list.RDS object, which contains an R list of MetaIntegrator objects.

```
data_list <- readRDS('data_list_v1.RDS')
MIobj <- data_list[[1]]
sapply(MIobj, class)
```

```
## $expr
## [1] "matrix" "array"
##
## $keys
## [1] "character"
##
## $pheno
## [1] "data.frame"
##
## $class
## [1] "numeric"
##
## $formattedName
## [1] "character"
```

Sample-level information information is contained in the Class and Pathogen fields of `$pheno` $Class contains the following categories: {Healthy, Convalescent, Virus, Bacteria, Other.Infectious, Other.Non-Infectious, Vir+Bact}

```
head(MIobj$pheno) %>% dplyr::select(geo_accession, Class)
```

```
##            geo_accession Class
## GSM2712389    GSM2712389 Virus
## GSM2712390    GSM2712390 Virus
## GSM2712391    GSM2712391 Virus
## GSM2712392    GSM2712392 Virus
## GSM2712393    GSM2712393 Virus
## GSM2712394    GSM2712394 Virus
```

```
table(MIobj$pheno$Class)
```

```
##
## Healthy   Virus
##      52     107
```

$Pathogen contains information about the underlying pathogen when available. "Healthy" is included in the Pathogen column to enable table(Class, Pathogen) cross-checking

```
head(MIobj$pheno) %>% dplyr::select(geo_accession, Pathogen)
```

```
##            geo_accession        Pathogen
## GSM2712389    GSM2712389 Influenza virus
## GSM2712390    GSM2712390 Influenza virus
## GSM2712391    GSM2712391 Influenza virus
## GSM2712392    GSM2712392 Influenza virus
## GSM2712393    GSM2712393 Influenza virus
## GSM2712394    GSM2712394 Influenza virus
```

```
table(MIobj$pheno$Pathogen)
```

```
##
##         Healthy Influenza virus
##              52             107
```

```
table(MIobj$pheno$Class, MIobj$pheno$Pathogen)
```

```
##
##           Healthy Influenza virus
##   Healthy      52               0
##   Virus         0             107
```

0.2 Time-Series Sample Metadata

Time-series studies are identifiable by the presence of SubjectID and Time.Point fields within $pheno. These fields indicate which samples came from each subject and at what time point. Time.Point is an ordered factor. Note: for GSE73072, "Symptoms" is a logical vector indicating whether subjects developed symptoms. For all other compendium studies, this information was not readily availabile.

```
MIobj2 <- data_list$GSE61754_GPL10558
table(MIobj2$pheno$SubjectID, MIobj2$pheno$Time.Point)
```

```
##
##            Pre-Exposure 12 hours 24 hours 48 hours
```

```
##   subjectL            1        1        1        1
##   subjectM            1        1        1        1
##   subjectN            1        1        1        1
##   subjectO            1        1        1        1
##   subjectP            1        1        1        1
##   subjectQ            1        1        1        1
##   subjectR            1        1        1        1
##   subjectS            1        1        1        1
##   subjectT            1        1        1        1
##   subjectU            1        1        1        1
##   subjectV            1        1        1        1
```

0.3 Additional Sample Metadata

Additional metadata was used throughout the manuscript that is not provided in the data_list file. This information includes metadata describing cohort age groups, bacterial cell wall characteristics, viral envelope/genome characteristics, and acute/chronic exposure timing.

Cohort age groups can be found within `darpa_compendium_metadata_v1.xlsx` (the master study-level metadata table) for studies where this information was available. Bacterial and viral pathogen characteristics can be easily annotated using the `$pheno$Pathogen` field. The annotation versions used for the manuscript can also be found within `bacterial_acute_annotation.xlsx` and `annotated_pathogen_df.xlsx` Acute/chronic labeling was only applicable to viral infections, and dataset labels can be found in `viral_timing_annotations.xlsx`.

0.4 Expression matrix processing metadata

Most datasets within the compendium were reprocessed from raw files. Expression matrices were taken directly from GEO if one of the following criteria were met: i) no gene annotation file provided ii) data came from an uncommon platform (see manuscript list) iii) the authors indicated that batch correction was part of their pre-processing pipeline

Here, we infer which datasets were reprocessed vs. taken from GEO by correlating expression based on matched geo_accession

```
# There should be a data.frame indicating which pipeline a dataset came through: is it from GEO or from
# Quantile normalized?
```

Note: datasets taken from GEO may not be quantile normalized, but this normalization is part of the neqc and rma methods used for data reprocessing.

Note: for compatbility with MetaIntegrator and geometric mean scoring, all datasets should be log-normalized and shifted to a minimum expression value of 1.

Chapter 1: Slicing and dicing the compendium

This section contains information about accessing segments of the compendium. For example, you may be interested in studies profiling viral infection, or infections caused by pathogens.

1.A Identifying classes of interest

Viral and bacterial infection datasets are stored together in the data_list object, and can be accessed programmatically by selecting studies that contain a minimum number of samples matching a $Class value:

```r
getClassSum <- function(MIobj, input_class, minimum_N = 4){
  class_match <- grepl(pattern = input_class, x = MIobj$pheno$Class)
  return(ifelse(sum(class_match) > minimum_N, T, F))
}

bacterial_study_ix <- sapply(data_list, getClassSum, 'Bacteria')
head(bacterial_study_ix)
```

```
##   GSE101702_GPL21185   GSE103119_GPL10558   GSE103842_GPL10558
##                FALSE                 TRUE                FALSE
## GSE110106_GPL10558_1 GSE110106_GPL10558_2        GSE1124_GPL96
##                 TRUE                 TRUE                FALSE
```

```r
first_bacterial_study <- which(bacterial_study_ix)[1]
MIobj3_pheno <- data_list[[first_bacterial_study]]$pheno #%>% filter(Class %in% c('Bacteria', 'Healthy'
table(MIobj3_pheno$Class, MIobj3_pheno$Pathogen)
```

```
##
##           Adenovirus;Bocavirus;Coronavirus Adenovirus;Mycoplasma;Rhinovirus
##   Bact+Vir                                0                                1
##   Bacteria                                0                                0
##   Healthy                                 0                                0
##   Virus                                   1                                0
##
##           Adenovirus;Respiratory syncytial virus Adenovirus;Rhinovirus
##   Bact+Vir                                      0                     0
##   Bacteria                                      0                     0
##   Healthy                                       0                     0
##   Virus                                         1                     5
##
##           Adenovirus;Streptococcus pneumoniae
##   Bact+Vir                                   1
##   Bacteria                                   0
##   Healthy                                    0
##   Virus                                      0
##
##           Bocavirus;Human parainfluenza virus;Rhinovirus Bocavirus;Rhinovirus
##   Bact+Vir                                              0                    0
##   Bacteria                                              0                    0
##   Healthy                                               0                    0
##   Virus                                                 1                    2
##
##           Coronavirus Coronavirus;Respiratory syncytial virus Healthy
##   Bact+Vir           0                                      0       0
##   Bacteria           0                                      0       0
##   Healthy            0                                      0      38
##   Virus              1                                      1       0
##
##           Human metapneumovirus;Respiratory syncytial virus;Rhinovirus
##   Bact+Vir                                                            0
##   Bacteria                                                            0
##   Healthy                                                             0
##   Virus                                                               1
##
```

```
##           Human metapneumovirus;Rhinovirus
##   Bact+Vir                                0
##   Bacteria                                0
##   Healthy                                 0
##   Virus                                  12
##
##           Human metapneumovirus;Rhinovirus;Streptococcus pneumoniae
##   Bact+Vir                                                        2
##   Bacteria                                                        0
##   Healthy                                                         0
##   Virus                                                           0
##
##           Human metapneumovirus;Rhinovirus;Streptococcus pyogenes
##   Bact+Vir                                                      3
##   Bacteria                                                      0
##   Healthy                                                       0
##   Virus                                                         0
##
##           Human parainfluenza virus Human parainfluenza virus;Mycoplasma
##   Bact+Vir                         0                                     2
##   Bacteria                         0                                     0
##   Healthy                          0                                     0
##   Virus                            5                                     0
##
##           Human parainfluenza virus;Respiratory syncytial virus
##   Bact+Vir                                                    0
##   Bacteria                                                    0
##   Healthy                                                     0
##   Virus                                                       1
##
##           Human parainfluenza virus;Respiratory syncytial virus;Rhinovirus;Staphylococcus aureus
##   Bact+Vir                                                                                     1
##   Bacteria                                                                                     0
##   Healthy                                                                                      0
##   Virus                                                                                        0
##
##           Human parainfluenza virus;Rhinovirus
##   Bact+Vir                                   0
##   Bacteria                                   0
##   Healthy                                    0
##   Virus                                      3
##
##           Human parainfluenza virus;Staphylococcus aureus
##   Bact+Vir                                              1
##   Bacteria                                              0
##   Healthy                                               0
##   Virus                                                 0
##
##           Human parainfluenza virus;Streptococcus pneumoniae Mycoplasma
##   Bact+Vir                                                  1          0
##   Bacteria                                                  0         30
##   Healthy                                                   0          0
##   Virus                                                     0          0
##
```

```
##            Mycoplasma;Rhinovirus Respiratory syncytial virus
##   Bact+Vir                     8                           0
##   Bacteria                     0                           0
##   Healthy                      0                           0
##   Virus                        0                          11
##
##            Respiratory syncytial virus;Rhinovirus Rhinovirus
##   Bact+Vir                                       0          0
##   Bacteria                                       0          0
##   Healthy                                        0          0
##   Virus                                          2         30
##
##            Rhinovirus;Streptococcus pneumoniae Streptococcus pneumoniae
##   Bact+Vir                                   2                         0
##   Bacteria                                   0                         4
##   Healthy                                    0                         0
##   Virus                                      0                         0
##
##            Streptococcus pyogenes
##   Bact+Vir                      0
##   Bacteria                      1
##   Healthy                       0
##   Virus                         0
```

Note: searching for one class does not preclude the inclusion of other classes in selected datasets. The $class vector must be manually set to {0, 1} for {control, case} respectively. helperFunctions.R from the compendium GitHub contains a function "filterMIobj" that accepts a logical vector to easily subset MIobj datasets.

Note: semicolons indicate co-infections with multiple pathogens.

1.B Identifying pathogens of interest

```
getPathogenSum <- function(MIobj, input_class, minimum_N = 4){
  path_match <- grepl(pattern = input_class, x = MIobj$pheno$Pathogen)
  return(ifelse(sum(path_match) > minimum_N, T, F))
}

influenza_study_ix <- sapply(data_list, getPathogenSum, 'flu')
head(influenza_study_ix)
```

```
##   GSE101702_GPL21185   GSE103119_GPL10558   GSE103842_GPL10558
##                 TRUE                 TRUE                FALSE
## GSE110106_GPL10558_1 GSE110106_GPL10558_2        GSE1124_GPL96
##                FALSE                FALSE                FALSE
```

```
first_flu_study <- which(influenza_study_ix)[10]
MIobj4_pheno <- data_list[[first_flu_study]]$pheno
table(MIobj4_pheno$Class, MIobj4_pheno$Pathogen)
```

```
##
##            Escherichia coli Healthy Influenza virus Staphylococcus aureus
##   Bacteria               29       0               0                    31
##   Healthy                 0       6               0                     0
##   Virus                   0       0              18                     0
```

```
##
##            Streptococcus pneumoniae
##   Bacteria                       13
##   Healthy                         0
##   Virus                           0
```

The same logic for `$Class` applies to querying individual pathogens in `$Pathogen`.

Note, regular expression searches may not always be specific. In the example above, `flu` will return `Influenza virus` as well as `Human parainfluenza virus` and `Haemophilus influenzae`. In some cases, confirming against `$pheno$Class` will resolve unintended selections, but in other cases (as for parainfluenza virus) labels should be carefully reviewed.

1.C Evaluating non-viral/non-bacterial conditions

Evaluation of non-infectious conditions, as well as parasitic infections, requires datasets that are not stored in `data_list_v1.RDS`. Datasets profiling parasitic infections, aging, obesity, and others can be found in `parasite_list.RDS`, `aging_list.RDS`, `obesity_list.RDS`, and `noninfectious_data_list.RDS` respectively.

```
your_data_dir <- '~/Documents/darpa-manuscript-data/data_list_final/'
aging_list <- readRDS(paste0(your_data_dir, 'aging_list.RDS'))
obesity_list <- readRDS(paste0(your_data_dir, 'obesity_list.RDS'))
parasite_list <- readRDS(paste0(your_data_dir, 'parasite_list.RDS'))
noninfectious_list <- readRDS(paste0(your_data_dir, '../noninfectious_data_list.RDS'))

paste0('Aging studies: ', length(aging_list))
```

```
## [1] "Aging studies: 6"
```

```
paste0('Obesity studies: ', length(obesity_list))
```

```
## [1] "Obesity studies: 8"
```

```
paste0('Parasite studies: ', length(parasite_list))
```

```
## [1] "Parasite studies: 13"
```

```
paste0('All noninfectious studies: ', length(noninfectious_list))
```

```
## [1] "All noninfectious studies: 34"
```

Note: `noninfectious_list.RDS` is a superset of aging, obesity, and all comorbidities curated for the COVID-19 project including COPD, pollution, etc.

```
all(names(aging_list) %in% names(noninfectious_list))
```

```
## [1] TRUE
```

```
all(names(obesity_list) %in% names(noninfectious_list))
```

```
## [1] TRUE
```

Condition information can be found in `$pheno$Standardized.Exposure` for each noninfectious study:

```
lapply(noninfectious_list, function(x){
  data.frame('Study' = x$formattedName, 'Condition' = unique(x$pheno$Standardized.Exposure))
}) %>%
  bind_rows() %>%
  arrange(Condition)
```

```
##                 Study                Condition
## 1  GSE101710_GPL10558                    Aging
## 2   GSE59635_GPL10558                    Aging
## 3   GSE59654_GPL10558                    Aging
## 4   GSE59743_GPL10558                    Aging
## 5    GSE63117_GPL7020                    Aging
## 6   GSE65219_GPL15988                    Aging
## 7   GSE83864_GPL10558            Air Pollution
## 8   GSE69683_GPL13158                   Asthma
## 9     GSE42057_GPL570                     COPD
## 10    GSE54837_GPL570                     COPD
## 11    GSE56766_GPL570                     COPD
## 12  GSE94916_GPL20844                     COPD
## 13   GSE20680_GPL4133 Coronary Artery Disease
## 14   GSE20681_GPL4133 Coronary Artery Disease
## 15  GSE131793_GPL6244             Hypertension
## 16   GSE19617_GPL6480             Hypertension
## 17    GSE22356_GPL570             Hypertension
## 18   GSE33463_GPL6947             Hypertension
## 19  GSE38267_GPL13607             Hypertension
## 20       GSE703_GPL80             Hypertension
## 21   GSE110551_GPL570                  Obesity
## 22    GSE18897_GPL570                  Obesity
## 23   GSE41233_GPL6947                  Obesity
## 24  GSE53232_GPL11532                  Obesity
## 25  GSE55205_GPL10558                  Obesity
## 26  GSE56960_GPL13667                  Obesity
## 27  GSE69039_GPL10558                  Obesity
## 28  GSE83223_GPL10558                  Obesity
## 29   GSE23323_GPL6480                  Smoking
## 30  GSE68526_GPL10904                  Smoking
## 31    GSE87072_GPL570                  Smoking
## 32  GSE23561_GPL10775         Type 2 Diabetes
## 33  GSE69528_GPL10558         Type 2 Diabetes
## 34   GSE87005_GPL6480         Type 2 Diabetes
```

1.D Adding datasets to the compendium

Datasets in the compendium follow the MetaIntegrator format, with additional `$pheno` columns to improve usability. A vignette for this package including a detailed description of this data format can be found here: https://cran.r-project.org/web/packages/MetaIntegrator/vignettes/MetaIntegrator.html

Briefly, each dataset is represented by a named list containing the following objects:

1. formattedName: a string with the formal name, usually GSE#_GPL#, where # indicates an integer. GSE# specifies GEO accession, and GPL# specifies GEO platform accession. Names may be followed by another underscore indicating cohorts or subsets from the original study
2. expr: a log-transformed expression matrix (minimum value should be 0), where column names indicate samples and row names indicate genes
3. pheno: a metadata table with rownames indicating samples. these names must match the column names of expr The additional `$pheno` columns necessary for consistent usage within the compendium are as follows: Class: a required field indicating one of {Virus, Bacteria, Other.NonInfectious, Other.Infectious, Healthy, Convalescent, Vir+Bact} for each sample Pathogen: a vector containing information about

the underlying causative pathogen, if available. Pathogen names are semi-standardized (non-ontology mapped, but should be unique within the compendium)

For time-series studies SubjectID: a vector indicating which samples came from the each subject over time Time.Point: an ordered factor indicating the chronological order of time points

4. keys: a named vector mapping rownames (genes) in `$expr` to symbols. Within the compendium, we use ENTREZ IDs for both key values. Studies should be max summarized
5. class: a binary vector indicating whether a sample belongs to case or control group. Names should match `$pheno` rownames. These values change depending on robustness/cross-reactivity evaluations

Note: The MetaIntegrator package includes a `checkDatasetObject` function to determine if data has been formatted correctly. Note2: GEOQuery allows you to pull datasets from GEO. MetaIntegrator has a formatting function that may be of interest.

Chapter 2: Balancing metadata for ML tasks

In the manuscript, we describe manually splitting datasets by metadata including platform, age, tissue, sample size, and other factors. This section contains a discussion of factors to consider, as well as my recommended best (most efficient) practices.

The goal of balancing is to ensure approximately even representation of various technical factors across discovery and validation splits. For platform, this can be considered by both manufacturer (Illumina vs. Affymetrix vs. Other), as well as the platform year (which should also be confounded by dataset publication year). Datasets may profile adults, children, or a mixture of age groups, and samples may have been derived from either whole blood or PBMCs (which may impact, for example, neutrophil gene expression measurements). Sample sizes will vary depending on the samples selected per dataset (e.g., if you want all viral samples or just one samples profiling one virus). Other factors to consider include geographic information about sample collection (indicative of exposure history + genetic background), sex ratio within cohorts, and clinical metadata including infection severity. Please note that these 'other factors' are NOT annotated across the compendium.

!!Intangible Experience Alert!! DO NOT MANUALLY BALANCE METADATA FOR ML TASKS THAT USE MORE THAN ~8 DATASETS For influenza-prediction tasks, we found that randomly assigning datasets and manually checking balance was MUCH FASTER than manual assignments

!!Intangible Experience Alert!! Some data accessions are part of a GEO SuperSeries, meaning they come from the same authors/labs. GEO Accessions that are continuous integers are highly suspect and should be verified. In practice, you can also identify datasets from similar labs by checking for metadata column name similarities (matches above 80% should be inspected).

Chapter 3: Evaluating datasets using MetaIntegrator, e.g. MRSA/MSSA

The evaluation framework borrows heavily from the MetaIntegrator package. Data formats and analytical methods are consistent with those described in Haynes et al. 2015. A vignette for this package can be found here: https://cran.r-project.org/web/packages/MetaIntegrator/vignettes/MetaIntegrator.html

The following section describes wrapper functions for evaluating the compendium using this package.

To benchmark a signature for a particular pathogen, we need to

i) Identify samples profiling responses to the pathogen (e.g., MRSA)

ii) Format for calculations

iii) Identify samples profiling other pathogens (e.g., other bacteria, viruses, etc.)

iiia) Set the class vectors in each MetaIntegrator object to reflect the desired contrast (e.g., MRSA vs. healthy, non-MRSA bacteria vs. healthy, etc.)

iiib) Filter based on sample sizes

iv) utilize MetaIntegrator functionality to compute signature scores and AUROCs

3.A Formatting datasets for each evaluation

i) Identify samples profiling responses to the pathogen (e.g., MRSA)

```
# for an input MIobj and  pathogen (character)
# this function returns a logical vector indicating whether a dataset
# contains the pathogen of interest
findPathogen <- function(MIobj, pathogen){
  pathogen_vector <- MIobj$pheno$Pathogen
  pathogen_vector <- pathogen_vector[!is.na(pathogen_vector)]
  if(sum(pathogen_vector == pathogen) > 0){
    return(T)
  } else {
    return(F)
  }
}

# identify datasets with staph infections
staph_ix <- sapply(data_list, findPathogen, pathogen = 'Staphylococcus aureus')
print(which(staph_ix))
```

```
##    GSE11908_GPL96    GSE11908_GPL97  GSE13015_GPL6106  GSE13015_GPL6947
##                13                14                16                17
##    GSE16129_GPL96  GSE30119_GPL6947 GSE40396_GPL10558   GSE6269_GPL2507
##                18                42                58                82
##    GSE6269_GPL570     GSE6269_GPL96 GSE69528_GPL10558    GSE6377_GPL201
##                83                84                94               124
```

```
# subset the full data_list down to relevant studies
staph_list <- data_list[staph_ix]
```

ii) format for calculations

```
# for an input MIobj and pathogen (character),
# this function returns the subset of subjects infected by the pathogen
# as well as healthy controls, with a modified $class vector for use with
# MetaIntegrator functions
formatPathogenContrast <- function(MIobj, pathogen){
  #print(MIobj$formattedName)
  keep_ix <- !is.na(MIobj$pheno$Pathogen) & !is.na(MIobj$pheno$Class)
  if(sum(keep_ix) == 0){print(paste0(MIobj$formattedName, ' has no valid samples')); return(NULL)}
  MIobj <- filterMIobj(MIobj, keep_ix)

  keep_ix <- MIobj$pheno$Pathogen == pathogen | MIobj$pheno$Class %in% c("Healthy", "Convalescent")
  if(sum(keep_ix) == 0){print(paste0(MIobj$formattedName, ' has no valid samples')); return(NULL)}
  MIobj <- filterMIobj(MIobj, keep_ix)
  contrast_vector <- as.numeric(!MIobj$pheno$Class %in% c('Healthy', 'Convalescent'))
  names(contrast_vector) <- rownames(MIobj$pheno)
  MIobj$class <- contrast_vector
```

```
  return(MIobj)
}

# format MIobjs
staph_list <- lapply(staph_list, formatPathogenContrast, pathogen = 'Staphylococcus aureus')

# filter out studies that do not have at least N cases and N controls
staph_N = 4
# return a logical indicating whether MIobj contains at least N cases and N controls
filterSampleSizes <- function(MIobj, N){
  if(is.null(MIobj)){return(F)}
  keep_ix <- !is.na(MIobj$pheno$Class) & !is.na(MIobj$class)
  MIobj <- filterMIobj(MIobj, keep_ix)
  return(length(unique(MIobj$class)) == 2 & min(table(MIobj$pheno$Class)) >= N)
}
sample_ix <- sapply(staph_list, filterSampleSizes, N = staph_N)
sample_ix
```

```
##    GSE11908_GPL96    GSE11908_GPL97  GSE13015_GPL6106  GSE13015_GPL6947
##             FALSE             FALSE             FALSE             FALSE
##    GSE16129_GPL96  GSE30119_GPL6947 GSE40396_GPL10558   GSE6269_GPL2507
##              TRUE              TRUE              TRUE             FALSE
##    GSE6269_GPL570     GSE6269_GPL96 GSE69528_GPL10558    GSE6377_GPL201
##             FALSE              TRUE              TRUE              TRUE
```

```
staph_list <- staph_list[sample_ix]
```

iii) Identify samples profiling other pathogens (e.g., other bacteria, viruses, etc.)

To evaluate the performance of a signature using samples which do NOT profile the pathogen of interest, we can remove the pathogen of interest from the compendium

```
# for an input MIobj, pathogen (character), and target_class (character)
# this function returns the subset of subjects with infecetions of type target_class that are NOT infec
# as well as healthy controls, with a modified $class vector for use with
# MetaIntegrator functions
removePathogen <- function(MIobj, pathogen, target_class){
  #print(MIobj$formattedName)
  keep_ix <- !is.na(MIobj$pheno$Class)
  if(sum(keep_ix) == 0){print(paste0(MIobj$formattedName, ' has no valid samples')); return(NULL)}
  MIobj <- filterMIobj(MIobj, keep_ix)

  keep_ix <- MIobj$pheno$Pathogen != pathogen & MIobj$pheno$Class %in% c("Healthy", "Convalescent", tar
  if(sum(keep_ix) == 0){print(paste0(MIobj$formattedName, ' has no valid samples')); return(NULL)}
  MIobj <- filterMIobj(MIobj, keep_ix)
  contrast_vector <- as.numeric(!MIobj$pheno$Class %in% c('Healthy', 'Convalescent'))
  names(contrast_vector) <- rownames(MIobj$pheno)
  MIobj$class <- contrast_vector
  return(MIobj)
}

MIobj <- data_list$GSE6269_GPL570
head(MIobj$pheno  %>% dplyr::select(Pathogen, Class), 10)
```

```
##                          Pathogen    Class
## GSM173175          Influenza virus    Virus
```

```
## GSM173177          Influenza virus    Virus
## GSM173210 Streptococcus pneumoniae Bacteria
## GSM173266 Streptococcus pneumoniae Bacteria
## GSM173269 Streptococcus pneumoniae Bacteria
## GSM173272 Streptococcus pneumoniae Bacteria
## GSM173276 Streptococcus pneumoniae Bacteria
## GSM173279 Streptococcus pneumoniae Bacteria
## GSM173281    Staphylococcus aureus Bacteria
## GSM173282    Staphylococcus aureus Bacteria
```

```
MIobj <- removePathogen(MIobj, 'Staphylococcus aureus', 'Bacteria')
head(MIobj$pheno  %>% dplyr::select(Pathogen, Class), 10)
```

```
##                           Pathogen    Class
## GSM173210 Streptococcus pneumoniae Bacteria
## GSM173266 Streptococcus pneumoniae Bacteria
## GSM173269 Streptococcus pneumoniae Bacteria
## GSM173272 Streptococcus pneumoniae Bacteria
## GSM173276 Streptococcus pneumoniae Bacteria
## GSM173279 Streptococcus pneumoniae Bacteria
```

```
bacteria_list <- lapply(data_list, removePathogen, pathogen = 'Staphylococcus aureus', target_class = '
```

```
## [1] "GSE18090_GPL570 has no valid samples"
## [1] "GSE50834_GPL10558 has no valid samples"
## [1] "GSE97741_GPL10558 has no valid samples"
```

```
sample_ix <- sapply(bacteria_list, filterSampleSizes, N = 4)
bacteria_list <- bacteria_list[sample_ix]

sapply(bacteria_list[1:10], function(x){table(x$pheno$Pathogen, x$pheno$Class)})
```

```
## $GSE103119_GPL10558
##
##                            Bacteria Healthy
##   Healthy                         0      38
##   Mycoplasma                     30       0
##   Streptococcus pneumoniae        4       0
##   Streptococcus pyogenes          1       0
##
## $GSE110106_GPL10558_1
##
##                                              Bacteria Healthy
##   Chlamydia trachomatis                            16       0
##   Chlamydia trachomatis;Neisseria gonorrhoeae       5       0
##   Healthy                                           0      36
##   Neisseria gonorrhoeae                             4       0
##
## $GSE110106_GPL10558_2
##
##                                              Bacteria Healthy
##   Chlamydia trachomatis                             7       0
##   Chlamydia trachomatis;Neisseria gonorrhoeae       3       0
##   Healthy                                           0      27
##   Neisseria gonorrhoeae                             3       0
##
```

```
## $GSE116014_GPL10558
##
##                             Bacteria Healthy
##   Healthy                          0      24
##   Mycobacterium tuberculosis      58       0
##
## $GSE13015_GPL6947
##
##                                Bacteria Healthy
##   Aeromonas hydrophila                1       0
##   Burkholderia pseudomallei          16       0
##   Corynebacterium                     1       0
##   Enterococcus                        1       0
##   Enterococcus faecium                1       0
##   Escherichia coli                    4       0
##   Healthy                             0       5
##   Salmonella                          1       0
##   Streptococcus Non-Group A or B      1       0
##
## $GSE19439_GPL6947
##
##                             Bacteria Healthy
##   Healthy                          0      12
##   Mycobacterium tuberculosis      13      17
##
## $GSE19442_GPL6947
##
##                             Bacteria Convalescent
##   Mycobacterium tuberculosis      20           31
##
## $GSE19444_GPL6947
##
##                             Bacteria Convalescent Healthy
##   Healthy                          0            0      12
##   Mycobacterium tuberculosis      21           21       0
##
## $GSE22098_GPL6947
##
##                 Bacteria Healthy
##   Healthy              0      23
##   Staphylococcus      40       0
##   Streptococcus       12       0
##   Unknown              0      58
##
## $GSE23140_GPL6254
##
##                           Bacteria Healthy
##   Healthy                        0       4
##   Streptococcus pneumoniae       4       0
```

```
virus_list <- lapply(data_list, removePathogen, pathogen = 'Staphylococcus aureus', target_class = 'Viru
```

```
## [1] "GSE9960_GPL570 has no valid samples"
```

```r
sample_ix <- sapply(virus_list, filterSampleSizes, N = 4)
virus_list <- virus_list[sample_ix]
sapply(virus_list[1:10], function(x){table(x$pheno$Pathogen, x$pheno$Class)})
```

```
## $GSE101702_GPL21185
##
##                   Healthy Virus
##   Healthy              52     0
##   Influenza virus       0   107
##
## $GSE103119_GPL10558
##
##                                                                 Healthy Virus
##   Adenovirus;Bocavirus;Coronavirus                                    0     1
##   Adenovirus;Respiratory syncytial virus                              0     1
##   Adenovirus;Rhinovirus                                               0     5
##   Bocavirus;Human parainfluenza virus;Rhinovirus                      0     1
##   Bocavirus;Rhinovirus                                                0     2
##   Coronavirus                                                         0     1
##   Coronavirus;Respiratory syncytial virus                             0     1
##   Healthy                                                            38     0
##   Human metapneumovirus;Respiratory syncytial virus;Rhinovirus        0     1
##   Human metapneumovirus;Rhinovirus                                    0    12
##   Human parainfluenza virus                                           0     5
##   Human parainfluenza virus;Respiratory syncytial virus               0     1
##   Human parainfluenza virus;Rhinovirus                                0     3
##   Respiratory syncytial virus                                         0    11
##   Respiratory syncytial virus;Rhinovirus                              0     2
##   Rhinovirus                                                          0    30
##
## $GSE103842_GPL10558
##
##                               Healthy Virus
##   Healthy                          12     0
##   Respiratory syncytial virus       0    62
##
## $GSE117827_GPL23126
##
##                               Healthy Virus
##   Coxsackievirus                    0     2
##   Enterovirus                       0     2
##   Healthy                          12     0
##   Respiratory syncytial virus       0    10
##   Rhinovirus                        0    24
##
## $GSE1739_GPL201
##
##            Healthy Virus
##   Healthy        4     0
##   SARS-CoV       0    10
##
## $GSE2171_GPL201
##
##                                Healthy Virus
```

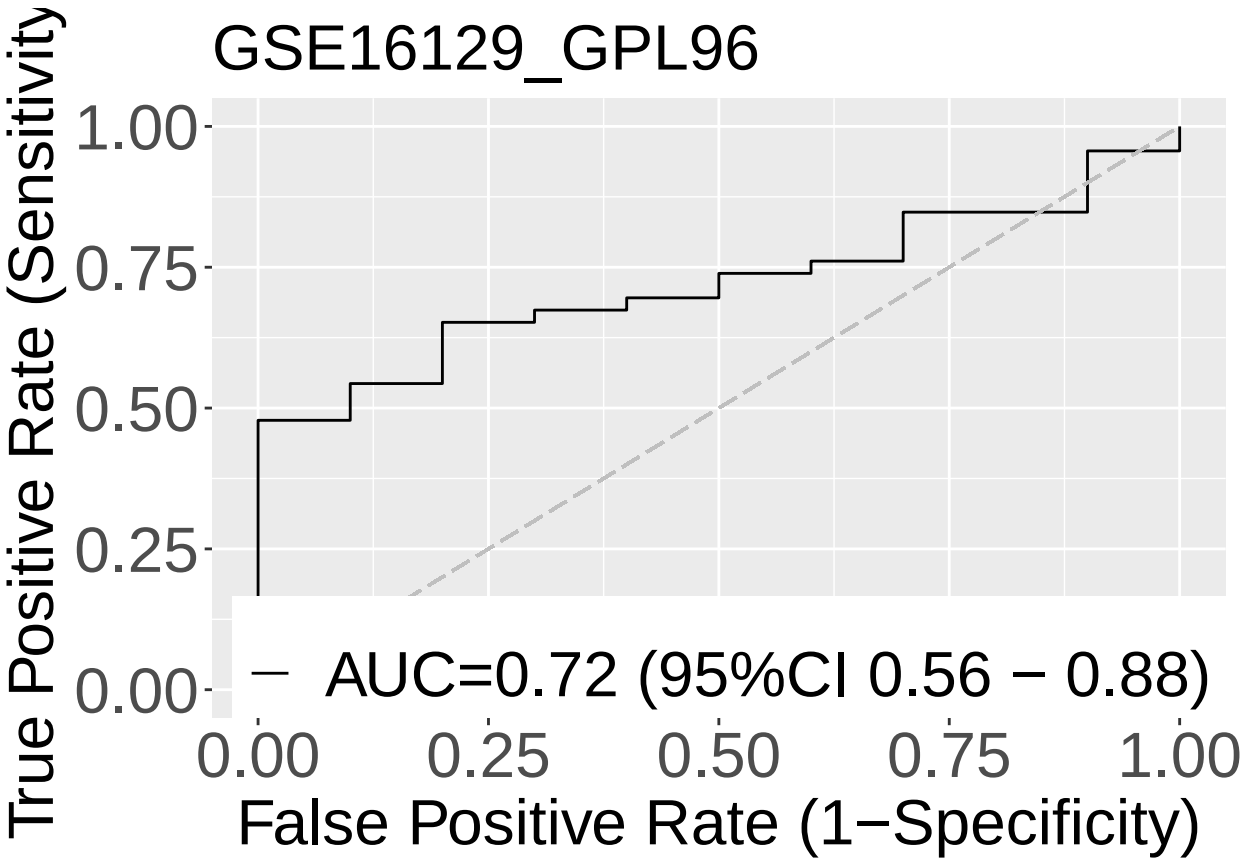
```
##   Healthy                            12     0
##   Human immunodeficiency virus        0    22
##
## $GSE2729_GPL8300
##
##             Convalescent Healthy Virus
##   Healthy               0       8     0
##   Rotavirus             5       0    10
##
## $GSE29333_GPL6884
##
##                                     Healthy Virus
##   Healthy                                 8     0
##   Human T-Lymphotropic virus Type 1       0    27
##
## $GSE29333_GPL6947
##
##                                     Healthy Virus
##   Healthy                                 9     0
##   Human T-Lymphotropic virus Type 1       0    30
##
## $GSE29366_GPL6884
##
##                    Healthy Virus
##   Healthy               12     0
##   Influenza virus        0    19
```

3.B Generating robustness AUROCs for an input signature

Consider an arbitrary signature containing 'IFI27' (down) and 'CTSB' (up). Note: MetaIntegrator requires a properly formatted MetaFilter object, which is a bit of a pain.

```
sig <- list()
sig$posGeneNames <- '1508' #CTSB
sig$negGeneNames <- '3429' #IFI27
sig$filterDescription <- 'test'
sig$FDRThresh <- 0
sig$effectSizeThresh <- 0
sig$numberStudiesThresh <- 1
sig$isLeaveOneOut <- F
sig$heterogeneityPvalThresh <- 0
sig$timestamp <- Sys.time()

MIobj <- staph_list$GSE16129_GPL96
MetaIntegrator::rocPlot(filterObject = sig, datasetObject = MIobj)
```

```
## Used  1 of  1  pos genes, and  1  of  1  neg genes
## For dataset GSE16129_GPL96, AUC = 0.72
```




```
AUC = calculateROC(labels = MIobj$class, predictions = calculateScore(filterObject = sig, datasetObject
```

```
## Used  1 of  1  pos genes, and  1  of  1  neg genes
```

```
print(AUC$auc)
```

```
##           [,1]
## [1,] 0.7195652
```

NOTE FROM DARPA PROGRAM: THIS IS NOT EFFICIENT!!!

Use the following function for quick calculations.

```
calculateAUROC <- function(MIobj, signature, method = 'geomMean', suppressMessages = F){
  labels <- MIobj$class
  scores <- calculateScoreRobust(filterObject = signature, datasetObject = MIobj, method = method, suppr
  if(all(sapply(scores, is.na))){
    roc <- NA
  } else {
    roc <- calculateROC(labels = labels, predictions = scores, AUConly = T)
  }
  return(as.numeric(roc))
}

calculateScoreRobust <- function(filterObject, datasetObject, suppressMessages = T, method = 'geomMean'
  if(!method %in% c('original', 'geomMean')){
    stop('method must be original or geomMean')
  }
```

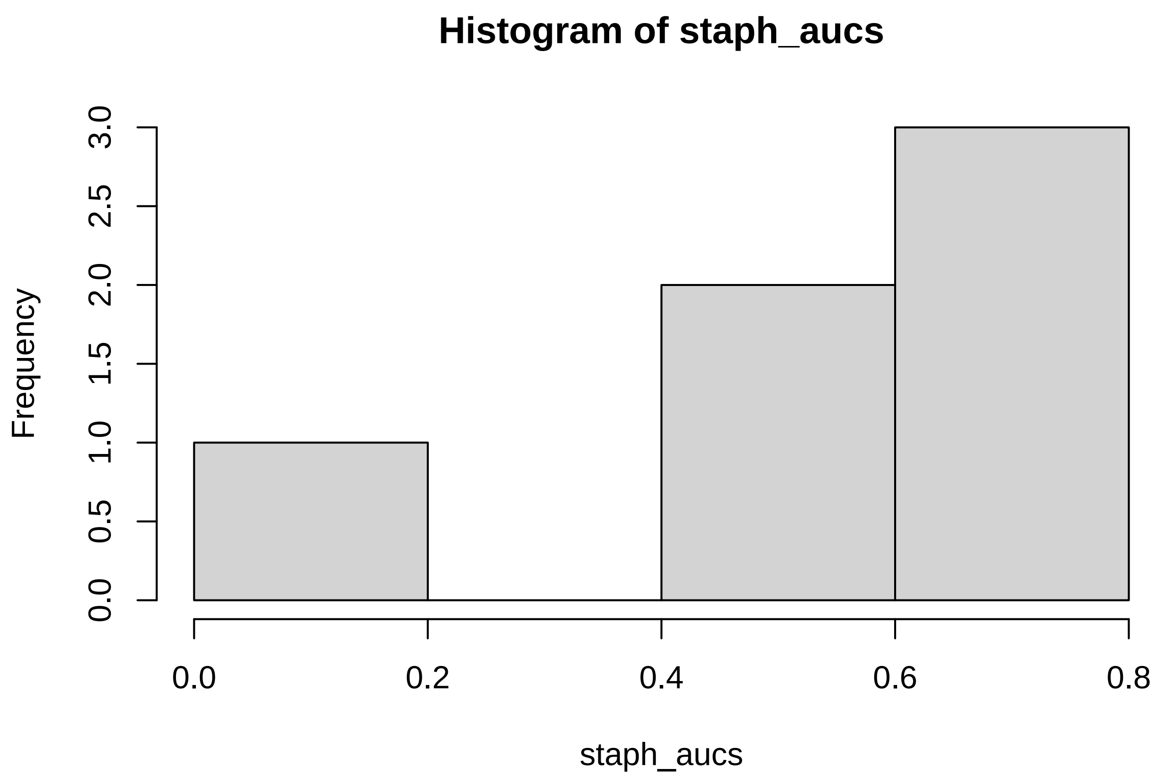
```
if(method == 'original'){
  totalScore = calculateScore(filterObject = filterObject, datasetObject = datasetObject, suppressMess
} else {
  # method options: geomMean, geomMean_exp2, mean, mean_exp2
  # assumes data is being entered on a log2 scale

  datasetObjectmin <- min(datasetObject$expr, na.rm = TRUE)
  if (datasetObjectmin < 0) {
    datasetObject$expr <- datasetObject$expr + abs(datasetObjectmin) + 1
  }

  # pull list of genes present in data set
  expr_genes <- datasetObject$keys
  filterObject$posGeneNames <- setdiff(filterObject$posGeneNames,'')
  filterObject$negGeneNames <- setdiff(filterObject$negGeneNames,'')
  pos_genes = intersect(filterObject$posGeneNames, expr_genes)
  neg_genes = intersect(filterObject$negGeneNames, expr_genes)

  # for reporting fraction of genes used
  if (!suppressMessages){
    N_pos_str <- paste0(length(pos_genes), ' of ', length(filterObject$posGeneNames))
    N_neg_str <- paste0(length(neg_genes), ' of ', length(filterObject$negGeneNames))
    print(paste0('Used ', N_pos_str, ' pos genes and ', N_neg_str, ' neg genes'))
  }


  if(length(pos_genes) == 0 & length(neg_genes) == 0){
    if (!suppressMessages){
      print('no common genes between data and signature')
    }
    return(NA)
  }

  posScore = .calculate_scores(datasetObject, pos_genes, method = method)
  negScore = .calculate_scores(datasetObject, neg_genes, method = method)

  if(grepl(pattern = 'exp2', x = method)){
    totalScore = posScore/negScore
  } else {
    totalScore = posScore - negScore
  }


  if (sum(abs(totalScore), na.rm = T) != 0){
    if(zScore){
      totalScore = as.numeric(scale(totalScore))
    } else {
      totalScore = as.numeric(totalScore)
    }

  }
}
```

```
  return(totalScore)
}


# implemented w input from Antonio Cappuccio!
.calculate_scores = function(datasetObject, genes, method = 'geomMean'){

  # find the genes to average
  genes_idx = which(datasetObject$keys %in% genes)

  # if there aren't any genes, return 0
  if (length(genes_idx) == 0){
    if(method == 'mean'){
      output_val = 1
    } else {
      output_val = 0
    }

    return(rep(output_val, ncol(datasetObject$expr)))
  }

  # filter expr to sig genes
  gene_expr = datasetObject$expr[genes_idx,]

  #in case only one gene is present, take the vector itself
  if (is.null(nrow(gene_expr))){
    sample_scores = gene_expr
  }

  if (!is.null(nrow(gene_expr))){
    sample_scores = apply(gene_expr, 2, function(method, x){switch(method,
                                                                   geomMean = geom_mean(x),
                                                                   geomMean_exp2 = geom_mean_exp2(x),
                                                                   mean = mean(x, na.rm = T),
                                                                   mean_exp2 = mean_exp2(x))
    }, method = method)}

  return(sample_scores)
}
```

```
staph_aucs <- sapply(X = staph_list, FUN = calculateAUROC, signature = sig)
```

```
## [1] "Used 1 of 1 pos genes and 1 of 1 neg genes"
## [1] "Used 1 of 1 pos genes and 1 of 1 neg genes"
## [1] "Used 1 of 1 pos genes and 1 of 1 neg genes"
## [1] "Used 1 of 1 pos genes and 1 of 1 neg genes"
## [1] "Used 1 of 1 pos genes and 1 of 1 neg genes"
## [1] "Used 1 of 1 pos genes and 1 of 1 neg genes"
```

```
hist(staph_aucs)
```



3.C Generating cross-reactivity AUROCs for an input signature

```
non_staph_bact_aucs <- sapply(X = bacteria_list, FUN = calculateAUROC, signature = sig)
```

```
## [1] "Used 1 of 1 pos genes and 1 of 1 neg genes"
## [1] "Used 1 of 1 pos genes and 1 of 1 neg genes"
## [1] "Used 1 of 1 pos genes and 1 of 1 neg genes"
## [1] "Used 1 of 1 pos genes and 1 of 1 neg genes"
## [1] "Used 1 of 1 pos genes and 1 of 1 neg genes"
## [1] "Used 1 of 1 pos genes and 1 of 1 neg genes"
## [1] "Used 1 of 1 pos genes and 1 of 1 neg genes"
## [1] "Used 1 of 1 pos genes and 1 of 1 neg genes"
## [1] "Used 1 of 1 pos genes and 1 of 1 neg genes"
## [1] "Used 1 of 1 pos genes and 1 of 1 neg genes"
## [1] "Used 1 of 1 pos genes and 1 of 1 neg genes"
## [1] "Used 1 of 1 pos genes and 1 of 1 neg genes"
## [1] "Used 1 of 1 pos genes and 1 of 1 neg genes"
## [1] "Used 1 of 1 pos genes and 1 of 1 neg genes"
## [1] "Used 1 of 1 pos genes and 1 of 1 neg genes"
## [1] "Used 1 of 1 pos genes and 1 of 1 neg genes"
## [1] "Used 1 of 1 pos genes and 1 of 1 neg genes"
## [1] "Used 1 of 1 pos genes and 1 of 1 neg genes"
## [1] "Used 1 of 1 pos genes and 1 of 1 neg genes"
## [1] "Used 1 of 1 pos genes and 1 of 1 neg genes"
## [1] "Used 1 of 1 pos genes and 1 of 1 neg genes"
## [1] "Used 1 of 1 pos genes and 1 of 1 neg genes"
## [1] "Used 1 of 1 pos genes and 1 of 1 neg genes"
## [1] "Used 1 of 1 pos genes and 1 of 1 neg genes"
```

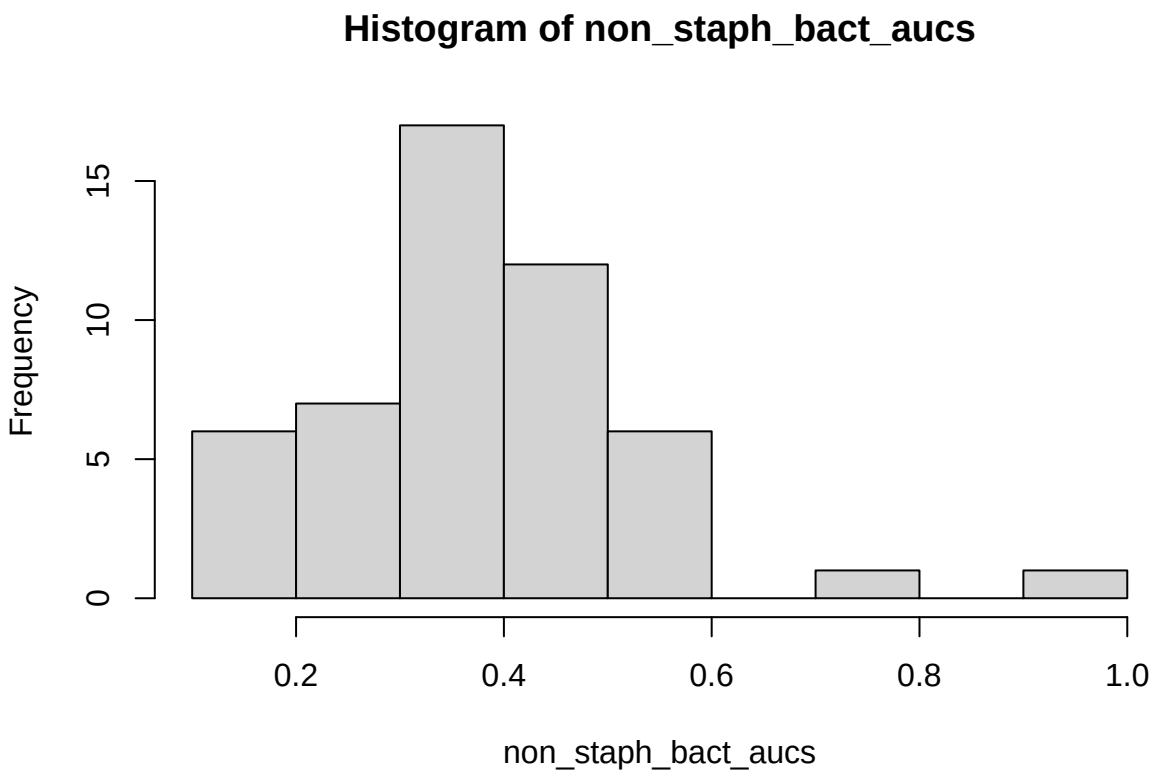
```
## [1] "Used 1 of 1 pos genes and 1 of 1 neg genes"
## [1] "Used 1 of 1 pos genes and 1 of 1 neg genes"
## [1] "Used 1 of 1 pos genes and 1 of 1 neg genes"
## [1] "Used 1 of 1 pos genes and 1 of 1 neg genes"
## [1] "Used 1 of 1 pos genes and 1 of 1 neg genes"
## [1] "Used 1 of 1 pos genes and 1 of 1 neg genes"
## [1] "Used 1 of 1 pos genes and 1 of 1 neg genes"
## [1] "Used 1 of 1 pos genes and 1 of 1 neg genes"
## [1] "Used 1 of 1 pos genes and 1 of 1 neg genes"
## [1] "Used 1 of 1 pos genes and 1 of 1 neg genes"
## [1] "Used 1 of 1 pos genes and 1 of 1 neg genes"
## [1] "Used 1 of 1 pos genes and 1 of 1 neg genes"
## [1] "Used 1 of 1 pos genes and 1 of 1 neg genes"
## [1] "Used 1 of 1 pos genes and 1 of 1 neg genes"
## [1] "Used 1 of 1 pos genes and 1 of 1 neg genes"
## [1] "Used 1 of 1 pos genes and 1 of 1 neg genes"
## [1] "Used 1 of 1 pos genes and 1 of 1 neg genes"
## [1] "Used 1 of 1 pos genes and 1 of 1 neg genes"
## [1] "Used 1 of 1 pos genes and 1 of 1 neg genes"
## [1] "Used 1 of 1 pos genes and 1 of 1 neg genes"
## [1] "Used 1 of 1 pos genes and 1 of 1 neg genes"
## [1] "Used 1 of 1 pos genes and 1 of 1 neg genes"
## [1] "Used 1 of 1 pos genes and 1 of 1 neg genes"
## [1] "Used 1 of 1 pos genes and 1 of 1 neg genes"
## [1] "Used 1 of 1 pos genes and 1 of 1 neg genes"
## [1] "Used 1 of 1 pos genes and 1 of 1 neg genes"
## [1] "Used 0 of 1 pos genes and 0 of 1 neg genes"
## [1] "no common genes between data and signature"
```

```
virus_aucs <- sapply(X = virus_list, FUN = calculateAUROC, signature = sig)
```

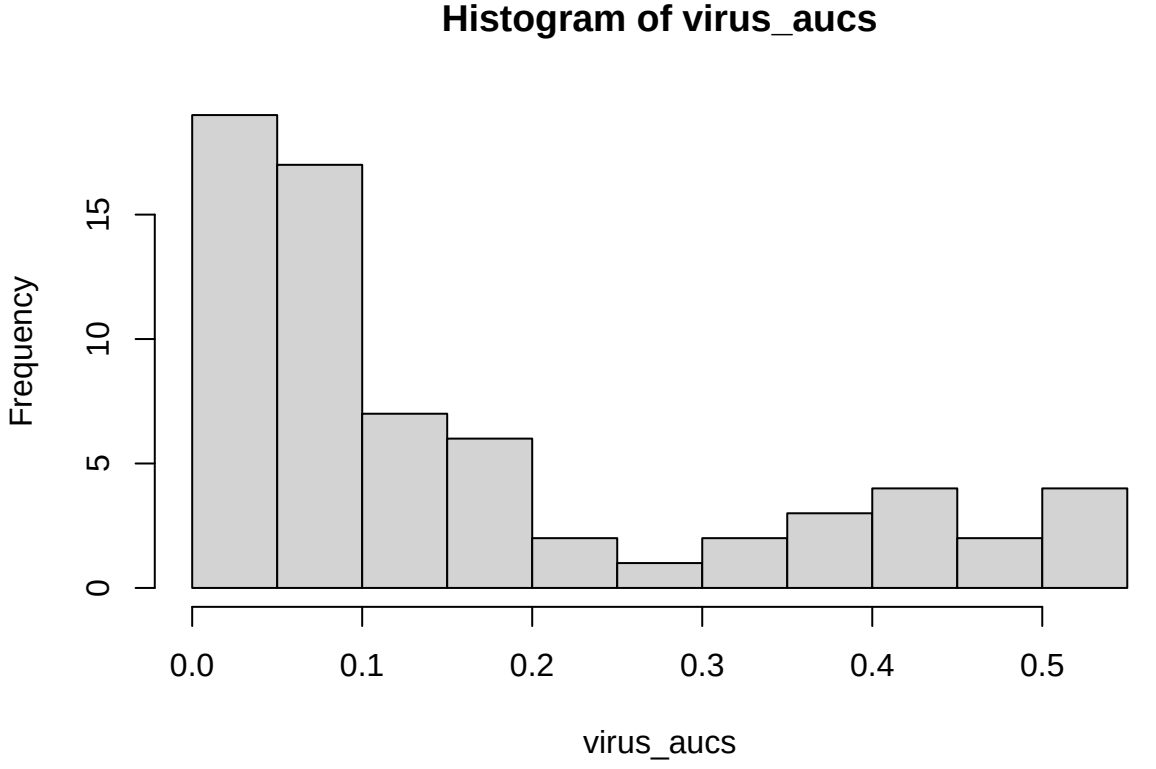
```
## [1] "Used 1 of 1 pos genes and 1 of 1 neg genes"
## [1] "Used 1 of 1 pos genes and 1 of 1 neg genes"
## [1] "Used 1 of 1 pos genes and 1 of 1 neg genes"
## [1] "Used 1 of 1 pos genes and 1 of 1 neg genes"
## [1] "Used 1 of 1 pos genes and 1 of 1 neg genes"
## [1] "Used 1 of 1 pos genes and 1 of 1 neg genes"
## [1] "Used 1 of 1 pos genes and 1 of 1 neg genes"
## [1] "Used 1 of 1 pos genes and 1 of 1 neg genes"
## [1] "Used 1 of 1 pos genes and 1 of 1 neg genes"
## [1] "Used 1 of 1 pos genes and 1 of 1 neg genes"
## [1] "Used 1 of 1 pos genes and 1 of 1 neg genes"
## [1] "Used 1 of 1 pos genes and 1 of 1 neg genes"
## [1] "Used 1 of 1 pos genes and 1 of 1 neg genes"
## [1] "Used 0 of 1 pos genes and 0 of 1 neg genes"
## [1] "no common genes between data and signature"
## [1] "Used 1 of 1 pos genes and 1 of 1 neg genes"
## [1] "Used 1 of 1 pos genes and 1 of 1 neg genes"
## [1] "Used 1 of 1 pos genes and 1 of 1 neg genes"
## [1] "Used 1 of 1 pos genes and 1 of 1 neg genes"
## [1] "Used 1 of 1 pos genes and 1 of 1 neg genes"
## [1] "Used 1 of 1 pos genes and 1 of 1 neg genes"
## [1] "Used 1 of 1 pos genes and 1 of 1 neg genes"
## [1] "Used 1 of 1 pos genes and 1 of 1 neg genes"
## [1] "Used 1 of 1 pos genes and 1 of 1 neg genes"
```

```
## [1] "Used 1 of 1 pos genes and 1 of 1 neg genes"
## [1] "Used 1 of 1 pos genes and 1 of 1 neg genes"
## [1] "Used 1 of 1 pos genes and 1 of 1 neg genes"
## [1] "Used 1 of 1 pos genes and 1 of 1 neg genes"
## [1] "Used 1 of 1 pos genes and 1 of 1 neg genes"
## [1] "Used 1 of 1 pos genes and 1 of 1 neg genes"
## [1] "Used 1 of 1 pos genes and 1 of 1 neg genes"
## [1] "Used 1 of 1 pos genes and 1 of 1 neg genes"
## [1] "Used 1 of 1 pos genes and 1 of 1 neg genes"
## [1] "Used 1 of 1 pos genes and 1 of 1 neg genes"
## [1] "Used 1 of 1 pos genes and 1 of 1 neg genes"
## [1] "Used 1 of 1 pos genes and 1 of 1 neg genes"
## [1] "Used 1 of 1 pos genes and 1 of 1 neg genes"
## [1] "Used 1 of 1 pos genes and 1 of 1 neg genes"
## [1] "Used 1 of 1 pos genes and 1 of 1 neg genes"
## [1] "Used 1 of 1 pos genes and 1 of 1 neg genes"
## [1] "Used 1 of 1 pos genes and 1 of 1 neg genes"
## [1] "Used 1 of 1 pos genes and 1 of 1 neg genes"
## [1] "Used 1 of 1 pos genes and 1 of 1 neg genes"
## [1] "Used 1 of 1 pos genes and 1 of 1 neg genes"
## [1] "Used 1 of 1 pos genes and 1 of 1 neg genes"
## [1] "Used 1 of 1 pos genes and 1 of 1 neg genes"
## [1] "Used 1 of 1 pos genes and 1 of 1 neg genes"
## [1] "Used 0 of 1 pos genes and 0 of 1 neg genes"
## [1] "no common genes between data and signature"
## [1] "Used 1 of 1 pos genes and 1 of 1 neg genes"
## [1] "Used 1 of 1 pos genes and 1 of 1 neg genes"
## [1] "Used 1 of 1 pos genes and 1 of 1 neg genes"
## [1] "Used 1 of 1 pos genes and 0 of 1 neg genes"
## [1] "Used 1 of 1 pos genes and 1 of 1 neg genes"
## [1] "Used 1 of 1 pos genes and 1 of 1 neg genes"
## [1] "Used 1 of 1 pos genes and 1 of 1 neg genes"
## [1] "Used 1 of 1 pos genes and 1 of 1 neg genes"
## [1] "Used 1 of 1 pos genes and 1 of 1 neg genes"
## [1] "Used 1 of 1 pos genes and 1 of 1 neg genes"
## [1] "Used 1 of 1 pos genes and 1 of 1 neg genes"
## [1] "Used 1 of 1 pos genes and 1 of 1 neg genes"
## [1] "Used 1 of 1 pos genes and 1 of 1 neg genes"
## [1] "Used 1 of 1 pos genes and 1 of 1 neg genes"
## [1] "Used 1 of 1 pos genes and 1 of 1 neg genes"
## [1] "Used 1 of 1 pos genes and 1 of 1 neg genes"
## [1] "Used 1 of 1 pos genes and 1 of 1 neg genes"
## [1] "Used 1 of 1 pos genes and 1 of 1 neg genes"
## [1] "Used 1 of 1 pos genes and 1 of 1 neg genes"
## [1] "Used 1 of 1 pos genes and 1 of 1 neg genes"
## [1] "Used 1 of 1 pos genes and 1 of 1 neg genes"
## [1] "Used 1 of 1 pos genes and 1 of 1 neg genes"
```

```
hist(non_staph_bact_aucs)
```




```
hist(virus_aucs)
```



3.D Plotting AUROC curves

For this functionality, we rely on the MetaIntegrator package

```
meta_obj <- list()
meta_obj$originalData <- staph_list
pooledROCPlot(metaObject = meta_obj, filterObject = sig)
```

```
## Warning in regularize.values(x, y, ties, missing(ties), na.rm = na.rm):
## collapsing to unique 'x' values

## Warning in regularize.values(x, y, ties, missing(ties), na.rm = na.rm):
## collapsing to unique 'x' values

## Warning in regularize.values(x, y, ties, missing(ties), na.rm = na.rm):
## collapsing to unique 'x' values

## Warning in regularize.values(x, y, ties, missing(ties), na.rm = na.rm):
## collapsing to unique 'x' values

## Warning in regularize.values(x, y, ties, missing(ties), na.rm = na.rm):
## collapsing to unique 'x' values

## Warning in regularize.values(x, y, ties, missing(ties), na.rm = na.rm):
## collapsing to unique 'x' values
```