Meta-mass shift chemical profiling of metabolomes from coral reefs

Hartmann, A.C.; Petras, D.; Quinn, R.A.; Protsyuk, I.; Archer, F.I.; Ransome, E.; Williams, G.J.; Bailey, B.A.; Vermeij, M.J.A.; Alexandrov, T.; Dorrestein, P.C.; Rohwer, F.L.

DOI
10.1073/pnas.1710248114

Publication date
2017

Document Version
Other version

Published in
Proceedings of the National Academy of Sciences of the United States of America

R code for Random Forests permutation test

```
library(gdata)
library(randomForest)

# mass shift data file, 'original_data'
# column 1: whole number associated with each mass shift;
 'bin_number' (1,2,3,...n)
# column 2: mass shift; 'mass_shift'
# column 3...n: mass shift data for each sample;
 'sample_1'...'sample_n',

# sample names data file; 'sample_names'
# column 1: sample names from original_data; 'sample'
# column 2: type of sample; 'class'


# read original data
orig.df <- read.xls("original_data.xlsx", stringsAsFactors = FALSE)

# extract bin ids into vector
bin.id <- orig.df$bin_number
# extract mass values and create column names
mod.col <- paste("mass_shift", orig.df$mass_shift, sep = "_")

# create transposed matrix of abundances and rename columns
orig.mat <- t(orig.df[, -(1:2)])
colnames(orig.mat) <- mod.col

# function takes a matrix and vector of bins (as long as number of
 columns in 'mat')
# returns matrix of summed values for each bin
sumBins <- function(mat, bins) {
  sums <- tapply(1:ncol(mat), bins, function(i) {
    bin.mat <- orig.mat[, i, drop = FALSE]
    rowSums(bin.mat, na.rm = TRUE)
  })
  do.call(cbind, sums)
}

# this is the empirical matrix of summed values
orig.bin.mat <- sumBins(orig.mat, bin.id)

# this is one matrix of randomly assigned bins (with bin frequency
 kept constant)
ran.bin.mat1 <- sumBins(orig.mat, sample(bin.id))

# read sample-class assignments
sample.names <- read.xls("sample_names.xlsx", stringsAsFactors =
 FALSE)
rownames(sample.names) <- sample.names$sample
row.class <- factor(sample.names[rownames(orig.bin.mat), "class"])

# the class frequencies do not vary, so calculate them here:
```

```
freq = table(row.class)
n = min(ceiling(freq / 2))
n = max(n, 4)
n = rep(n, length(freq))

# setup a randomForest function for your observed and null models
rfFunc <- function(mat, y, n) {
  randomForest(
    mat, y, proximity = TRUE, importance = TRUE,
    ntree = 10000, sampsize = n, replace = FALSE
  )
}

# run your observed randomForest
obs.rf <- rfFunc(orig.bin.mat, row.class, n)
obs.oob <- obs.rf$err.rate[nrow(obs.rf$err.rate), 1]

# use lapply to collect random forest models for your null
 distribution:
null.rf <- lapply(1:10, function(i) {
  cat(i, "\n")
  rfFunc(sumBins(orig.mat, sample(bin.id)), row.class, n)
})

# get whatever you want from models by looping through null.rf
null.oob.dist <- sapply(null.rf, function(x)
 x$err.rate[nrow(x$err.rate), 1])

hist(null.oob.dist)
abline(v = obs.oob, lty = "dashed", col = "red")
```