**Informative Approach on Differential Analysis for Time-course Metagenomic Sequencing Data**

This is a brief introduction about how to use the proposed method for detection of significantly differentially abundant features and varying time-interval for time-course metagenomic sequencing data between different conditions.

**1 Data input format for comparison of two different metagenomic conditions:**

Suppose the data set contains n features, m samples, k time points under two conditions. The elements in a count matrix C (m, n, k, h), cijdg corresponds to the number of reads (or relative abundance) of feature i in sample j at time d for each condtion, where h =2. Data set is tab-delimited format and name of feature should be clarified at the first column. The example format is listed below:

#################################################################################

## condition1 | condition2 ##

## time1 ... timek | time1 ... timek  ##

## sample1 ... samplem ... sample1 ... samplem | sample1 ... samplem ... sample1 ... samplem ##

## feature1 c1111 … c1m11 … c11k1 … c1mk1 | c1112 … c1m12 … c11k2 … c1mk2 ##

## feature1 c2111 … c2m11 … c21k1 … c2mk1 | c2112 … c2m12 … c21k2 … c2mk2 ##

## ... | ##

## featuren cn111 … cnm11 … cn1k1 … cnmk1 | cn112 … cnm12 … cn1k2 … cnmk2 ##

#################################################################################

A small of data set sample for 100 features count matrix is included on the website: *small.csv*.

**2. R commands**

Open R:

2.1 Input the source file *metaDprof.R*

> source("metaDprof.R")

2.2 Load a feature count matrix

> newdat=read.csv("example.csv", row.names=1, header=TRUE)

> newdat= as.matrix(newdat)

2.3 Set initial parameters

> n.sample = 5 # sample size;

> time.point = 10 # time point;

> f = 100 # number of feature;

> n.group= 2 # number of group;

> B= 500 # permutation iteration

> rownames(newdat)<-paste("feature", c(1:f), sep="")

> Time = as.integer(c(rep(c(1:time.point), each = n.sample),rep(c(1:time.point), each = n.sample)))

> Group = factor(rep(c(1, 2), c(time.point \* n.sample, time.point \* n.sample)))

> ID = factor(rep(c(1:n.sample), time.point \* n.group))

2.4 Analyze the example data

**# for one feature**

> f1\_out = metaDprof(newdat,1,t=time.point, n.sample=n.sample, Time = Time, Group = Group, ID = ID, norm=TRUE, log = TRUE, interval = TRUE)

> f1\_out

$feature

[1] "feature1"

$pvalue

[1] 0

$timeinterval

start end

interval1 4 10

In this example, log2 transformation was applied to the inputted dataset. TMM was used as normalization method. The output is a list with the names of each feature along with its permutation test p-value and detected varying time-interval.

**# for all features**

daf = daf.out (newdat, f, time.point, n.sample, n.group, B = B, Time = Time, Group = Group,

ID = ID, norm=TRUE, log = TRUE, interval = TRUE, adj.method = 'BH')

*# output DAFs with detected time-interval and sign for each unit interval*

> daf

feature pvalue timeinterval.start timeinterval.end timeinterval.sign padj

1 feature1 0.000 4 10 1 1 1 1 1 1 0.0000000

2 feature2 0.000 5 10 1 1 1 1 1 0.0000000

3 feature3 0.000 4 10 1 1 1 1 1 1 0.0000000

4 feature4 0.000 4 10 1 1 1 1 1 1 0.0000000

5 feature5 0.000 4 10 1 1 1 1 1 1 0.0000000

6 feature6 0.000 4 10 1 1 1 1 1 1 0.0000000

7 feature7 0.000 4 10 1 1 1 1 1 1 0.0000000

8 feature8 0.000 4 10 1 1 1 1 1 1 0.0000000

9 feature9 0.000 4 10 1 1 1 1 1 1 0.0000000

10 feature10 0.000 4 10 1 1 1 1 1 1 0.0000000

11 feature24 0.048 4 9 -1 -1 -1 -1 -1 0.3428571

12 feature43 0.048 1 3 1 1 0.3428571

13 feature99 0.000 1 4 -1 -1 -1 0.0000000

*#time-interval visualization: showing feature more abundant to what condition*

> Timeplot(daf)

