LDM Package Version 3.0

Yi-Juan Hu and Glen A. Satten

November 23, 2021

Contents

Changes in Version 2.1

- Added functionalities for analyzing matched-set data using the LDM or PERMANOVA (Zhu et al. 2020)
- Added functionalities for testing presence-absence associations using the LDM or PERMANOVA (Hu et al., 2021; Hu and Satten, 2021)
- Fixed some minor bugs
- The results here were computed from R 3.6.0, while the results in vignette v1.0 were computed from R 3.4.4

Changes in Version 3.0

- Added an option for mediation analysis using the LDM (`test.mediation` in `ldm`)
- Added an option for parallel computing (`n.cores` in `ldm` and `permanovaFL`)
- Removed the options `test.global` and `test.otu` and always perform both global and OTU tests (one can set `n.perm.max=10000` if one is interested in the global tests only and wants to cap the permutation at 10000).

1 Overview

The LDM package implements the Linear Decomposition Model (Hu and Satten 2020), which provides a single analysis path that includes global tests of any effect of the microbiome, tests of the effects of individual OTUs (operational taxonomic units) or ASVs (amplicon sequence variants) while accounting for multiple testing by controlling the false discovery rate (FDR), and a connection to distance-based ordination. It accommodates multiple covariates (e.g., clinical outcomes, environmental factors, treatment groups), either continuous or discrete (with ≥ 2 levels), as well as interaction terms to be tested either singly or in combination, allows for adjustment of confounding covariates, and uses permutation-based p-values that can control for clustered data (e.g., repeated measurements on the same individual). It gives results for both the frequency (i.e., relative abundance) and arcsine-root-transformed frequency data, and can give an “omnibus” test that combines results from analyses conducted on the two scales. In follow-up articles, we extended the LDM to test presence-absence associations (Hu et al. 2021), analyze matched sets of samples (Zhu et al. 2020), and test mediation effects of the microbiome (Yue and Hu 2021).

The core function of the LDM package is `ldm`, which can be used to test association between covariates of interest and the microbiome, possibly after adjusting for confounding covariates. Several additional functions are included in the LDM package. The function `permanovaFL` implements the PERMANOVA-FL test (our version of the PERMANOVA test); like the function `ldm`, `permanovaFL` allows adjustment of confounding covariates, control of clustered data, test of presence-absence associations (Hu and Satten 2021) and analysis of matched sets of samples (Zhu et al., 2020); unlike the implementation of PERMANOVA in `adonis` and `adonis2` in the R package `vegan`, `permanovaFL` adopts the permutation scheme described by Freedman and Lane (1983). The function `adjust.data.by.covariates` produces the adjusted distance matrix and OTU table after removing the effects of covariates (e.g., confounders). The functions `jaccard.mean` and `unifrac.mean` computes the expected value of the Jaccard and unweighted UniFrac distance matrices over rarefaction replicates. The function `avgdist.squared` computes the average of squared distance matrices based on multiple rarefied OTU tables.

Download the latest version of the package from `GitHub at https://github.com/yijuanhu/LDM` to a local hard drive and install and load the package:

```
> install.packages("LDM_3.0.tar.gz", repos=NULL)
> library(LDM)
```

2 Example dataset 1: throat microbiome data

The throat microbiome data included in this package contain 60 subjects with 28 smokers and 32 non-smokers. Microbiome data were collected from right and left nasopharynx and oropharynx region to form an OTU table with 856 OTUs. The phylogenetic tree was constructed using UPGMA on the K80 distance matrix of the OTUs. Metadata include smoking status, packs smoked per year, age, gender, and antibiotic use (we added a binary variable `AntibioticUse` which takes value 1 if `AntibioticUsePast3Months_TimeFromAntibioticUsage` is not equal to `"None"` and 0 if otherwise). For further information on these data see Hu and Satten (2020) and references therein. We first prepare this dataset:

```
> data(throat.tree)    # phylogenetic tree; only needed if UniFrac distances are requested
> data(throat.otu.tab) # OTU table
> data(throat.meta)    # metadata (covariates of interest, confounding covariates, etc.)
```

The OTU table should have rows corresponding to samples and columns corresponding to OTUs (`ldm` will transpose the OTU table if the number of rows is not equal to the length of the covariates in the metadata but this consistency check will fail in the unlikely case that the number of OTUs and samples are equal). Note that samples with zero reads for *every* OTU, possibly representing negative controls, are analyzed as they are, but a warning message will be sent. OTUs with zero reads in *every* sample will be automatically removed by `ldm` before any further analysis is carried out.

It is *essential* that the OTU table and the metadata have the same number of samples and that these occur *in the same order* in the OTU table and metadata. Note also that fitting the LDM assumes no missing data; samples having a NA value for any covariate in the formula (including confounders) are removed by `ldm` before any further analysis is carried out. Similarly, if a `cluster.id` variable is specified (see Section 8), samples having `cluster.id=NA` are removed.

As in the analysis presented in Hu and Satten (2020), we filtered out OTUs with presence in fewer than 5 samples, leaving 233 OTUs for analysis:

```
> otu_presence = which(colSums(throat.otu.tab>0)>=5)
> throat.otu.tab5 = throat.otu.tab[,otu_presence]
> dim(throat.otu.tab5)
[1]  60 233
```

The filtered OTU table `throat.otu.tab5` is also available in the LDM package. Although this criterion is commonly used, its use is not necessary to run `ldm`. Note that the taxonomy table, if it exists, should be subsetted accordingly.

3 Ordination using distances adjusted for covariates

(Users who read for the testing functionalities of the LDM can skip this section). The function `adjust.data.by.covariates` produces (1) the adjusted OTU table and (2) the adjusted distance matrix, both having the effects of the covariates (e.g., confounders) removed using projection, without fitting or testing the LDM. These are also part of the adjustment of covariates in the LDM. The adjusted distance matrix can be used to perform ordination in which the effects of confounding covariates are removed. As an example, we perform distance-based ordination and visualize whether the samples from the throat data are clustered by smoking status after removing the confounding effects from gender and antibiotic use:

```
> adj.data <- adjust.data.by.covariates(formula=~Sex+AntibioticUse, data=throat.meta,
                                        otu.table=throat.otu.tab5, dist.method="bray")
> PCs <- eigen(adj.data$adj.dist, symmetric=TRUE)
> PCs.percentage = signif(PCs$values[1:2]/sum(PCs$values),3)*100

> par(mfrow=c(1,1), mar=c(5, 4, 4, 8) + 0.1, pty="s", xpd=TRUE)

> color = rep("blue", length(throat.meta$SmokingStatus))
> color[which(throat.meta$SmokingStatus=="Smoker")] = "red"

> plot(PCs$vectors[,1], PCs$vectors[,2],
     main="Ordination based on Bray-Curtis distance",
     xlab=paste("PC1 (", PCs.percentage[1], "%)", sep=""),
     ylab=paste("PC2 (", PCs.percentage[2], "%)", sep=""),
     xlim=c(-0.4, 0.4), ylim=c(-0.4, 0.4),
     col=color, pch=21)
> ordiellipse(ord=PCs$vectors, groups=factor(throat.meta$SmokingStatus, exclude=c()),
            conf=0.9, col=c("blue", "red"))
> legend(x="topright", inset=c(-0.5, 0.4), bty="n",
       legend=c("smokers","non-smokers"), pch=21,
       col=c("red","blue"), lty=0)
```

The resulting plot is shown in Figure 1. By removing confounding directions from the distance matrix, any clustering pattern by smoking status is not due to correlation between smoking and gender and/or antibiotic use.

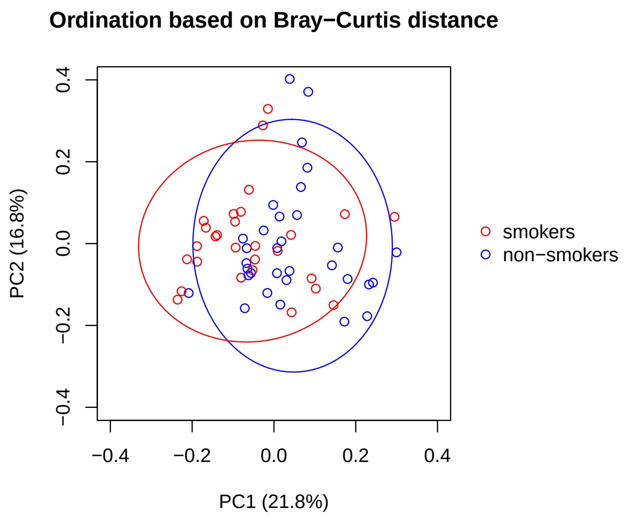

Figure 1 Ordination of the throat microbiome samples based on the Bray-Curtis distance after removing the effects of gender and antibiotic use. The ellipses are 90% confidence limits.

4 Writing formulas for `ldm` and `permanovaFL`

The function `ldm` uses a formula interface that is similar to other regression models in R, but with some differences. The general form of a formula for `LDM` is:

```
otu.table | (confounders) ~ (first set of covariates) + (second set) + (last set)
```

The left hand side of the formula gives the OTU table and any confounders that need to be controlled for. The generic formula in the absence of confounders is

```
otu.table ~ (first set of covariates) + (second set) + (last set)
```

On the right hand side of the formula, parentheses are used to group covariates that should be considered together as a single "submodel". The covariates in each submodel are considered jointly (i.e., leading to a joint p-value for each submodel), and in order from left to right such that the variance explained by each submodel is that part that remains after the previous submodels have been fit (i.e., are controlled for). If a submodel contains only a single covariate, then the parentheses are optional. Similarly, since the confounders are always treated as a single submodel, the parentheses around the confounders is also optional. The formula

```
otu.table ~ (confounders) + (first set of covariates) + (second set) + (last set)
```

will fit the same model as the first formula statement in this section. The only difference is that moving `(confounders)` to the right hand side of the model will produce a p-value for testing the confounders; specifying the confounders on the left hand side of the formula suppresses this p-value, which can speed up the `ldm`, both by reducing the number of test statistics calculated in each permutation and by possibly reducing the number of permutations required to satisfy all the stopping criteria.

The standard syntax for formulas in R can be used to expand single covariates into a model matrices with multiple columns. For example, the formula

```
otu.table ~ as.factor(a) + b*c
```

will have two submodels, the first with as many columns (degrees of freedom) as `a` has levels, and the second (using the standard R syntax for interactions) with columns corresponding to `(b, c, b×c)`.

The function `permanovaFL` uses the same formula interface as `ldm`. In addition, `permanovaFL` allows the OTU table on the left hand side of the formula to be replaced by a pre-calculated distance matrix.

We now give an example of a formula that can be fit to the throat data. We first wish to test the association of the microbiome with smoking status and then test the association of the microbiome with pack per year after removing the association with smoking status, both tests accounting for the confounding effects of sex and antibiotic use. Thus we specify this formula in the `ldm`:

```
throat.otu.tab5 | (Sex+AntibioticUse) ~ SmokingStatus + PackYears
```

5 Fitting the LDM model without testing

(Users who read for the testing functionalities of the LDM can skip this section). We can fit the LDM to the observed data (i.e., obtaining the variance explained [VE] without testing their significance) by calling `ldm` with `n.perm.max=0`. This is useful if we only want to see how much variability is explained by each set of covariates, or if we want factor loadings from OTUs. Unless otherwise specified, `ldm` automatically fits the OTU data at both the frequency (i.e., relative abundance) scale and the arcsin-root-transformed frequency scale.

Because we would also like to compare the VE by covariates to those by an ordination of the data using distance, an externally-calculated distance matrix can be used, or `ldm` can calculate the distance internally. Current distances supported include all of those supported by `vegdist` in the `vegan` package (i.e., "manhattan", "euclidean", "canberra", "bray", "kulczynski", "jaccard", "gower", "altGower", "morisita", "horn", "mountford", "raup" , "binomial", "chao", "cao", "mahalanobis") as well as "hellinger", "unwt-unifrac", and "wt-unifrac". The default choice for distance is the Bray-Curtis distance. Note that when we internally call `vegdist` for its supported distances without changing any of the default settings, the OTU table is converted to the frequency scale if `scale.otu.table=TRUE`; the parameter `binary=TRUE` if a presence-absence-type of distance is requested. The Hellinger distance measure ("hellinger") takes the form 0.5*E, where E is the Euclidean distance between the square-root-transformed frequency data. The UniFrac distances ("unwt-unifrac" and "wt-unifrac") are calculated by internally calling `GUniFrac` in the `GUniFrac` package. For the throat microbiome data, we choose the default Bray-Curtis distance:

```
> fit <- ldm(formula=throat.otu.tab5|(Sex+AntibioticUse)~SmokingStatus+PackYears,
             data=throat.meta,
             n.perm.max=0)            # parameter for fitting the LDM without testing

> ###  frequency scale
> fit$VE.global.freq.submodels        # VE for submodels 1 and 2
[1] 0.13891274 0.02751291
> fit$VE.otu.freq.submodels[1,1:3]    # Contribution of OTUs 1-3 to VE for submodel 1
        4695         4194         5160
3.780166e-07 1.138872e-07 2.666018e-03
> fit$F.global.freq     # F statistics for VE of submodels 1 and 2
            [,1]
[1,] 0.049056264
[2,] 0.009716031
> fit$F.otu.freq[1,1:3] # F statistics for contribution of OTUs 1-3 in submodel 1
       4695        4194        5160
0.098296641 0.000170493 0.041144505

> ###  arcsin-root transformed scale
> fit$VE.global.tran.submodels       # VE for submodels 1 and 2
[1] 1.2829126 0.3691146
```

The LDM performs a full decomposition (similar to SVD) of the OTU table; the singular values are returned in `d.freq` and `d.tran` (frequency and arcsin-root-transformed scale). It can be informative to make a scree plot to see how much variability is explained by submodel variables compared to the residual variability. The following code accomplishes this; the VE corresponding to the submodels is plotted in red, while residual components are plotted in black. In the code that follows, we handle the possibility of multiple terms per submodel by plotting a single point for each submodel; the value plotted is the average VE (i.e., averaged over each term in each submodel). We could also plot the VE for each term in each submodel by plotting the appropriate values of `d.freq`^{2}.

```
> scree.freq<-c(fit$VE.global.freq.submodels/fit$VE.df.submodels,fit$VE.global.freq.residuals)
> color <- c(rep("red", 2), rep("black", length(scree.freq)-2))
> plot(scree.freq/sum(scree.freq), main="Frequency Scale",
       xlab="Component", ylab="Proportion of total sum of squares", col=color)

> scree.tran<-c(fit$VE.global.tran.submodels/fit$VE.df.submodels,fit$VE.global.tran.residuals)
> color <- c(rep("red", 2), rep("black", length(scree.tran)-2))
> plot(scree.tran/sum(scree.tran), main="Arcsin-Root Scale",
       xlab="Component", ylab="", col=color)
```

The resulting plots are shown in Figure 2. We can see that although the VE by `SmokingStatus` does not rank the highest compared to other directions in the whole space (i.e., the augmented X matrix in the manuscript, which is denoted by `x` in the output of `ldm`), it still appears substantial. By contrast, the VE by `PackYears` seems very small. Although these results are suggestive, we must use permutation to decide which VEs are significant. The function `ldm` also returns the right singular vectors from the decomposition (`v.freq` and `v.tran`) which can be used to construct biplots (see Satten et al. 2017 for additional details).

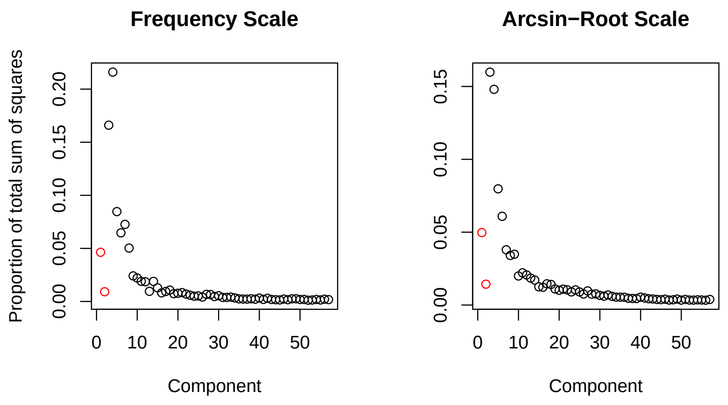

Figure 2 VEs when fitting the LDM to the throat microbiome data. The first two red circles represent VEs by `SmokingStatus` and `PackYears`, respectively. The black circles represent VEs (ordered by their Bray-Curtis eigenvalues) by residual components, usually referred to as the scree plot.

6 Testing relative-abundance associations using `ldm` and `permanovaFL`

The function `ldm` simultaneously provides global tests of any effect of the microbiome and OTU-specific tests. Thus the function `ldm` uses two sequential stopping criteria for the permutation procedure. For the global test, `ldm` uses the stopping rule of Besag and Clifford (1991), which stops permutation when a pre-specified minimum number of rejections (`n.rej.stop` with default 100; "rejection" occurs when the permutation statistic exceeded the observed test statistic) has been reached. For the OTU-specific tests, `ldm` uses the stopping rule of Sandve et al. (2011), which stops permutation when every OTU test has either reached the pre-specified number of rejections (`n.rej.stop` with default 100) or yielded a q-value that is below the nominal FDR level (`fdr.nominal` with default 0.1). As a convention, we call a test "*stopped*" if the corresponding stopping criterion for that test has been satisfied. For the OTU-specific tests, there exists an upper bound for the number of permutations for the algorithm to stop, which is $J \times L_{\min} \times \alpha^{-1}$, where J is the number of OTUs, $L_{\min}$ is the pre-specified minimum number of rejections, and α is the nominal FDR. For example, when $L_{\min} = 100$ and $\alpha = 0.1$, the upper bound is $1000J$, which is 233000 for this example dataset. If the maximum number of permutations `n.perm.max` takes the default value `NULL`, then the maximum number of permutations for testing OTUs is determined in this way, while the maximum number of permutations for the global test is fixed at 5000. Conversely, if a numeric value for `n.perm.max` is specified, this value is used for both the global tests and the OTU-specific tests. All tests are "terminated" when a user-specified `n.perm.max` have been generated; if this number is too small, some tests may not have "stopped". Tests that require a large number of permutations are those that are close to the threshold for acceptance or rejection. The function `permanovaFL` only uses the stopping rule of Besag and Clifford (1991) as it only implements a global test.

Further, unless specified otherwise by the user, `ldm` automatically performs tests on two data scales: the frequency (i.e., relative abundance) scale and the arcsin-root-transformed frequency scale. The omnibus test reports the most powerful of the two scales (after properly adjusting for the comparison).

We now give the simplest example of association testing using all default values for the `ldm` parameters. We test the association of the throat microbiome with smoking status and then test the association with pack per year after removing the association with smoking status, both tests accounting for the confounding effects of sex and antibiotic use:

```
> res.ldm <- ldm(formula=throat.otu.tab5|(Sex+AntibioticUse)~SmokingStatus+PackYears,
                 data=throat.meta, seed=67817)

> res.ldm$n.perm.completed
[1] 130000
```

```
> res.ldm$global.tests.stopped
[1] FALSE

> res.ldm$otu.tests.stopped
[1] TRUE

> res.ldm$p.global.omni
           [,1]
[1,] 0.00339932
[2,] 0.70120000

> res.ldm$detected.otu.omni
[[1]]
[1] "2434" "1490" "4703" "3538"
[[2]]
character(0)

> res.ldm$n.detected.otu.omni
[[1]]
[1] 4
[[2]]
[1] 0
```

With 5000 permutations, the stopping criterion of obtaining at least 100 rejections has not been met by the global test. The OTU tests stopped (met the early stopping criteria) after 130000 permutations. According to the omnibus test results, smoking status (after accounting for the confounders) is significantly associated with the microbiome (p-value = 0.0034) while pack per year after accounting for smoking status (and the confounders) is not (p-value = 0.701). Note that, when the true global p-value is very small, a larger number of permutations may be required to "accurately" estimate it and there is no upper bound for `n.perm.max` to guarantee the stop of the algorithm; if the value of `global.tests.stopped=FALSE`, then the reported p-value may be used as an upper bound. At the nominal FDR 0.1, we find four OTUs (with column names "2434", "'1490", "4703", and "3538" in `throat.otu.tab5`) that contribute to the association with smoking status and no OTU for pack per year. These results are consistent with the level of significance in the corresponding global tests.

Using the code below, we provide a summary table for the detected OTUs (or OTUs with the top smallest p-values), which includes the raw p-value, adjusted p-value (by the Benjamini-Hochberg [1995] [BH] procedure), population-level mean relative abundance, directions of covariate effects, and OTU name (can be substituted by taxonomy assignment); the OTUs are ordered by the raw p-values:

```
### order the detected OTUs by their p-values
> w1 = match(res.ldm$detected.otu.omni[[1]], colnames(res.ldm$q.otu.omni))
# remove [[1]] if there is only one covariate after "~" in the formula for ldm
```

```
> o = w1[order(res.ldm$p.otu.omni[1, w1])]

# If no OTU is detected, we can still provide a summary table for the top (e.g., 10) OTUs
# by re-defining o = order(res.ldm$p.otu.omni[1,])[1:10]

> summary.tab = data.frame(raw.pvalue=signif(res.ldm$p.otu.omni[1,o],3),
                           adj.pvalue=signif(res.ldm$q.otu.omni[1,o],3),
                           mean.freq=signif(res.ldm$mean.freq[o],3),
                           direction=t(ifelse(res.ldm$beta[1,]>0, "+", "-"))[o,],
                           otu.name=colnames(res.ldm$q.otu.omni)[o], # can use taxonomy assignment
                           row.names=NULL)
> colnames(summary.tab)[4] = paste("direction.", rownames(res.ldm$beta[1,]), sep="")
> summary.tab
  raw.pvalue adj.pvalue  mean.freq direction.SmokingStatusNonSmoker otu.name
1   0.000279     0.0583     0.0638                                -     1490
2   0.001000     0.0745     0.0373                                -     2434
3   0.001070     0.0745     0.0109                                -     3538
4   0.001700     0.0887     0.0129                                -     4703
```

It requires some care to interpret the directions of covariate effects. For example, `direction.SmokingStatusNonSmoker = '-'` means that the `NonSmoker` level in the `SmokingStatus` covariate is associated with lower relative abundances than the `Smokers` level, for all four OTUs.

For the detected OTUs, we can also plot their relative abundance data by smoking status, which are displayed in Figure 3. We first obtain the relative abundance data adjusted for the confounding covariates by using `adjust.data.by.covariates`:

```
> adj.data.1 <- adjust.data.by.covariates(formula= ~ Sex+AntibioticUse, # "~ 1" if no cov
                                    data=throat.meta, otu.table=throat.otu.tab5,
                                    center.otu.table=FALSE) # suppress centering
                                                            # to retain the original range

> for (i in 1:res.ldm$n.detected.otu.omni[[1]]) {
     boxplot(adj.data.1$y.freq[,o[i]]~throat.meta$SmokingStatus,
        main=paste("Top", i, "OTU"), ylab="Relative Abundance", xlab="Smoking Status")
   }
```

The `permanovaFL` function implements the PERMANOVA-FL test (our implementation of the PERMANOVA test). Like the PERMANOVA test implemented elsewhere (e.g., the `adonis` or `adonis2` functions in the R package `vegan`), our `permanovaFL` only allows testing of global hypotheses. Here we use the default `n.perm.max=5000`, i.e., 5000 for the maximum number of permutations.

```
> res.perm <- permanovaFL(throat.otu.tab5|(Sex+AntibioticUse)~SmokingStatus+PackYears,
                          data=throat.meta, dist.method="bray", seed=82955)

> res.perm
$F.statistics
          [,1]
```

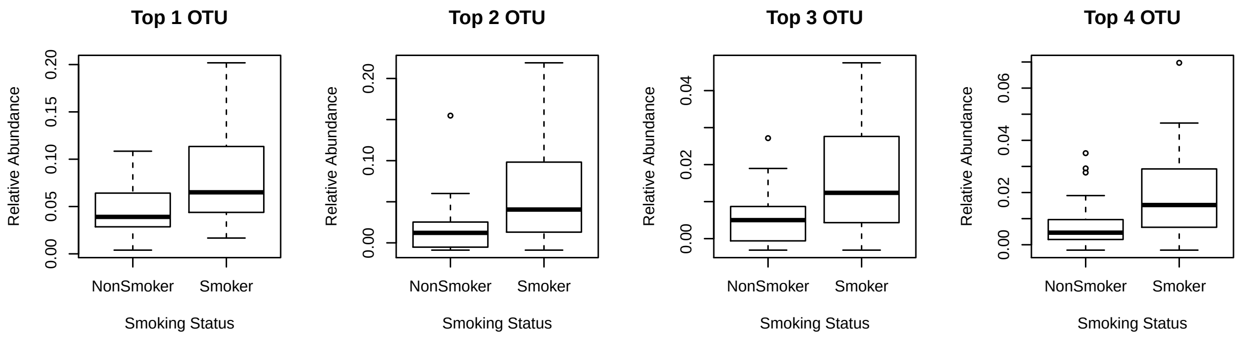

Figure 3 Box plot of relative abundances (adjusted for sex and antibiotic use).

```
[1,] 2.9354949
[2,] 0.8020703

$p.permanova
            [,1]
[1,] 0.00219956
[2,] 0.64500000

$n.perm.completed
[1] 5000

$permanova.stopped
[1] FALSE

$seed
[1] 82955
```

The global test results are consistent with those from `ldm`: smoking status (after accounting for the confounders) is significantly associated with the microbiome (p-value = 0.0022) while pack per year after accounting for smoking status (and the confounders) is not (p-value = 0.645).

7 Testing presence-absence associations using `ldm` and `permanovaFL`

The functions `ldm` and `permanovaFL` have implemented functionalities to test associations with the presence and absence of bacteria. This type of tests are often confounded by the library size (total number of reads per sample). A frequently used solution is *rarefaction*, which corresponds to subsampling reads of all samples to a common rarefaction depth that is often the lowest observed library size (after removing outliers). However, a single rarefaction typically results in a substantial loss of reads and hence statistical power. It also introduces a stochastic component to the analysis, so the results depend on the rarefaction replicate sampled.

In Hu et al. (2021), we presented two ways of extending the LDM F-statistic to aggregate information over multiple rarefactions: the LDM-F(R), which uses the average of the F-statistic over R rarefaction replicates as a test statistic, and the LDM-A, which averages the numerator and denominator of the F statistic separately and then uses the ratio of these averages as a test statistic. In particular, in LDM-A, the average of the numerator/denominator over *all* rarefaction replicates (equivalently, the expectation of the numerator/denominator over rarefaction) can be calculated in a closed form, and thus it is recommended over LDM-F(R). We have implemented both LDM-A and LDM-F(R) in `ldm`. Note that there is only one data scale, which is presence or absence (i.e., binary).

First, we illustrate the LDM-A method using the throat microbiome data, by calling `ldm` with `n.rarefy="all"`:

```
> res.ldmA <- ldm(formula=throat.otu.tab5 | (Sex+AntibioticUse) ~ SmokingStatus+PackYears,
                  data=throat.meta, seed=67817,
                  n.rarefy="all")      # parameter for requesting LDM-A

> res.ldmA$p.global.pa
[1] 0.00859828 0.75900000

> res.ldmA$detected.otu.pa
[[1]]
[1] "4363" "411"  "3954"
[[2]]
character(0)
```

According to the global results, smoking status (after accounting for the confounders) is significantly associated with the presence and absence of some bacteria (p-value = 0.0086) while pack per year after accounting for smoking status (and the confounders) is not (p-value = 0.759). At the nominal FDR 0.1, we find three OTUs that are differentially present among smokers and non-smokers. Using the code below, we provide a summary table for the detected OTUs, which includes the raw p-value, adjusted p-value (by BH), population-averaged probability of presence, directions of covariate effects, and OTU name (can be substituted by taxonomy assignment); the OTUs are ordered by the raw p-values.

```
### order the detected OTUs by their p-values
> w1 = match(res.ldmA$detected.otu.pa[[1]], colnames(res.ldmA$q.otu.pa))
# remove [[1]] if there is only one covariate after "~" in the formula for ldm
> o = w1[order(res.ldmA$p.otu.pa[1,w1])]
```

```
# If no OTU is detected, we can still provide a summary table for the top (e.g., 10) OTUs
# by re-defining o = order(res.ldmA$p.otu.pa[1,])[1:10]

> summary.ldmA.tab = data.frame(raw.pvalue=signif(res.ldmA$p.otu.pa[1,o],3),
                            adj.pvalue=signif(res.ldmA$q.otu.pa[1,o],3),
                            prob.presence=signif(colMeans(res.ldmA$phi)[o],3),
                            direction=t(ifelse(res.ldmA$beta[1,]>0, "+", "-"))[o,],
                            otu.name=colnames(res.ldmA$q.otu.pa)[o],
                            row.names=NULL)
> colnames(summary.ldmA.tab)[4] = paste("direction.", rownames(res.ldmA$beta[1,]), sep="")
> summary.ldmA.tab
  raw.pvalue adj.pvalue prob.presence direction.SmokingStatusNonSmoker otu.name
1   0.000687     0.0986         0.881                                +     3954
2   0.001240     0.0986         0.119                                -      411
3   0.001410     0.0986         0.100                                -     4363
```

It requires some care to interpret the directions of covariate effects. For example, `direction.SmokingStatusNonSmoker = '+'` means that the `NonSmoker` level in the `SmokingStatus` covariate is associated with higher chance of presence than the `Smokers` level for OTU "3954". For the detected OTUs, we can also plot the probability of presence in individual samples by smoking status, which are displayed in Figure 4. We first obtain the probability of presence of an OTU in each sample over rarefaction (denoted by `phi` in the `ldm` output), and adjust it for the confounding covariates by using the projection operation in the linear regression function `lm`:

```
> adj.phi = t(t(lm(res.ldmA$phi~Sex+AntibioticUse, data=throat.meta)$resid) #"~ 0" if no cov
           + colMeans(res.ldmA$phi))

> for (i in c(1:res.ldmA$n.detected.otu.pa[[1]])) {
      boxplot(adj.phi[,o[i]] ~ throat.meta$SmokingStatus,
         main=paste("Top", i, "OTU"), ylab="Chance of Presence", xlab="Smoking Status")
  }
```

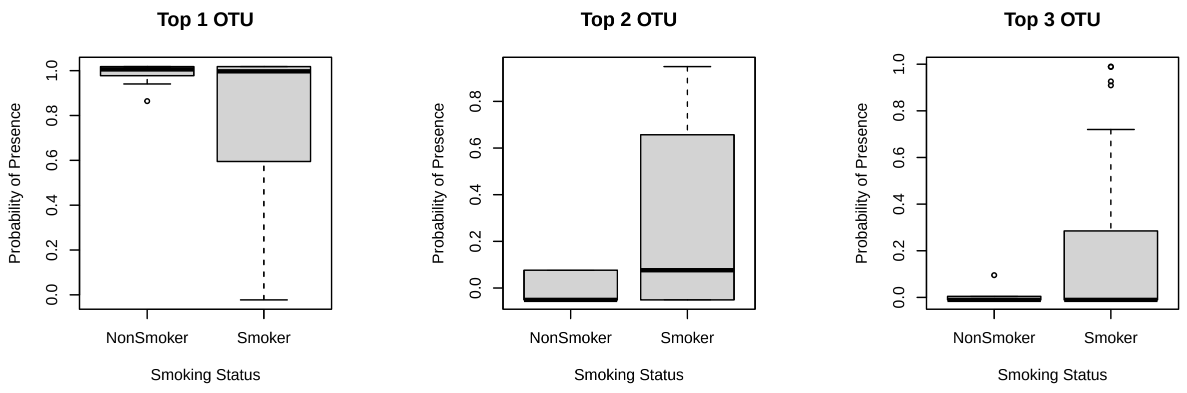

Figure 4 Box plot of probability of presence (adjusted for sex and antibiotic use).

Next, we illustrate the LDM-F(R) method, which can invoked by calling `ldm` with `binary=TRUE` and `n.rarefy=5`:

```
> res.ldmF <- ldm(formula=throat.otu.tab5 | (Sex+AntibioticUse) ~ SmokingStatus+PackYears,
                  data=throat.meta, seed=67817,
                  binary=TRUE, n.rarefy=5)      # parameters for requesting LDM-F(5)

> res.ldmF$p.global.pa                 # columns:R=1-5; rows: SmokingStatus, PackYears
          [,1]       [,2]       [,3]       [,4]       [,5]
[1,] 0.0079984 0.00839832 0.00739852 0.00739852 0.00739852
[2,] 0.7414000 0.74720000 0.77620000 0.82960000 0.83360000

> res.ldmF$detected.otu.pa
character(0)
```

In Hu and Satten (2021), we presented four ways of extending the PERMANOVA F-statistic to aggregate information over multiple rarefactions: the PERMANOVA-F(R), which uses the average of the F-statistic over R rarefaction replicates as a test statistic, the PERMANOVA-D2(R) and PERMANOVA-D(R), which average the element-wise squared and unsquared distance matrices, respectively, computed for each of R rarefaction replicates for use in calculating the F-statistic, and the PERMANOVA-D2-A$_o$, which calculates the expectation of the Jaccard or unweighted UniFrac distance matrix over *all* rarefaction replicates using the oth ($o = 1$ or 2) order approximation by the delta method. Like LDM-A, PERMANOVA-D2-A$_o$ does not perform any actual rarefaction and essentially uses all data in a non-stochastic manner, and thus it is recommended over the other three methods.

First, we illustrate the PERMANOVA-D2-A$_2$ method, which can be invoked by first calling `jaccard.mean` or `unifrac.mean` for calculating the expected squared Jaccard or unweighted UniFrac distance matrix and then calling `permanovaFL` with `square.dist=FALSE`:

```
> res.jaccard <- jaccard.mean( throat.otu.tab5 )
> dist.mean.sq <- res.jaccard$jac.mean.sq.o2
# res.unifrac <- unifrac.mean(throat.otu.tab5, throat.tree)
# dist.mean.sq <- res.unifrac$unifrac.mean.sq.o2

> res.perm.D2A2 <- permanovaFL(dist.mean.sq | (Sex+AntibioticUse) ~ SmokingStatus+PackYears,
                              data=throat.meta, seed=82955,
                              square.dist=FALSE)  # parameters for requesting PERMANOVA-D2-A2
> res.perm.D2A2$p.permanova
           [,1]
[1,] 0.00319936
[2,] 0.92340000
```

The expected squared Jaccard or unweighted UniFrac distance matrix, generated as part of the PERMANOVA-D2-A_2 method, can be used as if it were an ordinary squared distance matrix, for example, for performing ordination. The code below generated the ordination plot based on the Jaccard distance (after removing confounding directions), which is shown in Figure 5:

```
> adj.dist.mean.sq <- adjust.data.by.covariates(formula= ~ Sex+AntibioticUse,
                                                data=throat.meta,
                                                dist=dist.mean.sq, square.dist=FALSE)$adj.dist

> PCs <- eigen(adj.dist.mean.sq, symmetric=TRUE)
> color = rep("blue", length(throat.meta$SmokingStatus))
> w = which(throat.meta$SmokingStatus=="Smoker")
> color[w] = "red"

> plot(PCs$vectors[,1], PCs$vectors[,2], xlab="PC1", ylab="PC2",
     col=color, main="Smokers vs. non-smokers")
> legend(x="topleft", legend=c("smokers","non-smokers"), pch=c(21,21),
       col=c("red","blue"), lty=0)
> ordiellipse(ord=PCs$vectors, groups=factor(throat.meta$SmokingStatus, exclude=c()),
            conf=0.9, col=c("blue", "red"))
```

Then, we illustrate the other three methods.

```
### PERMANOVA-F(R)
> res.perm.F <- permanovaFL(throat.otu.tab5 | (Sex+AntibioticUse) ~ SmokingStatus+PackYears,
                             data=throat.meta, seed=82955,
                             dist.method="jaccard", binary=TRUE,            # parameters
                             n.rarefy=5)                   # for requesting PERMANOVA-F(5)

> res.perm.F$p.permanova
           [,1]       [,2]       [,3]       [,4]       [,5]
[1,] 0.00159968 0.00159968 0.00259948 0.00239952 0.00239952
[2,] 0.91560000 0.93460000 0.86560000 0.88860000 0.89240000


### PERMANOVA-D2(R)
> dist.avg.D2 <- avgdist.squared(throat.otu.tab5, dist.method="jaccard", n.rarefy=100)
```

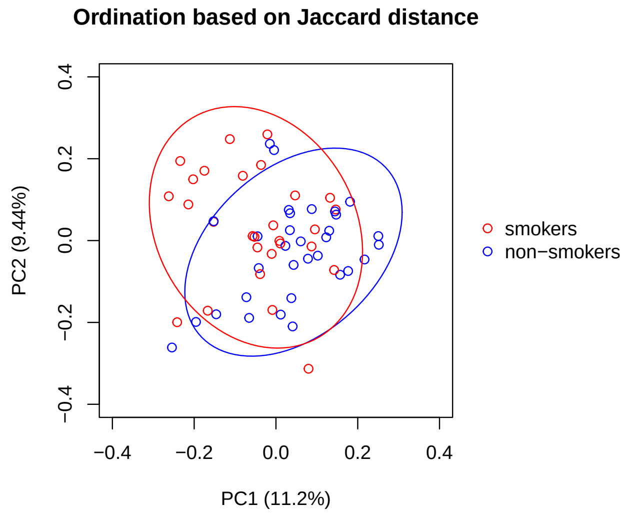

Figure 5 Ordination of the throat microbiome samples based on the expected squared Jaccard distance over rarefaction, which has the effects of gender and antibiotic use removed by projection. The ellipses are 90% confidence limits.

```
> res.perm.D2 <- permanovaFL(dist.avg.D2 | (Sex+AntibioticUse) ~ SmokingStatus+PackYears,
                             data=throat.meta, seed=82955,
                             square.dist=FALSE)  # parameter for requesting PERMANOVA-D2(100)
> res.perm.D2$p.permanova
           [,1]
[1,] 0.00339932
[2,] 0.92720000


### PERMANOVA-D(R)
> dist.avg.D <- as.matrix(avgdist(x=throat.otu.tab5, sample=min(rowSums(throat.otu.tab5)),
                          dmethod="jaccard", tree=NULL, iterations=100, binary=TRUE))
> res.perm.D <- permanovaFL(dist.avg.D | (Sex+AntibioticUse) ~ SmokingStatus+PackYears,
                            data=throat.meta, seed=82955)
> res.perm.D$p.permanova
           [,1]
[1,] 0.0029994
[2,] 0.9202000
```

8 Analyzing correlated samples with complex (restricted) permutations

The functions `ldm` and `permanovaFL` accommodate a variety of complex (i.e., restricted) permutations through the `permute` package. For unclustered data, `perm.within.type` governs the type of permutation. The default value, `permute.within.type="free"` performs unrestricted permutations, equivalent to using `sample(n.obs, replace=FALSE)`. Other allowed values are "series", "grid" and "none". If `perm.within.type="grid"` is specified, then values of `perm.within.nrow` and `perm.within.ncol` must be specified. For more information on the effect of choosing `perm.within.type` to a value other than "free", see the documentation for the `permute` package. If desired, permutations can also be restricted to groups of observations by specifying `strata=strata.id`, which restricts permutations to those that preserve the value of `strata.id` for each observation.

The functions `ldm` and `permanovaFL` use the complex (restricted) permutation schemes to analyze clustered data that preserves the sample correlations. Clusters in the data are identified by having a common value of `cluster.id`. There are two parameters that control the type of permutations possible with clustered data: `perm.between.type` and `perm.within.type`. The allowable values of `perm.within.type` are as described in the previous paragraph, while the allowable values of `perm.between.type` are "free", "none" and "series". To permute only *within* clusters, choose `perm.between.type="none"` and `perm.within.type="free"`. To permute only *across* clusters, choose `perm.between.type="free"` and `perm.within.type="none"`. To permute *both within and between* clusters, set both to "free". Note that if `perm.between.type` has a value other than "none" then all clusters must have the same size. Permutations of clustered data may be further restricted to preserve the value of `strata.id` by specifying `strata=strata.id`. Note that the value of `strata.id` must be the same for each member of a single cluster. If there are only a few observed cluster sizes, permuting within strata defined by cluster size is a way to analyze unbalanced, clustered data.

Users may also use the `how` function in the `permute` package to manually specify a restricted permutation. This is accomplished by setting `how=how.var` where `how.var` is an object of class `how` as produced by the `permute` package. Any permutation scheme allowed by `permute` can be specified in this way.

8.1 Example dataset 2: simulated data of repeated samples

To illustrate the analysis of repeated samples (i.e., samples within a cluster have the same value for the covariates of interest) using `ldm` and `permanovaFL`, we simulated a dataset based on the relative abundances and overdispersion observed in the throat microbiome data. It consists of data from 50 individuals, each of whom contributed 2 samples. Twenty-five individuals were cases (`Y.between` = 1), and the remaining ones were controls (`Y.between` = 0). Metadata include the case-control status (`Y.between`), a confounding covariate (`X.between`), and a unique individual identifier (`ID`). We use "between" in covariate names to indicate that these covariates are between-individual variables. See Hu and Satten (2020) for more details in generating this dataset. We first prepare this dataset:

```
> library(LDM)
> data(sim.otu.tab) # zero columns have been excluded
> data(sim.meta)
>
> otu_presence = which(colSums(sim.otu.tab>0)>=5)    # optionally remove OTUs that occur
> sim.otu.tab5 = sim.otu.tab[,otu_presence]          # in fewer than 5 individuals
> dim(sim.otu.tab5)
[1] 100 593
```

The OTU table contains 593 OTUs after excluding OTUs having less than 5 presence in the sample. To test the association of the microbiome with the case-control status `Y.between` while controlling for the confounder `X.between`, while also accounting for the clustering structure (so that the case-control status is always the same for all samples from the same individual), we specify `cluster.id=ID`, `perm.between.type="free"`, and `perm.within.type="none"` and use the code:

```
> res.ldm.repeated <- ldm(formula=sim.otu.tab5 | X.between ~ Y.between,
                          data=sim.meta, seed=34794,
                          cluster.id=ID,               # parameters for requesting analysis of
                          perm.within.type="none", perm.between.type="free") # repeated samples
> res.ldm.repeated$n.perm.completed
[1] 61000
> res.ldm.repeated$global.tests.stopped
[1] FALSE
> res.ldm.repeated$otu.tests.stopped
[1] TRUE
> res.ldm.repeated$p.global.omni
[1] 0.00019996
> res.ldm.repeated$detected.otu.omni
 [1] "1583" "3067" "5522" "5443" "2122" "2334" "2245" "2213" "330"  "2434" "3619" "4036"
[13] "922"  "3954" "3108" "799"  "3957" "4599" "596"
> res.ldm.repeated$n.detected.otu.omni
[1] 19
```

With 61000 permutations, the OTU-specific tests have met the stopping criterion and detect 19 OTUs at the nominal FDR 0.1. The global test has not met the stopping criterion of obtaining at least 100 rejections with 5000 permutations; however, we know the p-value is going to be very small around 0.00019996 and there is no need to add more permutations to make it more precise.

The syntax of using `permanovaFL` for this type of data is very similar to `ldm`, except that the method for calculating the distance matrix (`dist.method`) should be specified:

```
> res.perm.repeated <- permanovaFL(formula=sim.otu.tab5 | X.between ~ Y.between,
                         data=sim.meta, dist.method="bray", seed=34794,
                         cluster.id=ID,               # parameters for requesting analysis of
                         perm.within.type="none", perm.between.type="free") # repeated samples
> res.perm.repeated$p.permanova
           [,1]
[1,] 0.00019996
```

8.2 Example dataset 3: simulated data of matched sets of samples

To illustrate the analysis of matched sets of samples (i.e., samples within a cluster have different values for the covariates of interest) using `ldm` and `permanovaFL`, we added to the dataset in Section

8.1 a within-individual case-control status `Y.within` and a within-individual covariate `X.within`. In particular, `Y.within` is always 0 for one sample and 1 for the other sample from the same individual. To test the association of the microbiome with the case-control status `Y.within` while controlling for the potential confounder `X.within`, while also accounting for the clustering structure (so that the case-control status is always different for the two samples from the same individual), we specify `cluster.id=ID`, `perm.between.type="none"`, and `perm.within.type="free"`. As described in Zhu et al. (2020), the indicator variables for individuals should be adjusted as covariates to constrain the test within individuals.

```
> res.ldm.matched <- ldm(formula=sim.otu.tab5 | as.factor(ID) + X.within ~ Y.within,
                       data=sim.meta, seed=34794,
                       cluster.id=ID,            # parameters for requesting analysis of
                       perm.within.type="free", perm.between.type="none") # matched sets
> res.ldm.matched$n.perm.completed
[1] 71000
> res.ldm.matched$global.tests.stopped
[1] TRUE
> res.ldm.matched$otu.tests.stopped
[1] TRUE
> res.ldm.matched$p.global.omni
      [,1]
[1,] 0.435
> res.ldm.matched$detected.otu.omni
character(0)
```

With 71000 permutations, both the global test and OTU-specific tests have met the stopping criterion. The global test is non-significant with p-value 0.435 and the OTU-specific tests detect 0 significant OTUs.

Again, the syntax of using `permanovaFL` for this type of data is very similar to `ldm`, except that the method for calculating the distance matrix (`dist.method`) should be specified:

```
> res.perm.matched <- permanovaFL(formula=sim.otu.tab5 | as.factor(ID) + X.within ~ Y.within,
                          data=sim.meta, dist.method="bray", seed=34794,
                          cluster.id=ID,            # parameters for requesting analysis of
                          perm.within.type="free", perm.between.type="none") # matched sets
> res.perm.matched$p.permanova
          [,1]
[1,] 0.3275
```

9 Using `ldm` for performing a simple linear regression with permutation-based inference

In essence, `ldm` performs a simple linear regression with permutation-based inference at each taxon. In fact, `ldm` can be generally applied to data (in addition to the OTU table data) that do not follow normal distributions. In particular, `ldm` can be applied to analyze an alpha diversity measure or the relative abundance of a particular OTU in association of covariates of interest. Note that both measurements are univariate variables. To this end, we should suppress `test.otu` and `scale.otu.table` and turn on `freq.scale.only`.

```
### The alpha diversity for richness
> richness = matrix(rowSums(throat.otu.tab5>0), ncol=1)
> res.alpha.ldm <- ldm(richness | (Sex+AntibioticUse) ~ SmokingStatus+PackYears,
                     data=throat.meta, seed=67817,
                     scale.otu.table=FALSE, freq.scale.only=TRUE) # parameters for
                                           # requesting a simple linear regression
> res.alpha.ldm$p.global.freq
      [,1]
[1,] 0.528
[2,] 0.896

### The relative abundance of one OTU
> one.otu = throat.otu.tab5[,5,drop=FALSE]/rowSums(throat.otu.tab5)
> res.1otu.ldm <- ldm(one.otu | (Sex+AntibioticUse) ~ SmokingStatus+PackYears,
                  data=throat.meta, seed=67817,
                  scale.otu.table=FALSE, freq.scale.only=TRUE)
> res.1otu.ldm$p.global.freq
      [,1]
[1,] 0.909
[2,] 0.959
```

10 Using `ldm` for mediation analysis of the microbiome

To call a mediation analysis using the LDM, the formula takes the specific form

```
otu.table ~ exposure + outcome
```

or most generally

```
otu.table | (set of confounders) ~ (set of exposures) + (set of outcomes),
```

in which there should be exactly two terms on the right hand side of the regression, corresponding to the exposure(s) and the outcome(s), the outcome(s) must appear after the exposure(s), and the covariates or confounders must appear after "|".

We have simulated an outcome variable `Outcome` and added it to the metadata of the throat microbiome dataset. Here we test the mediation effects of the microbiome, at both the community level and the OTU level, that mediate the effect of `SmokingStatus` on `Outcome`.

```
> res.ldm.med <- ldm(formula = throat.otu.tab5 | (Sex+AntibioticUse) ~ SmokingStatus+Outcome,
                      data=throat.meta, seed=67817,
                      test.mediation=TRUE)          # parameter for requesting
                                                    # mediation analysis (LDM-med)
> res.ldm.med$n.perm.completed
[1] 40000
> res.ldm.med$global.tests.stopped
[1] FALSE
> res.ldm.med$otu.tests.stopped
[1] TRUE
> res.ldm.med$med.p.global.omni
[1] 0.00079992
> res.ldm.med$med.detected.otu.omni
[1] "2434" "171"  "4703"
```

We found that the microbiome played a significant mediation role and OTUs "2434", "171", and "4703" are the specific mediators. From the same run, we can obtain the bivariate associations between `SmokingStatus` and the microbiome, the microbiome and `Outcome` conditional on `SmokingStatus`, both at the community level and the OTU levels. Both bivariate associations have very small p-values, which are consistent with the existence of mediation effects.

```
> res.ldm.med$p.global.omni
           [,1]
[1,] 0.00089991
[2,] 0.00569943
> res.ldm.med$p.otu.omni[,res.ldm.med$med.detected.otu.omni]
             2434          171         4703
[1,] 0.0002499931 0.0067333333 0.0003055471
[2,] 0.0003199872 0.0000399984 0.0002799888
```

11 References

Hu YJ, Satten GA (2020). Testing hypotheses about the microbiome using the linear decomposition model (LDM). *Bioinformatics*, 36(14):4106–4115.

Zhu Z, Satten GA, Caroline M, and Hu YJ (2020). Analyzing matched sets of microbiome data using the LDM and PERMANOVA. *bioRxiv*, 2020.03.06.980367; doi: https://doi.org/10.1101/2020.03.06.980367. In press for *Microbiome*.

Hu YJ, Lane A, Satten GA (2021). A rarefaction-based extension of the LDM for testing presence-absence associations in the microbiome. *Bioinformatics*, btab012, https://doi.org/10.1093/bioinformatics/btab012.

Hu YJ, Satten GA (2021). A rarefaction-without-resampling extension of PERMANOVA for testing presence-absence associations in the microbiome. Under review.

Yue, Y, Hu, YJ (2021) A new approach to testing mediation of the microbiome using the LDM. *bioRxiv*, https://doi.org/10.1101/2021.11.12.468449.

Satten GA et al. (2017) Restoring the duality between principal components of a distance matrix and linear combinations of predictors, with application to studies of the microbiome. *PLoS One*, 12, e0168131.

Besag J, Clifford P (1991). Sequential Monte Carlo p-values. *Biometrika*, 78(2):301–304.

Benjamini Y and Hochberg Y (1995). Controlling the false discovery rate: a practical and powerful approach to multiple testing. *Journal of the royal statistical society*, Series B (Methodological), 289–300.

Sandve GK, Ferkingstad E, Nygard S (2011). Sequential Monte Carlo multiple testing. *Bioinformatics*, 27(23):3235–3241.

Freedman D, Lane D (1983). A nonstochastic interpretation of reported significance levels. *Journal of Business & Economic Statistics*, 1(4):292–298.