Microbiome Data Simulation

Dahun Seo

2024-09-06

차 례

1 Goal of proposed method

Goal

- The proposed methodology aims to identify features with significant signals for the outcome through an outcome model, while minimizing false discoveries.
- It focuses on variable selection utilizing phylogenetic tree information.
- Additionally, the method emphasizes obtaining valid confidence intervals for the estimator, laying the foundation for mediation analysis.

Main references

- Constructing predictive microbial signatures at multiple taxonomic levels, JASA (2017)
- Exact post-selection inference for the generalized lasso path, EJS (2018)

2 Simulation Study

2.1 The logistic normal (LN) distribution

- generate random binary tree with p variables
- calculate cophenetic distanace between variables
- define variance-covariance matrix using distance matrix

$$\Sigma_{ij} = \exp(-d_{ij})/2$$

- generate data from multivariate normal distribution modelled by covairates

$$M_i \sim \mathcal{N}_p(\mu_i, \Sigma)$$

where

$$\mu_i = \alpha_0 + \alpha_t \text{treatment}_i + \alpha_{\text{Sex}}\text{Sex}_i + \alpha_{\text{Age}}\text{Age}_i$$

- transformation to compositional data

$$Z_{ij} = \frac{\exp(M_{ij})}{\sum_{j=1}^{p} \exp(M_{ij})}$$

- log transformation or not

$$Z_{ij} = \log(Z_{ij})$$

- generate outcome variable

$$Y_i = \beta_0 + \beta_t \text{treatment}_i + \beta_{\text{Sex}}\text{Sex}_i + \beta_{\text{Age}}\text{Age}_i + Z_i^\top \beta + \varepsilon$$

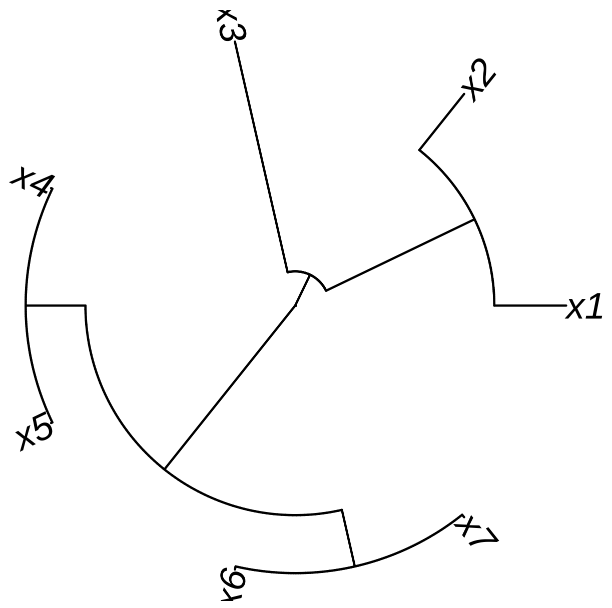
where $\varepsilon \sim \mathcal{N}(0, \sigma^2)$.

```r
# The logistic normal (LN) distribution
n <- 20  # n: sample size
p <- 7  # p: number of features
noise_sigma <- 1  # noise_sigma: noise level for response

# covariate generation
set.seed(1234)
sex <- sample(c(0, 1), n, replace = TRUE)  # sex: 0 or 1
age <- rnorm(n, mean = 50, sd = 10)  # age: N(50, 10^2)

# treatment generation
treatment <- sample(c(0, 1), n, replace = TRUE)  # treatment: 0 or 1

# parameters for normal distribution
base_mu <- rep(0, p)
```

```r
# Create a random binary tree for the p features
random_tree <- ape::rcoal(p)
random_tree$tip.label <- paste0("x", 1:p)
```

```r
ape::plot.phylo(random_tree, type = "fan", main = paste("Random Binary Tree with",
    p, "Variables"))
```

Random Binary Tree with 7 Variables



```r
# Compute the distance matrix using cophenetic
# distances
dist_matrix <- stats::cophenetic(random_tree)

# variance-covariance setting
sigma <- exp(-dist_matrix)/2

# sigma <- matrix(0, p, p) gamma <- 0.5 for(i in
# 1:nrow(sigma)){ for(j in 1:nrow(sigma)){
# sigma[i,j] <- gamma^(abs(i-j)) } }

# coefficients alpha_sex <- rnorm(p, mean = 0.2,
# sd = 0.1) alpha_age <- rnorm(p, mean = 0.01, sd
# = 0.005) alpha_treatment <- rnorm(p, mean =
# 0.2, sd = 0.1)

# coefficients
alpha_sex <- 0
```

```r
alpha_age <- 0
alpha_treatment <- 0

# calculate mu for each i
mu_matrix <- matrix(NA, n, p)
for (i in 1:n) {
    mu_matrix[i, ] <- base_mu + alpha_treatment * treatment[i] +
        alpha_sex * sex[i] + alpha_age * age[i]
}

# generation of logistic normal samples
z <- matrix(NA, n, p)
for (i in 1:n) {
    normal_sample <- MASS::mvrnorm(1, mu_matrix[i,
        ], sigma)
    exp_sample <- exp(normal_sample)
    z[i, ] <- exp_sample/sum(exp_sample)
}

# label
colnames(z) <- paste0("x", 1:p)
```

```r
head(z)
#>              x1         x2         x3         x4         x5         x6
#> [1,] 0.06100137 0.09236474 0.42762350 0.11503238 0.09685259 0.10361977
#> [2,] 0.15059782 0.14995610 0.26655681 0.09599289 0.09945985 0.12263442
#> [3,] 0.24166867 0.37835709 0.14581799 0.07939978 0.08187447 0.03575922
#> [4,] 0.20135813 0.11890826 0.05566617 0.16827019 0.15954574 0.14064553
#> [5,] 0.15445884 0.13187171 0.05885872 0.17393257 0.15786184 0.14164710
#> [6,] 0.23121194 0.12654486 0.06223345 0.11069139 0.10633769 0.15381712
#>              x7
#> [1,] 0.10350565
#> [2,] 0.11480211
#> [3,] 0.03712278
#> [4,] 0.15560597
#> [5,] 0.18136921
#> [6,] 0.20916355
apply(z, 1, sum)
#>  [1] 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
```

```r
# log transformation
z <- log(z)

# coefficients
beta_non_zero <- c(-3, 3, 2.5, -1, -1.5, 3, 3, -2,
    -2, -2, 1, -1, 3, -2, -1)

if (length(beta_non_zero) >= p) {
    beta <- beta_non_zero[1:p]
} else {
    beta <- c(beta_non_zero, rep(0, p - length(beta_non_zero)))
}

beta_sex <- 1
beta_age <- 1
beta_treatment <- 1
base_y <- rep(1, n)

y <- base_y + beta_treatment * treatment + beta_sex *
    sex + beta_age * age + as.vector(z %*% beta) +
    stats::rnorm(n, 0, sd = noise_sigma)
```

2.2 The Dirichlet-multinomial (DM) distribution

- generate random binary tree with p variables
- calculate cophenetic distanace between variables
- define variance-covariance matrix using distance matrix

$$\Sigma_{ij} = \text{exp}(-d_{ij})/2$$

- generate a sample x_0 from multivariate normal distribution $\mathcal{N}_p(0, \Sigma)$.
- generate samples t_i from dirichlet distribution Dirichlet(α_i) where $\alpha_i = x_0 \times \text{exp}(\alpha_t \text{treatment}_i + \alpha_{\text{Sex}} \text{Sex}_i + \alpha_{\text{Age}} \text{Age}_i)$
- generate sequencing depth n_i from negative biomial distribution
- generage count data from multinomial distribution

$$M_i \sim \text{Multi}(n_i, t_i)$$

- transformation to compositional data

$$Z_{ij} = \frac{M_{ij}}{\sum_{j=1}^{p} M_{ij}}$$

- add the psudeo-count 0.5 or not
- log transformation or not

$$Z_{ij} = \text{log}(Z_{ij})$$

- generate outcome variable

$$Y_i = \beta_0 + \beta_t \text{treatment}_i + \beta_{\text{Sex}} \text{Sex}_i + \beta_{\text{Age}} \text{Age}_i + Z_i^\top \beta + \varepsilon$$

where $\varepsilon \sim \mathcal{N}(0, \sigma^2)$.

```r
# The Dirichlet-multinomial (DM) distribution
n <- 20  # n: sample size
p <- 7  # p: number of features
noise_sigma <- 1  # noise_sigma: noise level for response
depth <- stats::rnbinom(n, mu = 10000, size = 25)

# covariate generation
set.seed(1234)
sex <- sample(c(0, 1), n, replace = TRUE)  # sex: 0 or 1
age <- rnorm(n, mean = 50, sd = 10)  # age: N(50, 10^2)

# treatment generation
treatment <- sample(c(0, 1), n, replace = TRUE)  # treatment: 0 or 1

# Create a random binary tree for the p features
random_tree <- ape::rcoal(p)
```

```r
random_tree$tip.label <- paste0("x", 1:p)

# Compute the distance matrix using cophenetic
# distances
dist_matrix <- stats::cophenetic(random_tree)

# variance-covariance setting
cov_matrix <- exp(-dist_matrix)/2

# coefficients base_alpha <- rep(1, p) alpha_sex
# <- rnorm(p, mean = 0.2, sd = 0.1) alpha_age <-
# rnorm(p, mean = 0.01, sd = 0.005)
# alpha_treatment <- rnorm(p, mean = 0.2, sd =
# 0.1)

# coefficients base_alpha <- rep(1, p) alpha_sex
# <- 0 alpha_age <- 0 alpha_treatment <- 0

# coefficients
base_alpha <- exp(MASS::mvrnorm(1, mu = rep(0, p),
    Sigma = cov_matrix))
alpha_sex <- 0
alpha_age <- 0
alpha_treatment <- 0

# calculate mu for each i
alpha_matrix <- matrix(NA, n, p)
for (i in 1:n) {
    alpha_matrix[i, ] <- base_alpha * exp(alpha_treatment *
        treatment[i] + alpha_sex * sex[i] + alpha_age *
        age[i])
}

# generation of Dirichlet-multinomial samples
z <- matrix(NA, nrow = n, ncol = p)
for (i in 1:n) {
    dirichlet_sample <- dirmult::rdirichlet(1, alpha_matrix[i,
        ])
    z[i, ] <- rmultinom(1, size = depth[i], prob = dirichlet_sample)
}
```

```r
# label
colnames(z) <- paste0("x", 1:p)

# add the psudeo-count 0.5
z <- ifelse(z == 0, 0.5, z)

# calculate proportion
z <- z/rowSums(z)
```

```r
# check
head(z)
#>               x1         x2        x3         x4         x5         x6
#> [1,] 0.002236136 0.07289803 0.5538462 0.09919499 0.05044723 0.11797853
#> [2,] 0.034240562 0.03538191 0.5242318 0.07357331 0.14960492 0.06918349
#> [3,] 0.031239831 0.11291897 0.4870648 0.06906931 0.07541490 0.02635861
#> [4,] 0.041269111 0.01287504 0.3372801 0.09575813 0.03851017 0.31704794
#> [5,] 0.090833179 0.07867879 0.2391909 0.36982743 0.03507144 0.06708109
#> [6,] 0.060583726 0.05173939 0.4834906 0.04584316 0.15079599 0.13163325
#>              x7
#> [1,] 0.10339893
#> [2,] 0.11378402
#> [3,] 0.19793362
#> [4,] 0.15725946
#> [5,] 0.11931713
#> [6,] 0.07591392
apply(z, 1, sum)
#>  [1] 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
```

```r
# log transformation
z <- log(z)

# coefficients
beta_non_zero <- c(-3, 3, 2.5, -1, -1.5, 3, 3, -2,
    -2, -2, 1, -1, 3, -2, -1)

if (length(beta_non_zero) >= p) {
    beta <- beta_non_zero[1:p]
} else {
    beta <- c(beta_non_zero, rep(0, p - length(beta_non_zero)))
}
```

```r
beta_sex <- 1
beta_age <- 1
beta_treatment <- 1
base_y <- rep(1, n)

y <- base_y + beta_treatment * treatment + beta_sex *
    sex + beta_age * age + as.vector(z %*% beta) +
    stats::rnorm(n, 0, sd = noise_sigma)
```

3 performance measures

3.1 empirical FDR

$$\widehat{\text{FDR}} = \mathbb{E}_N \left[\frac{|\{j : \beta_j = 0 \text{ and } j \in \widehat{S}\}}{|\widehat{S}| \vee 1} \right]$$

where $\mathbb{E}_N$ denotes the empirical average over replicated simulation.

3.2 empirical power

$$\widehat{\text{Power}} = \mathbb{E}_N \left[\frac{|\{j : \beta_j \neq 0 \text{ and } j \in \widehat{S}\}}{|S^*|} \right]$$

where $\mathbb{E}_N$ denotes the empirical average over replicated simulation.

3.3 empirical coverage?

3.4 model mse?

4 Competitive method

- Compositional knockoff filter for high‐dimensional regression analysis of microbiome data, Biometrics (2021)