Lab 4: Analysis of Microbiome Data in R/Bioconductor

Omics Data Science

Himel Mallick

Contents

Setup

Install the following R packages to get started.

```r
library(ggplot2)
library(Maaslin2)
library(MMUPHin)
library(kableExtra)
library(tidyverse)
library(reshape2)
library(phyloseq)
library(SummarizedExperiment)
library(MultiAssayExperiment)
library(ggplot2)
library(vegan)
library(ecodist)
library(HMP2Data)
library(curatedMetagenomicData)
library(microbiome)
library(igraph)
library(network)
library(SpiecEasi)  #devtools::install_github('zdk123/SpiecEasi')
library(Matrix)
library(igraph)
library(WGCNA)
library(gsEasy)
theme_set(theme_bw())
```

Introduction

The composition of microbial species in a human body is essential for maintaining human health, and it is associated with a number of diseases including obesity, inflammatory bowel diseases, and cancer, among others. Over the last decade, microbiome data analysis almost entirely shifted towards using samples taken directly from various sites of human body and to explore a large number of microbes using 16S or whole metagenome sequencing. This technological shift has led to a radical change in the data collected and led to the creation of the Human Microbiome Project (HMP) in 2008. With the growth and success of the microbiomics field, the size and complexity, and availability of the microbiome data in any given experiment have increased exponentially.

Bioconductor provides curated resources of microbiome data. Most microbiome data are generated either by targeted amplicon sequencing (usually of variable regions of the 16S ribosomal RNA gene) or by metagenomic shotgun sequencing (MGX). These two approaches are analyzed by different sequence analysis tools, but downstream statistical and ecological analysis can involve any of the following types of data:

- taxonomic abundance at different levels of the taxonomic hierarchy
- abundance of microbial genes and gene families

Load the IBD Microbiome Data from the HMP2Data Package

Let's load the 16S microbiome data consisting of 3 groups: Crohn's disease, ulcerative colitis and controls. Rows are OTU IDs and columns are sample names.

```
data("IBD16S_tax")
colnames(IBD16S_tax)
#> [1] "Kingdom" "Phylum"  "Class"   "Order"   "Family"  "Genus"
```

Load the 16S sample annotation data as a matrix, rows are samples, columns are annotations:

```
data("IBD16S_samp")
colnames(IBD16S_samp) %>%
    head()
#> [1] "Project"       "sample_id"     "subject_id"    "site_sub_coll"
#> [5] "data_type"     "week_num"
```

The IBD `phyloseq` object can be loaded as follows. Note that, similar to `SummarizedExperiment` `phyloseq` is a container for microbiome data.

```
IBD <- IBD16S()
IBD
#> phyloseq-class experiment-level object
#> otu_table()   OTU Table:         [ 982 taxa and 178 samples ]
#> sample_data() Sample Data:       [ 178 samples by 490 sample variables ]
#> tax_table()   Taxonomy Table:    [ 982 taxa by 6 taxonomic ranks ]
```

Alpha diversity analysis

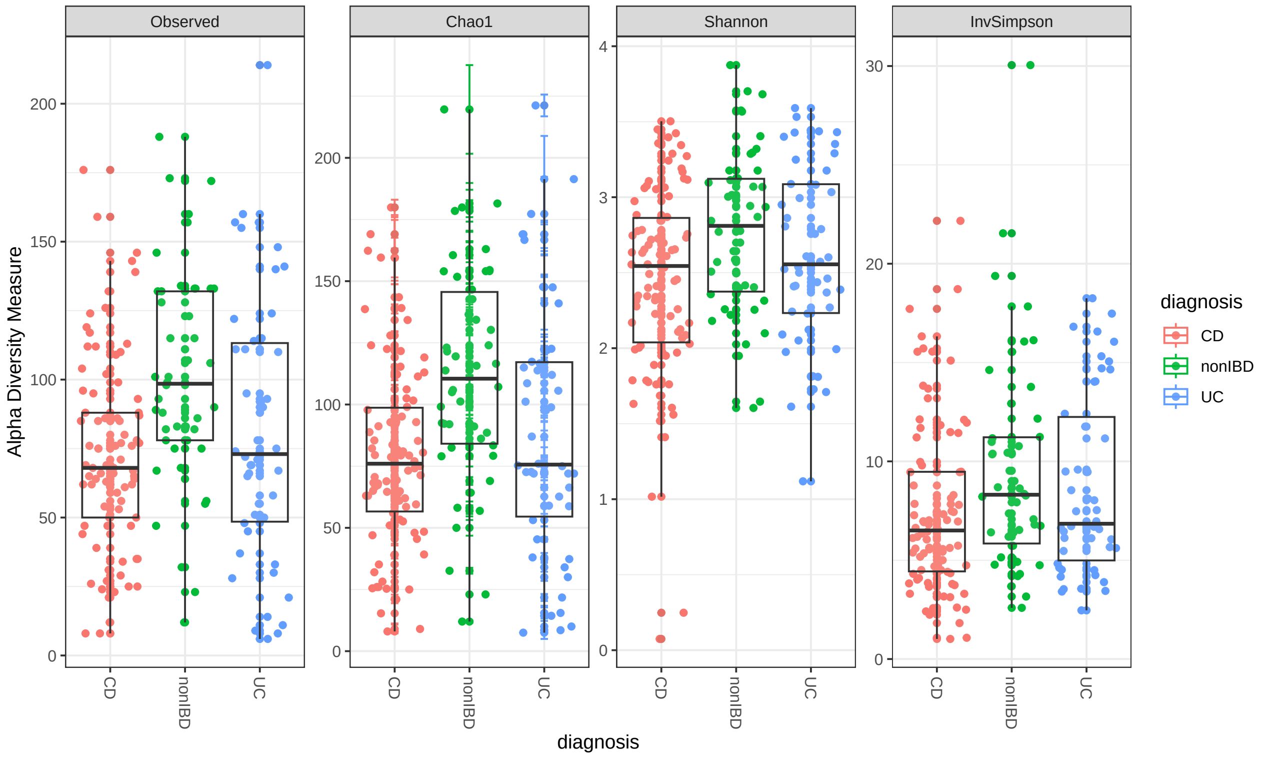
Alpha Diversity Definitions Alpha diversity is a within-sample diversity measurement that models the richness and/or eveness of the microbial community. Different alpha diversities emphasize different aspect of the community. Commonly used alpha diversities include:

Diversity	Description	Formula
Observed Species	No. of unique taxa in a sample	Number of species
Chao 1	Adding a correction to observed species	$S_{ob} + \frac{n_1^2}{2n_2}$
Shannon	Both richness and evenness	$-\sum_{i=1}^{s} p_i \log(p_i)$

Diversity	Description	Formula
Simpson reciprocal	Both richness and evenness	$\frac{N(N-1)}{n(n-1)}$
Pielou's evenness	evenness	$\frac{H}{H_{max}}$

Comparing Alpha Diversities between Groups We can use "plot_richness" from phyloseq package to plot the alpha diversities.

```r
p = plot_richness(IBD, x = "diagnosis", color = "diagnosis",
    measures = c("Observed", "InvSimpson", "Shannon", "Chao1")) +
    geom_jitter()
p + geom_boxplot(data = p$data, aes(x = diagnosis, y = value,
    color = NULL), alpha = 0.1)
```



Once we summarize the whole community through a one-dimensional summary statistics (alpha diversity), all the traditional methods for hypothesis testing are applicable.

```r
alpha = estimate_richness(IBD, measures = c("Observed", "InvSimpson",
    "Shannon"))
alpha_df = data.frame(sample_data(IBD)[, c("sample_id", "subject_id")],
    alpha)
rownames(alpha_df) = NULL
kable(head(alpha_df), digits = 3, caption = "Some Alpha Diversities of the IBD dataset") %>%
    kable_styling(bootstrap_options = c("striped", "hover"),
        full_width = F, position = "center")
```

Table 2: Some Alpha Diversities of the IBD dataset

sample_id	subject_id	Observed	Shannon	InvSimpson
206534	M2008	112	2.572	5.590

206536	M2008	96	2.534	5.565
206538	M2008	80	2.655	6.452
206547	M2014	77	2.733	7.030
206548	M2014	69	2.632	6.556
206561	M2021	63	3.357	16.324

Beta diversity analysis

Beta Diversity Definitions

Beta diversity describes how samples vary against each other. Beta diversity is typically the thinking behind "clustering" algorithms that show differences or similarities among samples. For example, we may be intrested in the differences in gut microbiome between non IBD and IBD patients.

List of the commonly used beta-diversities:

Diversity	Description	Formula
UniFrac	Qualitative, Phylogenetics-based	$\sum_{i=1}^n \frac{b_i \lvert I(p_i^A>0)-I(p_i^B>0)\rvert}{\sum_{i=1}^n b_i}$
Weighted UniFrac	Quantitative,Phylogenetics-based	$\frac{\sum_{i=1}^n b_i \lvert p_i^A-p_i^B\rvert}{\sum_{i=1}^n b_i(p_i^A+p_i^B)}$
Generalized UniFrac	Compromise between the previous two	$\frac{\sum_{i=1}^n b_i(p_i^A+p_i^B)^\alpha \lvert\frac{p_i^A-p_i^B}{p_i^A+p_i^B}\rvert}{\sum_{i=1}^n b_i(p_i^A+p_i^B)^\alpha}$
Bray-Curtis	Quantitative	$1-\frac{2C_{i,j}}{S_i+S_j}$
Jaccard	Qualitative	$\frac{\lvert A\cap B\rvert}{\lvert A\cup B\rvert}$

```r
par(mfrow = c(1, 2))
jac <- as.matrix(distance(t(otu_table(IBD)), method = "jaccard"))
mod1 <- vegan::betadisper(as.dist(jac), as.factor(IBD16S_samp$diagnosis))
bc = as.matrix(ecodist::bcdist(t(otu_table(IBD))))
```

Comparing Beta Diversities between Groups

Permutational Multivariate Analysis of Variance Permutational multivariate analysis of variance (PERMANOVA), a non-parametric multivariate statistical test, is one commonly used for assessing the association between microbiome community and a phenotype (e.g., IBD or not).

```r
IBD0 = subset_samples(IBD, week_num == 0 & diagnosis %in% c("nonIBD",
    "CD"))

jac0 <- as.matrix(distance(t(otu_table(IBD0)), method = "jaccard"))
bc0 = as.matrix(ecodist::bcdist(t(otu_table(IBD0))))
CD = as.numeric(sample_data(IBD0)$diagnosis == "CD")
gender = as.numeric(factor(sample_data(IBD0)$subject_gender))

fit = vegan::adonis(bc0 ~ gender + CD, permutations = 500)
kable(fit$aov.tab, digits = 3, caption = "PERMANOVA result using Bray-Curtis distance") %>%
    kable_styling(bootstrap_options = c("striped", "hover"),
        full_width = F)
```

Table 4: PERMANOVA result using Bray-Curtis distance

	Df	SumsOfSqs	MeanSqs	F.Model	R2	Pr(>F)
gender	1	0.421	0.421	1.288	0.017	0.182
CD	1	0.740	0.740	2.260	0.030	0.018
Residuals	71	23.237	0.327	NA	0.952	NA
Total	73	24.398	NA	NA	1.000	NA

Differential Abundance Analysis and Microbe Set Enrichment Analysis

In this section, we will use the R package `MaAsLin2` to conduct differential abundance analysis of microbiome data. MaAsLin2 is comprehensive R package for efficiently determining multivariable association between phenotypes, environments, exposures, covariates, and microbial multi-omic features. MaAsLin2 relies on general linear models to accommodate most modern epidemiological study designs, including cross-sectional and longitudinal, and offers a variety of data exploration, normalization, and transformation methods.

Similar to gene set enrichment analyses for genes, an obvious next step following differential abundance analysis in microbiome studies is to conduct enrichment for microbe sets, known as microbe set enrichment analysis (MSEA). Similar to GSEA, the primary goal of MSEA is to detect the modest but coordinated changes in pre-specified sets of related microbes. Such a set might include all the microbes in a specific pathway or microbial genes that have been shown to be co-regulated based on previously published studies. Like GSEA, MSEA aggregates the per-feature statistics across microbes within a microbe set. This corresponds to the hypothesis that many relevant phenotype differences are manifested by small but consistent changes in a set of features.

Input data for DA and MSEA

We will use the publicly available Inflammatory Bowel Diseases (IBD) microbiome data from the `curatedMetagenomicData` package. We aim to conduct DA and MSEA analysis based on the taxonomic profiles.

As an aside, a few words about the `curatedMetagenomicData` package which contains the following:

- Processed data from whole-metagenome shotgun metagenomics, with manually-curated metadata, as integrated and documented Bioconductor objects
- ~80 fields of specimen metadata from original papers, supplementary files, and websites, with manual curation to standardize annotations
- Processing of data through the MetaPhlAn2 pipeline for taxonomic abundance, and HUMAnN2 pipeline for metabolic analysis
- These represent ~100TB of raw sequencing data, but the processed data provided are much smaller.

We will first prepare the input feature table and sample metadata for relative abundance data and subset the data to baseline samples to conduct differential analysis.

```r
##################### Load species data #

se_relative <- sampleMetadata |>
    filter(study_name == "HMP_2019_ibdmdb") |>
    returnSamples("relative_abundance", rownames = "short")

########################### Create sample metadata #

sample_metadata <- colData(se_relative) %>%
    as.data.frame() %>%
    filter(visit_number == 1) %>%
    .[, c("age", "disease", "antibiotics_current_use")]  ### check

# Set reference
sample_metadata$disease <- as.factor(sample_metadata$disease)
sample_metadata$disease <- relevel(sample_metadata$disease, "healthy")

########################### Create Species Features #

feature_species_t <- as.data.frame(assay(se_relative))
rownames(feature_species_t) <- sub(".*s__", "", rownames(feature_species_t))

############################## Subset to baseline samples #

feature_species <- as.data.frame(t(feature_species_t))
feature_species <- feature_species[rownames(sample_metadata),
    ]
feature_species <- feature_species/100
rm(feature_species_t)
rm(se_relative)
```

In the first step, we will use the `Maaslin2` function from the `Maaslin2` package to fit a multivariable regression model for testing the association between microbial species abundance versus IBD diagnosis. The analysis method we use here is "LM", which is the default. Age and antibiotics usage will also be adjusted as a covariate in the model.

```
fit_data = Maaslin2(input_data = feature_species, input_metadata = sample_metadata,
    normalization = "NONE", output = "output_species", fixed_effects = c("disease",
        "age", "antibiotics_current_use"))
#> [1] "Creating output folder"
#> [1] "Creating output feature tables folder"
#> [1] "Creating output fits folder"
#> [1] "Creating output figures folder"
#> 2024-12-28 23:20:46.16683 INFO::Writing function arguments to log file
#> 2024-12-28 23:20:46.174 INFO::Verifying options selected are valid
#> 2024-12-28 23:20:46.568236 INFO::Determining format of input files
#> 2024-12-28 23:20:46.568758 INFO::Input format is data samples as rows and metadata samples as rows
#> 2024-12-28 23:20:46.572621 INFO::Formula for fixed effects: expr ~  disease + age + antibiotics_curr
#> 2024-12-28 23:20:46.572993 INFO::Factor detected for categorial metadata 'disease'. Provide a refere
#> 2024-12-28 23:20:46.573306 INFO::Filter data based on min abundance and min prevalence
#> 2024-12-28 23:20:46.573523 INFO::Total samples in data: 136
#> 2024-12-28 23:20:46.573736 INFO::Min samples required with min abundance for a feature not to be fil
#> 2024-12-28 23:20:46.57649 INFO::Total filtered features: 452
#> 2024-12-28 23:20:46.576864 INFO::Filtered feature names from abundance and prevalence filtering: Clo
#> 2024-12-28 23:20:46.57829 INFO::Total filtered features with variance filtering: 0
#> 2024-12-28 23:20:46.578543 INFO::Filtered feature names from variance filtering:
#> 2024-12-28 23:20:46.578765 INFO::Running selected normalization method: NONE
#> 2024-12-28 23:20:46.57904 INFO::Applying z-score to standardize continuous metadata
#> 2024-12-28 23:20:46.581097 INFO::Running selected transform method: LOG
#> 2024-12-28 23:20:46.582678 INFO::Running selected analysis method: LM
#> 2024-12-28 23:20:46.589796 INFO::Fitting model to feature number 1, Phocaeicola.vulgatus
#> 2024-12-28 23:20:46.592511 INFO::Fitting model to feature number 2, Bacteroides.uniformis
#> 2024-12-28 23:20:46.593606 INFO::Fitting model to feature number 3, Bacteroides.thetaiotaomicron
#> 2024-12-28 23:20:46.59462 INFO::Fitting model to feature number 4, Faecalibacterium.prausnitzii
#> 2024-12-28 23:20:46.595621 INFO::Fitting model to feature number 5, Roseburia.faecis
#> 2024-12-28 23:20:46.596604 INFO::Fitting model to feature number 6, Bacteroides.caccae
#> 2024-12-28 23:20:46.597597 INFO::Fitting model to feature number 7, Enterocloster.clostridioformis
#> 2024-12-28 23:20:46.598577 INFO::Fitting model to feature number 8, Bacteroides.fragilis
#> 2024-12-28 23:20:46.599556 INFO::Fitting model to feature number 9, Fusicatenibacter.saccharivorans
#> 2024-12-28 23:20:46.600618 INFO::Fitting model to feature number 10, Flavonifractor.plautii
#> 2024-12-28 23:20:46.601659 INFO::Fitting model to feature number 11, Dialister.invisus
#> 2024-12-28 23:20:46.602671 INFO::Fitting model to feature number 12, Ruminococcus.bicirculans
#> 2024-12-28 23:20:46.603939 INFO::Fitting model to feature number 13, Blautia.sp..CAG.257
#> 2024-12-28 23:20:46.604918 INFO::Fitting model to feature number 14, X.Ruminococcus..gnavus
#> 2024-12-28 23:20:46.606003 INFO::Fitting model to feature number 15, Dorea.longicatena
#> 2024-12-28 23:20:46.606978 INFO::Fitting model to feature number 16, Anaerobutyricum.hallii
#> 2024-12-28 23:20:46.607972 INFO::Fitting model to feature number 17, Enterocloster.bolteae
#> 2024-12-28 23:20:46.609006 INFO::Fitting model to feature number 18, Eubacterium.sp..CAG.38
#> 2024-12-28 23:20:46.610039 INFO::Fitting model to feature number 19, Lacrimispora.saccharolytica
#> 2024-12-28 23:20:46.611022 INFO::Fitting model to feature number 20, Intestinimonas.butyriciproducen
#> 2024-12-28 23:20:46.611972 INFO::Fitting model to feature number 21, Anaerostipes.hadrus
#> 2024-12-28 23:20:46.612959 INFO::Fitting model to feature number 22, Anaerotignum.lactatifermentans
#> 2024-12-28 23:20:46.61399 INFO::Fitting model to feature number 23, Ruthenibacterium.lactatiformans
#> 2024-12-28 23:20:46.615018 INFO::Fitting model to feature number 24, Lawsonibacter.asaccharolyticus
#> 2024-12-28 23:20:46.616011 INFO::Fitting model to feature number 25, Clostridium.bolteae.CAG.59
#> 2024-12-28 23:20:46.616964 INFO::Fitting model to feature number 26, X.Clostridium..symbiosum
#> 2024-12-28 23:20:46.617948 INFO::Fitting model to feature number 27, Anaerotruncus.colihominis
#> 2024-12-28 23:20:46.618979 INFO::Fitting model to feature number 28, Agathobaculum.butyriciproducens
#> 2024-12-28 23:20:46.620014 INFO::Fitting model to feature number 29, Escherichia.coli
#> 2024-12-28 23:20:46.621068 INFO::Fitting model to feature number 30, Eisenbergiella.massiliensis
#> 2024-12-28 23:20:46.622012 INFO::Fitting model to feature number 31, Enterocloster.citroniae
#> 2024-12-28 23:20:46.622993 INFO::Fitting model to feature number 32, Bacteroides.stercoris
#> 2024-12-28 23:20:46.624012 INFO::Fitting model to feature number 33, Monoglobus.pectinilyticus
#> 2024-12-28 23:20:46.625039 INFO::Fitting model to feature number 34, Alistipes.putredinis
#> 2024-12-28 23:20:46.626022 INFO::Fitting model to feature number 35, Parabacteroides.distasonis
#> 2024-12-28 23:20:46.626965 INFO::Fitting model to feature number 36, Roseburia.hominis
#> 2024-12-28 23:20:46.627951 INFO::Fitting model to feature number 37, Bifidobacterium.longum
#> 2024-12-28 23:20:46.628989 INFO::Fitting model to feature number 38, X.Eubacterium..siraeum
#> 2024-12-28 23:20:46.630068 INFO::Fitting model to feature number 39, Parabacteroides.merdae
#> 2024-12-28 23:20:46.631082 INFO::Fitting model to feature number 40, Roseburia.inulinivorans
#> 2024-12-28 23:20:46.632057 INFO::Fitting model to feature number 41, Bacteroides.cellulosilyticus
#> 2024-12-28 23:20:46.633047 INFO::Fitting model to feature number 42, Bacteroides.ovatus
#> 2024-12-28 23:20:46.634062 INFO::Fitting model to feature number 43, Blautia.wexlerae
#> 2024-12-28 23:20:46.635131 INFO::Fitting model to feature number 44, Bifidobacterium.adolescentis
#> 2024-12-28 23:20:46.636157 INFO::Fitting model to feature number 45, Lachnospira.eligens
#> 2024-12-28 23:20:46.637181 INFO::Fitting model to feature number 46, X.Clostridium..leptum
#> 2024-12-28 23:20:46.638181 INFO::Fitting model to feature number 47, Eggerthella.lenta
#> 2024-12-28 23:20:46.639132 INFO::Fitting model to feature number 48, Bacteroides.xylanisolvens
#> 2024-12-28 23:20:46.640138 INFO::Fitting model to feature number 49, Phocaeicola.dorei
#> 2024-12-28 23:20:46.641186 INFO::Fitting model to feature number 50, Enterocloster.lavalensis
#> 2024-12-28 23:20:46.64222 INFO::Fitting model to feature number 51, X.Eubacterium..rectale
#> 2024-12-28 23:20:46.64324 INFO::Fitting model to feature number 52, Akkermansia.muciniphila
#> 2024-12-28 23:20:46.644353 INFO::Fitting model to feature number 53, Acidaminococcus.intestini
#> 2024-12-28 23:20:46.6454 INFO::Fitting model to feature number 54, X.Ruminococcus..torques
#> 2024-12-28 23:20:46.646395 INFO::Fitting model to feature number 55, Bacteroides.salyersiae
#> 2024-12-28 23:20:46.647364 INFO::Fitting model to feature number 56, Sellimonas.intestinalis
#> 2024-12-28 23:20:46.648326 INFO::Fitting model to feature number 57, Bacteroides.faecis
#> 2024-12-28 23:20:46.649297 INFO::Fitting model to feature number 58, Roseburia.intestinalis
#> 2024-12-28 23:20:46.650263 INFO::Fitting model to feature number 59, Collinsella.aerofaciens
#> 2024-12-28 23:20:46.651277 INFO::Fitting model to feature number 60, Bacteroides.intestinalis
#> 2024-12-28 23:20:46.652319 INFO::Fitting model to feature number 61, Bacteroides.faecis.CAG.32
#> 2024-12-28 23:20:46.653357 INFO::Fitting model to feature number 62, X.Ruminococcus..lactaris
#> 2024-12-28 23:20:46.654395 INFO::Fitting model to feature number 63, Prevotella.copri
#> 2024-12-28 23:20:46.655418 INFO::Fitting model to feature number 64, Eisenbergiella.tayi
#> 2024-12-28 23:20:46.656456 INFO::Fitting model to feature number 65, Klebsiella.pneumoniae
#> 2024-12-28 23:20:46.657488 INFO::Fitting model to feature number 66, Intestinibacter.bartlettii
#> 2024-12-28 23:20:46.658535 INFO::Fitting model to feature number 67, Erysipelatoclostridium.ramosum
#> 2024-12-28 23:20:46.659576 INFO::Fitting model to feature number 68, Barnesiella.intestinihominis
#> 2024-12-28 23:20:46.660602 INFO::Fitting model to feature number 69, Blautia.obeum
#> 2024-12-28 23:20:46.661624 INFO::Fitting model to feature number 70, Bacteroides.finegoldii
#> 2024-12-28 23:20:46.662584 INFO::Fitting model to feature number 71, Parasutterella.excrementihomini
#> 2024-12-28 23:20:46.663627 INFO::Fitting model to feature number 72, Phascolarctobacterium.faecium
#> 2024-12-28 23:20:46.664655 INFO::Fitting model to feature number 73, Lachnospira.pectinoschiza
#> 2024-12-28 23:20:46.665677 INFO::Fitting model to feature number 74, Clostridium.sp..CAG.58
#> 2024-12-28 23:20:46.666752 INFO::Fitting model to feature number 75, Bilophila.wadsworthia
#> 2024-12-28 23:20:46.667781 INFO::Fitting model to feature number 76, Turicimonas.muris
#> 2024-12-28 23:20:46.668805 INFO::Fitting model to feature number 77, Proteobacteria.bacterium.CAG.13
#> 2024-12-28 23:20:46.669858 INFO::Fitting model to feature number 78, Anaeromassilibacillus.sp..An250
#> 2024-12-28 23:20:46.670899 INFO::Fitting model to feature number 79, Hungatella.hathewayi
#> 2024-12-28 23:20:46.671926 INFO::Fitting model to feature number 80, Alistipes.finegoldii
#> 2024-12-28 23:20:46.672967 INFO::Fitting model to feature number 81, Ruminococcus.bromii
#> 2024-12-28 23:20:46.674004 INFO::Fitting model to feature number 82, Odoribacter.splanchnicus
#> 2024-12-28 23:20:46.675032 INFO::Fitting model to feature number 83, Butyricimonas.virosa
#> 2024-12-28 23:20:46.676059 INFO::Fitting model to feature number 84, Oscillibacter.sp..CAG.241
#> 2024-12-28 23:20:46.677016 INFO::Fitting model to feature number 85, Alistipes.indistinctus
#> 2024-12-28 23:20:46.678008 INFO::Fitting model to feature number 86, Coprococcus.comes
#> 2024-12-28 23:20:46.679099 INFO::Fitting model to feature number 87, Gemmiger.formicilis
#> 2024-12-28 23:20:46.680213 INFO::Fitting model to feature number 88, Holdemania.filiformis
#> 2024-12-28 23:20:46.681278 INFO::Fitting model to feature number 89, Firmicutes.bacterium.CAG.83
#> 2024-12-28 23:20:46.68236 INFO::Fitting model to feature number 90, Dorea.formicigenerans
#> 2024-12-28 23:20:46.683411 INFO::Fitting model to feature number 91, Collinsella.intestinalis
#> 2024-12-28 23:20:46.684474 INFO::Fitting model to feature number 92, Oscillibacter.sp..57_20
#> 2024-12-28 23:20:46.685517 INFO::Fitting model to feature number 93, Firmicutes.bacterium.CAG.94
#> 2024-12-28 23:20:46.68655 INFO::Fitting model to feature number 94, Dielma.fastidiosa
#> 2024-12-28 23:20:46.687588 INFO::Fitting model to feature number 95, Roseburia.sp..CAG.471
#> 2024-12-28 23:20:46.688623 INFO::Fitting model to feature number 96, X.Clostridium..innocuum
#> 2024-12-28 23:20:46.689657 INFO::Fitting model to feature number 97, Haemophilus.parainfluenzae
#> 2024-12-28 23:20:46.690689 INFO::Fitting model to feature number 98, Veillonella.dispar
#> 2024-12-28 23:20:46.691717 INFO::Fitting model to feature number 99, Veillonella.parvula
#> 2024-12-28 23:20:46.692781 INFO::Fitting model to feature number 100, Veillonella.infantium
#> 2024-12-28 23:20:46.693808 INFO::Fitting model to feature number 101, Streptococcus.salivarius
#> 2024-12-28 23:20:46.694836 INFO::Fitting model to feature number 102, Enterocloster.aldenensis
#> 2024-12-28 23:20:46.695863 INFO::Fitting model to feature number 103, Veillonella.atypica
#> 2024-12-28 23:20:46.696886 INFO::Fitting model to feature number 104, Phocaeicola.plebeius
#> 2024-12-28 23:20:46.69793 INFO::Fitting model to feature number 105, Paraprevotella.xylaniphila
#> 2024-12-28 23:20:46.699284 INFO::Fitting model to feature number 106, Bacteroides.eggerthii
#> 2024-12-28 23:20:46.703557 INFO::Fitting model to feature number 107, Alistipes.shahii
#> 2024-12-28 23:20:46.704513 INFO::Fitting model to feature number 108, Butyricimonas.synergistica
#> 2024-12-28 23:20:46.705471 INFO::Fitting model to feature number 109, Haemophilus.sp..HMSC71H05
#> 2024-12-28 23:20:46.706417 INFO::Fitting model to feature number 110, Eubacterium.ramulus
#> 2024-12-28 23:20:46.70736 INFO::Fitting model to feature number 111, Coprobacter.fastidiosus
#> 2024-12-28 23:20:46.708308 INFO::Fitting model to feature number 112, Lactobacillus.rogosae
#> 2024-12-28 23:20:46.709243 INFO::Fitting model to feature number 113, Phocaeicola.massiliensis
#> 2024-12-28 23:20:46.710216 INFO::Fitting model to feature number 114, Streptococcus.parasanguinis
#> 2024-12-28 23:20:46.71121 INFO::Fitting model to feature number 115, Ruminococcaceae.bacterium.D16
#> 2024-12-28 23:20:46.712216 INFO::Fitting model to feature number 116, Veillonella.sp..T11011.6
#> 2024-12-28 23:20:46.713252 INFO::Fitting model to feature number 117, Klebsiella.variicola
#> 2024-12-28 23:20:46.714244 INFO::Fitting model to feature number 118, Bacteroides.galacturonicus
#> 2024-12-28 23:20:46.715277 INFO::Fitting model to feature number 119, Clostridium.sp..CAG.299
#> 2024-12-28 23:20:46.716281 INFO::Fitting model to feature number 120, Eubacterium.ventriosum
#> 2024-12-28 23:20:46.717314 INFO::Fitting model to feature number 121, Bifidobacterium.bifidum
#> 2024-12-28 23:20:46.718451 INFO::Fitting model to feature number 122, Bifidobacterium.pseudocatenula
#> 2024-12-28 23:20:46.719494 INFO::Fitting model to feature number 123, Coprococcus.eutactus
#> 2024-12-28 23:20:46.720455 INFO::Fitting model to feature number 124, Eubacterium.sp..CAG.251
#> 2024-12-28 23:20:46.721424 INFO::Fitting model to feature number 125, Phocaeicola.coprocola
#> 2024-12-28 23:20:46.72244 INFO::Fitting model to feature number 126, Paraprevotella.clara
#> 2024-12-28 23:20:46.723459 INFO::Fitting model to feature number 127, Coprococcus.catus
#> 2024-12-28 23:20:46.733034 INFO::Counting total values for each feature
#> 2024-12-28 23:20:46.738078 INFO::Writing filtered data to file output_species/features/filtered_data
#> 2024-12-28 23:20:46.744604 INFO::Writing filtered, normalized data to file output_species/features/f
#> 2024-12-28 23:20:46.750832 INFO::Writing filtered, normalized, transformed data to file output_speci
#> 2024-12-28 23:20:46.760636 INFO::Writing residuals to file output_species/fits/residuals.rds
#> 2024-12-28 23:20:46.764323 INFO::Writing fitted values to file output_species/fits/fitted.rds
#> 2024-12-28 23:20:46.767451 INFO::Writing all results to file (ordered by increasing q-values): outpu
#> 2024-12-28 23:20:46.769142 INFO::Writing the significant results (those which are less than or equal
#> 2024-12-28 23:20:46.769649 INFO::Writing heatmap of significant results to file: output_species/heat
#> 2024-12-28 23:20:46.798 INFO::Writing association plots (one for each significant association) to ou
#> 2024-12-28 23:20:46.799304 INFO::Plotting associations from most to least significant, grouped by me
#> 2024-12-28 23:20:46.799585 INFO::Plotting data for metadata number 1, age
#> 2024-12-28 23:20:46.800057 INFO::Creating scatter plot for continuous data, age vs Alistipes.indisti
#> 2024-12-28 23:20:46.869958 INFO::Creating scatter plot for continuous data, age vs Ruminococcus.bici
#> 2024-12-28 23:20:46.949713 INFO::Creating scatter plot for continuous data, age vs Lacrimispora.sacc
#> 2024-12-28 23:20:47.003931 INFO::Creating scatter plot for continuous data, age vs Ruminococcus.brom
#> 2024-12-28 23:20:47.058074 INFO::Creating scatter plot for continuous data, age vs Bifidobacterium.p
#> 2024-12-28 23:20:47.116434 INFO::Creating scatter plot for continuous data, age vs X.Ruminococcus..g
#> 2024-12-28 23:20:47.181036 INFO::Creating scatter plot for continuous data, age vs Lachnospira.elige
#> 2024-12-28 23:20:47.236121 INFO::Creating scatter plot for continuous data, age vs Haemophilus.parai
#> 2024-12-28 23:20:47.297516 INFO::Creating scatter plot for continuous data, age vs Butyricimonas.vir
#> 2024-12-28 23:20:47.353575 INFO::Creating scatter plot for continuous data, age vs Lawsonibacter.asa
#> 2024-12-28 23:20:47.42027 INFO::Creating scatter plot for continuous data, age vs X.Eubacterium..sir
#> 2024-12-28 23:20:47.476219 INFO::Creating scatter plot for continuous data, age vs Butyricimonas.syn
#> 2024-12-28 23:20:47.533639 INFO::Creating scatter plot for continuous data, age vs Monoglobus.pectin
#> 2024-12-28 23:20:47.589651 INFO::Creating scatter plot for continuous data, age vs Veillonella.atypi
#> 2024-12-28 23:20:48.253357 INFO::Plotting data for metadata number 2, disease
#> 2024-12-28 23:20:48.254406 INFO::Creating boxplot for categorical data, disease vs Alistipes.putredi
#> 2024-12-28 23:20:48.312665 INFO::Creating boxplot for categorical data, disease vs Gemmiger.formicil
#> 2024-12-28 23:20:48.361037 INFO::Creating boxplot for categorical data, disease vs X.Ruminococcus..t
#> 2024-12-28 23:20:48.409011 INFO::Creating boxplot for categorical data, disease vs Ruminococcus.bici
#> 2024-12-28 23:20:48.46329 INFO::Creating boxplot for categorical data, disease vs Sellimonas.intesti
#> 2024-12-28 23:20:48.524217 INFO::Creating boxplot for categorical data, disease vs X.Clostridium..le
#> 2024-12-28 23:20:48.573384 INFO::Creating boxplot for categorical data, disease vs Alistipes.shahii
#> 2024-12-28 23:20:48.995776 INFO::Plotting data for metadata number 3, antibiotics_current_use
#> 2024-12-28 23:20:48.996486 INFO::Creating boxplot for categorical data, antibiotics_current_use vs C
#> 2024-12-28 23:20:49.044094 INFO::Creating boxplot for categorical data, antibiotics_current_use vs X
#> 2024-12-28 23:20:49.104534 INFO::Creating boxplot for categorical data, antibiotics_current_use vs A
```

Let's visually examine the results of the MaAsLin2 analysis. Note that, MaAsLin2 produces an output folder which contains the statistically significant results as well as plots. **PLEASE EXAMINE THE OUTPUT FOLDER FIRST BEFORE RETURNING TO THE REST OF THE LAB!!!**

In the second step, we will use the residuals extracted from the model for the MSEA analysis.

```
datExpr <- as.data.frame(t(fit_data$residuals))
```

Note that, unlike gene expression studies, we do not have well-defined signatures or modules for microbiome data. Here, we will construct data-driven modules using WGCNA. By working on the residuals, we want to eliminate the effect of the covariates so that the modules are only biologically related as the effect of disease and other covariates has been removed. First, we will check whether there are outliers in our data.

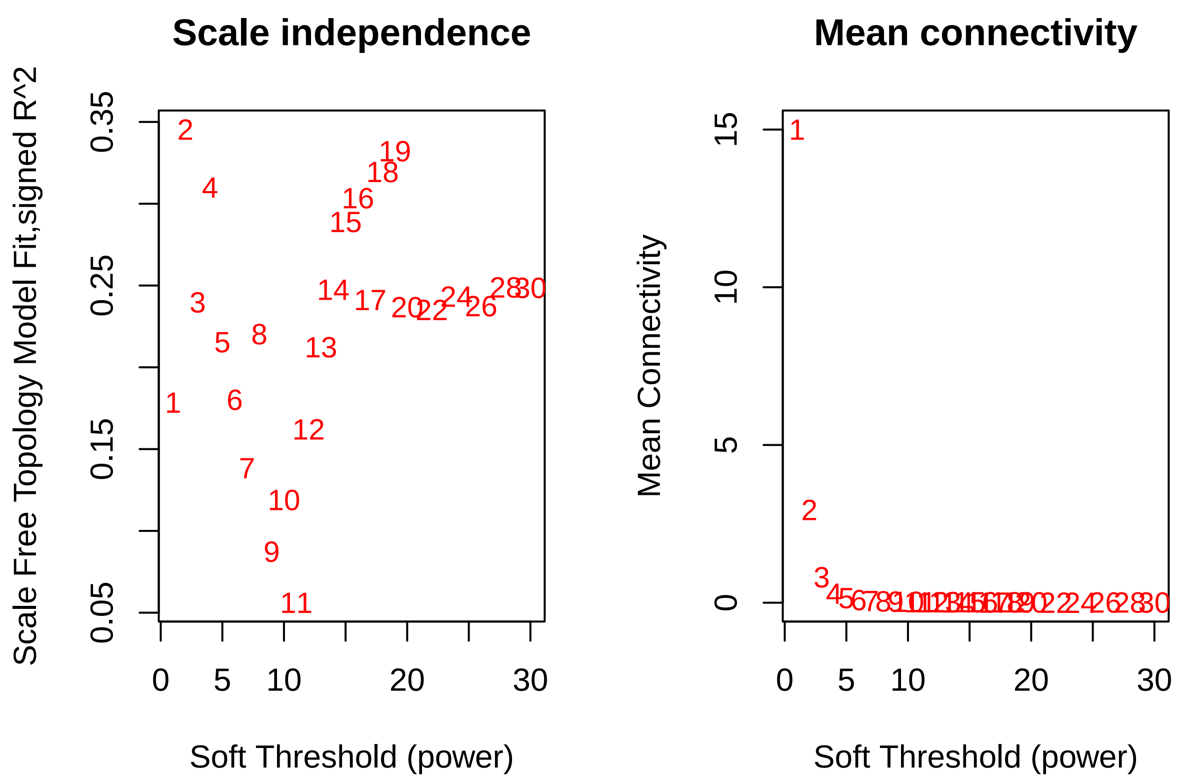
```
gsg = goodSamplesGenes(datExpr, verbose = 3)
#>  Flagging genes and samples with too many missing values...
#>   ..step 1
#>   ..Excluding 6 samples from the calculation due to too many missing genes.
#>   ..step 2
gsg$allOK
#> [1] FALSE
```

If the last statement returns TRUE, all genes have passed the cuts. If not, we need to remove the offending genes and samples from the data.

```
if (!gsg$allOK) {
    if (sum(!gsg$goodGenes) > 0)
        printFlush(paste("Removing genes:", paste(names(datExpr)[!gsg$goodGenes],
            collapse = ", ")))
    if (sum(!gsg$goodSamples) > 0)
        printFlush(paste("Removing samples:", paste(rownames(datExpr)[!gsg$goodSamples],
            collapse = ", ")))
    datExpr = datExpr[gsg$goodSamples, gsg$goodGenes]
}
#> Removing samples: CSM5MCVB_P, CSM79HNY_P, ESM5GEYY_P, ESM718U9_P, MSM6J2N6_P, MSM9VZLX_P
```

After removing the outliers, we need to choose a suitable soft threshold parameter for creating the modules, where the correlation values are corrected based on a correction factor. This factor can enhance the differences between strong and weak correlations, which making the weak values become closer to zero. This power value must produce a graph similar to a scale-free network. We can use the mean connectivity graphic for the selection of this power parameter.

```
# Choose a set of soft threshold parameters
powers = c(c(1:20), seq(from = 22, to = 30, by = 2))
sft = pickSoftThreshold(datExpr, powerVector = powers, verbose = 5,
    dataIsExpr = T, RsquaredCut = 0.3)
#> pickSoftThreshold: will use block size 127.
#>  pickSoftThreshold: calculating connectivity for given powers...
#>    ..working on genes 1 through 127 of 127
#>    Power SFT.R.sq  slope truncated.R.sq  mean.k. median.k.  max.k.
#> 1      1    0.1790 -0.972        0.85700 15.00000  1.48e+01 23.0000
#> 2      2    0.3450 -0.729        0.63500  2.95000  2.85e+00  5.6900
#> 3      3    0.2400 -4.130        0.12700  0.80400  7.36e-01  2.6100
#> 4      4    0.3100 -4.870        0.19900  0.29200  2.14e-01  1.6700
#> 5      5    0.2150 -4.030       -0.00880  0.13700  7.02e-02  1.2200
#> 6      6    0.1800 -2.930       -0.05130  0.07910  2.46e-02  0.9570
#> 7      7  0.1380 -2.240       -0.10100  0.05320  9.33e-03  0.7900
#> 8      8  0.2200 -2.790       -0.00259  0.03940  3.71e-03  0.6690
#> 9      9  0.0872 -1.450       -0.09130  0.03100  1.50e-03  0.5760
#> 10    10  0.1190 -1.630       -0.07660  0.02540  6.73e-04  0.5020
#> 11    11  0.0562 -1.250       -0.03470  0.02130  2.66e-04  0.4410
#> 12    12  0.1620 -2.150       -0.07720  0.01810  1.12e-04  0.3910
#> 13    13  0.2120 -2.350       -0.00949  0.01560  4.84e-05  0.3480
#> 14    14  0.2480 -2.410        0.04950  0.01360  2.12e-05  0.3110
#> 15    15  0.2890 -2.470        0.15200  0.01190  9.53e-06  0.2810
#> 16    16  0.3030 -2.420        0.18400  0.01050  4.33e-06  0.2580
#> 17    17  0.2410 -2.380        0.02450  0.00933  1.98e-06  0.2380
#> 18    18  0.3190 -2.640        0.15400  0.00830  9.03e-07  0.2180
#> 19    19  0.3320 -2.600        0.17700  0.00741  4.03e-07  0.2010
#> 20    20  0.2370 -2.050        0.02180  0.00664  1.70e-07  0.1840
#> 21    22  0.2350 -2.000        0.07520  0.00536  3.40e-08  0.1560
#> 22    24  0.2430 -2.230        0.09560  0.00437  6.98e-09  0.1310
#> 23    26  0.2370 -2.120        0.26200  0.00358  1.44e-09  0.1110
#> 24    28  0.2490 -2.080        0.26200  0.00294  2.93e-10  0.0937
#> 25    30  0.2490 -2.290        0.30100  0.00243  5.67e-11  0.0791

# Scale-free topology fit index as a function of the
# soft-thresholding power
par(mfrow = c(1, 2))
cex1 = 0.9
plot(sft$fitIndices[, 1], -sign(sft$fitIndices[, 3]) * sft$fitIndices[,
    2], xlab = "Soft Threshold (power)", ylab = "Scale Free Topology Model Fit,signed R^2",
    type = "n", main = paste("Scale independence"))
text(sft$fitIndices[, 1], -sign(sft$fitIndices[, 3]) * sft$fitIndices[,
    2], labels = powers, cex = cex1, col = "red")

# This line corresponds to using an R^2 cut-off of h
abline(h = 0.9, col = "red")

# Mean connectivity as a function of the soft-thresholding
# power
plot(sft$fitIndices[, 1], sft$fitIndices[, 5], xlab = "Soft Threshold (power)",
    ylab = "Mean Connectivity", type = "n", main = paste("Mean connectivity"))
text(sft$fitIndices[, 1], sft$fitIndices[, 5], labels = powers,
    cex = cex1, col = "red")
```



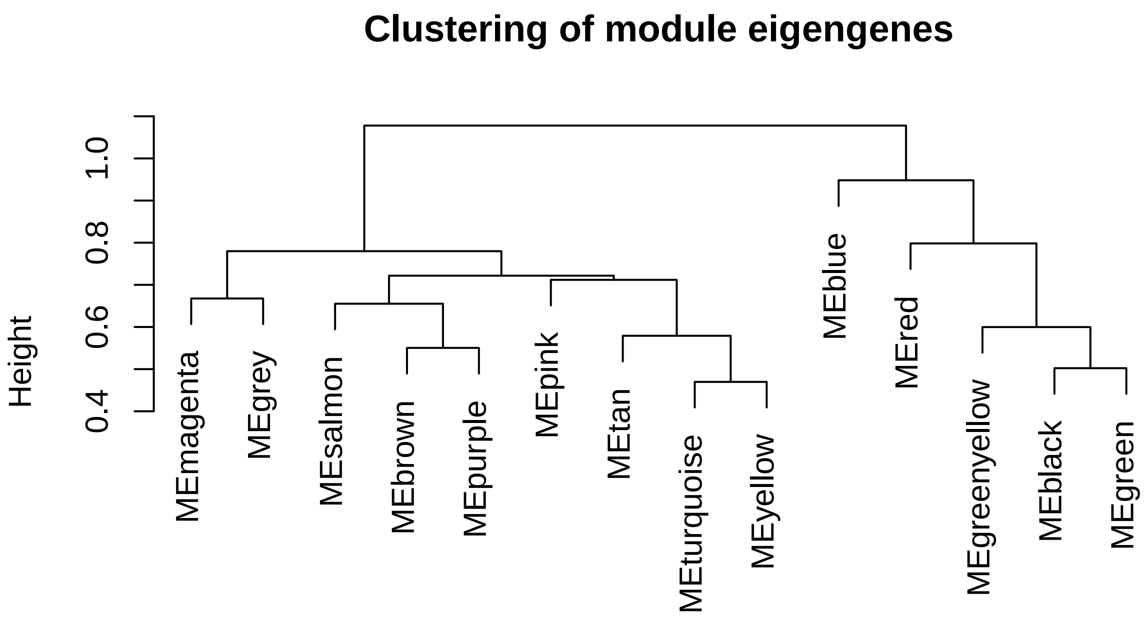
```
sft$powerEstimate
#> [1] 2
```

In this step, we will conduct an one-step module detection based on the soft threshold parameter we selected above with `blockwiseModules` function.

```
power = sft$powerEstimate
net = blockwiseModules(datExpr, power = power, corFnc = "bicor",
    corOptions = list(maxPOutliers = 0.1), networkType = "unsigned",
    maxBlockSize = ncol(datExpr), minModuleSize = 3, TOMType = "unsigned",
    reassignThreshold = 0, mergeCutHeight = 0, verbose = 3)
#>  Calculating module eigengenes block-wise from all genes
#>    Flagging genes and samples with too many missing values...
#>     ..step 1
#>  ..Working on block 1 .
#>     TOM calculation: adjacency..
#>     ..will not use multithreading.
#>      Fraction of slow calculations: 0.000000
#>     ..connectivity..
#>     ..matrix multiplication (system BLAS)..
#>     ..normalization..
#>     ..done.
#>  ....clustering..
#>  ....detecting modules..
#>  ....calculating module eigengenes..
#>  ....checking kME in modules..
#>      ..removing 3 genes from module 1 because their KME is too low.
#>      ..removing 3 genes from module 2 because their KME is too low.
#>      ..removing 2 genes from module 3 because their KME is too low.
#>      ..removing 1 genes from module 5 because their KME is too low.
#>      ..removing 1 genes from module 7 because their KME is too low.
#>  ..merging modules that are too close..
#>      mergeCloseModules: Merging modules whose distance is less than 0
#>        Calculating new MEs...

#################### How many modules #

ncol(net$MEs)  # 14
#> [1] 14
table(net$colors)
#>
#>       black        blue       brown       green greenyellow        grey
#>           7          14          13          12           5          10
#>     magenta        pink      purple         red      salmon         tan
#>           6           6           5          11           3           5
#>   turquoise      yellow
#>          18          12
```

Note that, we have detected 14 modules. Let's visualize them.

```
########################## Plot module dendrogram #

eigenGenes <- net$MEs
MEDiss = 1 - cor(eigenGenes)
METree = hclust(as.dist(MEDiss), method = "average")
plot(METree, main = "Clustering of module eigengenes", xlab = "",
    sub = "")
```



Next, we will create mapping files for the MSEA analysis

```
########################################### Re-calculate
########################################### modules and
########################################### find hub genes
########################################### #
```

```r
moduleColors <- net$colors
MEs0 = moduleEigengenes(datExpr, moduleColors)$eigengenes
modules_data = orderMEs(MEs0)

######################## Create mapping file #

feature_by_modules <- as.data.frame(net$colors)
feature_by_modules <- rownames_to_column(feature_by_modules)
colnames(feature_by_modules) <- c("Feature", "Module")
features_mapping <- feature_by_modules
features_mapping$Module <- paste("ME", features_mapping$Module,
    sep = "")
```

Now, we will run the MSEA analysis on the predefined set of signatures. We will use the same wrapper from Lab 2.

```r
################
# MSEA Wrapper #
################

run_MSEA <- function(geneSet, # A list
                     ranked_features, # Ranked list of featured
                     filter.count = 3,
                     seed = 1234,
                     fdr.correction = 'BH') {


  ###################
  # Filter out sets #
  ##################

  geneSet0 <- geneSet
  cond <- sapply(geneSet0, function(x) length(x) > filter.count)
  geneSet <- geneSet0[cond]
  lengthGeneSet <- as.data.frame(reshape2::melt(lapply(geneSet, function(x) length(x))))
  colnames(lengthGeneSet) <- c('Freq','Set')

  ################
  # Classic MSEA #
  ################

  set.seed(seed)
  enrichment <- as.data.frame(sapply(geneSet, function(set) gset(S = set, r = ranked_features)))
  colnames(enrichment)<-'ES'
  enrichment <- rownames_to_column(enrichment, 'Set')
  enrichment <- merge(enrichment, lengthGeneSet, 'Set')
  enrichment$qval <- p.adjust(enrichment$ES, fdr.correction)

  ##########
  # Return #
  ##########

  return(enrichment)
}
```

Before running the MSEA, we also need to rank the differential analysis results and prepare the MSEA input as before.

```
################### Rank DA results #

results <- fit_data$results %>%
    filter(metadata == "disease")
results$qval <- p.adjust(results$pval, "BH")
sum(results$qval < 0.05)
#> [1] 0
results <- results[order(results$qval, decreasing = FALSE), ]

################### MSEA Processing #

library(topGO)

module_map <- features_mapping
mod.gs <- tapply(module_map$Module, module_map$Feature, as.character)
geneSet <- inverseList(mod.gs)
geneSet
#> $MEblack
#> [1] "Anaeromassilibacillus.sp..An250" "Anaerotruncus.colihominis"
#> [3] "Blautia.wexlerae"                "Eisenbergiella.tayi"
#> [5] "Firmicutes.bacterium.CAG.94"     "Ruthenibacterium.lactatiformans"
#> [7] "Sellimonas.intestinalis"
#>
#> $MEblue
#>  [1] "Clostridium.sp..CAG.58"         "Erysipelatoclostridium.ramosum"
#>  [3] "Haemophilus.parainfluenzae"     "Intestinibacter.bartlettii"
#>  [5] "Klebsiella.pneumoniae"          "Klebsiella.variicola"
#>  [7] "Lawsonibacter.asaccharolyticus" "Streptococcus.parasanguinis"
#>  [9] "Streptococcus.salivarius"       "Veillonella.atypica"
#> [11] "Veillonella.dispar"             "Veillonella.infantium"
#> [13] "Veillonella.parvula"            "Veillonella.sp..T11011.6"
#>
#> $MEbrown
#>  [1] "Alistipes.finegoldii"        "Alistipes.indistinctus"
#>  [3] "Alistipes.putredinis"        "Alistipes.shahii"
#>  [5] "Bacteroides.xylanisolvens"   "Bilophila.wadsworthia"
#>  [7] "Firmicutes.bacterium.CAG.83" "Odoribacter.splanchnicus"
#>  [9] "Oscillibacter.sp..57_20"     "Oscillibacter.sp..CAG.241"
#> [11] "Phocaeicola.dorei"           "Ruminococcus.bromii"
#> [13] "X.Eubacterium..siraeum"
#>
#> $MEgreen
#>  [1] "Blautia.sp..CAG.257"            "Clostridium.bolteae.CAG.59"
#>  [3] "Eggerthella.lenta"              "Eisenbergiella.massiliensis"
#>  [5] "Enterocloster.aldenensis"       "Enterocloster.bolteae"
#>  [7] "Enterocloster.citroniae"        "Enterocloster.clostridioformis"
#>  [9] "Enterocloster.lavalensis"       "Flavonifractor.plautii"
#> [11] "Lacrimispora.saccharolytica"    "X.Clostridium..symbiosum"
#>
```

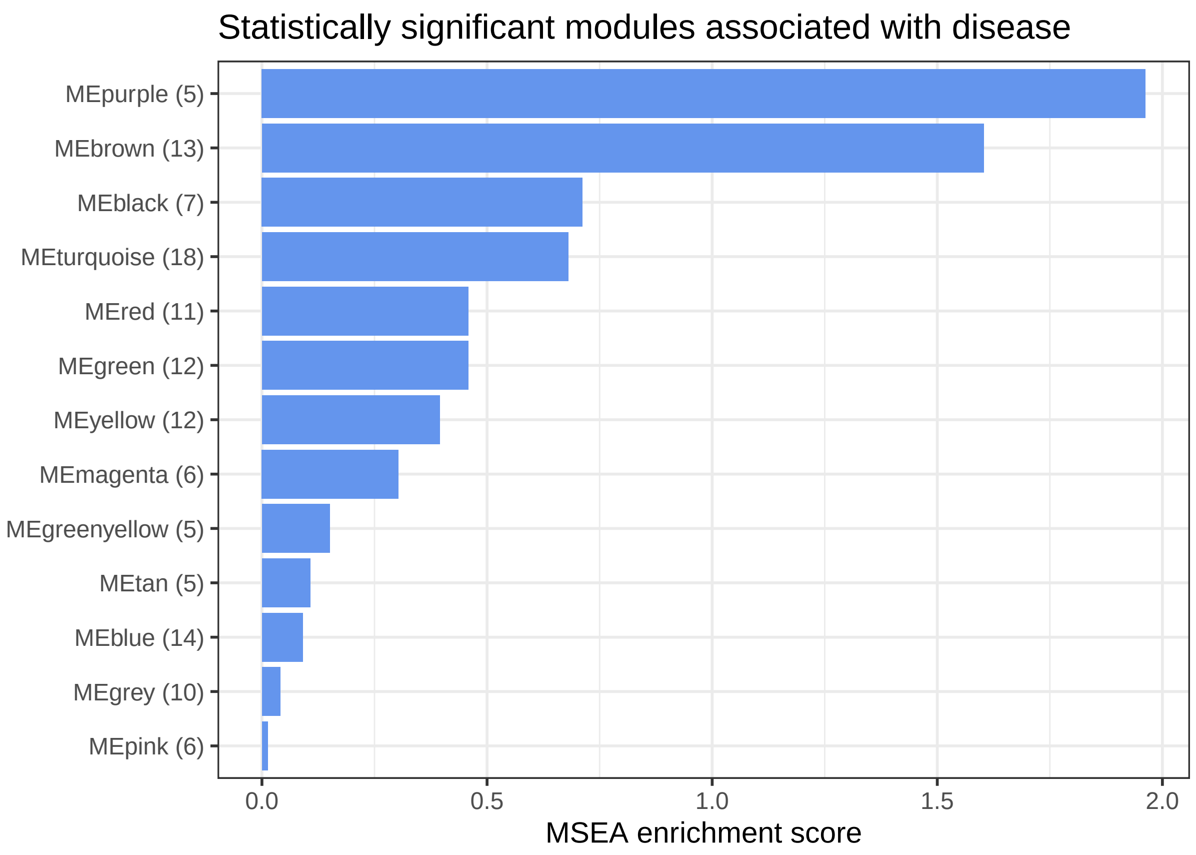
```
#> $MEgreenyellow
#> [1] "Coprobacter.fastidiosus" "Escherichia.coli"
#> [3] "Hungatella.hathewayi"    "X.Clostridium..innocuum"
#> [5] "X.Ruminococcus..gnavus"
#>
#> $MEgrey
#>  [1] "Anaerotignum.lactatifermentans"   "Bacteroides.cellulosilyticus"
#>  [3] "Bacteroides.fragilis"             "Collinsella.intestinalis"
#>  [5] "Eubacterium.sp..CAG.38"           "Haemophilus.sp..HMSC71H05"
#>  [7] "Intestinimonas.butyriciproducens" "Lachnospira.eligens"
#>  [9] "Roseburia.intestinalis"           "Roseburia.sp..CAG.471"
#>
#> $MEmagenta
#> [1] "Bacteroides.finegoldii"     "Paraprevotella.clara"
#> [3] "Paraprevotella.xylaniphila" "Phocaeicola.coprocola"
#> [5] "Phocaeicola.plebeius"       "Prevotella.copri"
#>
#> $MEpink
#> [1] "Bacteroides.galacturonicus"        "Bifidobacterium.pseudocatenulatum"
#> [3] "Eubacterium.sp..CAG.251"           "Lachnospira.pectinoschiza"
#> [5] "Lactobacillus.rogosae"             "Phocaeicola.massiliensis"
#>
#> $MEpurple
#> [1] "Barnesiella.intestinihominis" "Butyricimonas.synergistica"
#> [3] "Butyricimonas.virosa"         "Coprococcus.eutactus"
#> [5] "Ruminococcus.bicirculans"
#>
#> $MEred
#>  [1] "Akkermansia.muciniphila"           "Bacteroides.intestinalis"
#>  [3] "Clostridium.sp..CAG.299"           "Dialister.invisus"
#>  [5] "Dielma.fastidiosa"                 "Holdemania.filiformis"
#>  [7] "Monoglobus.pectinilyticus"         "Parasutterella.excrementihominis"
#>  [9] "Proteobacteria.bacterium.CAG.139" "Turicimonas.muris"
#> [11] "X.Clostridium..leptum"
#>
#> $MEsalmon
#> [1] "Bacteroides.faecis"              "Bacteroides.faecis.CAG.32"
#> [3] "Phascolarctobacterium.faecium"
#>
#> $MEtan
#> [1] "Bacteroides.salyersiae"       "Bifidobacterium.adolescentis"
#> [3] "Bifidobacterium.bifidum"      "Bifidobacterium.longum"
#> [5] "Collinsella.aerofaciens"
#>
#> $MEturquoise
#>  [1] "Agathobaculum.butyriciproducens" "Anaerobutyricum.hallii"
#>  [3] "Anaerostipes.hadrus"             "Bacteroides.eggerthii"
#>  [5] "Blautia.obeum"                   "Coprococcus.catus"
#>  [7] "Coprococcus.comes"               "Dorea.formicigenerans"
#>  [9] "Dorea.longicatena"               "Eubacterium.ramulus"
#> [11] "Faecalibacterium.prausnitzii"    "Fusicatenibacter.saccharivorans"
#> [13] "Gemmiger.formicilis"             "Roseburia.faecis"
#> [15] "Roseburia.hominis"               "Roseburia.inulinivorans"
#> [17] "X.Eubacterium..rectale"          "X.Ruminococcus..torques"
#> 
#> $MEyellow
#>  [1] "Acidaminococcus.intestini"     "Bacteroides.caccae"
#>  [3] "Bacteroides.ovatus"            "Bacteroides.stercoris"
#>  [5] "Bacteroides.thetaiotaomicron"  "Bacteroides.uniformis"
#>  [7] "Eubacterium.ventriosum"        "Parabacteroides.distasonis"
#>  [9] "Parabacteroides.merdae"        "Phocaeicola.vulgatus"
#> [11] "Ruminococcaceae.bacterium.D16" "X.Ruminococcus..lactaris"
```

We are finally ready to run the MSEA.

```
MSEA <- run_MSEA(geneSet, results$feature)
# MSEA <- select(MSEA, c('Set', 'Freq', 'ES'),
# everything())
MSEA <- MSEA[, c("Set", "Freq", "ES", setdiff(names(MSEA), c("Set",
    "Freq", "ES")))]  ### Check
colnames(MSEA) <- c("ID", "Size", "pval", "qval")
MSEA$ID <- paste(MSEA$ID, " (", MSEA$Size, ")", sep = "")
```

We can use the bar plot to visualize the MSEA results.

```
p <- MSEA %>%
    arrange(-pval) %>%
    mutate(ID = factor(ID, levels = ID)) %>%
    ggplot(aes(y = -log10(pval), x = ID)) + geom_bar(stat = "identity",
    fill = "cornflowerblue") + theme_bw() + coord_flip() + ggtitle("Statistically significant modules a
    xlab("") + ylab("MSEA enrichment score")
print(p)
```



```
# Print the most significant modules

geneSet[["MEpurple"]]
#> [1] "Barnesiella.intestinihominis" "Butyricimonas.synergistica"
#> [3] "Butyricimonas.virosa"         "Coprococcus.eutactus"
#> [5] "Ruminococcus.bicirculans"
geneSet[["MEbrown"]]
#>  [1] "Alistipes.finegoldii"         "Alistipes.indistinctus"
#>  [3] "Alistipes.putredinis"         "Alistipes.shahii"
#>  [5] "Bacteroides.xylanisolvens"    "Bilophila.wadsworthia"
#>  [7] "Firmicutes.bacterium.CAG.83" "Odoribacter.splanchnicus"
#>  [9] "Oscillibacter.sp..57_20"      "Oscillibacter.sp..CAG.241"
#> [11] "Phocaeicola.dorei"            "Ruminococcus.bromii"
#> [13] "X.Eubacterium..siraeum"
```

Based on the MSEA results, we see that there 13 modules enriched for the outcome of interest.

Ecological network construction

The widely reported compositionality bias in similarity measures can be fixed with SpiecEasi or SparCC; the implementations are available via the SpiecEasi package. Note that the execution is slow. See the phyloseq tutorial for additional network visualization tools.

Analysis of American Gut data

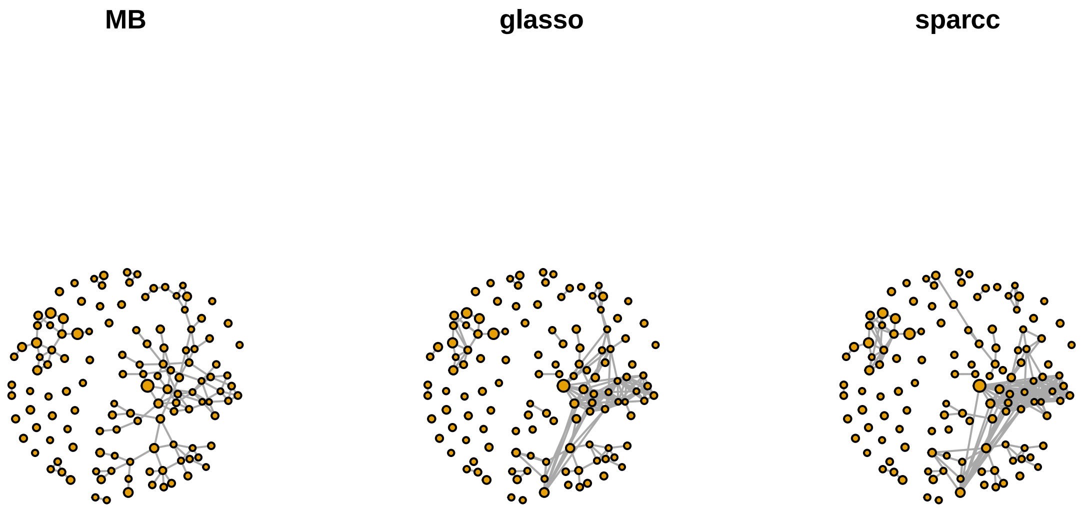
Now let's apply SpiecEasi directly to the American Gut data. Don't forget that the normalization is performed internally in the `spiec.easi` function. Also, we should use a larger number of stars repetitions for real data.

We can pass in arguments to the inner stars selection function as a list via the parameter `pulsar.params`. If you have more than one processor available, you can also supply a number to `ncores`. Also, let's compare results from the MB and glasso methods as well as SparCC (correlation).

```
data(amgut1.filt)
se.mb.amgut <- spiec.easi(amgut1.filt, method = "mb", lambda.min.ratio = 0.01,
    nlambda = 20, pulsar.params = list(rep.num = 50))
se.gl.amgut <- spiec.easi(amgut1.filt, method = "glasso", lambda.min.ratio = 0.01,
    nlambda = 20, pulsar.params = list(rep.num = 50))
sparcc.amgut <- sparcc(amgut1.filt)

## Define arbitrary threshold for SparCC correlation matrix
## for the graph
sparcc.graph <- abs(sparcc.amgut$Cor) >= 0.3
diag(sparcc.graph) <- 0

sparcc.graph <- Matrix(sparcc.graph, sparse = TRUE)

## Create igraph objects
ig.mb <- adj2igraph(getRefit(se.mb.amgut))
ig.gl <- adj2igraph(getRefit(se.gl.amgut))
ig.sparcc <- adj2igraph(sparcc.graph)
```

Visualize using igraph plotting:

```
## set size of vertex proportional to clr-mean
vsize <- rowMeans(clr(amgut1.filt, 1)) + 6
am.coord <- layout.fruchterman.reingold(ig.mb)

par(mfrow = c(1, 3))
plot(ig.mb, layout = am.coord, vertex.size = vsize, vertex.label = NA,
    main = "MB")
plot(ig.gl, layout = am.coord, vertex.size = vsize, vertex.label = NA,
    main = "glasso")
plot(ig.sparcc, layout = am.coord, vertex.size = vsize, vertex.label = NA,
    main = "sparcc")
```



Homework 4

- Repeat the DA and MSEA analysis using the same data with `Tweedieverse` and compare the results with `MaAsLin2`. Note that, we are only interested in the effect of IBD disease status. Make sure the comparison is made with respect to IBD differential abundance while adjusting for age and antibiotic usage.
- By changing the `dataType` to `pathway_abundance` in the `curatedMetagenomicData()` function, repeat the analysis above (both DA and MSEA) to find modules of pathways enriched in IBD. Perform a comparative study of `MaAsLin2` and `Tweedieverse`. Note that, you need to prepare the WGCNA modules to run the MSEA as we don't have access to curated 'microbe sets'.
- The SPIEC-EASI method demonstrated above uses graphical LASSO to estimate the microbiome correlation network. `BAnOCC` is a method which improves upon SPIEC-EASI by considering a Bayesian graphical LASSO approach. Repeat the network analysis above using `BAnOCC` on the American Gut data and compare the results graphically.

Session information

```
sessionInfo()
#> R version 4.4.2 (2024-10-31)
#> Platform: aarch64-apple-darwin20
#> Running under: macOS Sequoia 15.2
#> 
#> Matrix products: default
#> BLAS:   /Library/Frameworks/R.framework/Versions/4.4-arm64/Resources/lib/libRblas.0.dylib
#> LAPACK: /Library/Frameworks/R.framework/Versions/4.4-arm64/Resources/lib/libRlapack.dylib;  LAPACK v
#> 
#> locale:
#> [1] en_US.UTF-8/en_US.UTF-8/en_US.UTF-8/C/en_US.UTF-8/en_US.UTF-8
#> 
#> time zone: Asia/Kolkata
#> tzcode source: internal
#> 
#> attached base packages:
#> [1] stats4    stats     graphics  grDevices utils     datasets  methods  
#> [8] base     
#> 
#> other attached packages:
#>  [1] topGO_2.56.0                    SparseM_1.84-2                 
#>  [3] GO.db_3.19.1                    AnnotationDbi_1.66.0           
#>  [5] graph_1.82.0                    gsEasy_1.5                     
#>  [7] WGCNA_1.73                      fastcluster_1.2.6              
#>  [9] dynamicTreeCut_1.63-1           Matrix_1.7-1                   
#> [11] SpiecEasi_1.1.3                 network_1.19.0                 
#> [13] igraph_2.1.2                    microbiome_1.26.0              
#> [15] curatedMetagenomicData_3.12.0   TreeSummarizedExperiment_2.12.0
#> [17] Biostrings_2.72.1               XVector_0.44.0                 
#> [19] SingleCellExperiment_1.26.0     HMP2Data_1.18.0                
#> [21] ecodist_2.1.3                   vegan_2.6-8                    
#> [23] lattice_0.22-6                  permute_0.9-7                  
#> [25] MultiAssayExperiment_1.30.3     SummarizedExperiment_1.34.0    
#> [27] Biobase_2.64.0                  GenomicRanges_1.56.2           
#> [29] GenomeInfoDb_1.40.1             IRanges_2.38.1                 
#> [31] S4Vectors_0.42.1                BiocGenerics_0.50.0            
#> [33] MatrixGenerics_1.16.0          matrixStats_1.4.1
#> [35] phyloseq_1.48.0                reshape2_1.4.4
#> [37] lubridate_1.9.4                forcats_1.0.0
#> [39] stringr_1.5.1                  dplyr_1.1.4
#> [41] purrr_1.0.2                    readr_2.1.5
#> [43] tidyr_1.3.1                    tibble_3.2.1
#> [45] tidyverse_2.0.0                kableExtra_1.4.0
#> [47] MMUPHin_1.18.1                 Maaslin2_1.18.0
#> [49] ggplot2_3.5.1                  knitr_1.49
#>
#> loaded via a namespace (and not attached):
#>   [1] fs_1.6.5                    DirichletMultinomial_1.46.0
#>   [3] RColorBrewer_1.1-3          httr_1.4.7
#>   [5] doParallel_1.0.17           tools_4.4.2
#>   [7] backports_1.5.0             R6_2.5.1
#>   [9] lazyeval_0.2.2              mgcv_1.9-1
#>  [11] rhdf5filters_1.16.0         withr_3.0.2
#>  [13] gridExtra_2.3               preprocessCore_1.66.0
#>  [15] cli_3.6.3                   formatR_1.14
#>  [17] logging_0.10-108            biglm_0.9-3
#>  [19] labeling_0.4.3              mvtnorm_1.3-2
#>  [21] robustbase_0.99-4-1         pbapply_1.7-2
#>  [23] systemfonts_1.1.0           yulab.utils_0.1.8
#>  [25] foreign_0.8-87              svglite_2.1.3
#>  [27] scater_1.32.1               decontam_1.24.0
#>  [29] huge_1.3.5                  VGAM_1.1-12
#>  [31] rstudioapi_0.17.1           impute_1.78.0
#>  [33] RSQLite_2.3.9               generics_0.1.3
#>  [35] shape_1.4.6.1               biomformat_1.32.0
#>  [37] ggbeeswarm_0.7.2            DECIPHER_3.0.0
#>  [39] abind_1.4-8                 lifecycle_1.0.4
#>  [41] yaml_2.3.10                 rhdf5_2.48.0
#>  [43] SparseArray_1.4.8           BiocFileCache_2.12.0
#>  [45] Rtsne_0.17                  grid_4.4.2
#>  [47] blob_1.2.4                  ExperimentHub_2.12.0
#>  [49] crayon_1.5.3                beachmat_2.20.0
#>  [51] KEGGREST_1.44.1             pillar_1.10.0
#>  [53] optparse_1.7.5              pulsar_0.3.11
#>  [55] codetools_0.2-20            glue_1.8.0
#>  [57] data.table_1.16.4           vctrs_0.6.5
#>  [59] png_0.1-8                   treeio_1.28.0
#>  [61] gtable_0.3.6                assertthat_0.2.1
#>  [63] cachem_1.1.0                xfun_0.49
#>  [65] S4Arrays_1.4.1              mime_0.12
#>  [67] coda_0.19-4.1               pcaPP_2.0-5
#>  [69] survival_3.8-3              pheatmap_1.0.12
#>  [71] iterators_1.0.14            tinytex_0.54
#>  [73] bluster_1.14.0              nlme_3.1-166
#>  [75] bit64_4.5.2                 filelock_1.0.3
#>  [77] irlba_2.3.5.1               vipor_0.4.7
#>  [79] rpart_4.1.23                colorspace_2.1-1
#>  [81] DBI_1.2.3                   Hmisc_5.2-1
#>  [83] nnet_7.3-19                 ade4_1.7-22
#>  [85] tidyselect_1.2.1            bit_4.5.0.1
#>  [87] compiler_4.4.2              curl_6.0.1
#>  [89] ontologyIndex_2.12          glmnet_4.1-8
#>  [91] htmlTable_2.4.3             BiocNeighbors_1.22.0
#>  [93] xml2_1.3.6                  DelayedArray_0.30.1
#>  [95] checkmate_2.3.2             scales_1.3.0
#>  [97] DEoptimR_1.1-3-1            rappdirs_0.3.3
#>  [99] digest_0.6.37               rmarkdown_2.29
#> [101] htmltools_0.5.8.1           pkgconfig_2.0.3
#> [103] base64enc_0.1-3             sparseMatrixStats_1.16.0
#> [105] dbplyr_2.5.0                fastmap_1.2.0
#> [107] rlang_1.1.4                 htmlwidgets_1.6.4
#> [109] UCSC.utils_1.0.0            DelayedMatrixStats_1.26.0
#> [111] farver_2.1.2                jsonlite_1.8.9
#> [113] BiocParallel_1.38.0         statnet.common_4.10.0
#> [115] BiocSingular_1.20.0         magrittr_2.0.3
#> [117] Formula_1.2-5               scuttle_1.14.0
#> [119] GenomeInfoDbData_1.2.12     Rhdf5lib_1.26.0
#> [121] munsell_0.5.1               Rcpp_1.0.13-1
#> [123] ape_5.8-1                   viridis_0.6.5
#> [125] stringi_1.8.4               zlibbioc_1.50.0
#> [127] MASS_7.3-61                 AnnotationHub_3.12.0
#> [129] plyr_1.8.9                  parallel_4.4.2
#> [131] ggrepel_0.9.6               splines_4.4.2
#> [133] hash_2.2.6.3                multtest_2.60.0
#> [135] hms_1.1.3                   ScaledMatrix_1.12.0
#> [137] BiocVersion_3.19.1          evaluate_1.0.1
#> [139] BiocManager_1.30.25         tzdb_0.4.0
#> [141] foreach_1.5.2               getopt_1.20.4
#> [143] rsvd_1.0.5                  tidytree_0.4.6
#> [145] viridisLite_0.4.2           memoise_2.0.1
#> [147] beeswarm_0.4.0              cluster_2.1.8
#> [149] timechange_0.3.0            mia_1.12.0
```