DspikeIn with Phyloseq

Mitra Ghotbi

2025-06-20

Required Packages

```
#                          INSTALL CRAN PACKAGES

# Install missing CRAN packages
install.packages(setdiff(
  c(
    "stats", "dplyr", "ggplot2", "flextable", "ggpubr",
    "randomForest", "ggridges", "ggalluvial", "tibble",
    "matrixStats", "RColorBrewer", "ape", "rlang",
    "scales", "magrittr", "phangorn", "igraph", "tidyr",
    "xml2", "data.table", "reshape2", "vegan", "patchwork", "officer"
  ),
  installed.packages()[, "Package"]
))

# Load CRAN packages
lapply(c(
  "stats", "dplyr", "ggplot2", "flextable", "ggpubr", "randomForest",
  "ggridges", "ggalluvial", "tibble", "matrixStats", "RColorBrewer",
  "ape", "rlang", "scales", "magrittr", "phangorn", "igraph", "tidyr",
  "xml2", "data.table", "reshape2", "vegan", "patchwork", "officer"
), library, character.only = TRUE)


#                          INSTALL BIOCONDUCTOR PACKAGES


# Install BiocManager if not installed
if (!requireNamespace("BiocManager", quietly = TRUE)) install.packages("BiocManager")

# Install missing Bioconductor packages
BiocManager::install(setdiff(
  c(
```

```
    "phyloseq", "msa", "DESeq2", "ggtree", "edgeR",
    "Biostrings", "DECIPHER", "microbiome", "limma",
    "S4Vectors", "SummarizedExperiment", "TreeSummarizedExperiment"
  ),
  installed.packages()[, "Package"]
))

# Load Bioconductor packages
lapply(
  c(
    "phyloseq", "msa", "DESeq2", "edgeR", "Biostrings", "ggtree", "DECIPHER",
    "microbiome", "limma", "S4Vectors", "SummarizedExperiment", "TreeSummarizedExperiment"
  ),
  library,
  character.only = TRUE
)


#                          INSTALL DspikeIn FROM GITHUB


# To access the DspikeIn vignette for a detailed tutorial, use vignette("DspikeIn"), or browse
devtools::install_github("mghotbi/DspikeIn", build_vignettes = TRUE, dependencies = TRUE)
library(DspikeIn)
browseVignettes("DspikeIn")
vignette("DspikeIn")

## or

if (!requireNamespace("devtools", quietly = TRUE)) install.packages("devtools")
devtools::install_github("mghotbi/DspikeIn")

# Load DspikeIn only if installed
if ("DspikeIn" %in% installed.packages()[, "Package"]) {
  library(DspikeIn)
} else {
  stop("DspikeIn installation failed. Check errors above.")
}
```

Acknowledgments

The development of the DspikeIn package was made possible through the generous and pioneering efforts of the R and Bioconductor communities. We gratefully acknowledge the developers and maintainers of the following open-source packages, whose tools and infrastructure underpin our work: **Core infrastructure & data manipulation:** methods, stats, utils, graphics, grDevices,

data.table, dplyr, tibble, tidyr, reshape2, matrixStats, rlang, S4Vectors, grid, officer, xml2 **Statistical analysis & modeling:** DESeq2, edgeR, limma, randomForest, microbiome **Phylogenetics & microbiome structure:** phyloseq, TreeSummarizedExperiment, SummarizedExperiment, phangorn, ape, DECIPHER, msa, Biostrings **Network and graph analysis:** igraph, ggraph **Visualization & layout design:** ggplot2, ggrepel, ggpubr, ggnewscale, ggalluvial, ggtree, ggtreeExtra, ggstar, ggridges, patchwork, scales, RColorBrewer, flextable

These tools collectively empowered us to build a reproducible, modular, and extensible platform for robust absolute abundance quantification in microbial community analysis. We further acknowledge the broader scientific community working on absolute microbial quantification, spike-in calibration, and compositional data analysis, whose foundational insights directly informed the design and conceptual framework of DspikeIn.

DspikeIn

The DspikeIn package supports both phyloseq and TreeSummarizedExperiment formats to streamline microbial quantification across diverse experimental setups. It accommodates either a single spike-in taxon or synthetic community taxa with variable or equal spike-in volumes and copy numbers. The package offers a comprehensive suite of tools for AA quantification, addressing challenges through ten core functions: 1) validation of spiked species, 2) data preprocessing, 3) system-specific spiked species retrieval, 4) scaling factor calculation, 5) conversion to absolute abundance, 6) bias correction and normalization, 7) performance assessment, and 8) taxa exploration and filtering 9) network topology assessment 10) further analyses and visualization.

DspikeIn requirements

DspikeIn works with 7 taxonomic ranks

```
#             To remove strain from the taxonomic ranks

# DspikeIn works with 7 taxonomic ranks
# colnames(taxmat) <- c("Kingdom", "Phylum", "Class", "Order", "Family", "Genus", "Species")

library(phyloseq)
# Function to remove strain information from taxonomy columns
remove_strain_info <- function(tax_table) {
  # Define the regex pattern for common strain identifiers
  pattern <- "Strain.*|strain.*|\\s*\\[.*\\]|\\s*\\(.*\\)" # Adjust the regex to match specifi
  # Apply the pattern to each column of the taxonomy table
  for (col in colnames(tax_table)) {
    tax_table[, col] <- gsub(pattern, "", tax_table[, col]) # Remove strain info
    tax_table[, col] <- trimws(tax_table[, col]) # Trim trailing whitespace
  }
  return(tax_table)
}
```

```r
# Step 1: Extract the taxonomy table
taxonomy <- tax_table(ps)
# Step 2: Remove strain information (including the `Strain` column)
cleaned_taxonomy <- remove_strain_info(taxonomy)
# Remove the `Strain` column if it exists
if ("Strain" %in% colnames(cleaned_taxonomy)) {
  cleaned_taxonomy <- cleaned_taxonomy[, colnames(cleaned_taxonomy) != "Strain"]
}
# Step 3: Update the taxonomy table in the phyloseq object
tax_table(ps) <- cleaned_taxonomy
# Step 4: Verify the changes
print(head(tax_table(ps))) # Display the first few rows



#             To add species rank to the taxonomic ranks


library(phyloseq)

# Step 1: Extract taxonomy table safely
taxonomy <- as.data.frame(as.matrix(tax_table(ps)))
if (!"Genus" %in% colnames(taxonomy)) {
  stop("The 'Genus' column is missing in the taxonomy table.")
}

# Step 2: Handle potential missing genera (optional but recommended)
taxonomy$Genus[is.na(taxonomy$Genus) | taxonomy$Genus == ""] <- "Unknown"
# Step 3: Create a new 'species' column
taxonomy$species <- paste0(taxonomy$Genus, "_OTU", seq_len(nrow(taxonomy)))
# Step 4: Assign back to phyloseq object (must coerce back to matrix)
tax_table(ps) <- tax_table(as.matrix(taxonomy))
```

Build phyloseq or TreeSummarizedExperiment file

for more information please refer to https://github.com/joey711/phyloseq & https://github.com/markrobinsonuzh/TreeSummarizedExperiment

```r
#                     Build phyloseq


otu <- read.csv("otu.csv", header = TRUE, sep = ",", row.names = 1)
# taxonomic rank need to be capilalized, only the first letter of each rank
tax <- read.csv("tax.csv", header = TRUE, sep = ",", row.names = 1)
# Ensure 'spiked.volume' column is present and correctly formatted in metadata
```

```
meta <- read.csv("metadata.csv", header = TRUE, sep = ",")

# Convert data to appropriate formats
meta <- as.data.frame(meta)
taxmat <- as.matrix(tax)
otumat <- as.matrix(otu)
colnames(taxmat) <- c("Kingdom", "Phylum", "Class", "Order", "Family", "Genus", "Species")
OTU <- otu_table(otumat, taxa_are_rows = TRUE)
TAX <- phyloseq::tax_table(taxmat)

# Check
row.names(meta) <- sample_names(OTU)
metadata <- sample_data(meta)
# Build phyloseq obj
physeq <- phyloseq(OTU, TAX, metadata)

# Follow the next steps if tree and reference files are included
MyTree <- read.tree("tree.nwk")
reference_seqs <- readDNAStringSet(file = "dna-sequences.fasta", format = "fasta")

physeq_16SOTU <- merge_phyloseq(physeq, reference_seqs, MyTree)
physeq_16SOTU <- tidy_phyloseq_tse(physeq_16SOTU)

saveRDS(physeq_16SOTU, file = "physeq_16SOTU.rds")
physeq_16SOTU <- readRDS("physeq_16SOTU.rds")



#                       Build TreeSummarizedExperiment (TSE)


otu <- read.csv("otu.csv", header = TRUE, sep = ",", row.names = 1)
otu_mat <- as.matrix(otu) # Convert to matrix
tax <- read.csv("tax.csv", header = TRUE, sep = ",", row.names = 1)
colnames(tax) <- c("Kingdom", "Phylum", "Class", "Order", "Family", "Genus", "Species")
tax_mat <- as.matrix(tax) # Convert to matrix
meta <- read.csv("metadata.csv", header = TRUE, sep = ",", row.names = 1)
reference_seqs <- readDNAStringSet("dna-sequences.fasta", format = "fasta")
tse <- TreeSummarizedExperiment(
  assays = list(counts = otu_mat), # OTU table
  rowData = tax_mat, # Taxonomy information
  colData = meta, # Sample metadata
  rowTree = MyTree, # Phylogenetic tree
  rowSeqs = reference_seqs # Reference sequences
)
```

Do all detected sample spike-in sequences cluster with the reference, and are their branch lengths

statistically similar, supporting a common ancestor?

spike-in validation

All sample-derived sequences are forming a clade with the reference. We look for a monophyletic grouping of spike-in OTUs The clade is strongly supported (bootstrap around 100 percentage). The branch lengths and distances are in a biologically plausible range.

```
# Use the Neighbor-Joining method  based on a Jukes-Cantor distance matrix and plot the tree w
# we compare the Sanger read of Tetragenococcus halophilus with the FASTA sequence of Tetragen

library(Biostrings)
library(phyloseq)
library(DspikeIn)

# Get path to external data folder
extdata_path <- system.file("extdata", package = "DspikeIn")
list.files(extdata_path)
```

```
## [1] "Complete.graphml" "NoBasid.graphml"  "NoHubs.graphml"   "Ref.fasta"        "Sample.fast
```

```
data("physeq_16SOTU", package = "DspikeIn")

physeq_16SOTU <- tidy_phyloseq_tse(physeq_16SOTU)

physeq_16SOTU <- phyloseq::prune_taxa(
  get_tax_table(physeq_16SOTU)$Kingdom %in% c("Bacteria", "Archaea"),
  physeq_16SOTU
)


# Subset the phyloseq object to include only Tetragenococcus species first
# Tetragenococcus <- subset_taxa(physeq_16SOTU, Genus=="Tetragenococcus")
# Tetragenococcus <- subset_taxa(Tetragenococcus, !is.na(taxa_names(Tetragenococcus)) & taxa_n
# tree <- phy_tree(Tetragenococcus)
# ref_sequences_Tetragenococcus <- refseq(Tetragenococcus)
# library(Biostrings)
# writeXStringSet(ref_sequences_Tetragenococcus, "Sample.fasta.fasta")



ref_fasta <- system.file("extdata", "Ref.fasta", package = "DspikeIn")
sample_fasta <- system.file("extdata", "Sample.fasta", package = "DspikeIn")
```

```
result <- validate_spikein_clade(
  reference_fasta = ref_fasta,
  sample_fasta = sample_fasta,
  bootstrap = 200,
  output_prefix = NULL)
```

```
## use default substitution matrix
```

```
# result$tree_plot
```

Did spike-ins behave as expected across all samples?

Tip labels= OTU/ASV names Branch length numbers= Actual evolutionary distances (small = very similar) Prevalence stars How frequently the OTU occurs across samples Blue bar ring= Log10 mean abundance Outer colored tiles= The metadata variable you choose (e.g., Animal.type)

```
data("physeq_16SOTU", package = "DspikeIn")
library(ggstar)
library(ggplot2)
# filter your object to only include spike-in taxa beforehand:
# change the OTU IDs for easy detection
# Big stars = detected in many samples
# Small stars = rarely detected
# log10(Mean Abundance) Bars= Color intensity reflects mean abundance.
# The log-transformed average abundance of each OTU across all samples where it appears.
# Extreme blue may signal unintended over-representation.
# Metadata Ring = factor of your interest e.g. Animal.type
# Each OTU is colored by where it was observed.
# Branch length numbers=    Actual evolutionary distances (small = very similar)


spikein <- phyloseq::subset_taxa(physeq_16SOTU, Genus == "Tetragenococcus")
taxa_names(spikein) <- paste0("OTU", seq_len(ntaxa(spikein)))

# Visualize
ps <- plot_spikein_tree_diagnostic(
  obj = spikein,
  metadata_var = "Animal.type",
  output_prefix = "tetragenococcus_diag"
)
```

Pre_processing

merges monophyletic ASVs/OTUs

```r
# merges monophyletic ASVs/OTUs

# The function Pre_processing_species() merges ASVs/OTUs
# of a species using "sum" or "max" methods, preserving taxonomic,
# phylogenetic, and sequencing data.


#                       Load the phyloseq objs/ TSE obj


library(phyloseq)
library(DspikeIn)
library(TreeSummarizedExperiment)
library(SummarizedExperiment)

data("physeq_16SOTU", package = "DspikeIn")

# TSE format
# tse_16SOTU <- convert_phyloseq_to_tse(physeq_16SOTU)
# physeq_16SOTU <- convert_tse_to_phyloseq(tse_16SOTU)

physeq_16SOTU <- DspikeIn::tidy_phyloseq_tse(physeq_16SOTU) # make it tidy

# for those prefer to use TSE format
tse_16SOTU<-convert_phyloseq_to_tse(physeq_16SOTU)

# Check if metadata contains spiked volumes with this format
colnames(sample_data(physeq_16SOTU))
```

```
##  [1] "sample.id"                                "X16S.biosample"
##  [3] "dna.biosample"                            "data.type"
##  [5] "ampliconlibrary.quantification.ng.ul"     "plate.ID"
##  [7] "well.location"                            "Env.broad.scale"
##  [9] "Host.taxon"                               "Host.genus"
## [11] "Host.species"                             "Animal.type"
## [13] "Animal.ecomode"                           "Clade.Order"
## [15] "Family"                                   "Diet"
## [17] "Diet.Detailed"                            "Habitat"
## [19] "Metamorphosis"                            "Reproduction"
## [21] "Ecoregion.III"                            "Ecoregion.IV"
## [23] "Site"                                     "sample.name"
## [25] "biosample.parent"                         "data.type.1"
## [27] "ampliconlibrary.quantification.ng.ul.1"   "plate.ID.1"
## [29] "well.location.1"                          "sample.or.blank"
## [31] "sample.spiked.blank"                      "spiked.volume"
## [33] "swab.presence"                            "MK.spike"
```

Spiked species and related parameters for 16S (Phyloseq format)

```
#                        PREREQUISITE FOR 16S & CALCULATE SPIKED %


# Define spiked species and related parameters**

library(flextable)
library(DspikeIn)
# Define spike-in parameters
spiked_cells <- 1847
species_name <- spiked_species <- c("Tetragenococcus_halophilus", "Tetragenococcus_sp.")
merged_spiked_species <- "Tetragenococcus_halophilus"

# If you prefer Genus-level matching
# species_name <- spiked_species <- c("Tetragenococcus")
# merged_spiked_species <- "Tetragenococcus"


# Subset taxa for spiked species
Tetragenococcus <- phyloseq::subset_taxa(
  physeq_16SOTU,
  Species %in% species_name
)

# Get taxon hashcodes
hashcodes <- row.names(phyloseq::tax_table(Tetragenococcus))

# Subset samples based on spiked volume
spiked_16S_OTU_spiked <- phyloseq::subset_samples(
  physeq_16SOTU,
  spiked.volume %in% c("2", "1")
)

# Merge OTUs derived from spiked species
# File will be saved to tempdir() (no cleanup needed)
output_rds <- file.path(tempdir(), "merged_physeq_sum.rds")

Spiked_16S_sum_scaled <- Pre_processing_species(
  spiked_16S_OTU_spiked,
  species_name,
  merge_method = "sum",
  output_file = output_rds
)

# Calculate the spike-in percentage across samples
```

```
Perc <- calculate_spike_percentage(
  Spiked_16S_sum_scaled,
  merged_spiked_species,
  passed_range = c(0.1, 20)
)
```

Spiked species and related parameters for ITS

```
#                       PREREQUISITE FOR ITS & CALCULATE SPIKED %


# Define spiked species and related parameters**


# Define the spiked species
# spiked_cells <- 733
# species_name <- spiked_species <- merged_spiked_species <- "Dekkera_bruxellensis"

# Subset taxa for spiked species
# Dekkera <- phyloseq::subset_taxa(
# physeq_ITSOTU,
# Species %in% species_name)

# hashcodes <- row.names(phyloseq::tax_table(Dekkera))

# Subset samples based on spiked volume
# physeq_ITSOTU_spiked <- phyloseq::subset_samples(physeq_ITSOTU, spiked.volume %in% c("2", "1

# if TSE format
# tse_ITSOTU <- convert_phyloseq_to_tse(physeq_ITSOTU)
# physeq_ITSOTU_spiked <- tse_ITSOTU[, tse_ITSOTU$spiked.volume %in% c("2", "1")]
```

Calculating Scaling Factors (Phyloseq object)

```
#                       CALCULATE SCALING FACTORS


# Calculate spike-in scaling factors
result <- calculate_spikeIn_factors(
  obj = Spiked_16S_sum_scaled,
  spiked_cells = spiked_cells,
```

```
  merged_spiked_species = merged_spiked_species
)

# View extracted outputs
result$spiked_species_merged # Merged spiked species name
```

```
## phyloseq-class experiment-level object
## otu_table()   OTU Table:         [ 1 taxa and 264 samples ]
## sample_data() Sample Data:       [ 264 samples by 34 sample variables ]
## tax_table()   Taxonomy Table:    [ 1 taxa by 7 taxonomic ranks ]
## refseq()      DNAStringSet:      [ 1 reference sequences ]
```

```
result$spiked_species_reads # Total reads detected for the spike
```

```
##                                      Sample Spiked_Reads
## spiked.blank.20433_S84 spiked.blank.20433_S84            8
## spiked.blank.20817_S84 spiked.blank.20817_S84        47066
## Std2uL.20625_S84             Std2uL.20625_S84        62433
## StdSwab1uL.20624_S72     StdSwab1uL.20624_S72        17639
## STP1719.20422_S47           STP1719.20422_S47        14549
## STP213.20423_S59             STP213.20423_S59           83
## STP268.20424_S71             STP268.20424_S71           17
## STP544.20419_S11             STP544.20419_S11         2259
## STP570.20420_S23             STP570.20420_S23          822
## STP579.20421_S35             STP579.20421_S35         1759
## STP614.20418_S94             STP614.20418_S94            0
## UHM1000.20604_S22           UHM1000.20604_S22          118
## UHM1001.20609_S82           UHM1001.20609_S82           93
## UHM1007.20622_S48           UHM1007.20622_S48          118
## UHM1009.20614_S47           UHM1009.20614_S47          116
## UHM1010.20621_S36           UHM1010.20621_S36          979
## UHM1011.20606_S46           UHM1011.20606_S46          130
## UHM1024.20620_S24           UHM1024.20620_S24          994
## UHM1026.20607_S58           UHM1026.20607_S58          396
## UHM1028.20613_S35           UHM1028.20613_S35          898
## UHM1032.20605_S34           UHM1032.20605_S34         1616
## UHM1033.20619_S12           UHM1033.20619_S12        37416
## UHM1034.20616_S71           UHM1034.20616_S71            4
## UHM1035.20611_S11           UHM1035.20611_S11          210
## UHM1036.20612_S23           UHM1036.20612_S23          249
## UHM1052.20615_S59           UHM1052.20615_S59          150
## UHM1060.20723_S1             UHM1060.20723_S1        24617
## UHM1065.20724_S13           UHM1065.20724_S13         8179
## UHM1068.20732_S14           UHM1068.20732_S14          243
## UHM1069.20742_S39           UHM1069.20742_S39          343
## UHM1070.20725_S25           UHM1070.20725_S25         1741
```

```
## UHM1071.20733_S26          UHM1071.20733_S26          56
## UHM1072.20734_S38          UHM1072.20734_S38        1486
## UHM1073.20735_S50          UHM1073.20735_S50          39
## UHM1075.20726_S37          UHM1075.20726_S37        1179
## UHM1077.20736_S62          UHM1077.20736_S62          73
## UHM1078.20727_S49          UHM1078.20727_S49         173
## UHM1080.20737_S74          UHM1080.20737_S74          69
## UHM1081.20728_S61          UHM1081.20728_S61        1772
## UHM1088.20738_S86          UHM1088.20738_S86         240
## UHM1090.20739_S3            UHM1090.20739_S3        5303
## UHM1093.20729_S73          UHM1093.20729_S73         333
## UHM1095.20730_S85          UHM1095.20730_S85        6219
## UHM1097.20623_S60          UHM1097.20623_S60        6050
## UHM1099.20608_S70          UHM1099.20608_S70          24
## UHM1100.20788_S21          UHM1100.20788_S21          75
## UHM1102.20789_S33          UHM1102.20789_S33         108
## UHM1104.20790_S45          UHM1104.20790_S45          83
## UHM1105.20791_S57          UHM1105.20791_S57         148
## UHM1109.20531_S1            UHM1109.20531_S1         116
## UHM1110.20568_S65          UHM1110.20568_S65          50
## UHM1113.20792_S69          UHM1113.20792_S69          79
## UHM1114.20793_S81          UHM1114.20793_S81         361
## UHM1115.20794_S93          UHM1115.20794_S93         181
## UHM1117.20795_S10          UHM1117.20795_S10          61
## UHM1118.20796_S22          UHM1118.20796_S22         248
## UHM1120.20797_S34          UHM1120.20797_S34         718
## UHM1124.20798_S46          UHM1124.20798_S46         583
## UHM1126.20799_S58          UHM1126.20799_S58         193
## UHM1128.20800_S70          UHM1128.20800_S70         183
## UHM1140.20555_S4            UHM1140.20555_S4         130
## UHM1145.20801_S82          UHM1145.20801_S82          45
## UHM1163.20405_S33          UHM1163.20405_S33         150
## UHM1164.20402_S92          UHM1164.20402_S92           3
## UHM1169.20552_S63          UHM1169.20552_S63        6591
## UHM1171.20579_S7            UHM1171.20579_S7          97
## UHM1176.20404_S21          UHM1176.20404_S21          67
## UHM1177.20546_S86          UHM1177.20546_S86        1558
## UHM1182.20576_S66          UHM1182.20576_S66         233
## UHM1210.20802_S94          UHM1210.20802_S94          98
## UHM1212.20803_S11          UHM1212.20803_S11         306
## UHM1217.20804_S23          UHM1217.20804_S23         188
## UHM1218.20805_S35          UHM1218.20805_S35         151
## UHM1219.20806_S47          UHM1219.20806_S47          57
## UHM1220.20807_S59          UHM1220.20807_S59          84
## UHM1221.20808_S71          UHM1221.20808_S71         105
## UHM1222.20809_S83          UHM1222.20809_S83        3039
## UHM1223.20810_S95          UHM1223.20810_S95          69
## UHM1225.20811_S12          UHM1225.20811_S12         229
```

```
## UHM1227.20812_S24          UHM1227.20812_S24          871
## UHM1228.20813_S36          UHM1228.20813_S36           69
## UHM1237.20814_S48          UHM1237.20814_S48          413
## UHM1240.20566_S41          UHM1240.20566_S41          683
## UHM1246.20815_S60          UHM1246.20815_S60          573
## UHM1247.20816_S72          UHM1247.20816_S72          235
## UHM1248.20575_S54          UHM1248.20575_S54         3287
## UHM1256.20570_S89          UHM1256.20570_S89          901
## UHM1260.20596_S21          UHM1260.20596_S21          614
## UHM1270.20577_S78          UHM1270.20577_S78          217
## UHM1271.20397_S32          UHM1271.20397_S32          893
## UHM1272.20398_S44          UHM1272.20398_S44         1529
## UHM1274.20554_S87          UHM1274.20554_S87          467
## UHM1275.20597_S33          UHM1275.20597_S33         1203
## UHM1282.20599_S57          UHM1282.20599_S57         3820
## UHM1287.20543_S50          UHM1287.20543_S50         1979
## UHM1291.20416_S70          UHM1291.20416_S70          261
## UHM1296.20550_S39          UHM1296.20550_S39           76
## UHM1319.20561_S76          UHM1319.20561_S76         1567
## UHM1324.20413_S34          UHM1324.20413_S34          433
## UHM1327.20545_S74          UHM1327.20545_S74           73
## UHM1328.20572_S18          UHM1328.20572_S18         3325
## UHM1334.20417_S82          UHM1334.20417_S82          155
## UHM1338.20399_S56          UHM1338.20399_S56          760
## UHM1341.20602_S93          UHM1341.20602_S93         8050
## UHM1356.20541_S26          UHM1356.20541_S26          810
## UHM1380.20580_S19          UHM1380.20580_S19          308
## UHM1383.20594_S92          UHM1383.20594_S92          421
## UHM1385.20563_S5            UHM1385.20563_S5        20431
## UHM1399.20756_S17          UHM1399.20756_S17          918
## UHM1400.20757_S29          UHM1400.20757_S29          316
## UHM1401.20758_S41          UHM1401.20758_S41         2414
## UHM1402.20759_S53          UHM1402.20759_S53          166
## UHM1403.20760_S65          UHM1403.20760_S65          303
## UHM1405.20761_S77          UHM1405.20761_S77          214
## UHM1406.20762_S89          UHM1406.20762_S89          213
## UHM1414.20763_S6            UHM1414.20763_S6          107
## UHM1419.20764_S18          UHM1419.20764_S18          488
## UHM1427.20389_S31          UHM1427.20389_S31          103
## UHM1428.20390_S43          UHM1428.20390_S43            0
## UHM1429.20391_S55          UHM1429.20391_S55           29
## UHM1430.20392_S67          UHM1430.20392_S67           11
## UHM1432.20393_S79          UHM1432.20393_S79           18
## UHM1435.20388_S19          UHM1435.20388_S19            0
## UHM162.20560_S64            UHM162.20560_S64          704
## UHM198.20585_S79            UHM198.20585_S79         3649
## UHM20.3314_S52                UHM20.3314_S52           43
## UHM20.3315_S64                UHM20.3315_S64          349
```

```
## UHM204.20409_S81          UHM204.20409_S81          66
## UHM206.20410_S93          UHM206.20410_S93           0
## UHM207.20593_S80          UHM207.20593_S80         200
## UHM208.20411_S10          UHM208.20411_S10         391
## UHM211.20406_S45          UHM211.20406_S45         164
## UHM215.20408_S69          UHM215.20408_S69           9
## UHM216.20429_S36          UHM216.20429_S36          63
## UHM219.20430_S48          UHM219.20430_S48       21041
## UHM236.20431_S60          UHM236.20431_S60          41
## UHM238.20407_S57          UHM238.20407_S57         120
## UHM245.20538_S85          UHM245.20538_S85         108
## UHM252.20558_S40          UHM252.20558_S40        1090
## UHM267.20400_S68          UHM267.20400_S68         127
## UHM274.20581_S31          UHM274.20581_S31        2025
## UHM276.20586_S91          UHM276.20586_S91        1169
## UHM280.20401_S80          UHM280.20401_S80         389
## UHM286.20425_S83          UHM286.20425_S83          30
## UHM289.20426_S95          UHM289.20426_S95           4
## UHM294.20427_S12          UHM294.20427_S12         610
## UHM298.20600_S69          UHM298.20600_S69         404
## UHM325.20548_S15          UHM325.20548_S15         584
## UHM337.20412_S22          UHM337.20412_S22         261
## UHM354.20535_S49          UHM354.20535_S49         355
## UHM356.20415_S58          UHM356.20415_S58         411
## UHM369.20773_S31          UHM369.20773_S31         233
## UHM370.20774_S43          UHM370.20774_S43          88
## UHM372.20775_S55          UHM372.20775_S55          30
## UHM373.20776_S67          UHM373.20776_S67          84
## UHM374.20777_S79          UHM374.20777_S79          69
## UHM375.20778_S91          UHM375.20778_S91         709
## UHM377.20779_S8            UHM377.20779_S8         477
## UHM38.3376_S36              UHM38.3376_S36           0
## UHM386.20781_S32          UHM386.20781_S32         234
## UHM387.20782_S44          UHM387.20782_S44          58
## UHM414.20583_S55          UHM414.20583_S55          85
## UHM418.20765_S30          UHM418.20765_S30         270
## UHM422.20766_S42          UHM422.20766_S42        1144
## UHM425.20767_S54          UHM425.20767_S54         118
## UHM426.20534_S37          UHM426.20534_S37        4598
## UHM428.20544_S62          UHM428.20544_S62         454
## UHM429.20559_S52          UHM429.20559_S52        1283
## UHM435.20547_S3            UHM435.20547_S3        1718
## UHM437.20768_S66          UHM437.20768_S66          49
## UHM439.20564_S17          UHM439.20564_S17          71
## UHM44.3526_S31              UHM44.3526_S31        1281
## UHM445.20569_S77          UHM445.20569_S77          26
## UHM447.20783_S56          UHM447.20783_S56          64
## UHM448.20769_S78          UHM448.20769_S78         424
```

```
## UHM45.3539_S92               UHM45.3539_S92           0
## UHM454.20770_S90           UHM454.20770_S90         278
## UHM455.20785_S80           UHM455.20785_S80         787
## UHM458.20786_S92           UHM458.20786_S92          64
## UHM459.20787_S9             UHM459.20787_S9         120
## UHM461.20771_S7             UHM461.20771_S7         242
## UHM467.20772_S19           UHM467.20772_S19          47
## UHM470.20533_S25           UHM470.20533_S25        1220
## UHM476.20414_S46           UHM476.20414_S46        1135
## UHM478.20549_S27           UHM478.20549_S27         429
## UHM479.20551_S51           UHM479.20551_S51       14412
## UHM481.20403_S9             UHM481.20403_S9         227
## UHM482.20590_S44           UHM482.20590_S44         233
## UHM483.20603_S10           UHM483.20603_S10          91
## UHM519.20582_S43           UHM519.20582_S43         136
## UHM520.20573_S30           UHM520.20573_S30         253
## UHM746.21478_S117         UHM746.21478_S117       85168
## UHM747.21477_S106         UHM747.21477_S106       82287
## UHM748.21467_S170         UHM748.21467_S170       55291
## UHM748.21487_S129         UHM748.21487_S129       36462
## UHM749.21479_S128         UHM749.21479_S128       62780
## UHM759.21466_S159         UHM759.21466_S159       76702
## UHM759.21486_S118         UHM759.21486_S118       16500
## UHM775.21485_S107         UHM775.21485_S107       43046
## UHM776.21482_S161         UHM776.21482_S161       58165
## UHM777.21484_S183         UHM777.21484_S183       23485
## UHM779.21468_S181         UHM779.21468_S181       80976
## UHM779.21488_S140         UHM779.21488_S140           9
## UHM782.21480_S139         UHM782.21480_S139       38354
## UHM810.21472_S138         UHM810.21472_S138       94712
## UHM811.21471_S127         UHM811.21471_S127       30104
## UHM813.21481_S150         UHM813.21481_S150       64962
## UHM818.21469_S105         UHM818.21469_S105       50417
## UHM818.21489_S151         UHM818.21489_S151          19
## UHM819.21473_S149         UHM819.21473_S149       83424
## UHM820.21470_S116         UHM820.21470_S116       66075
## UHM820.21490_S162         UHM820.21490_S162         515
## UHM827.21474_S160         UHM827.21474_S160       83467
## UHM829.21476_S182         UHM829.21476_S182       69357
## UHM832.21483_S172         UHM832.21483_S172       68494
## UHM836.20385_S78           UHM836.20385_S78           0
## UHM837.20386_S90           UHM837.20386_S90           0
## UHM838.20387_S7             UHM838.20387_S7          11
## UHM891.20384_S66           UHM891.20384_S66          85
## UHM892.20532_S13           UHM892.20532_S13        1164
## UHM893.20595_S9             UHM893.20595_S9          16
## UHM894.20540_S14           UHM894.20540_S14          36
## UHM895.20536_S61           UHM895.20536_S61        2116
```

```
## UHM896.20601_S81             UHM896.20601_S81            0
## UHM897.20591_S56             UHM897.20591_S56            0
## UHM898.20394_S91             UHM898.20394_S91            0
## UHM899.20588_S20             UHM899.20588_S20           24
## UHM900.20395_S8               UHM900.20395_S8            0
## UHM901.20542_S38             UHM901.20542_S38           95
## UHM902.20584_S67             UHM902.20584_S67         1217
## UHM903.20587_S8               UHM903.20587_S8          470
## UHM904.20567_S53             UHM904.20567_S53           45
## UHM905.20598_S45             UHM905.20598_S45           89
## UHM906.20565_S29             UHM906.20565_S29            0
## UHM907.20592_S68             UHM907.20592_S68           30
## UHM908.20396_S20             UHM908.20396_S20            0
## UHM909.20557_S28             UHM909.20557_S28           87
## UHM910.20562_S88             UHM910.20562_S88           34
## UHM965.20537_S73             UHM965.20537_S73          464
## UHM966.20743_S51             UHM966.20743_S51          461
## UHM967.20744_S63             UHM967.20744_S63            0
## UHM968.20571_S6               UHM968.20571_S6        33012
## UHM969.20745_S75             UHM969.20745_S75           30
## UHM971.20746_S87             UHM971.20746_S87           10
## UHM973.20578_S90             UHM973.20578_S90           46
## UHM974.20432_S72             UHM974.20432_S72            0
## UHM975.20747_S4               UHM975.20747_S4           33
## UHM977.20748_S16             UHM977.20748_S16          100
## UHM978.20749_S28             UHM978.20749_S28           91
## UHM979.20750_S40             UHM979.20750_S40           35
## UHM980.20731_S2               UHM980.20731_S2          248
## UHM981.20539_S2               UHM981.20539_S2          766
## UHM982.20740_S15             UHM982.20740_S15          161
## UHM983.20556_S16             UHM983.20556_S16        12547
## UHM984.20751_S52             UHM984.20751_S52           26
## UHM985.20752_S64             UHM985.20752_S64          797
## UHM988.20753_S76             UHM988.20753_S76          603
## UHM989.20754_S88             UHM989.20754_S88           17
## UHM991.20755_S5               UHM991.20755_S5         1864
## UHM993.20741_S27             UHM993.20741_S27           52
## UHM996.20610_S94             UHM996.20610_S94            0
## UHM997.20553_S75             UHM997.20553_S75            0
## UHM998.20618_S95             UHM998.20618_S95         2610
## UHM999.20617_S83             UHM999.20617_S83           72
```

```
scaling_factors <- result$scaling_factors
head(scaling_factors) # Vector of scaling factors per sample
```

```
## spiked.blank.20433_S84 spiked.blank.20817_S84       Std2uL.20625_S84   StdSwab1uL.20624_S72
##           230.87500000             0.03924277             0.02958371             0.05235558
```

```
##       STP213.20423_S59
##             22.25301205
```

Convert relative counts to absolute counts

```
#             Convert relative counts to absolute counts


# absolute counts=relative counts×sample-specific scaling factor


# Convert to absolute counts
absolute <- convert_to_absolute_counts(Spiked_16S_sum_scaled, scaling_factors)

# Extract processed data
absolute_counts <- absolute$absolute_counts
physeq_absolute <- absolute$obj_adj

physeq_absolute <- tidy_phyloseq_tse(physeq_absolute)

# View absolute count data
head(absolute_counts)
```

```
##                                  spiked.blank.20433_S84 spiked.blank.20817_S84 Std2uL.20625_
## 020e00d90ba97c5898944ab6f7b1b7c9                      0                      0
## b00466354053c9065c8aa3d6fbb33eaa                      0                      0
## f872c4bf84bcf44434fa2023788f6517                      0                      0
##                                  STP1719.20422_S47 STP213.20423_S59 STP268.20424_S71 STP544
## 020e00d90ba97c5898944ab6f7b1b7c9                 0                 0                 0
## b00466354053c9065c8aa3d6fbb33eaa                 0                 0                 0
## f872c4bf84bcf44434fa2023788f6517                 0                 0                 0
##                                  STP579.20421_S35 STP614.20418_S94 UHM1000.20604_S22 UHM100
## 020e00d90ba97c5898944ab6f7b1b7c9                0                0                 0
## b00466354053c9065c8aa3d6fbb33eaa                0                0                 0
## f872c4bf84bcf44434fa2023788f6517                0                0                 0
##                                  UHM1007.20622_S48 UHM1009.20614_S47 UHM1010.20621_S36 UHM10
## 020e00d90ba97c5898944ab6f7b1b7c9                 0                 0                 0
## b00466354053c9065c8aa3d6fbb33eaa                 0                 0                 0
## f872c4bf84bcf44434fa2023788f6517                 0                 0                 0
##                                  UHM1024.20620_S24 UHM1026.20607_S58 UHM1028.20613_S35 UHM10
## 020e00d90ba97c5898944ab6f7b1b7c9                 0                 0                 0
## b00466354053c9065c8aa3d6fbb33eaa                 0                 7                 0
## f872c4bf84bcf44434fa2023788f6517                 0                 0                 0
##                                  UHM1033.20619_S12 UHM1034.20616_S71 UHM1035.20611_S11 UHM10
## 020e00d90ba97c5898944ab6f7b1b7c9                 0               462                 0
```

```
## b00466354053c9065c8aa3d6fbb33eaa                 0                 0                 0
## f872c4bf84bcf44434fa2023788f6517                 0               693                 0
##                                  UHM1052.20615_S59 UHM1060.20723_S1 UHM1065.20724_S13 UHM106
## 020e00d90ba97c5898944ab6f7b1b7c9                 0                 0                 0
## b00466354053c9065c8aa3d6fbb33eaa                 0                 0                 0
## f872c4bf84bcf44434fa2023788f6517                 0                 0                 0
##                                  UHM1069.20742_S39 UHM1070.20725_S25 UHM1071.20733_S26 UHM10
## 020e00d90ba97c5898944ab6f7b1b7c9                 0                 0                 0
## b00466354053c9065c8aa3d6fbb33eaa                 0                 0                 0
## f872c4bf84bcf44434fa2023788f6517                 0                 0                 0
##                                  UHM1073.20735_S50 UHM1075.20726_S37 UHM1077.20736_S62 UHM10
## 020e00d90ba97c5898944ab6f7b1b7c9                 0                 0                 0
## b00466354053c9065c8aa3d6fbb33eaa                 0                 0                 0
## f872c4bf84bcf44434fa2023788f6517                 0                 0                 0
##                                  UHM1080.20737_S74 UHM1081.20728_S61 UHM1088.20738_S86 UHM10
## 020e00d90ba97c5898944ab6f7b1b7c9                 0                 0                 0
## b00466354053c9065c8aa3d6fbb33eaa                 0                 0                 0
## f872c4bf84bcf44434fa2023788f6517                 0                 0                 0
##                                  UHM1093.20729_S73 UHM1095.20730_S85 UHM1097.20623_S60 UHM10
## 020e00d90ba97c5898944ab6f7b1b7c9                 0                 0                 0
## b00466354053c9065c8aa3d6fbb33eaa                 0                 0                 0
## f872c4bf84bcf44434fa2023788f6517                 0                 0                 0
##                                  UHM1100.20788_S21 UHM1102.20789_S33 UHM1104.20790_S45 UHM11
## 020e00d90ba97c5898944ab6f7b1b7c9                 0                 0                 0
## b00466354053c9065c8aa3d6fbb33eaa                 0                 0                 0
## f872c4bf84bcf44434fa2023788f6517                 0                 0                 0
##                                  UHM1109.20531_S1 UHM1110.20568_S65 UHM1113.20792_S69 UHM111
## 020e00d90ba97c5898944ab6f7b1b7c9                0                 0                 0
## b00466354053c9065c8aa3d6fbb33eaa                0                 0                 0
## f872c4bf84bcf44434fa2023788f6517                0                 0                 0
##                                  UHM1115.20794_S93 UHM1117.20795_S10 UHM1118.20796_S22 UHM11
## 020e00d90ba97c5898944ab6f7b1b7c9                 0                 0                 0
## b00466354053c9065c8aa3d6fbb33eaa                 0                 0                 0
## f872c4bf84bcf44434fa2023788f6517                 0                 0                 0
##                                  UHM1124.20798_S46 UHM1126.20799_S58 UHM1128.20800_S70 UHM11
## 020e00d90ba97c5898944ab6f7b1b7c9                 0                 0                 0
## b00466354053c9065c8aa3d6fbb33eaa                 0                 0                 0
## f872c4bf84bcf44434fa2023788f6517                 0                 0                 0
##                                  UHM1145.20801_S82 UHM1163.20405_S33 UHM1164.20402_S92 UHM11
## 020e00d90ba97c5898944ab6f7b1b7c9                 0                 0                 0
## b00466354053c9065c8aa3d6fbb33eaa                 0                 0                 0
## f872c4bf84bcf44434fa2023788f6517                 0                 0                 0
##                                  UHM1171.20579_S7 UHM1176.20404_S21 UHM1177.20546_S86 UHM118
## 020e00d90ba97c5898944ab6f7b1b7c9                0                 0                 0
## b00466354053c9065c8aa3d6fbb33eaa                0                 0                 0
## f872c4bf84bcf44434fa2023788f6517                0                 0                 0
##                                  UHM1210.20802_S94 UHM1212.20803_S11 UHM1217.20804_S23 UHM12
## 020e00d90ba97c5898944ab6f7b1b7c9                 0                 0                 0
```

```
## b00466354053c9065c8aa3d6fbb33eaa                 0                 0                 0
## f872c4bf84bcf44434fa2023788f6517                 0                 0                 0
##                                  UHM1219.20806_S47 UHM1220.20807_S59 UHM1221.20808_S71 UHM1
## 020e00d90ba97c5898944ab6f7b1b7c9                 0                 0                 0
## b00466354053c9065c8aa3d6fbb33eaa                 0                 0                 0
## f872c4bf84bcf44434fa2023788f6517                 0                 0                 0
##                                  UHM1223.20810_S95 UHM1225.20811_S12 UHM1227.20812_S24 UHM1
## 020e00d90ba97c5898944ab6f7b1b7c9                 0                 0                 0
## b00466354053c9065c8aa3d6fbb33eaa                 0                 0                 0
## f872c4bf84bcf44434fa2023788f6517                 0                 0                 0
##                                  UHM1237.20814_S48 UHM1240.20566_S41 UHM1246.20815_S60 UHM1
## 020e00d90ba97c5898944ab6f7b1b7c9                 0                 0                 0
## b00466354053c9065c8aa3d6fbb33eaa                 0                 0                 0
## f872c4bf84bcf44434fa2023788f6517                 0                 0                 0
##                                  UHM1248.20575_S54 UHM1256.20570_S89 UHM1260.20596_S21 UHM1
## 020e00d90ba97c5898944ab6f7b1b7c9                 0                 0                 0
## b00466354053c9065c8aa3d6fbb33eaa                 0                 0                 0
## f872c4bf84bcf44434fa2023788f6517                 0                 0                 0
##                                  UHM1271.20397_S32 UHM1272.20398_S44 UHM1274.20554_S87 UHM1
## 020e00d90ba97c5898944ab6f7b1b7c9                 0                 0                 0
## b00466354053c9065c8aa3d6fbb33eaa                 0                 0                 0
## f872c4bf84bcf44434fa2023788f6517                 0                 0                 0
##                                  UHM1282.20599_S57 UHM1287.20543_S50 UHM1291.20416_S70 UHM1
## 020e00d90ba97c5898944ab6f7b1b7c9                 0                 0                 0
## b00466354053c9065c8aa3d6fbb33eaa                 0                 0                 0
## f872c4bf84bcf44434fa2023788f6517                 0                 0                 0
##                                  UHM1319.20561_S76 UHM1324.20413_S34 UHM1327.20545_S74 UHM1
## 020e00d90ba97c5898944ab6f7b1b7c9                 0                 0                 0
## b00466354053c9065c8aa3d6fbb33eaa                 0                 0                 0
## f872c4bf84bcf44434fa2023788f6517                 0                 0                 0
##                                  UHM1334.20417_S82 UHM1338.20399_S56 UHM1341.20602_S93 UHM1
## 020e00d90ba97c5898944ab6f7b1b7c9                 0                 0                 0
## b00466354053c9065c8aa3d6fbb33eaa                 0                 0                 0
## f872c4bf84bcf44434fa2023788f6517                 0                 0                 0
##                                  UHM1380.20580_S19 UHM1383.20594_S92 UHM1385.20563_S5 UHM139
## 020e00d90ba97c5898944ab6f7b1b7c9                 0                 0                 0
## b00466354053c9065c8aa3d6fbb33eaa                 0                 0                 0
## f872c4bf84bcf44434fa2023788f6517                 0                 0                 0
##                                  UHM1400.20757_S29 UHM1401.20758_S41 UHM1402.20759_S53 UHM14
## 020e00d90ba97c5898944ab6f7b1b7c9                 0                 0                 0
## b00466354053c9065c8aa3d6fbb33eaa                 0                 0                 0
## f872c4bf84bcf44434fa2023788f6517                 0                 0                 0
##                                  UHM1405.20761_S77 UHM1406.20762_S89 UHM1414.20763_S6 UHM141
## 020e00d90ba97c5898944ab6f7b1b7c9                 0                 0                 0
## b00466354053c9065c8aa3d6fbb33eaa                 0                 0                 0
## f872c4bf84bcf44434fa2023788f6517                 0                 0                 0
##                                  UHM1427.20389_S31 UHM1428.20390_S43 UHM1429.20391_S55 UHM14
## 020e00d90ba97c5898944ab6f7b1b7c9                 0                 0                 0
```

```
## b00466354053c9065c8aa3d6fbb33eaa                 0                 0                 0
## f872c4bf84bcf44434fa2023788f6517                 0                 0                 0
##                                  UHM1432.20393_S79 UHM1435.20388_S19 UHM162.20560_S64 UHM198
## 020e00d90ba97c5898944ab6f7b1b7c9                 0                 0                 0
## b00466354053c9065c8aa3d6fbb33eaa                 0                 0                 0
## f872c4bf84bcf44434fa2023788f6517                 0                 0                 0
##                                  UHM20.3315_S64 UHM204.20409_S81 UHM206.20410_S93 UHM207.205
## 020e00d90ba97c5898944ab6f7b1b7c9              0                 0                 0
## b00466354053c9065c8aa3d6fbb33eaa              0                 0                 0
## f872c4bf84bcf44434fa2023788f6517              0                 0                 0
##                                  UHM211.20406_S45 UHM215.20408_S69 UHM216.20429_S36 UHM219.1
## 020e00d90ba97c5898944ab6f7b1b7c9                0                 0                 0
## b00466354053c9065c8aa3d6fbb33eaa                0                 0                 0
## f872c4bf84bcf44434fa2023788f6517                0                 0                 0
##                                  UHM238.20407_S57 UHM245.20538_S85 UHM252.20558_S40 UHM267.1
## 020e00d90ba97c5898944ab6f7b1b7c9                0                 0                 0
## b00466354053c9065c8aa3d6fbb33eaa                0                 0                 0
## f872c4bf84bcf44434fa2023788f6517                0                 0                 0
##                                  UHM276.20586_S91 UHM280.20401_S80 UHM286.20425_S83 UHM289.1
## 020e00d90ba97c5898944ab6f7b1b7c9                0                 0                 0
## b00466354053c9065c8aa3d6fbb33eaa                0                 0                 0
## f872c4bf84bcf44434fa2023788f6517                0                 0                 0
##                                  UHM298.20600_S69 UHM325.20548_S15 UHM337.20412_S22 UHM354.1
## 020e00d90ba97c5898944ab6f7b1b7c9                0                 0                 0
## b00466354053c9065c8aa3d6fbb33eaa                0                 0                 0
## f872c4bf84bcf44434fa2023788f6517                0                 0                 0
##                                  UHM369.20773_S31 UHM370.20774_S43 UHM372.20775_S55 UHM373.1
## 020e00d90ba97c5898944ab6f7b1b7c9                0                 0                 0
## b00466354053c9065c8aa3d6fbb33eaa                0                 0                 0
## f872c4bf84bcf44434fa2023788f6517                0                 0                 0
##                                  UHM375.20778_S91 UHM377.20779_S8 UHM38.3376_S36 UHM386.2078
## 020e00d90ba97c5898944ab6f7b1b7c9                0                0              0
## b00466354053c9065c8aa3d6fbb33eaa                0                0              0
## f872c4bf84bcf44434fa2023788f6517                0                0              0
##                                  UHM414.20583_S55 UHM418.20765_S30 UHM422.20766_S42 UHM425.1
## 020e00d90ba97c5898944ab6f7b1b7c9                0                 0                 0
## b00466354053c9065c8aa3d6fbb33eaa                0                 0                 0
## f872c4bf84bcf44434fa2023788f6517                0                 0                 0
##                                  UHM428.20544_S62 UHM429.20559_S52 UHM435.20547_S3 UHM437.20
## 020e00d90ba97c5898944ab6f7b1b7c9                0                 0               0
## b00466354053c9065c8aa3d6fbb33eaa                0                 0               0
## f872c4bf84bcf44434fa2023788f6517                0                 0               0
##                                  UHM44.3526_S31 UHM445.20569_S77 UHM447.20783_S56 UHM448.207
## 020e00d90ba97c5898944ab6f7b1b7c9              0                 0                 0
## b00466354053c9065c8aa3d6fbb33eaa              0                 0                 0
## f872c4bf84bcf44434fa2023788f6517              0                 0                 0
##                                  UHM454.20770_S90 UHM455.20785_S80 UHM458.20786_S92 UHM459.1
## 020e00d90ba97c5898944ab6f7b1b7c9                0                 0                 0
```

```
## b00466354053c9065c8aa3d6fbb33eaa                0                0                0
## f872c4bf84bcf44434fa2023788f6517                0                0                0
##                                  UHM467.20772_S19 UHM470.20533_S25 UHM476.20414_S46 UHM478.2
## 020e00d90ba97c5898944ab6f7b1b7c9                0                0                0
## b00466354053c9065c8aa3d6fbb33eaa                0                0                0
## f872c4bf84bcf44434fa2023788f6517                0                0                0
##                                  UHM481.20403_S9 UHM482.20590_S44 UHM483.20603_S10 UHM519.20
## 020e00d90ba97c5898944ab6f7b1b7c9               0                0                0
## b00466354053c9065c8aa3d6fbb33eaa               0                0                0
## f872c4bf84bcf44434fa2023788f6517               0                0                0
##                                  UHM746.21478_S117 UHM747.21477_S106 UHM748.21467_S170 UHM74
## 020e00d90ba97c5898944ab6f7b1b7c9                 0                 0                 0
## b00466354053c9065c8aa3d6fbb33eaa                 0                 0                 0
## f872c4bf84bcf44434fa2023788f6517                 0                 0                 0
##                                  UHM749.21479_S128 UHM759.21466_S159 UHM759.21486_S118 UHM77
## 020e00d90ba97c5898944ab6f7b1b7c9                 0                 0                 0
## b00466354053c9065c8aa3d6fbb33eaa                 0                 0                 0
## f872c4bf84bcf44434fa2023788f6517                 0                 0                 0
##                                  UHM776.21482_S161 UHM777.21484_S183 UHM779.21468_S181 UHM77
## 020e00d90ba97c5898944ab6f7b1b7c9                 0                 0                 0
## b00466354053c9065c8aa3d6fbb33eaa                 0                 0                 0
## f872c4bf84bcf44434fa2023788f6517                 0                 0                 0
##                                  UHM782.21480_S139 UHM810.21472_S138 UHM811.21471_S127 UHM81
## 020e00d90ba97c5898944ab6f7b1b7c9                 0                 0                 0
## b00466354053c9065c8aa3d6fbb33eaa                 0                 0                 0
## f872c4bf84bcf44434fa2023788f6517                 0                 0                 0
##                                  UHM818.21469_S105 UHM818.21489_S151 UHM819.21473_S149 UHM82
## 020e00d90ba97c5898944ab6f7b1b7c9                 0                 0                 0
## b00466354053c9065c8aa3d6fbb33eaa                 0                 0                 0
## f872c4bf84bcf44434fa2023788f6517                 0                 0                 0
##                                  UHM820.21490_S162 UHM827.21474_S160 UHM829.21476_S182 UHM83
## 020e00d90ba97c5898944ab6f7b1b7c9                 0                 0                 0
## b00466354053c9065c8aa3d6fbb33eaa                 0                 0                 0
## f872c4bf84bcf44434fa2023788f6517                 0                 0                 0
##                                  UHM836.20385_S78 UHM837.20386_S90 UHM838.20387_S7 UHM891.20
## 020e00d90ba97c5898944ab6f7b1b7c9                0                0                0
## b00466354053c9065c8aa3d6fbb33eaa                0                0                0
## f872c4bf84bcf44434fa2023788f6517                0                0                0
##                                  UHM893.20595_S9 UHM894.20540_S14 UHM895.20536_S61 UHM896.20
## 020e00d90ba97c5898944ab6f7b1b7c9               0                0                0
## b00466354053c9065c8aa3d6fbb33eaa               0                0                0
## f872c4bf84bcf44434fa2023788f6517               0                0                0
##                                  UHM898.20394_S91 UHM899.20588_S20 UHM900.20395_S8 UHM901.20
## 020e00d90ba97c5898944ab6f7b1b7c9                0                0                0
## b00466354053c9065c8aa3d6fbb33eaa                0                0                0
## f872c4bf84bcf44434fa2023788f6517                0                0                0
##                                  UHM903.20587_S8 UHM904.20567_S53 UHM905.20598_S45 UHM906.20
## 020e00d90ba97c5898944ab6f7b1b7c9               0                0                0
```

```
## b00466354053c9065c8aa3d6fbb33eaa               0                0                0
## f872c4bf84bcf44434fa2023788f6517               0                0                0
##                                  UHM908.20396_S20 UHM909.20557_S28 UHM910.20562_S88 UHM965.1
## 020e00d90ba97c5898944ab6f7b1b7c9                0                0                0
## b00466354053c9065c8aa3d6fbb33eaa                0                0                0
## f872c4bf84bcf44434fa2023788f6517                0                0                0
##                                  UHM967.20744_S63 UHM968.20571_S6 UHM969.20745_S75 UHM971.20
## 020e00d90ba97c5898944ab6f7b1b7c9                0               0                0
## b00466354053c9065c8aa3d6fbb33eaa                0               0                0
## f872c4bf84bcf44434fa2023788f6517                0               0                0
##                                  UHM974.20432_S72 UHM975.20747_S4 UHM977.20748_S16 UHM978.20
## 020e00d90ba97c5898944ab6f7b1b7c9                0               0                0
## b00466354053c9065c8aa3d6fbb33eaa                0               0                0
## f872c4bf84bcf44434fa2023788f6517                0               0                0
##                                  UHM980.20731_S2 UHM981.20539_S2 UHM982.20740_S15 UHM983.205
## 020e00d90ba97c5898944ab6f7b1b7c9               0               0                0
## b00466354053c9065c8aa3d6fbb33eaa               0               0                0
## f872c4bf84bcf44434fa2023788f6517               0               0                0
##                                  UHM985.20752_S64 UHM988.20753_S76 UHM989.20754_S88 UHM991.1
## 020e00d90ba97c5898944ab6f7b1b7c9                0                0                0
## b00466354053c9065c8aa3d6fbb33eaa                0                0                0
## f872c4bf84bcf44434fa2023788f6517                0                0                0
##                                  UHM996.20610_S94 UHM997.20553_S75 UHM998.20618_S95 UHM999.1
## 020e00d90ba97c5898944ab6f7b1b7c9                0                0                0
## b00466354053c9065c8aa3d6fbb33eaa                0                0                0
## f872c4bf84bcf44434fa2023788f6517                0                0                0
##  [ reached 'max' / getOption("max.print") -- omitted 3 rows ]
```

Summary Stat

```
#                       CALCULATE SPIKE PERCENTAGE & summary stat


# ** Calculate spike percentage & Generate summary statistics for absolute counts**

# Generate summary statistics for absolute counts
post_eval_summary <- calculate_summary_stats_table(absolute_counts)

# You may want to Back normal to check calculation accuracy
# the scaling factor was computed based on spiked species reads and fixed cell count.
# Multiplying the spiked species read count by this scaling factor restores the exact spiked c
# lets check it
# BackNormal <- calculate_spike_percentage(
#  physeq_absolute,
#  merged_spiked_species,
```

```
#  passed_range = c(0.1, 20)
# )
```

Filter unsuccessful spiked samples

```
# Optional, or you may do it at the end of the process
# ** Time to filter out unsuccessful spiked samples**

library(phyloseq)
library(dplyr)
library(tibble)
library(microbiome)

filtered_sample_data <- microbiome::meta(physeq_absolute) %>%
  as.data.frame() %>%
  tibble::rownames_to_column(var = "Sample") %>%
  dplyr::mutate(Sample = as.character(Sample)) %>%
  dplyr::left_join(Perc, by = "Sample")

filtered_sample_data <- tibble::column_to_rownames(filtered_sample_data, "Sample")

filtered_sample_data <- sample_data(as.data.frame(filtered_sample_data))

# Assign back to phyloseq obj
sample_data(physeq_absolute) <- filtered_sample_data
```

spiked species retrieval is system-dependent

The goal is to identify the range where, for example, the evenness of your community remains independent of spiked species retrieval—meaning the p-value should not be significant, and the R^2 value should be low, indicating minimal influence. Hill number is interpretable as “effective diversity” (number of abundant species).

```
# The acceptable range of spiked species retrieval is system-dependent
# Spiked species become centroid of the community (Distance to Centroid)
# Spiked species become dominant and imbalance the community (Evenness)

# What range of spiked species retrieval is appropriate for your system?
# Calculate Pielou's Evenness using Shannon index and species richness (Observed)
# Hill number q = 1 = exp(Shannon index), representing the effective number of equally abundan
# Unlike Pielou's evenness, this metric is not normalized by richness and it shows Effective n
```

```r
library(phyloseq)
library(vegan)
library(dplyr)
library(microbiome)
library(DspikeIn)

# 1. Alpha Diversity: Pielou's Evenness and Hill Number (q = 1)

# Estimate richness and Shannon index
alphab <- estimate_richness(
  physeq_absolute,
  measures = c("Observed", "Shannon")
)

# Calculate Pielou's Evenness = H' / S
alphab$Pielou_evenness <- alphab$Shannon / alphab$Observed

# Extract OTU matrix
otu_mat <- otu_table(physeq_absolute)
if (taxa_are_rows(physeq_absolute)) {
  otu_mat <- t(otu_mat)
}
otu_mat <- as(otu_mat, "matrix")

# Calculate Hill number q = 1
alphab$Hill_q1 <- exp(vegan::diversity(otu_mat, index = "shannon"))

# Add sample ID
alphab$Sample <- rownames(alphab)


# 2. Merge Alpha Diversity with Metadata
# ============================================================================

metadata <- as.data.frame(microbiome::meta(physeq_absolute))
metadata$Sample <- rownames(metadata)

metadata <- left_join(
  metadata,
  alphab[, c("Sample", "Observed", "Shannon", "Pielou_evenness", "Hill_q1")],
  by = "Sample"
)
metadata <- column_to_rownames(metadata, var = "Sample")
sample_data(physeq_absolute) <- sample_data(metadata)

# Check presence of spike-in reads
if (!"Spiked_Reads" %in% colnames(metadata)) {
```

```
  stop("Column 'Spiked_Reads' not found in metadata.")
}


# 3. Beta Diversity: Distance to Global Centroid (Full Bray-Curtis)
# ==============================================================================

# Convert OTU table to relative abundances
otu_mat_rel <- vegan::decostand(otu_mat, method = "total")

# centroid profile
centroid_profile <- colMeans(otu_mat_rel)

# Compute Bray-Curtis distance to centroid for each sample
dist_to_centroid <- apply(otu_mat_rel, 1, function(x) {
  vegan::vegdist(rbind(x, centroid_profile), method = "bray")[1]
})

# Add distance to metadata
metadata$Dist_to_Centroid <- dist_to_centroid[rownames(metadata)]

# Update phyloseq object
sample_data(physeq_absolute) <- sample_data(metadata)


# 4. Regression Plots: Diversity vs. Spike-in Reads
# ==============================================================================

# Pielou's Evenness
plot_object_pielou <- regression_plot(
  data = metadata,
  x_var = "Pielou_evenness",
  y_var = "Spiked_Reads",
  custom_range = c(0.1, 20, 30, 40, 50, 60, 100),
  plot_title = NULL
)

# Hill Number (q = 1)
plot_object_hill <- regression_plot(
  data = metadata,
  x_var = "Hill_q1",
  y_var = "Spiked_Reads",
  custom_range = c(0.1,10, 20, 30, 100),
  plot_title = NULL
)

# Distance to Global Centroid
```

```r
plot_object_centroid <- regression_plot(
  data = metadata,
  x_var = "Dist_to_Centroid",
  y_var = "Spiked_Reads",
  custom_range = c(0.1, 20, 30, 40, 50, 60, 100),
  plot_title = NULL
)


# Interpretation
# ================================================================================
# - Pielou's evenness is normalized by richness; useful for detecting imbalance.
# - Hill number q = 1 gives effective number of common species; sensitive to dominance.
# - Distance to centroid in full Bray-Curtis space shows deviation from the average community.
```

Calculate the percentage of spiked species retrieval

```r
# * Calculate the percentage of spiked species retrieval per sample*

absolute_abundance_16S_OTU_perc <- phyloseq::subset_samples(physeq_absolute, sample.or.blank !=

# Adjust the threshold range as needed based on system specifications

result_perc <- calculate_spike_percentage(
  absolute_abundance_16S_OTU_perc,
  merged_spiked_species,
  passed_range = c(0.1, 20)
)

# BackNormal
conc <- conclusion(absolute_abundance_16S_OTU_perc,
  merged_spiked_species,
  max_passed_range = 20,
  output_path
)

conc$full_report
```

```
##                Sample Total_Reads Spiked_Reads   Percentage Result
## 1   STP1719.20422_S47        2429         1847  76.03952244 failed
## 2    STP213.20423_S59      188314         1847   0.98080865 passed
## 3    STP268.20424_S71      757488         1847   0.24383225 passed
## 4    STP544.20419_S11        1913         1847  96.54992159 failed
## 5    STP570.20420_S23        5948         1847  31.05245461 failed
```

```
## 6    STP579.20421_S35        5452         1847  33.87747616 failed
## 7    STP614.20418_S94           5            0   0.00000000 failed
## 8   UHM1000.20604_S22       43347          924   2.13163541 passed
## 9   UHM1001.20609_S82       39309          924   2.35060673 passed
## 10  UHM1007.20622_S48       86418          924   1.06922169 passed
## 11  UHM1009.20614_S47      181742          924   0.50841303 passed
## 12  UHM1010.20621_S36        6123          924  15.09064184 passed
## 13  UHM1011.20606_S46       81017          924   1.14050138 passed
## 14  UHM1024.20620_S24        2690          924  34.34944238 failed
## 15  UHM1026.20607_S58        7494          924  12.32986389 passed
## 16  UHM1028.20613_S35        2620          924  35.26717557 failed
## 17  UHM1032.20605_S34        1410          924  65.53191489 failed
## 18  UHM1033.20619_S12         966          924  95.65217391 failed
## 19  UHM1034.20616_S71      782439          924   0.11809227 passed
## 20  UHM1035.20611_S11        8082          924  11.43281366 passed
## 21  UHM1036.20612_S23        8625          924  10.71304348 passed
## 22  UHM1052.20615_S59       98557          924   0.93752854 passed
## 23   UHM1060.20723_S1        2074         1847  89.05496625 failed
## 24  UHM1065.20724_S13        2384         1847  77.47483221 failed
## 25  UHM1068.20732_S14       19198         1847   9.62079383 passed
## 26  UHM1069.20742_S39       88462         1847   2.08790215 passed
## 27  UHM1070.20725_S25       11119         1847  16.61120604 passed
## 28  UHM1071.20733_S26       66892         1847   2.76116725 passed
## 29  UHM1072.20734_S38        4254         1847  43.41795957 failed
## 30  UHM1073.20735_S50      129333         1847   1.42809646 passed
## 31  UHM1075.20726_S37       12116         1847  15.24430505 passed
## 32  UHM1077.20736_S62      240090         1847   0.76929485 passed
## 33  UHM1078.20727_S49       48860         1847   3.78018829 passed
## 34  UHM1080.20737_S74      237852         1847   0.77653331 passed
## 35  UHM1081.20728_S61        2118         1847  87.20491029 failed
## 36  UHM1088.20738_S86       48261         1847   3.82710677 passed
## 37   UHM1090.20739_S3        4260         1847  43.35680751 failed
## 38  UHM1093.20729_S73       11145         1847  16.57245402 passed
## 39  UHM1095.20730_S85        3475         1847  53.15107914 failed
## 40  UHM1097.20623_S60        1645          924  56.17021277 failed
## 41  UHM1099.20608_S70      141219          924   0.65430289 passed
## 42  UHM1100.20788_S21      120156         1847   1.53716835 passed
## 43  UHM1102.20789_S33       41207         1847   4.48224816 passed
## 44  UHM1104.20790_S45       75900         1847   2.43346509 passed
## 45  UHM1105.20791_S57       29445         1847   6.27271184 passed
## 46   UHM1109.20531_S1       49710         1847   3.71555019 passed
## 47  UHM1110.20568_S65      257312         1847   0.71780562 passed
## 48  UHM1113.20792_S69      374668         1847   0.49296978 passed
## 49  UHM1114.20793_S81       25789         1847   7.16196828 passed
## 50  UHM1115.20794_S93       53637         1847   3.44351847 passed
## 51  UHM1117.20795_S10      120412         1847   1.53390028 passed
## 52  UHM1118.20796_S22       39826         1847   4.63767388 passed
## 53  UHM1120.20797_S34        6447         1847  28.64898402 failed
```

```
## 54  UHM1124.20798_S46       19988         1847   9.24054433 passed
## 55  UHM1126.20799_S58       56862         1847   3.24821498 passed
## 56  UHM1128.20800_S70       75987         1847   2.43067893 passed
## 57   UHM1140.20555_S4      144876         1847   1.27488335 passed
## 58  UHM1145.20801_S82      136223         1847   1.35586502 passed
## 59  UHM1163.20405_S33       66525         1847   2.77639985 passed
## 60  UHM1164.20402_S92      560873         1847   0.32930806 passed
## 61  UHM1169.20552_S63        2087         1847  88.50023958 failed
## 62   UHM1171.20579_S7       94275         1847   1.95916203 passed
## 63  UHM1176.20404_S21      153994         1847   1.19939738 passed
## 64  UHM1177.20546_S86        7001         1847  26.38194544 failed
## 65  UHM1182.20576_S66       19322         1847   9.55905186 passed
## 66  UHM1210.20802_S94      107631         1847   1.71604835 passed
## 67  UHM1212.20803_S11       35517         1847   5.20032660 passed
## 68  UHM1217.20804_S23       66837         1847   2.76343941 passed
## 69  UHM1218.20805_S35       42728         1847   4.32269238 passed
## 70  UHM1219.20806_S47       73751         1847   2.50437282 passed
## 71  UHM1220.20807_S59       85890         1847   2.15042496 passed
## 72  UHM1221.20808_S71       37595         1847   4.91288735 passed
## 73  UHM1222.20809_S83        5442         1847  33.93972804 failed
## 74  UHM1223.20810_S95      114277         1847   1.61624824 passed
## 75  UHM1225.20811_S12       48275         1847   3.82599689 passed
## 76  UHM1227.20812_S24       20964         1847   8.81034154 passed
## 77  UHM1228.20813_S36      147297         1847   1.25392914 passed
## 78  UHM1237.20814_S48       24999         1847   7.38829553 passed
## 79  UHM1240.20566_S41       35116         1847   5.25971067 passed
## 80  UHM1246.20815_S60       16128         1847  11.45213294 passed
## 81  UHM1247.20816_S72       26551         1847   6.95642349 passed
## 82  UHM1248.20575_S54        7646         1847  24.15642166 failed
## 83  UHM1256.20570_S89       35459         1847   5.20883274 passed
## 84  UHM1260.20596_S21       34301         1847   5.38468266 passed
## 85  UHM1270.20577_S78       12183         1847  15.16046951 passed
## 86  UHM1271.20397_S32       19173         1847   9.63333855 passed
## 87  UHM1272.20398_S44       10447         1847  17.67971667 passed
## 88  UHM1274.20554_S87       32434         1847   5.69464143 passed
## 89  UHM1275.20597_S33       15061         1847  12.26346192 passed
## 90  UHM1282.20599_S57        6536         1847  28.25887393 failed
## 91  UHM1287.20543_S50       10739         1847  17.19899432 passed
## 92  UHM1291.20416_S70       74217         1847   2.48864815 passed
## 93  UHM1296.20550_S39      207620         1847   0.88960601 passed
## 94  UHM1319.20561_S76       11679         1847  15.81471016 passed
## 95  UHM1324.20413_S34       45204         1847   4.08592160 passed
## 96  UHM1327.20545_S74      272806         1847   0.67703790 passed
## 97  UHM1328.20572_S18        9273         1847  19.91804163 passed
## 98  UHM1334.20417_S82       76551         1847   2.41277057 passed
## 99  UHM1338.20399_S56       26305         1847   7.02147881 passed
## 100 UHM1341.20602_S93        4830         1847  38.24016563 failed
## 101 UHM1356.20541_S26       21082         1847   8.76102837 passed
```

```
## 102 UHM1380.20580_S19       71993        1847   2.56552720 passed
## 103 UHM1383.20594_S92       28316        1847   6.52281396 passed
## 104  UHM1385.20563_S5        2166        1847  85.27239151 failed
## 105 UHM1399.20756_S17       11403        1847  16.19749189 passed
## 106 UHM1400.20757_S29       49300        1847   3.74645030 passed
## 107 UHM1401.20758_S41        4984        1847  37.05858748 failed
## 108 UHM1402.20759_S53       78898        1847   2.34099724 passed
## 109 UHM1403.20760_S65       48278        1847   3.82575914 passed
## 110 UHM1405.20761_S77       51126        1847   3.61264327 passed
## 111 UHM1406.20762_S89       90444        1847   2.04214763 passed
## 112  UHM1414.20763_S6      113476        1847   1.62765695 passed
## 113 UHM1419.20764_S18       22182        1847   8.32657109 passed
## 114 UHM1427.20389_S31       93753        1847   1.97007029 passed
## 115 UHM1428.20390_S43        4532           0   0.00000000 failed
## 116 UHM1429.20391_S55      245778        1847   0.75149118 passed
## 117 UHM1430.20392_S67      846266        1847   0.21825289 passed
## 118 UHM1432.20393_S79      415264        1847   0.44477730 passed
## 119 UHM1435.20388_S19        9292           0   0.00000000 failed
## 120  UHM162.20560_S64       37088        1847   4.98004745 passed
## 121  UHM198.20585_S79        5610        1847  32.92335116 failed
## 122    UHM20.3314_S52     1335639        1847   0.13828587 passed
## 123    UHM20.3315_S64      352109        1847   0.52455348 passed
## 124  UHM204.20409_S81      160159        1847   1.15322898 passed
## 125  UHM206.20410_S93         523           0   0.00000000 failed
## 126  UHM207.20593_S80       76828        1847   2.40407143 passed
## 127  UHM208.20411_S10       34089        1847   5.41817008 passed
## 128  UHM211.20406_S45       87029        1847   2.12228108 passed
## 129  UHM215.20408_S69     3947656        1847   0.04678726 failed
## 130  UHM216.20429_S36      207601        1847   0.88968743 passed
## 131  UHM219.20430_S48        1897        1847  97.36425936 failed
## 132  UHM236.20431_S60      199471        1847   0.92594914 passed
## 133  UHM238.20407_S57       64050        1847   2.88368462 passed
## 134  UHM245.20538_S85      554868        1847   0.33287196 passed
## 135  UHM252.20558_S40       14250        1847  12.96140351 passed
## 136  UHM267.20400_S68      183380        1847   1.00719817 passed
## 137  UHM274.20581_S31       13695        1847  13.48667397 passed
## 138  UHM276.20586_S91       21560        1847   8.56679035 passed
## 139  UHM280.20401_S80       35352        1847   5.22459833 passed
## 140  UHM286.20425_S83      130953        1847   1.41042970 passed
## 141  UHM289.20426_S95        6003        1847  30.76794936 failed
## 142  UHM294.20427_S12       17177        1847  10.75275077 passed
## 143  UHM298.20600_S69      101173        1847   1.82558588 passed
## 144  UHM325.20548_S15       24562        1847   7.51974595 passed
## 145  UHM337.20412_S22       32632        1847   5.66008826 passed
## 146  UHM354.20535_S49       58805        1847   3.14088938 passed
## 147  UHM356.20415_S58       43688        1847   4.22770555 passed
## 148  UHM369.20773_S31       79551        1847   2.32178100 passed
## 149  UHM370.20774_S43       88455        1847   2.08806738 passed
```

```
## 150  UHM372.20775_S55       21119         1847   8.74567925 passed
## 151  UHM373.20776_S67      113026         1847   1.63413728 passed
## 152  UHM374.20777_S79       96042         1847   1.92311697 passed
## 153  UHM375.20778_S91       19019         1847   9.71134129 passed
## 154   UHM377.20779_S8       29234         1847   6.31798591 passed
## 155    UHM38.3376_S36       29112            0   0.00000000 failed
## 156  UHM386.20781_S32       54216         1847   3.40674340 passed
## 157  UHM387.20782_S44       68442         1847   2.69863534 passed
## 158  UHM414.20583_S55      278181         1847   0.66395620 passed
## 159  UHM418.20765_S30       46656         1847   3.95876200 passed
## 160  UHM422.20766_S42       10409         1847  17.74425978 passed
## 161  UHM425.20767_S54       66208         1847   2.78969309 passed
## 162  UHM426.20534_S37        5428         1847  34.02726603 failed
## 163  UHM428.20544_S62       13527         1847  13.65417314 passed
## 164  UHM429.20559_S52       21213         1847   8.70692500 passed
## 165   UHM435.20547_S3       13365         1847  13.81967826 passed
## 166  UHM437.20768_S66      148167         1847   1.24656637 passed
## 167  UHM439.20564_S17       96401         1847   1.91595523 passed
## 168    UHM44.3526_S31        5511         1847  33.51478860 failed
## 169  UHM445.20569_S77       90006         1847   2.05208542 passed
## 170  UHM447.20783_S56      168350         1847   1.09711910 passed
## 171  UHM448.20769_S78       28120         1847   6.56827881 passed
## 172    UHM45.3539_S92       35221            0   0.00000000 failed
## 173  UHM454.20770_S90       43632         1847   4.23313165 passed
## 174  UHM455.20785_S80       15021         1847  12.29611877 passed
## 175  UHM458.20786_S92      102573         1847   1.80066879 passed
## 176   UHM459.20787_S9      106958         1847   1.72684605 passed
## 177   UHM461.20771_S7       77653         1847   2.37853013 passed
## 178  UHM467.20772_S19      195315         1847   0.94565190 passed
## 179  UHM470.20533_S25        9215         1847  20.04340749 failed
## 180  UHM476.20414_S46       20143         1847   9.16943851 passed
## 181  UHM478.20549_S27       27792         1847   6.64579735 passed
## 182  UHM479.20551_S51        2119         1847  87.16375649 failed
## 183   UHM481.20403_S9       15762         1847  11.71805608 passed
## 184  UHM482.20590_S44        2014         1847  91.70804369 failed
## 185  UHM483.20603_S10       90381         1847   2.04357110 passed
## 186  UHM519.20582_S43      220582         1847   0.83733034 passed
## 187  UHM520.20573_S30       68381         1847   2.70104269 passed
## 188 UHM746.21478_S117         927          924  99.67637540 failed
## 189 UHM747.21477_S106         924          924 100.00000000 failed
## 190 UHM748.21467_S170         924          924 100.00000000 failed
## 191 UHM748.21487_S129         928          924  99.56896552 failed
## 192 UHM749.21479_S128         925          924  99.89189189 failed
## 193 UHM759.21466_S159         924          924 100.00000000 failed
## 194 UHM759.21486_S118         938          924  98.50746269 failed
## 195 UHM775.21485_S107         923          923 100.00000000 failed
## 196 UHM776.21482_S161         923          923 100.00000000 failed
## 197 UHM777.21484_S183         925          924  99.89189189 failed
```

```
## 198 UHM779.21468_S181         928          924  99.56896552 failed
## 199 UHM779.21488_S140         924          924 100.00000000 failed
## 200 UHM782.21480_S139         924          924 100.00000000 failed
##  [ reached 'max' / getOption("max.print") -- omitted 60 rows ]
```

```
conc$summary_stats
```

Table 1: Summary Statistics of Spiked S

mean_total_reads_spiked	sd_total_reads_spiked	median_total_reads_spiked	mean_percentag
113,281.1	306,326.4	29,339.5	20.21071

```
# you may keep only passed data
# Filter to get only the samples that passed
passed_samples <- result_perc$Sample[result_perc$Result == "passed"]

# Subset the original phyloseq object to keep only the samples that passed
passed_physeq <- phyloseq::prune_samples(passed_samples, absolute_abundance_16S_OTU_perc)
```

Abundance-based Core microbiome

```
physeq_absolute <- absolute$obj_adj
pps_Abs <- DspikeIn::get_long_format_data(physeq_absolute)

# calculation for relative abundance needs sum of total reads
# total_reads <- sum(pps_Abs$Abundance)

# Generate an alluvial plot

alluvial_plot_abs <- alluvial_plot(
  data = pps_Abs,
  axes = c("Host.genus", "Ecoregion.III","Diet", "Animal.ecomode"),
  abundance_threshold = 10000,
  fill_variable = "Class",
  silent = TRUE,
  abundance_type = "absolute",
  top_taxa = 15,
  text_size = 4,
  legend_ncol = 1,
  custom_colors = DspikeIn::color_palette$light_MG # Use the color palette from DspikeIn
)
```

Data transform & Normalization

you may select to transform your data befor moving forward with Differential Abundance

```
# you may need to normalize/transform your data to reduce biases

ps <- physeq_16SOTU

# TC Normalization
result_TC <- normalization_set(ps, method = "TC", groups = "Host.species")
normalized_ps_TC <- result_TC$dat.normed
scaling_factors_TC <- result_TC$scaling.factor

# UQ Normalization
data("physeq_16SOTU", package = "DspikeIn")
ps <- physeq_16SOTU
result_UQ <- normalization_set(ps, method = "UQ", groups = "Host.species")
normalized_ps_UQ <- result_UQ$dat.normed
scaling_factors_UQ <- result_UQ$scaling.factor

# Median Normalization
data("physeq_16SOTU", package = "DspikeIn")
ps <- physeq_16SOTU
result_med <- normalization_set(ps, method = "med", groups = "Host.species")
normalized_ps_med <- result_med$dat.normed
scaling_factors_med <- result_med$scaling.factor

# DESeq Normalization
data("physeq_16SOTU", package = "DspikeIn")
ps <- physeq_16SOTU
ps_n <- remove_zero_negative_count_samples(ps)
result_DESeq <- normalization_set(ps_n, method = "DESeq", groups = "Animal.type")
normalized_ps_DESeq <- result_DESeq$dat.normed
scaling_factors_DESeq <- result_DESeq$scaling.factor

# Poisson Normalization
data("physeq_16SOTU", package = "DspikeIn")
ps <- physeq_16SOTU
result_Poisson <- normalization_set(ps, method = "Poisson", groups = "Host.genus")
normalized_ps_Poisson <- result_Poisson$dat.normed
scaling_factors_Poisson <- result_Poisson$scaling.factor

# Quantile Normalization
data("physeq_16SOTU", package = "DspikeIn")
ps <- physeq_16SOTU
result_QN <- normalization_set(ps, method = "QN")
normalized_ps_QN <- result_QN$dat.normed
```

```r
scaling_factors_QN <- result_QN$scaling.factor

# TMM Normalization
data("physeq_16SOTU", package = "DspikeIn")
ps <- physeq_16SOTU
result_TMM <- normalization_set(ps, method = "TMM", groups = "Animal.type")
normalized_ps_TMM <- result_TMM$dat.normed
scaling_factors_TMM <- result_TMM$scaling.factor

# CLR Normalization
data("physeq_16SOTU", package = "DspikeIn")
ps <- physeq_16SOTU
result_clr <- normalization_set(ps, method = "clr")
normalized_ps_clr <- result_clr$dat.normed
scaling_factors_clr <- result_clr$scaling.factor

# Rarefying
data("physeq_16SOTU", package = "DspikeIn")
ps <- physeq_16SOTU
result_rar <- normalization_set(ps, method = "rar")
normalized_ps_rar <- result_rar$dat.normed
scaling_factors_rar <- result_rar$scaling.factor

# CSS Normalization
data("physeq_16SOTU", package = "DspikeIn")
ps <- physeq_16SOTU
result_css <- normalization_set(ps, method = "css")
normalized_ps_css <- result_css$dat.normed
scaling_factors_css <- result_css$scaling.factor

# TSS Normalization
data("physeq_16SOTU", package = "DspikeIn")
ps <- physeq_16SOTU
result_tss <- normalization_set(ps, method = "tss")
normalized_ps_tss <- result_tss$dat.normed
scaling_factors_tss <- result_tss$scaling.factor

# RLE Normalization
data("physeq_16SOTU", package = "DspikeIn")
ps <- physeq_16SOTU
result_rle <- normalization_set(ps, method = "rle")
normalized_ps_rle <- result_rle$dat.normed
scaling_factors_rle <- result_rle$scaling.factor
```

Ridge plot

```
rf_physeq <- RandomForest_selected(
  physeq_16SOTU,
  response_var = "Host.genus",
  na_vars = c("Habitat", "Ecoregion.III", "Host.genus", "Diet")
)

ridge_physeq <- ridge_plot_it(rf_physeq, taxrank = "Family", top_n = 10) + scale_fill_manual(va
# ridge_physeq

result_css <- normalization_set(rf_physeq, method = "css")
normalized_ps_css <- result_css$dat.normed

ridge_physeq <- ridge_plot_it(normalized_ps_css, taxrank = "Family", top_n = 10) + scale_fill_m

# ridge_physeq
```

Further analysis

remove the spike-in sp before further analysis

```
absolute <- phyloseq::subset_taxa(physeq_absolute, Genus != "Tetragenococcus")
absolute <- subset_taxa(absolute, Family != "Chloroplast" & Order != "Chloroplast")
Caudate_abs <- phyloseq::subset_samples(absolute, Clade.Order == "Caudate")
Three_Genara_abs <- phyloseq::subset_samples(Caudate_abs, Host.genus %in% c("Desmognathus", "Pl
Three_Genara_abs_BlueRidge <- phyloseq::subset_samples(Three_Genara_abs, Ecoregion.III == "Blue
Desmog_Blue_Ins_16_abs <- phyloseq::subset_samples(Three_Genara_abs_BlueRidge, Host.genus == "D

results_DESeq2 <- perform_and_visualize_DA(
  obj = Desmog_Blue_Ins_16_abs,
  method = "DESeq2",
  group_var = "Host.taxon",
  contrast = c("Desmognathus monticola", "Desmognathus imitator"),
  output_csv_path = "DA_DESeq2.csv",
  target_glom = "Genus",
  significance_level = 0.05
)


head(results_DESeq2$results)
```

```
##    baseMean         logFC     lfcSE          stat    pvalue padj FDR    Significance
## 1 10.000000  9.105617e-08 0.2413445  3.772871e-07 0.9999997    1   1 Not Significant  Desmog
## 2  2.000000  1.334927e-07 0.5326085  2.506394e-07 0.9999998    1   1 Not Significant  Desmog
## 3  1.357143 -1.055569e-06 0.7488460 -1.409594e-06 0.9999989    1   1 Not Significant Desmogn
## 4  2.000000  1.334927e-07 0.5326085  2.506394e-07 0.9999998    1   1 Not Significant  Desmog
```

```
## 5 12.000000  8.255739e-08 0.2212147  3.732003e-07 0.9999997    1   1 Not Significant  Desmog
## 6 13.809524 -3.680398e-01 0.2446159 -1.504562e+00 0.1324366    1   1 Not Significant Desmogn
##                                OTU  Kingdom          Phylum               Class
## 1 b00466354053c9065c8aa3d6fbb33eaa Bacteria  Armatimonadota          uncultured         uncu
## 2 f872c4bf84bcf44434fa2023788f6517 Bacteria  Proteobacteria Gammaproteobacteria       Chroma
## 3 df13f71584d4a579c81d909eaba11a74 Bacteria      Firmicutes    Syntrophomonadia Syntrophomon
## 4 ed285eb1aac505a1f062b482300b69f7 Bacteria      Firmicutes     Symbiobacteriia  Symbiobacte
## 5 63f5509575600a9e7afb6847d6296976 Bacteria Gemmatimonadota    Gemmatimonadetes   Gemmatimon
## 6 fea92298310d6159915036da73e7a88a Bacteria Gemmatimonadota    Gemmatimonadetes   Gemmatimon
##             Genus Species
## 1      uncultured    <NA>
## 2 Thiophaeococcus    <NA>
## 3  Syntrophomonas    <NA>
## 4      uncultured    <NA>
## 5      uncultured    <NA>
## 6    Gemmatimonas    <NA>
```

```
results_DESeq2$obj_significant
```

```
## phyloseq-class experiment-level object
## otu_table()   OTU Table:         [ 27 taxa and 42 samples ]
## sample_data() Sample Data:       [ 42 samples by 34 sample variables ]
## tax_table()   Taxonomy Table:    [ 27 taxa by 7 taxonomic ranks ]
## phy_tree()    Phylogenetic Tree: [ 27 tips and 26 internal nodes ]
## refseq()      DNAStringSet:      [ 27 reference sequences ]
```

```
# results_DESeq2$plot
# results_DESeq2$bar_plot
```

```
# Relative abundance
data("physeq_16SOTU", package = "DspikeIn")
physeq_16SOTU <- tidy_phyloseq_tse(physeq_16SOTU)

relative <- phyloseq::subset_taxa(physeq_16SOTU, Genus != "Tetragenococcus")
Caudate_rel <- phyloseq::subset_samples(relative, Clade.Order == "Caudate")
Three_Genara_rel <- phyloseq::subset_samples(Caudate_rel, Host.genus %in% c("Desmognathus", "P
Three_Genara_rel_BlueRidge <- phyloseq::subset_samples(Three_Genara_rel, Ecoregion.III == "Blue
Desmog_Blue_Ins_16_rel <- phyloseq::subset_samples(Three_Genara_rel_BlueRidge, Host.genus == "

results_DESeq2_rel <- perform_and_visualize_DA(
  obj = Desmog_Blue_Ins_16_rel,
  method = "DESeq2",
  group_var = "Host.taxon",
  contrast = c("Desmognathus monticola", "Desmognathus imitator"),
  output_csv_path = "DA_DESeq2.csv",
  target_glom = "Genus",
```

```
  significance_level = 0.05
)

# head(results_DESeq2_rel$results) #  sig taxa
# results_DESeq2_rel$plot
# results_DESeq2_rel$bar_plot
# results_DESeq2_rel$obj_significant
```

Turnover (Presence/absence analysis)

taxonomic detection consistency between RA and AA OTU tables Presence/absence analysis to detect concordance between AA and RA profiles

```
# ------------------------------------------
# 1. Extract OTU tables
# ------------------------------------------
otu_rel <- otu_table(Desmog_Blue_Ins_16_rel)
otu_abs <- otu_table(Desmog_Blue_Ins_16_abs)

if (taxa_are_rows(Desmog_Blue_Ins_16_rel)) {
  otu_rel <- t(otu_rel)
  otu_abs <- t(otu_abs)
}

otu_rel <- as.matrix(otu_rel)
otu_abs <- as.matrix(otu_abs)

# ------------------------------------------
# 2. Convert to Presence/Absence
# ------------------------------------------
otu_rel_pa <- (otu_rel > 0) * 1
otu_abs_pa <- (otu_abs > 0) * 1

# ------------------------------------------
# 3. Identify common samples & taxa
# ------------------------------------------
shared_samples <- intersect(rownames(otu_rel_pa), rownames(otu_abs_pa))
shared_taxa <- intersect(colnames(otu_rel_pa), colnames(otu_abs_pa))

# Subset to common set
otu_rel_pa <- otu_rel_pa[shared_samples, shared_taxa]
otu_abs_pa <- otu_abs_pa[shared_samples, shared_taxa]

# ------------------------------------------
# 4. Compare presence/absence profiles
```

```r
# ----------------------------------------
# Shared presence across both AA and RA
shared_pa <- (otu_rel_pa + otu_abs_pa) == 2
total_present <- (otu_rel_pa + otu_abs_pa) >= 1

# Calculate percent agreement (global)
percent_shared <- sum(shared_pa) / sum(total_present)

cat("percent of shared taxa detections (AA vs RA):",
    round(100 * percent_shared, 1), "%\n")
```

```
## percent of shared taxa detections (AA vs RA): 99.6 %
```

Differential abundance (Single and Multilayer Pairwise)

```r
# DspikeIn: Differential Abundance Examples
# Using perform_and_visualize_DA() with multiple contrast scenarios


# Load example dataset from the package
data("physeq_16SOTU", package = "DspikeIn")



# 1. Single Contrast

res_single <- perform_and_visualize_DA(
  obj = physeq_16SOTU,
  method = "DESeq2",
  group_var = "Diet",
  contrast = c("Insectivore", "Carnivore"),
  target_glom = "Genus")



# 2. Single Factor - All Pairwise Contrasts

# level names for DESeq2 compatibility
sample_data(physeq_16SOTU)$Host.taxon <- factor(make.names(sample_data(physeq_16SOTU)$Host.taxo

# Get unique levels
host_levels <- levels(sample_data(physeq_16SOTU)$Host.taxon)

# contrast list
```

```r
contrast_named <- list(
  Host.taxon = combn(host_levels, 2, simplify = FALSE))

# multiple pairwise contrasts
res_multi <- perform_and_visualize_DA(
  obj = physeq_16SOTU,
  method = "DESeq2",
  group_var = "Host.taxon",
  contrast = contrast_named,
  target_glom = "Genus")



# 3. Single Factor - Selected Contrasts

contrast_list <- list(
  c("Insectivore", "Carnivore"),
  c("Omnivore", "Herbivore"))

res_selected <- perform_and_visualize_DA(
  obj = physeq_16SOTU,
  method = "DESeq2",
  group_var = "Diet",
  contrast = contrast_list,
  global_fdr = TRUE)



# 4. Multiple Factors - Selected Contrasts

contrast_named <- list(
  Diet = list(
    c("Insectivore", "Carnivore"),
    c("Omnivore", "Carnivore") ),
  Animal.type = list(
    c("Frog", "Salamander") ))

res_multi_factor <- perform_and_visualize_DA(
  obj = physeq_16SOTU,
  method = "DESeq2",
  contrast = contrast_named,
  target_glom = "Genus",
  significance_level = 0.01,
  global_fdr = TRUE)
```

```r
# 5. Multiple Factors - all pairwise contrasts

# Ensure clean factor levels
sample_data(physeq_16SOTU)$Host.taxon <- droplevels(factor(sample_data(physeq_16SOTU)$Host.taxo
sample_data(physeq_16SOTU)$Habitat <- droplevels(factor(sample_data(physeq_16SOTU)$Habitat))

#  pairwise contrasts
host_levels <- levels(sample_data(physeq_16SOTU)$Host.taxon)
habitat_levels <- levels(sample_data(physeq_16SOTU)$Habitat)

contrast_named <- list(
  Host.taxon = combn(host_levels, 2, simplify = FALSE),
  Habitat = combn(habitat_levels, 2, simplify = FALSE))

results_multi_factor <- perform_and_visualize_DA(
  obj = physeq_16SOTU,
  method = "DESeq2",
  contrast = contrast_named,
  target_glom = "Genus",
  significance_level = 0.01)



# 6. Multiple Factors - combined group contrasts (interactions)

# Create a combined grouping variable
sample_data(physeq_16SOTU)$ComboGroup <- factor(interaction(
  sample_data(physeq_16SOTU)$Animal.type,
  sample_data(physeq_16SOTU)$Diet,
  drop = TRUE))

#  selected contrasts
contrast_list <- list(
  c("Salamander.Insectivore", "Lizard.Insectivore"),
  c("Salamander.Carnivore", "Snake.Carnivore"),
  c("Salamander.Carnivore", "Frog.Carnivore"))

#  multi-contrast analysis
res_combo <- perform_and_visualize_DA(
  obj = physeq_16SOTU,
  method = "DESeq2",
  group_var = "ComboGroup",
  contrast = contrast_list,
  target_glom = "Genus",
  global_fdr = TRUE)
```

Visualization

```
#   Visualization of community composition


Rel <- phyloseq::subset_taxa(physeq_16SOTU, Genus != "Tetragenococcus")
Prok_OTU_spiked <- phyloseq::subset_samples(Rel, spiked.volume %in% c("2", "1"))
Prok_OTU_spiked <- phyloseq::subset_samples(Prok_OTU_spiked, sample.or.blank != "blank")
Prok_OTU_sal <- phyloseq::subset_samples(Prok_OTU_spiked, Animal.type == "Salamander")

Prok_OTU_sal <- tidy_phyloseq_tse(Prok_OTU_sal)

TB<-taxa_barplot(
  Prok_OTU_sal,
  target_glom = "Genus",
  custom_tax_names = NULL,
  normalize = TRUE,
  treatment_variable = "Habitat",
  abundance_type = "relative",
  x_angle = 25,
  fill_variable = "Family",
  facet_variable = "Diet",
  top_n_taxa = 20,
  palette = DspikeIn::color_palette$mix_MG,
  legend_size = 11,
  legend_columns = 1,
  x_scale = "free",
  xlab = NULL)
```

Microbial dynamics & NetWork comparision

```
# 1. Initialization and loading NetWorks for Comparision


# library(SpiecEasi)
# library(ggnet)
# library(igraph)
library(DspikeIn)
library(tidyr)
library(dplyr)
library(ggpubr)
library(igraph)
```

```r
# To create a microbial co-occurrence network, you can refer to the SpiecEasi package availabl
# SpiecEasi GitHub Repository  https://github.com/zdk123/SpiecEasi

# herp.Bas.rel.f is a merged phyloseq object for both bacterial and fungal domains
# spiec.easi(herp.Bas.rel.f, method='mb', lambda.min.ratio=1e-3, nlambda=250,pulsar.select=TRU

Complete <- load_graphml("Complete.graphml")
NoBasid <- load_graphml("NoBasid.graphml")
NoHubs <- load_graphml("NoHubs.graphml")

# result <- weight_Network(graph_path = "Complete.graphml")
# result

result_kk <- degree_network(
  graph_path = load_graphml("Complete.graphml"),
  save_metrics = TRUE,
  layout_type = "stress"
)
# print(result_kk$plot)
```

Network Topological Metrics

```r
# 2. Metrics Calculation


result_Complete <- node_level_metrics(Complete)
result_NoHubs <- node_level_metrics(NoHubs)
result_NoBasid <- node_level_metrics(NoBasid)

Complete_metrics <- result_Complete$metrics
Nohub_metrics <- result_NoHubs$metrics
Nobasid_metrics <- result_NoBasid$metrics

Complete_metrics <- data.frame(lapply(Complete_metrics, as.character), stringsAsFactors = FALSE
Nohub_metrics <- data.frame(lapply(Nohub_metrics, as.character), stringsAsFactors = FALSE)
Nobasid_metrics <- data.frame(lapply(Nobasid_metrics, as.character), stringsAsFactors = FALSE)


print(igraph::vcount(Complete)) # Number of nodes
```

```
## [1] 308
```

```r
print(igraph::ecount(Complete)) # Number of edges
```

```
## [1] 1144
```

```r
print(igraph::vcount(NoBasid))
```

```
## [1] 307
```

```r
print(igraph::ecount(NoBasid))
```

```
## [1] 1187
```

```r
print(igraph::vcount(NoHubs))
```

```
## [1] 286
```

```r
print(igraph::ecount(NoHubs))
```

```
## [1] 916
```

```r
metrics_scaled <- bind_rows(
  Complete_metrics %>% mutate(Network = "Complete"),
  Nohub_metrics %>% mutate(Network = "NoHubs"),
  Nobasid_metrics %>% mutate(Network = "NoBasid")
) %>%
  dplyr::mutate(dplyr::across(where(is.numeric), scale))

metrics_long_scaled <- metrics_scaled %>%
  tidyr::pivot_longer(cols = -c(Node, Network), names_to = "Metric", values_to = "Value")
```

bind the metrics to plot them

```r
# 3. Visualization


# Remove missing values
metrics_long_scaled <- na.omit(metrics_long_scaled)

# We visualize only six metrics
selected_metrics <- c(
  "Degree", "Closeness", "Betweenness",
  "EigenvectorCentrality", "PageRank", "Transitivity"
)
```

```r
metrics_long_filtered <- metrics_long_scaled %>%
  filter(Metric %in% selected_metrics) %>%
  mutate(
    Value = as.numeric(as.character(Value)),
    Network = recode(Network,
      "Complete" = "Complete Network",
      "NoHubs" = "Network & Module Hubs Removed",
      "NoBasid" = "Basidiobolus Subnetwork Removed" ) ) %>%
  na.omit() # Remove any NA if any

metrics_long_filtered$Network <- factor(metrics_long_filtered$Network,
  levels = c(
    "Complete Network",
    "Network & Module Hubs Removed",
    "Basidiobolus Subnetwork Removed" ))

# DspikeIn::color_palette$light_MG
network_colors <- c(
  "Complete Network" = "#F1E0C5",
  "Network & Module Hubs Removed" = "#D2A5A1",
  "Basidiobolus Subnetwork Removed" = "#B2C3A8")

# statistical comparisons a vs b
comparisons <- list(
  c("Complete Network", "Network & Module Hubs Removed"),
  c("Complete Network", "Basidiobolus Subnetwork Removed"),
  c("Network & Module Hubs Removed", "Basidiobolus Subnetwork Removed"))

networks_in_data <- unique(metrics_long_filtered$Network)
comparisons <- comparisons[sapply(comparisons, function(pair) all(pair %in% networks_in_data))]

met <- ggplot(metrics_long_filtered, aes(x = Network, y = Value, fill = Network)) +
  geom_boxplot(outlier.shape = NA) +
  geom_jitter(aes(color = Network),
    position = position_jitter(0.2), alpha = 0.2, size = 1.5 ) +
  scale_fill_manual(values = network_colors) +
  scale_color_manual(values = network_colors) +
  facet_wrap(~Metric, scales = "free_y", labeller = label_wrap_gen(width = 20)) +
  ggpubr::stat_compare_means(method = "wilcox.test", label = "p.signif", comparisons = comparis
  theme_minimal() +
  theme(
    axis.title.x = element_blank(),
    axis.title.y = element_blank(),
    axis.text.x = element_text(size = 10, angle = 10, hjust = 0.8),
    strip.text = element_text(size = 12, margin = margin(t = 15, b = 5)),
    legend.position = "top",
    legend.text = element_text(size = 13),
```

```
    legend.title = element_text(size = 13, face = "bold"),
    plot.title = element_text(size = 13, face = "bold")
  ) + labs(title = "Selected Node Metrics Across Networks", fill = "Network Type", color = "Net
```

To find first and second neighbors your target node

```
Complete <- load_graphml("Complete.graphml")
result2 <- extract_neighbors(
  graph = Complete,
  target_node = "OTU69:Basidiobolus_sp", mode = "all")
head(result2$summary)
```

```
##             Type                           Node
## 1 First Neighbor            OTU8:Mortierella_sp
## 2 First Neighbor           OTU13:Mortierella_sp
## 3 First Neighbor           OTU15:Mortierella_sp
## 4 First Neighbor            OTU16:Ascomycota_sp
## 5 First Neighbor            OTU18:Helotiales_sp
## 6 First Neighbor OTU19:Margaritispora_monticola
```

Compute Node-Level Metrics

Metric	Description
'Node'	Node name (character format)
'Degree'	Number of edges connected to the node
'Strength'	Sum of edge weights connected to the node
'Closeness'	Closeness centrality (normalized, based on shortest paths)
'Betweenness'	Betweenness centrality (normalized, measures control over network flow)
'EigenvectorCentrality'	Eigenvector centrality (importance based on connections to influential nodes)
'PageRank'	PageRank score (importance based on incoming links)
'Transitivity'	Local clustering coefficient (tendency of a node to form triangles)
'Coreness'	Node's coreness (from k-core decomposition)
'Constraint'	Burt's constraint (measures structural holes in a node's ego network)
'EffectiveSize'	Inverse of constraint (larger values = more non-redundant connections)
'Redundancy'	Sum of constraint values of a node's alters
'Community'	Community assignment from Louvain clustering

Metric	Description
‘Efficiency’	Global efficiency (average inverse shortest path length)
‘Local_Efficiency’	Local efficiency (subgraph efficiency for a node’s neighbors)
‘Within_Module_Connectivity’	Proportion of neighbors in the same community
‘Among_Module_Connectivity’	Proportion of neighbors in different communities

```
# Load required libraries
library(igraph)
library(dplyr)
library(tidyr)
library(ggplot2)
library(ggrepel)

# Compute Node-Level Metrics
completeMetrics <- node_level_metrics(Complete)
NoHubsMetrics <- node_level_metrics(NoHubs)
NoBasidMetrics <- node_level_metrics(NoBasid)

# completeMetrics$facet_plot

# Ensure each dataset has a "Network" column before combining
completeMetrics$metrics <- completeMetrics$metrics %>% mutate(Network = "Complete Network")
NoHubsMetrics$metrics <- NoHubsMetrics$metrics %>% mutate(Network = "Network & Module Hubs Remo
NoBasidMetrics$metrics <- NoBasidMetrics$metrics %>% mutate(Network = "Basidiobolus Subnetwork

# Combine All Data
combined_data <- bind_rows(
  completeMetrics$metrics,
  NoHubsMetrics$metrics,
  NoBasidMetrics$metrics
)


# Add Node Identifier if missing
if (!"Node" %in% colnames(combined_data)) {
  combined_data <- combined_data %>% mutate(Node = rownames(.))
}

# Convert `Network` into Factor
combined_data$Network <- factor(combined_data$Network, levels = c(
  "Complete Network",
  "Network & Module Hubs Removed",
  "Basidiobolus Subnetwork Removed"))

# Convert Data to Long Format
metrics_long <- combined_data %>%
```

```r
  pivot_longer(
    cols = c("Redundancy", "Efficiency", "Betweenness"),
    names_to = "Metric", values_to = "Value")


# Define Custom Colors and Shapes
  network_colors <- c(
  "Complete Network" = "#F1E0C5",
  "Network & Module Hubs Removed" = "#D2A5A1",
  "Basidiobolus Subnetwork Removed" = "#B2C3A8")

network_shapes <- c(
  "Complete Network" = 21,
  "Network & Module Hubs Removed" = 22,
  "Basidiobolus Subnetwork Removed" = 23)

# Determine Top 30% of Nodes to Label/Optional
metrics_long <- metrics_long %>%
  group_by(Network, Metric) %>%
  mutate(Label = ifelse(rank(-Value, ties.method = "random") / n() <= 0.3, Node, NA))

# ?quadrant_plot() can creat plot for indivisual network
# plot <- quadrant_plot(metrics, x_metric = "Degree", y_metric = "Efficiency")

# Create comparision Plots
create_metric_plot <- function(metric_name, data, title) {
  data_filtered <- data %>% filter(Metric == metric_name)
  median_degree <- median(data_filtered$Degree, na.rm = TRUE)
  median_value <- median(data_filtered$Value, na.rm = TRUE)

  ggplot(data_filtered, aes(x = Degree, y = Value, fill = Network)) +
    geom_point(aes(shape = Network), size = 3, stroke = 1, color = "black") +
    geom_text_repel(aes(label = Label), size = 3, max.overlaps = 50) +
    scale_fill_manual(values = network_colors) +
    scale_shape_manual(values = network_shapes) +
    geom_vline(xintercept = median_degree, linetype = "dashed", color = "black", size = 1) +
    geom_hline(yintercept = median_value, linetype = "dashed", color = "black", size = 1) +
    labs(
      title = title,
      x = "Degree",
      y = metric_name,
      fill = "Network",
      shape = "Network" ) +
    theme_minimal() +
    theme(
      plot.title = element_text(
        hjust = 0.5, size = 16, face = "bold",
```

```
        margin = margin(t = 10, b = 20) # Moves the title downward
      ),
      axis.title = element_text(size = 14, face = "bold"),
      legend.position = "top",
      legend.title = element_text(size = 14, face = "bold"),
      legend.text = element_text(size = 12)  )
}


# Generate Plots
plot_redundancy <- create_metric_plot("Redundancy", metrics_long, "Redundancy vs. Degree Across
plot_efficiency <- create_metric_plot("Efficiency", metrics_long, "Efficiency vs. Degree Across
plot_betweenness <- create_metric_plot("Betweenness", metrics_long, "Betweenness vs. Degree Ac

# Save Plots
# ggsave("plot_redundancy_20percent.png", plot_redundancy, width = 8, height = 6)
# ggsave("plot_efficiency_20percent.png", plot_efficiency, width = 8, height = 6)
# ggsave("plot_betweenness_20percent.png", plot_betweenness, width = 8, height = 6)

# Print Plots
# print(plot_redundancy)
# print(plot_efficiency)
# print(plot_betweenness)
```

Compute degree metrics and visualize the network

```
# Compute degree metrics and visualize the network
# Options: `"stress"` (default), `"graphopt"`, `"fr"`

result <- degree_network(graph_path = Complete, save_metrics = TRUE)
# print(result$metrics)
# print(result$plot)


# Compute network weights for different graph structures
NH <- weight_Network(graph_path = "NoHubs.graphml")
NB <- weight_Network(graph_path = "NoBasid.graphml")
C <- weight_Network(graph_path = "Complete.graphml")

# Extract metrics from the computed network weights
CompleteM <- C$metrics
NoHubsM <- NH$metrics
NoBasidM <- NB$metrics
```

```r
# Combine metrics into a single dataframe for comparison
df <- bind_rows(
  CompleteM %>% mutate(Group = "CompleteM"),
  NoHubsM %>% mutate(Group = "NoHubsM"),
  NoBasidM %>% mutate(Group = "NoBasidM")) %>%
  pivot_longer(cols = -Group, names_to = "Metric", values_to = "Value")

# Aggregate the total values by metric and group
df_bar <- df %>%
  group_by(Metric, Group) %>%
  summarise(Total_Value = sum(Value), .groups = "drop")

# Plot the metrics comparison
pg<-ggplot(df_bar, aes(x = Metric, y = log1p(Total_Value), fill = Group)) +
  geom_bar(stat = "identity", position = "dodge", alpha = 0.8) +
  theme_minimal(base_size = 14) +
  theme(axis.text.x = element_text(angle = 45, hjust = 1)) +
  scale_fill_manual(values = c("#F1E0C5", "#D2A5A1", "#B2C3A8")) +
  labs(
    title = "Total Network Metrics Comparison",
    x = "Metric",
    y = "Total Value (log-scaled)",
    fill = "Group" )
```

Small-World Index Analysis

To determine whether the complete network exhibited small-world topology, we computed the Small-World Index (SWI; σ) following the quantitative framework established by Humphries M, Gurney K, 2008. PLoS ONE 4 (Eq.1) (SWI; σ) = (Global clustering coefficient real/Global clustering coefficient random)/(Avg Path real/Avg Path random))

```r
library(igraph)
library(tidygraph)
library(ggraph)
library(DspikeIn)


# AA abundance
Complete<-load_graphml("Complete.graphml")
# NoHubs<-load_graphml("NoHubs.graphml")
# NoBasid<-load_graphml("NoBasid.graphml")

# RA abundance
# Complete_Rel<-load_graphml("~/Downloads/herp.spiecsym.Rel.graphml")
```

```r
# Degree distribution
# deg <- degree(Complete_Rel)
deg <- degree(Complete)

hist(deg, breaks = 30, main = "Degree Distribution", xlab = "Degree")
fit <- fit_power_law(deg + 1)  # Avoid zero degrees
print(fit)


# ---- empirical network ----
g_empirical = Complete
g_empirical = Complete_Rel



# ---- Calculate real network metrics ----
creal <- transitivity(g_empirical, type = "global")
E(g_empirical)$weight <- abs(E(g_empirical)$weight)
lreal <- mean_distance(g_empirical, directed = FALSE, unconnected = TRUE, weights = E(g_empiri

# ---- Generate 1000 Erdős-Rényi random graphs with same size ----
set.seed(42)  # for reproducibility
n_nodes <- vcount(g_empirical)
n_edges <- ecount(g_empirical)

crand_vals <- numeric(1000)
lrand_vals <- numeric(1000)

for (i in 1:1000) {
g_rand <- sample_gnm(n = n_nodes, m = n_edges, directed = FALSE)
if (!is_connected(g_rand)) next

  crand_vals[i] <- transitivity(g_rand, type = "global")
  lrand_vals[i] <- mean_distance(g_rand, directed = FALSE, unconnected = TRUE)
}

# ---- Calculate mean values across random graphs ----
crand <- mean(crand_vals, na.rm = TRUE)
lrand <- mean(lrand_vals, na.rm = TRUE)

# ---- Compute Small-World Index ( ) ----
sigma <- (creal / crand) / (lreal / lrand)


cat("Global clustering coefficient (real):", creal, "\n")
cat("Average path length (real):", lreal, "\n")
cat("Mean clustering coefficient (random):", crand, "\n")
```

```
cat("Mean path length (random):", lrand, "\n")
cat("Small-World Index ( ):", round(sigma, 2), "\n")
```

```
sessionInfo()
```

```
## R version 4.5.0 (2025-04-11)
## Platform: aarch64-apple-darwin20
## Running under: macOS Sequoia 15.5
##
## Matrix products: default
## BLAS:   /System/Library/Frameworks/Accelerate.framework/Versions/A/Frameworks/vecLib.framew
## LAPACK: /Library/Frameworks/R.framework/Versions/4.5-arm64/Resources/lib/libRlapack.dylib;
##
## 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: Europe/Berlin
## tzcode source: internal
##
## attached base packages:
## [1] stats4    stats     graphics  grDevices utils     datasets  methods   base
##
## other attached packages:
##  [1] ggrepel_0.9.6                   igraph_2.1.4                    ggpubr_0.6.0
##  [4] tidyr_1.3.1                     DspikeIn_0.99.11                rmarkdown_2.29
##  [7] mia_1.16.0                      MultiAssayExperiment_1.34.0     vegan_2.7-1
## [10] permute_0.9-7                   microbiome_1.30.0               tibble_3.3.0
## [13] dplyr_1.1.4                     flextable_0.9.9                 TreeSummarizedExperime
## [16] SingleCellExperiment_1.30.1     SummarizedExperiment_1.38.1     Biobase_2.68.0
## [19] GenomicRanges_1.60.0            MatrixGenerics_1.20.0           matrixStats_1.5.0
## [22] ggplot2_3.5.2                   ggstar_1.0.4                    phyloseq_1.52.0
## [25] Biostrings_2.76.0               GenomeInfoDb_1.44.0             XVector_0.48.0
## [28] IRanges_2.42.0                  S4Vectors_0.46.0                BiocGenerics_0.54.0
## [31] generics_0.1.4                  BiocStyle_2.36.0
##
## loaded via a namespace (and not attached):
##   [1] urlchecker_1.0.1           TH.data_1.1-3              vctrs_0.6.5                di
##   [5] BiocBaseUtils_1.10.0       rbiom_2.2.0                parallelly_1.45.0          ms
##   [9] MASS_7.3-65                fontLiberation_0.1.0       reshape2_1.4.4             ht
##  [13] foreach_1.5.2              withr_3.0.2                xfun_0.52                  gg
##  [17] ellipsis_0.3.2             survival_3.8-3             memoise_2.0.1              co
##  [21] ggbeeswarm_0.7.2           emmeans_1.11.1             profvis_0.4.0              sy
##  [25] ragg_1.4.0                 tidytree_0.4.6             zoo_1.8-14                 gg
##  [29] Formula_1.2-5              sys_3.4.3                  promises_1.3.3             ht
##  [33] rstatix_0.7.2              rhdf5filters_1.20.0        ps_1.9.1                   rl
##  [37] rstudioapi_0.17.1          UCSC.utils_1.4.0           miniUI_0.1.2               gg
##  [41] processx_3.8.6             curl_6.3.0                 ScaledMatrix_1.16.0        gg
```

```
##  [45] polyclip_1.10-7              randomForest_4.7-1.2         GenomeInfoDbData_1.2.14
##  [49] SparseArray_1.8.0            RBGL_1.84.0                  xtable_1.8-4
##  [53] desc_1.4.3                   ade4_1.7-23                  evaluate_1.0.4
##  [57] BiocFileCache_2.16.0         hms_1.1.3                    bookdown_0.43
##  [61] filelock_1.0.3               readxl_1.4.5                 readr_2.1.5
##  [65] later_1.4.2                  viridis_0.6.5                ggtree_3.16.0
##  [69] DECIPHER_3.4.0               XML_3.99-0.18                scuttle_1.18.0
##  [73] nlme_3.1-168                 iterators_1.0.14             decontam_1.28.0
##  [77] beachmat_2.24.0              stringi_1.8.7                biomformat_1.36.0
##  [81] plyr_1.8.9                   crayon_1.5.3                 abind_1.4-8
##  [85] ggtext_0.1.2                 gridGraphics_0.5-1           locfit_1.5-9.12
##  [89] bit_4.6.0                    sandwich_3.1-1               fastmatch_1.1-6
##  [93] multcomp_1.4-28              textshaping_1.0.1            BiocSingular_1.24.0
##  [97] slam_0.1-55                  bslib_0.9.0                  multtest_2.64.0
## [101] splines_4.5.0                Rcpp_1.0.14                  dbplyr_2.5.0
## [105] BiocCheck_1.44.2             cellranger_1.1.0             gridtext_0.1.5
## [109] blob_1.2.4                   utf8_1.2.6                   fs_1.6.6
## [113] pkgbuild_1.4.8               ggsignif_0.6.4               ggplotify_0.1.2
## [117] Matrix_1.7-3                 callr_3.7.6                  statmod_1.5.0
## [121] tweenr_2.0.3                 pkgconfig_2.0.3              tools_4.5.0
## [125] RSQLite_2.4.1                viridisLite_0.4.2            DBI_1.2.3
## [129] scales_1.4.0                 grid_4.5.0                   credentials_2.0.2
## [133] fillpattern_1.0.2            broom_1.0.8                  sass_0.4.10
## [137] patchwork_1.3.0              coda_0.19-4.1                BiocManager_1.30.26
## [141] carData_3.0-5                farver_2.1.2                 tidygraph_1.3.1
## [145] biocViews_1.76.0             yaml_2.3.10                  roxygen2_7.3.2
## [149] purrr_1.0.4                  lifecycle_1.0.4              rsconnect_1.4.2
## [153] mvtnorm_1.3-3                bluster_1.18.0               sessioninfo_1.2.3
## [157] BiocParallel_1.42.1          gtable_0.3.6                 ggridges_0.5.6
## [161] ape_5.8-1                    testthat_3.2.3               limma_3.64.1
## [165] edgeR_4.6.2                  bitops_1.0-9                 bit64_4.6.0-1
## [169] Rtsne_0.17                   yulab.utils_0.2.0            BiocNeighbors_2.2.0
## [173] jquerylib_0.1.4              lazyeval_0.2.2               shiny_1.10.0
## [177] rappdirs_0.3.3               tinytex_0.57                 glue_1.8.0
## [181] gdtools_0.4.2                RCurl_1.98-1.17              rprojroot_2.0.4
## [185] gridExtra_2.3                R6_2.6.1                     DESeq2_1.48.1
## [189] cluster_2.1.8.1              pkgload_1.4.0                Rhdf5lib_1.30.0
## [193] aplot_0.2.6                  DirichletMultinomial_1.50.0  DelayedArray_0.34.1
## [197] vipor_0.4.7                  ggforce_0.5.0                xml2_1.3.8
## [201] car_3.1-3                    rsvd_1.0.5                   fontquiver_0.2.1
## [205] htmlwidgets_1.6.4            RColorBrewer_1.1-3           rlang_1.1.6
## [209] uuid_1.2-1                   remotes_2.5.0                RUnit_0.4.33.1
## [213] phangorn_2.12.1              beeswarm_0.4.0
```

Dspikeln volume protocol

Spike-in volume Protocol;

The species Tetragenococcus halophilus (bacterial spike; ATCC33315) and Dekkera bruxellensis (fungal spike; WLP4642-White Labs) were selected as taxa to spike into gut microbiome samples as they were not found in an extensive collection of wildlife skin (GenBank BioProjects: PRJNA1114724, PRJNA 1114659) or gut microbiome samples. Stock cell suspensions of both microbes were grown in either static tryptic soy broth (T. halophilus) or potato dextrose broth (D. bruxellensis) for 72 hours then serially diluted and optical density (OD600) determined on a ClarioStar plate reader. Cell suspensions with an optical density of 1.0, 0.1, 0.01, 0.001 were DNA extracted using the Qiagen DNeasy Powersoil Pro Kit. These DNA isolations were used as standards to determine the proper spike in volume of cells to represent 0.1-10% of a sample (Rao et al., 2021b) Fecal pellets (3.1 ± 1.6 mg; range = 1 – 5.1 mg) from an ongoing live animal study using wood frogs (Lithobates sylvaticus) were used to standardize the input material for the development of this protocol. A total of (n=9) samples were used to validate the spike in protocol. Each fecal sample was homogenized in 1mL of sterile molecular grade water then 250uL of fecal slurry was DNA extracted as above with and without spiked cells. Two approaches were used to evaluate the target spike-in of 0.1-10%, the range of effective spike-in percentage described in (Rao et al., 2021b), including 1) an expected increase of qPCR cycle threshold (Ct) value that is proportional to the amount of spiked cells and 2) the expected increase in copy number of T. halophilus and D. bruxellensis in spiked vs. unspiked samples. A standard curve was generated using a synthetic fragment of DNA for the 16S-V4 rRNA and ITS1 rDNA regions of T. halophilus and D. bruxellensis, respectively. The standard curve was used to convert Ct values into log copy number for statistical analyses (detailed approach in[2, 3]) using the formula y = -0.2426x + 10.584 for T. halophilus and y = -0.3071x + 10.349 for D. bruxellensis, where x is the average Ct for each unknown sample. Quantitative PCR (qPCR) was used to compare known copy numbers from synthetic DNA sequences of T. halophilus and D. bruxellensis to DNA extractions of T. halophilus and D. bruxellensis independently, and wood frog fecal samples with and without spiked cells. SYBR qPCR assays were run at 20ul total volume including 10ul 2X Quantabio PerfeCTa SYBR Green Fastmix, 1ul of 10uM forward and reverse primers, 1ul of ArcticEnzymes dsDNAse master mix clean up kit, and either 1ul of DNA for D. bruxellensis or 3ul for T. halophilus. Different volumes of DNA were chosen for amplification of bacteria and fungi due to previous optimization of library preparation and sequencing steps [3]. The 515F [4] and 806R [5] primers were chosen to amplify bacteria and ITS1FI2 [6] and ITS2 for fungi, as these are the same primers used during amplicon library preparation and sequencing. Cycling conditions on an Agilent AriaMX consisted of 95 C for 3 mins followed by 40 cycles of 95 C for 15 sec, 60 C for 30 sec and 72 C for 30 sec. Following amplification, a melt curve was generated under the following conditions including 95 C for 30 sec, and a melt from 60 C to 90 C increasing in resolution of 0.5 C in increments of a 5 sec soak time. To validate the spike in protocol we selected two sets of fecal samples including 360 samples from a diverse species pool of frogs, lizards, salamanders and snakes and a more targeted approach of 122 fecal samples from three genera of salamanders from the Plethodontidae. (Supplemental Table #). FFecal samples were not weighed in the field, rather, a complete fecal pellet was diluted in an equal volume of sterile water and standardized volume of fecal slurry (250µL) extracted for independent samples.. A volume of 1ul T. halophilus (1847 copies) and 1ul D. bruxellensis (733 copies) were spiked into each fecal sample then DNA was extracted as above, libraries constructed, and amplicon sequenced on an Illumina MiSeq as in [7] .

https://github.com/mghotbi/DspikeIn