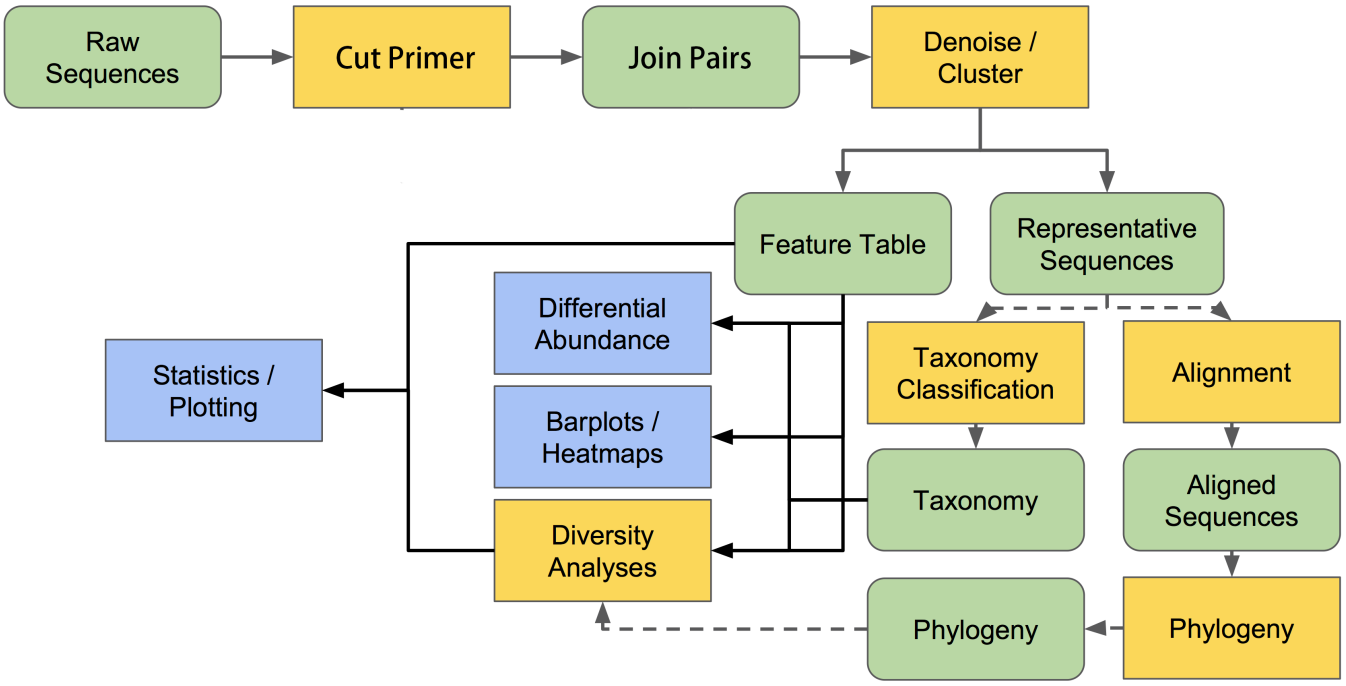
目录

1. 课程简介

扩增子测序的分析流程

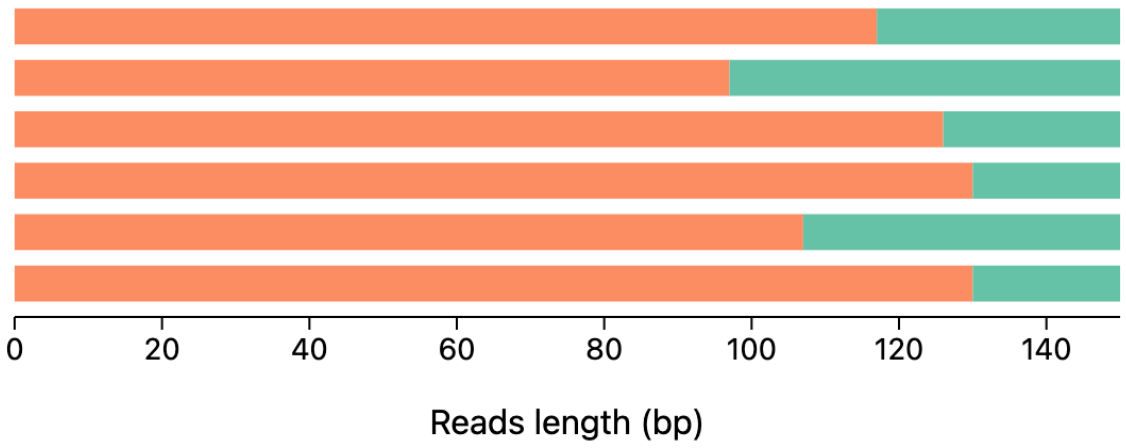


统一裁剪的弊端

非统一裁剪文章例子1

非统一裁剪文章例子2

- 每一对reads不合格的地方都是随机的，统一裁剪会导致某些合格的区域被过度裁剪，而且是影响了所有参与计算的样本。
- 导致分析过程繁琐，不同批次的分析都要人工去决定裁剪到哪个长度。
- 对于输入是指控过后的长短不一的数据，低于裁剪长度的read会被直接丢弃，但只要这对reads有足够的overlap区域就可以拼接起来，没必要丢弃。



适合人群

- 有一定的生信基础
- 会基础的linux和R语言
- 希望自己分析16S扩增子

你将学到什么?

- 扩增子测序基本原理
- 扩增子分析的基本流程和相关概念
- 一些实用的linux技巧、zshell命令
- 使用mamba搭建分析环境
- 相关软件参数的使用经验
- 相关R包的使用
- snakemake分析流程自动化

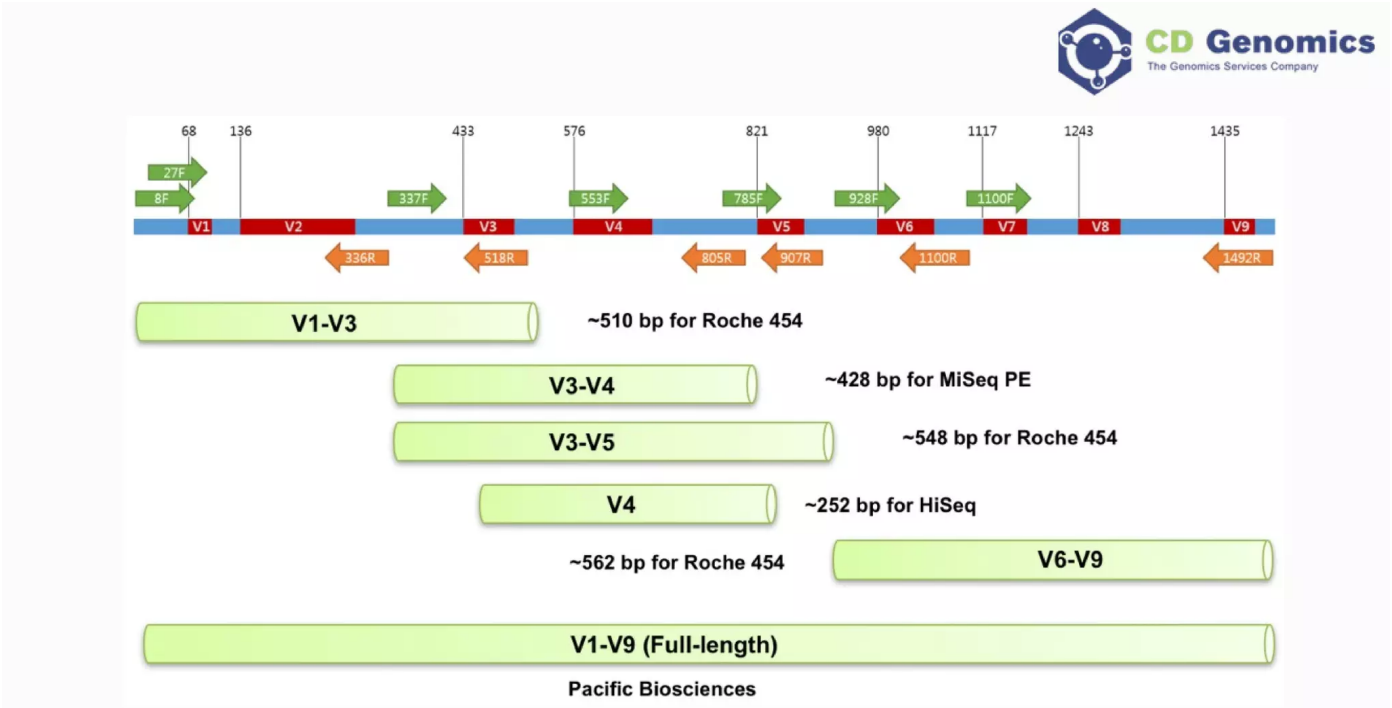
资料下载

网页版讲义更好用!

- 课程资料及网页版讲义：链接
- 添加微信进交流群，请备注来自CCtalk

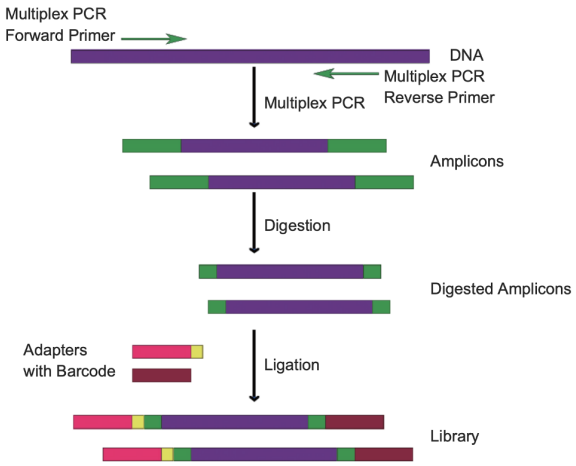
2. 16S rRNA基因扩增子测序的基本原理

原理ppt

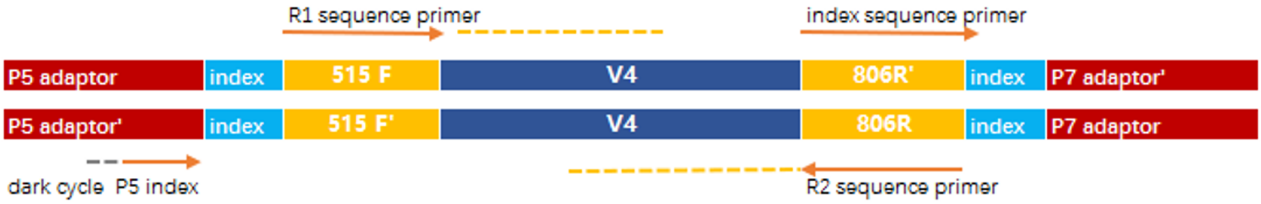


The suitable primer sets for various sequencing system

- 建库流程



VAHTS® AmpSeq Library Prep Kit V3

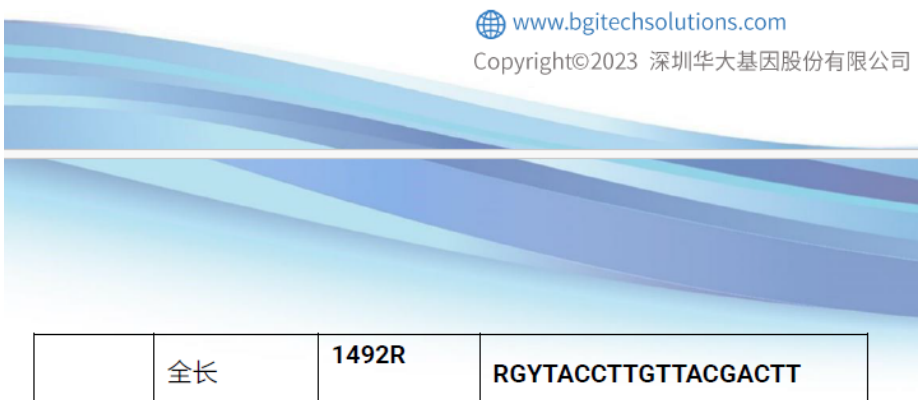


- 测序引物

3、测序策略

	扩增区域	引物名称	引物对
细菌	V4	515F	GTGCCAGCMGCCGCGGTAA
		806R	GGACTACHVGGGTWTCTAAT
	V1-V3	8F	AGAGTTTGATYMTGGCTCAG
		518R	ATTACCGCGGCTGCTGG
	V3-V4	341F	ACTCCTACGGGAGGCAGCAG
		806R	GGACTACHVGGGTWTCTAAT
	V4-V5	515F	GTGCCAGCMGCCGCGG
		907R	CCGTCAATTCMTTTRAGT
真菌	ITS1	its1	CTTGGTCATTTAGAGGAAGTAA
		its2	GCTGCGTTCTTCATCGATGC
	ITS2	its3	GCATCGATGAAGAACGCAGC
		its4	TCCTCCGCTTATTGATATGC

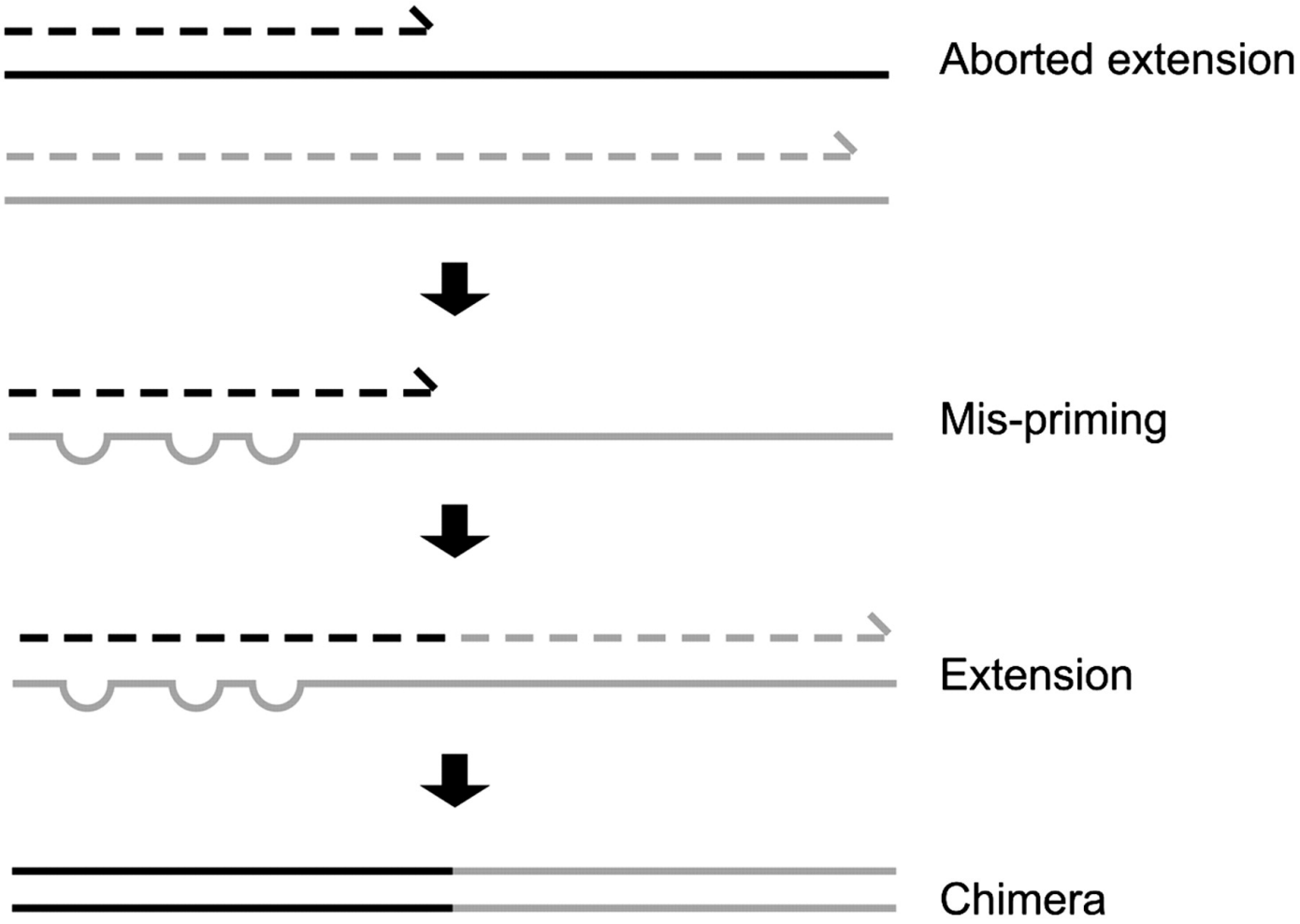
	扩增区域	引物名称	引物对
细菌	V4	515F	GTGCCAGCMGCCGCGGTAA
		806R	GGACTACHVGGGTWTCTAAT
	V3-V4	338F	ACTCCTACGGGAGGCAGCAG
		806R	GGACTACHVGGGTWTCTAAT
		27F	AGRGTTYGATYMTGGCTCAG



	全长	1492R	RGYTACCTTGTTACGACTT
真菌	ITS1	its1	CTTGGTCATTTAGAGGAAGTAA
		its2	GCTGCGTTCTTCATCGATGC
	ITS2	its3	GCATCGATGAAGAACGCAGC
		its4	TCCTCCGCTTATTGATATGC

- chimera(嵌合体)

参考文献



3. 搭建分析环境

3.1 配置一个舒适的zshell终端

- 安装zshell

```
sudo apt install zsh
```

- 更改用户的shell程序

```
chsh -s /bin/zsh
```

- 安装oh-my-zsh：官网

```
sh -c "$(wget https://raw.githubusercontent.com/ohmyzsh/ohmyzsh/master/tools/install.sh -O -)"
```

- 更改主题

```
# 修改配置文件
vim ~/.zshrc
# 修改主题名称
ZSH_THEME="ys"
```

- 添加自动提示插件

```
# 把下载好的插件移动到插件目录
mv zsh-autosuggestions/ ~/.oh-my-zsh/custom/plugins/
# 修改插件配置
vim ~/.zshrc
plugins=(git last-working-dir zsh-autosuggestions)
```

- 实用zshell命令

```
# 批量重命名

autoload -U zmv
zmv 'G([123]A).fq.gz' 'G-7D-$1.fq.gz'
zmv 'G(*).(*).gz' 'G-$1.$2kk.gz'

# 递归搜索

## 查找所有普通文件
print -l **/*(.)
## 查找所有后缀是gz结尾的文件
print -l **/*.gz
## 列出最近一天修改过内容的文件
print -l *(.m-1)
## 列出最近一个月没有读取过的文件
print -l *(.aM+1)
## 列出当前目录下小于 2k 的文件
print -l *(.Lk-2)

# 字符串截取

aa="1234"
## 取第一个
echo $aa[1]
## 取最后一个
echo $aa[-1]
## 取中间两个
echo $aa[2,3]

# 字符串拼接

% str1=abc
% str2=def

% str2+=$str1
```

```
% echo $str2
defabc

# 字符串替换

% str=abcabc

## 只替换找到的第一个
% echo ${str/bc/ef}
aefabc

# for循环
for i (*){
echo $i
}
# while循环
while {read i} {
echo $i
} < Flavo_accession.txt
```

3.2 安装mamba和sankemake

- Miniforge的github地址：Miniforge

```
# 下载安装脚本到本地
wget https://github.com/conda-forge/miniforge/releases/latest/download/Miniforge3-Linux-x86_64.sh
# 添加执行权限
chmod +x Miniforge3-Linux-x86_64.sh
# 运行脚本
./Miniforge3-Linux-x86_64.sh
# 更新
source ~/.zshrc
```

- snakemake

```
# 单独创建一个环境
mamba create -n snakemake -c bioconda snakemake graphviz
# 激活环境
mamba activate snakemake
```

3.3 扩增子相关软件安装

- 新建目录

```
mkdir 16S_rRNA_amplicon
cd 16S_rRNA_amplicon
```

- 创建环境依赖清单文件`env.yaml`

```yaml
channels:
  - conda-forge
  - bioconda
dependencies:
  - bioconductor-shortread
  - bioconductor-dada2
  - bioconductor-phyloseq
  - r-pheatmap
  - r-ggplot2
  - r-dplyr
  - r-phangorn
  - flash
  - cutadapt
  - mafft
  - fasttree
  - biopython
  - pandas
```

- 创建环境并且安装上述依赖

```
mamba env create -n 16S_rRNA_amplicon --file env.yaml
# 激活环境
mamba activate 16S_rRNA_amplicon
```

- 安装nodejs

[nodejs官网]

```
# 下载
wget https://nodejs.org/dist/v20.11.0/node-v20.11.0-linux-x64.tar.xz
# 解压
tar -xf node-v20.11.0-linux-x64.tar.xz
# 添加到环境变量
vim ~/.zshrc
export PATH=$PATH:/home/xjm/software/node-v20.11.0-linux-x64/bin

# 更新
source ~/.zshrc
```

- 安装puppeteer

[github]

```
# 创建一个目录
mkdir puppeteer
cd puppeteer
# 初始化工程
npm init
```

```
📌 🐶 # xjm @ lmit-X12DPi-N-T-6 in /userdata2/cc/TaiwanStrait-16S/spring_custom_pipeline/puppeteer [16:43:45]
👉 cat package.json
{
  "name": "puppeteer",
  "version": "1.0.0",
  "description": "",
  "main": "app.js",
  "scripts": {
    "test": "echo \"Error: no test specified\" && exit 1"
  },
  "author": "",
  "license": "ISC"
}

# 更改module模式

vim package.json

📌 🐶 # xjm @ lmit-X12DPi-N-T-6 in /userdata2/cc/TaiwanStrait-16S/spring_custom_pipeline/puppeteer [16:45:44]
👉 cat package.json
{
  "name": "puppeteer",
  "version": "1.0.0",
  "description": "",
  "type": "module",
  "main": "app.js",
  "scripts": {
    "test": "echo \"Error: no test specified\" && exit 1"
  },
  "author": "",
  "license": "ISC"
}

# 安装
npm i puppeteer
# 或者
pnpm i puppeteer
```

pnpm应该类似mamba之于conda

- 复制`app.js`

```
mv app.js puppeteer/
```

4. 分析流程拆解

4.1 去引物与裁剪低质量序列

准备示例数据

- 解压

```
tar -xzf example_data.tar.gz
```

- 目录结构：3个环境水体的采样站点，每个站点有3个平行，raw data

```
# example_data
├── S1-1
│   ├── S1-1_1.fq.gz
│   └── S1-1_2.fq.gz
├── S1-2
│   ├── S1-2_1.fq.gz
│   └── S1-2_2.fq.gz
├── S1-3
│   ├── S1-3_1.fq.gz
│   └── S1-3_2.fq.gz
├── S2-1
│   ├── S2-1_1.fq.gz
│   └── S2-1_2.fq.gz
├── S2-2
│   ├── S2-2_1.fq.gz
│   └── S2-2_2.fq.gz
├── S2-3
│   ├── S2-3_1.fq.gz
│   └── S2-3_2.fq.gz
├── S3-1
│   ├── S3-1_1.fq.gz
│   └── S3-1_2.fq.gz
├── S3-2
│   ├── S3-2_1.fq.gz
│   └── S3-2_2.fq.gz
└── S3-3
    ├── S3-3_1.fq.gz
    └── S3-3_2.fq.gz
```

- 测序区域为V3～V4区

`F_primer:` ACTCCTACGGGAGGCAGCAG

`R_primer:` GGACTACHVGGGTWTCTAAT

cutadapt

官方文档

```
# -g 5' adapter
# -G 5' adapter to be removed from R2
# -e 最大允许的错频率（0 <= E < 1）或者错配个数（E >= 1），因为你给定的引物对并不是百分百匹配的
# --discard-untrimmed 去掉那些没有检测到adapter的序列 这些序列往往是错误序列 测序错误或者是PCR扩增错误
# -q 裁剪3'端低质量碱基，没有裁剪会导致后续拼接overlap区域的错配偏高
# --cores 指定线程
cutadapt -q 20 --discard-untrimmed -e 4 --cores 32 -g ACTCCTACGGGAGGCAGCAG -G GGACTACHVGGGTWTCTAAT -o cutadapt_output/S1-1/S1-1_1.fq.gz -p cutadapt_output/S1-1/S1-1_2.fq.gz example_data/S1-1/S1-1_1.fq.gz example_data/S1-1/S1-1_2.fq.gz
```

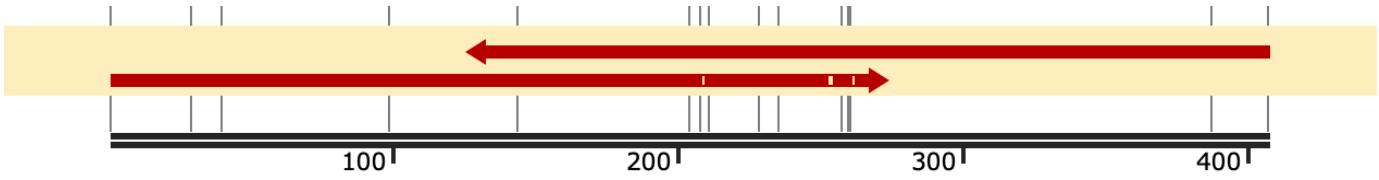
- Degenerate bases

IUPAC nucleotide code	Base
R	A or G
Y	C or T
S	G or C
W	A or T
K	G or T
M	A or C
B	C or G or T
D	A or G or T
H	A or C or T
V	A or C or G
N	any base

4.2 双端序列拼接

flash

官网



特别注意这个参数：-M 最大重叠长度，如果重叠区大于这个长度也会正常拼接，但是下面的错配比率的分母还是按照这个值去算而不是整个重叠区长度，会导致拼接失败的比例升高，非常鸡肋，你自己自动识别不就行 [default: 65]

```
# -m 两个reads之间可靠重叠所需的最小重叠长度 [default: 10]
# -M 最大重叠长度，如果重叠区大于这个长度也会正常拼接，但是下面的错配比率的分母还是按照这个值去算而不是整个重叠区长度，会导致拼接失败的比例升高 [default: 65]
# -x 两个读段的错配碱基数与重叠长度之间的最大允许比率 [default: 0.25]
# -o 输出结果前缀
# -d 输出目录
# -t 线程，如果想按输入文件的顺序输出就把线程设置为1
# -p 质量分数偏移值 [Default: 33]
flash -m 15 -M 250 -x 0.1 -o S1-1 -d flash_output/S1-1 -t 16 cutadapt_output/S1-1/S1-1_1.fq.gz cutadapt_output/S1-1/S1-1_2.fq.gz
```

4.3 dada2及后续流程分析

4.3.1 RStudio安装相关R包

```
.cran_packages <- c("dplyr", "phangorn")
.bioc_packages <- c("dada2", "phyloseq", "ShortRead")
if(!requireNamespace("BiocManager")){
  install.packages("BiocManager")
}

.inst <- .cran_packages %in% installed.packages()
if(any(!.inst)) {
   install.packages(.cran_packages[!.inst])
}
.inst <- .bioc_packages %in% installed.packages()
if(any(!.inst)) {
   BiocManager::install(.bioc_packages[!.inst])
}
```

4.3.2 生成去嵌合体的特征表与代表序列

- ASV (Amplicon Sequence Variants)
- 嵌合体

4.3.3 物种注释、去叶绿体和线粒体序列

> ## 自定义数据库格式

`assignTaxonomy(...)` 需要一个训练fasta文件（或压缩的fasta文件），其中每个序列对应的分类信息被编码在id行中，格式如下（第二个序列没有被分配到第6级）：

```
>Level1;Level2;Level3;Level4;Level5;Level6;
ACCTAGAAAGTCGTAGATCGAAGTTGAAGCATCGCCCGATGATCGTCTGAAGCTGTAGCATGAGTCGATTTTCACATTCAGGGATACCATAGGATAC
>Level1;Level2;Level3;Level4;Level5;
CGCTAGAAAGTCGTAGAAGGCTCGGAGGTTTGAAGCATCGCCCGATGGGATCTCGTTGCTGTAGCATGAGTACGGACATTCAGGGATCATAGGATAC
```

`assignSpecies(...)`和`addSpecies(...)`期望训练数据以 fasta 文件的形式提供（或压缩的 fasta 文件），其中 id 行的格式如下：

```
>ID Genus species
ACCTAGAAAGTCGTAGATCGAAGTTGAAGCATCGCCCGATGATCGTCTGAAGCTGTAGCATGAGTCGATTTTCACATTCAGGGATACCATAGGATAC
>ID Genus species
CGCTAGAAAGTCGTAGAAGGCTCGGAGGTTTGAAGCATCGCCCGATGGGATCTCGTTGCTGTAGCATGAGTACGGACATTCAGGGATCATAGGATAC
```

4.3.4 alpha多样性计算

1. 丰富度: **Richness**
   衡量一个生态系统有多少不同的物种
2. 均匀度: **Evenness**
   衡量生态系统中，不同物种之间数量的差异度

- Shannon
- Simpson's Index
- ACE和Chao1

```
# 辛普森多样性指数是基于在一个无限大的群落中，随机抽取两个个体，它们属于同一物种的概率
# Simpson's Index: D
# 这里的Simpson 算的应该是Simpson's Index of Diversity: 1-D
# "Observed", "Chao1", "ACE"基本值都是一样的，因为样本里面数目为1或者2的ASV基本没有
```

4.3.5 构建ASV系统发育树

- `mafft`多序列比对

```
mkdir mafft_output
mafft --auto dada2_single_end_output/ASV_seqs_no_chloro_mito.fasta > mafft_output/ASV_seqs_no_chloro_mito_align.fasta
```

- `FastTree`最大似然法构建进化树

```
mkdir fasttree_output
FastTree -gtr -nt mafft_output/ASV_seqs_no_chloro_mito_align.fasta > fasttree_output/ASV_seqs_no_chloro_mito_tree.nwk
```

4.3.6 beta多样性计算

Beta多样性是生物多样性研究中的一个重要概念，用于衡量不同生态系统或样本之间的物种多样性差异程度。它通常通过计算物种组成的差异性来衡量，可以帮助我们了解不同地点或环境中物种的多样性情况。

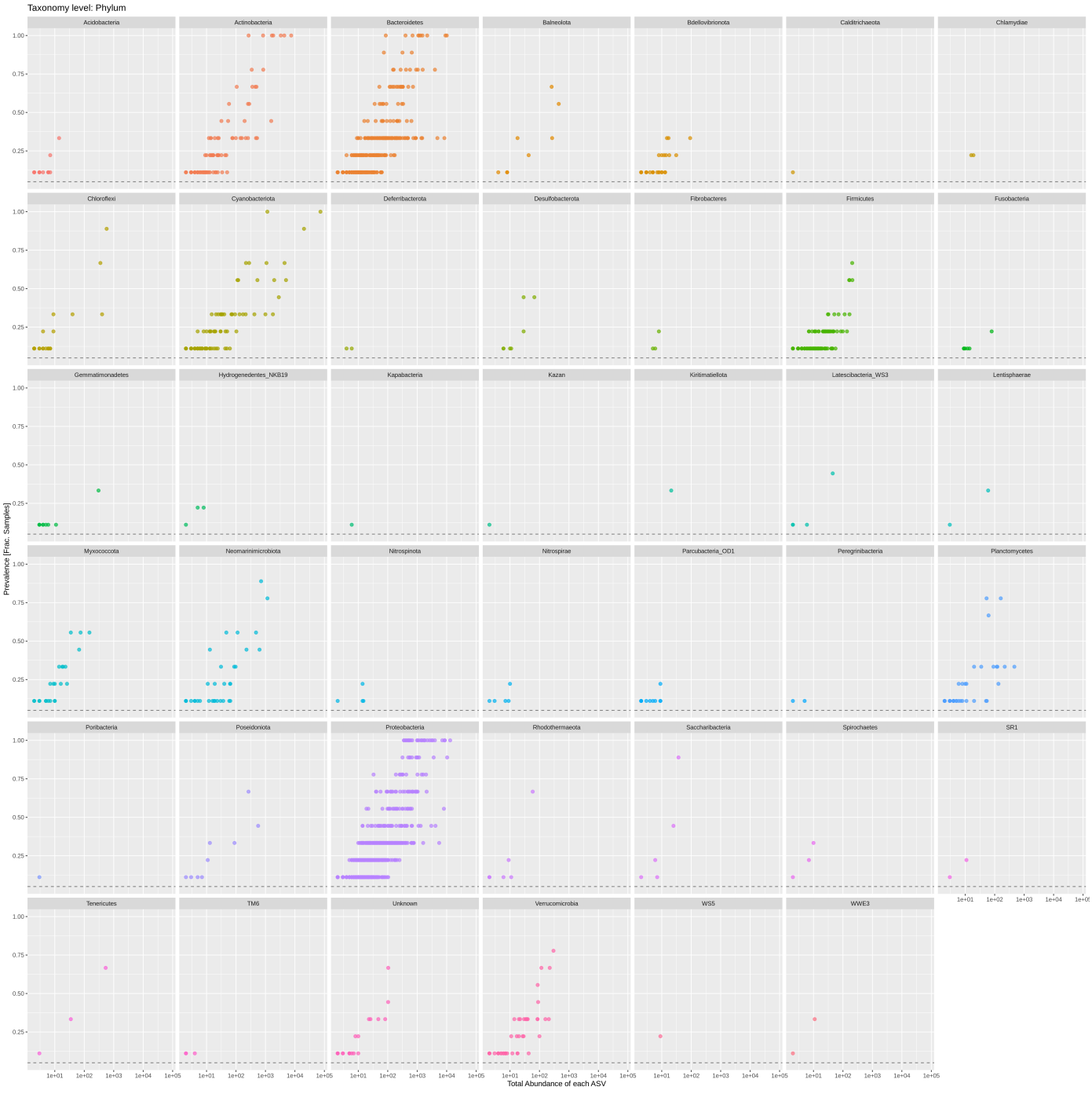
4.3.7 ASV流行度和总丰度计算

- 流行度: **Prevalence**

一个物种出现在越多的样品中，这个物种的流行度越高。

- 总丰度: **Total Abundance**

一个物种在其所有出现的样品中的丰度总和。



5. Snakemake分析流程自动化

5.1 准备基础工作目录和修改配置文件

- 复制基础工作目录

```
# 解压
tar -xzf custom_pipeline.tar.gz
```

```
# 整个工作目录结构
custom_pipeline
├── configs
│   └── 16S_rRNA_amplicon_config.yaml
├── taxonomy_databases
│   ├── silva_nr99_v138.1_train_Mitochondria_Chloroplast_set.fa
│   ├── silva_nr99_v138.1_train_set.fa.gz
│   └── silva_species_assignment_v138.1.fa.gz
└── workflow
    ├── 16S_rRNA_amplicon_EZ.smk
    ├── 16S_rRNA_amplicon_Silva.smk
    └── scripts
        ├── python
        │   └── ezbiocloud.py
        └── R
            ├── dada2
            │   ├── assign_taxonomy.R
            │   ├── dada2_single_reads_multi_samples.R
            │   └── downstream_analysis.R
            └── reads_length_frequency_distribution.R
```

- 修改`configs/16S_rRNA_amplicon_config.yaml`

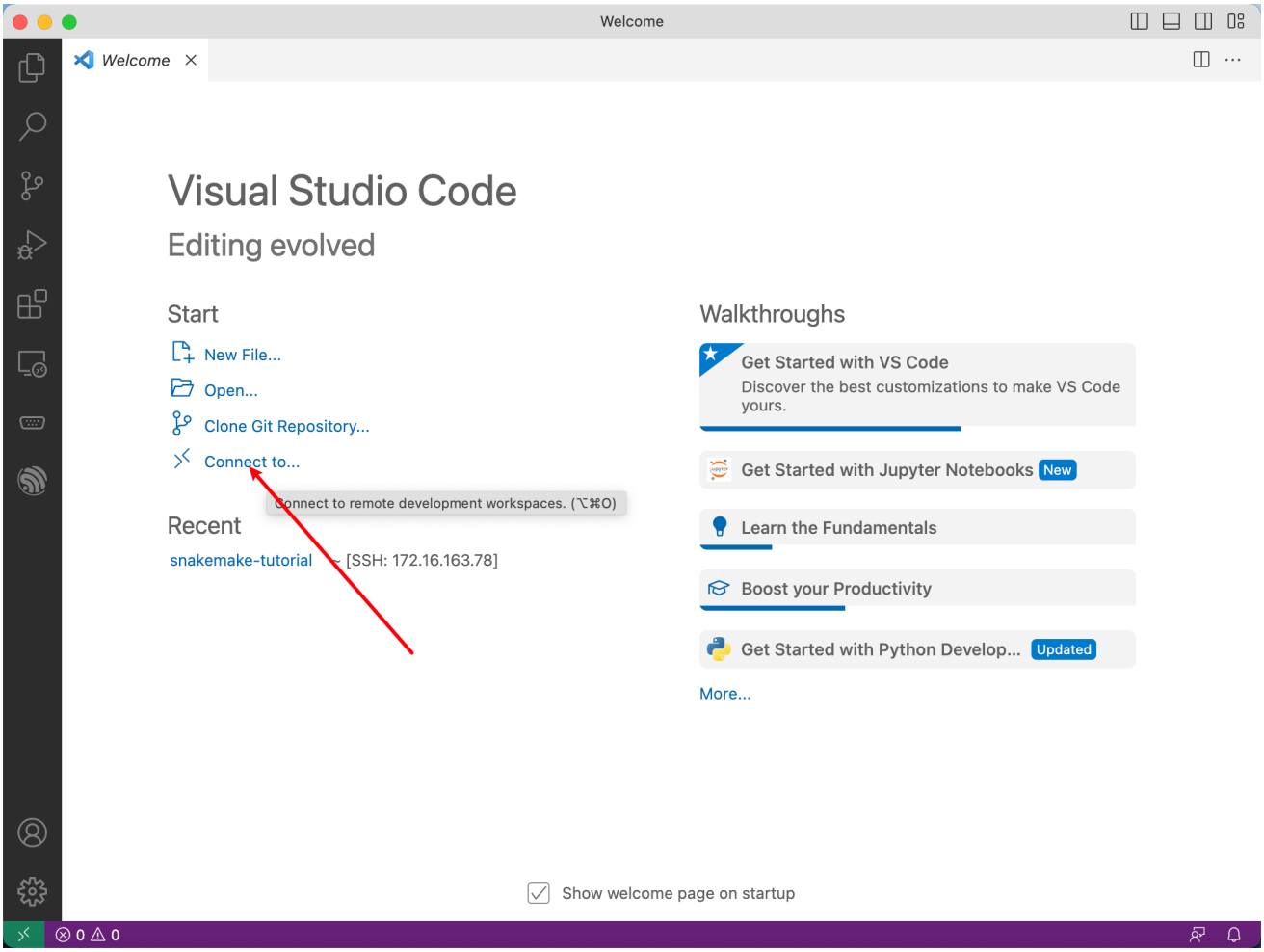
```
samples_dir: "samples_XJM"
f_suffix: "_1.fq.gz"
r_suffix: "_2.fq.gz"
env_name: "16S_rRNA_amplicon"
f_primer: "GTGCCAGCMGCCGCGG"
r_primer: "CCGTCAATTCMTTTRAGT"
phred_offset: 33 # 只有很早期是数据是64，现在都是33
minFoldParentOverAbundance: 1.5 # 嵌合体父母本丰度倍数
max_error_num: 4 # for cutadapt
error_rate_learning_nbases: 500000000 # 错误率学习的采样数量，会影响降噪准确度，1e8 大概能采样70000reads X 4样品
is_raw_data: false # 是否是raw data
taxon_mito_chloro:
"taxonomy_databases/silva_nr99_v138.1_train_Mitochondria_Chloroplast_set.fa"
db_taxon_file: "taxonomy_database/silva_nr99_v138.1_train_set.fa.gz"
db_spec_file: "taxonomy_database/silva_species_assignment_v138.1.fa.gz"
puppeteer_js: "puppeteer/app.js"
ez_account:
  username: "xxxxx@xxxx"
  password: "xxxxxxxx"
```

5.2 配置VS Code

软件下载

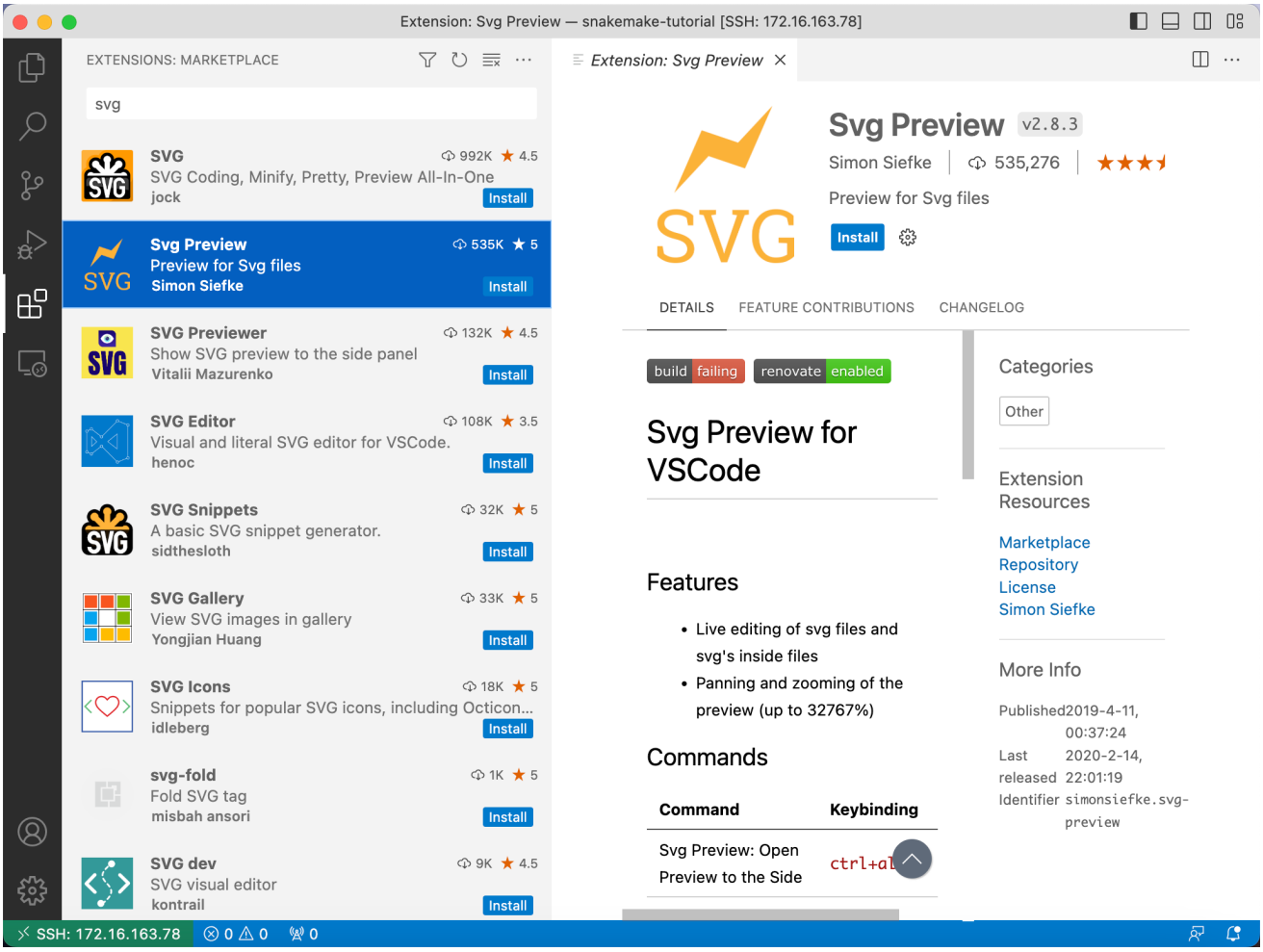
- VS Code 官方网站：链接

连接远程服务器

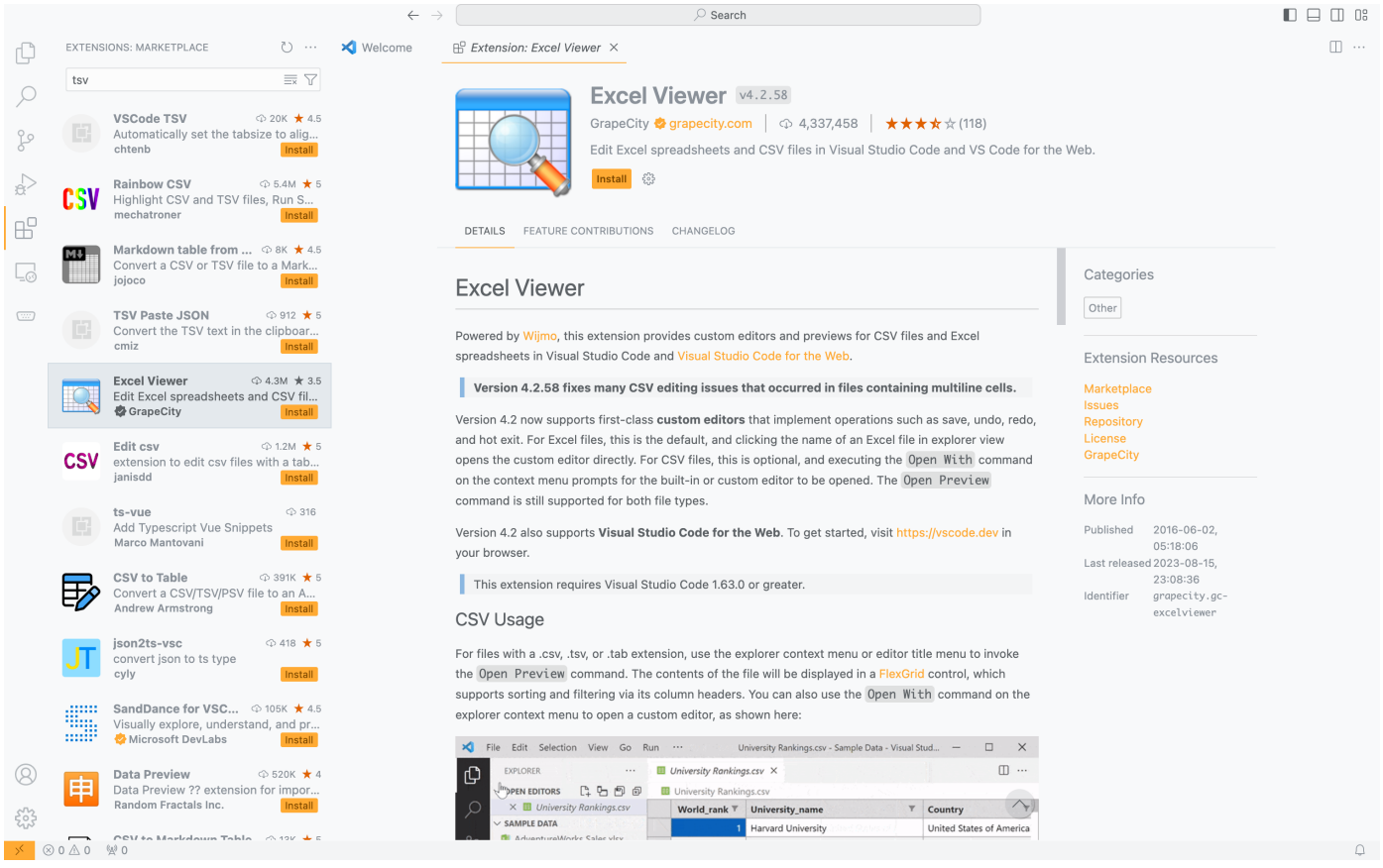


安装插件

- SVG查看器


Extension: Svg Preview — snakemake-tutorial [SSH: 172.16.163.78]
EXTENSIONS: MARKETPLACE
svg
SVG
SVG Coding, Minify, Pretty, Preview All-In-One
jock
Install
Svg Preview
Preview for Svg files
Simon Siefke
Install
SVG Previewer
Show SVG preview to the side panel
Vitalii Mazurenko
Install
SVG Editor
Visual and literal SVG editor for VSCode.
henoc
Install
SVG Snippets
A basic SVG snippet generator.
sidthesloth
Install
SVG Gallery
View SVG images in gallery
Yongjian Huang
Install
SVG Icons
Snippets for popular SVG icons, including Octicon...
idleberg
Install
svg-fold
Fold SVG tag
misbah ansori
Install
SVG dev
SVG visual editor
kontrail
Install
Extension: Svg Preview
Svg Preview v2.8.3
Simon Siefke 535,276
Preview for Svg files
Install
DETAILS
FEATURE CONTRIBUTIONS
CHANGELOG
build failing
renovate enabled
Svg Preview for VSCode
Features
Live editing of svg files and svg's inside files
Panning and zooming of the preview (up to 32767%)
Commands
Command
Keybinding
Svg Preview: Open Preview to the Side
ctrl+al
Categories
Other
Extension Resources
Marketplace
Repository
License
Simon Siefke
More Info
Published 2019-4-11, 00:37:24
Last released 2020-2-14, 22:01:19
Identifier simonsiefke.svg-preview
SSH: 172.16.163.78

- 表格查看器


EXTENSIONS: MARKETPLACE
tsv
Welcome
Extension: Excel Viewer
Excel Viewer v4.2.58
GrapeCity grapecity.com 4,337,458 (118)
Edit Excel spreadsheets and CSV files in Visual Studio Code and VS Code for the Web.
Install
DETAILS
FEATURE CONTRIBUTIONS
CHANGELOG
Excel Viewer
Powered by Wijmo, this extension provides custom editors and previews for CSV files and Excel spreadsheets in Visual Studio Code and Visual Studio Code for the Web.
Version 4.2.58 fixes many CSV editing issues that occurred in files containing multiline cells.
Version 4.2 now supports first-class custom editors that implement operations such as save, undo, redo, and hot exit. For Excel files, this is the default, and clicking the name of an Excel file in explorer view opens the custom editor directly. For CSV files, this is optional, and executing the Open With command on the context menu prompts for the built-in or custom editor to be opened. The Open Preview command is still supported for both file types.
Version 4.2 also supports Visual Studio Code for the Web. To get started, visit https://vscode.dev in your browser.
This extension requires Visual Studio Code 1.63.0 or greater.
CSV Usage
For files with a .csv, .tsv, or .tab extension, use the explorer context menu or editor title menu to invoke the Open Preview command. The contents of the file will be displayed in a FlexGrid control, which supports sorting and filtering via its column headers. You can also use the Open With command on the explorer context menu to open a custom editor, as shown here:
Categories
Other
Extension Resources
Marketplace
Issues
Repository
License
GrapeCity
More Info
Published 2016-06-02, 05:18:06
Last released 2023-08-15, 23:08:36
Identifier grapecity.gc-excelviewer

5.3 运行snakemake

- 激活环境

```
mamba activate snakemake
```

- 流程图

```
# Silva流程
snakemake -s workflow/16S_rRNA_amplicon_Silva.smk --dag | dot -Tsvg > workflow_Silva.svg
# EZ流程
snakemake -s workflow/16S_rRNA_amplicon_EZ.smk --dag | dot -Tsvg > workflow_EZ.svg
```

- 运行Silva流程

```
# 试运行
# -c 线程数量
# --use-conda 使用conda环境
snakemake -s workflow/16S_rRNA_amplicon_Silva.smk -np -c all --use-conda
# 真运行
snakemake -s workflow/16S_rRNA_amplicon_Silva.smk -p -c all --use-conda
```

- 运行EZ流程

> 优点：
> 1数据库更全，能注释到更多种水平的ASV
> 2各个分类水平的单词是最新的命名单词

```
# 试运行
snakemake -s workflow/16S_rRNA_amplicon_EZ.smk -np -c all --use-conda
# 真运行
snakemake -s workflow/16S_rRNA_amplicon_EZ.smk -p -c all --use-conda
```

5.4 结果文件说明

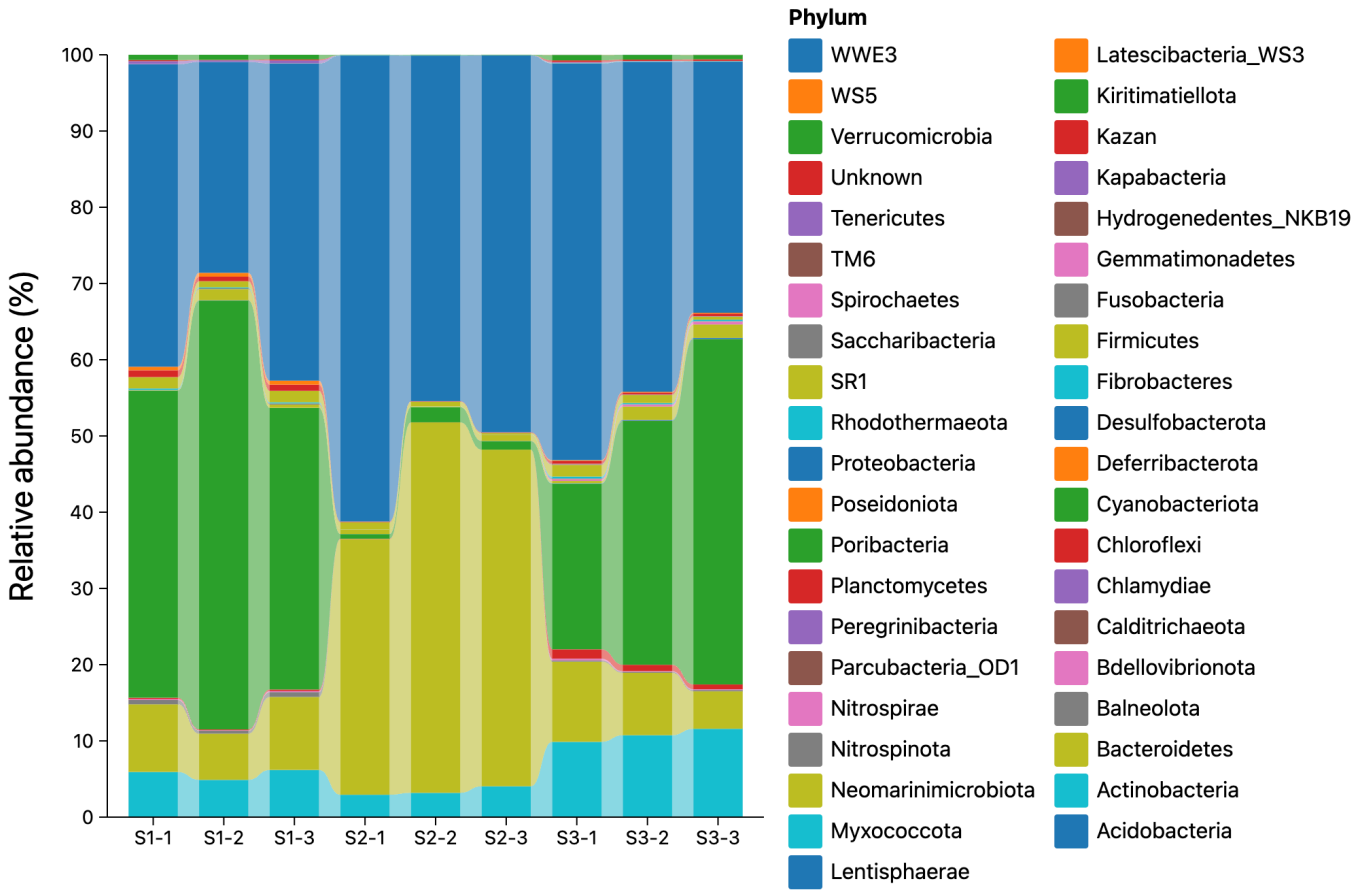
```
# benchmarks
└── dada2_single_end_reads_benchmark.tsv # 计算资源使用情况
# dada2_single_end_output
├── seqs
│   ├── ASV_seqs_nochim.fasta # 去掉嵌合体后的ASV序列
│   ├── ASV_seqs_no_chloro_mito.fasta # 去掉嵌合体、叶绿体和线粒体后的ASV序列
│   ├── ASV_seqs_with_chloro_mito.fasta # 叶绿体和线粒体的ASV序列
│   └── filtered_reads  [73 entries exceeds filelimit, not opening dir]
└── tables
    ├── ASV_taxon_no_chloro_mito.tsv # 去掉嵌合体、叶绿体和线粒体后的ASV物种分类
```

```
    ├── ASV_taxon_with_chloro_mito.tsv # 叶绿体和线粒体ASV物种分类
    ├── ASV_prevalence_totalAbundance.tsv # ASV流行度和总丰度表
    ├── seq_table_no_chloro_mito.biom # 用于导入qiime2的特征表
    ├── seq_table_no_chloro_mito.tsv # 去掉嵌合体、叶绿体和线粒体后的特征表
    ├── seq_table_prop_no_chloro_mito.tsv # 去掉嵌合体、叶绿体和线粒体后的相对丰度特征表，可用于后续的beta多样性分析
    └── seq_table_with_chloro_mito.tsv # 叶绿体和线粒体序列的特征表
# summary_output
├── beta_diversity # 不同降维和距离算法的beta多样性
│   ├── ord_nmds_bray_stress.tsv
│   ├── ord_nmds_bray.tsv
│   ├── ord_nmds_unifrac_stress.tsv
│   ├── ord_nmds_unifrac.tsv
│   ├── ord_nmds_wunifrac_stress.tsv
│   ├── ord_nmds_wunifrac.tsv
│   ├── ord_pcoa_bray_explained_variance_ratio.tsv
│   ├── ord_pcoa_bray.tsv
│   ├── ord_pcoa_unifrac_explained_variance_ratio.tsv
│   ├── ord_pcoa_unifrac.tsv
│   ├── ord_pcoa_wunifrac_explained_variance_ratio.tsv
│   └── ord_pcoa_wunifrac.tsv
├── cutadapter
│   ├── R1_length_frequency_distribution_after_cutting.tsv # R1 裁剪引物后的长度分布矩阵
│   ├── R1_length_frequency_distribution_before_cutting.tsv # R1 裁剪引物前的长度分布矩阵
│   ├── R2_length_frequency_distribution_after_cutting.tsv # R2 裁剪引物后的长度分布矩阵
│   └── R2_length_frequency_distribution_before_cutting.tsv # R2 裁剪引物前的长度分布矩阵
├── dada2
│   ├── assign_taxonomy_stats.tsv # 每一步处理所剩序列数量表 + 最终去除叶绿体和线粒体的所剩数量
│   └── denoising_stats.tsv # 每一步处理所剩序列数量表
├── alpha_diversity
│   └── alpha_diversity.tsv # 多种alpha多样性指标值表
└── stacked_bar_plot # 不同分类水平绘制堆叠柱状图的数据
    ├── Class.tsv
    ├── Family.tsv
    ├── Genus.tsv
    ├── Kingdom.tsv
    ├── Order.tsv
    ├── Phylum.tsv
    └── Species.tsv
# figure_output
├── alpha_diversity # 多种alpha多样性图
│   ├── ACE.png
│   ├── Chao1.png
│   ├── InvSimpson.png
│   ├── Observed.png
│   ├── Shannon.png
│   └── Simpson.png
├── cutadapter # 引物裁剪前后长度分布热图
│   ├── R1_length_frequency_distribution_after_cutting.png
│   ├── R1_length_frequency_distribution_before_cutting.png
│   ├── R2_length_frequency_distribution_after_cutting.png
```

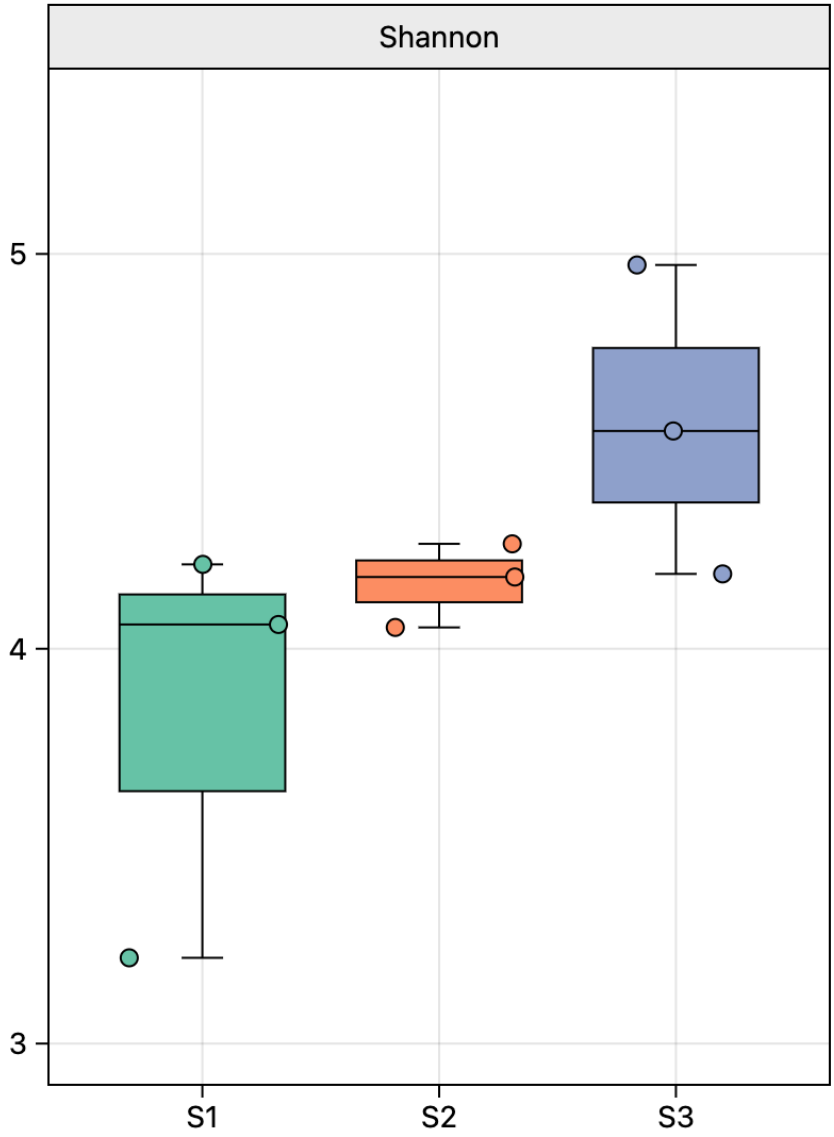
```
61  │   └── R2_length_frequency_distribution_before_cutting.png
62  └── prevalence # 不同分类水平的ASV流行度与总丰度的散点图
63      ├── Class.png
64      ├── Family.png
65      ├── Genus.png
66      ├── Kingdom.png
67      ├── Order.png
68      └── Phylum.png
```

6. 下游数据分析与可视化

6.1 绘制物种组成丰度堆叠柱状图



6.2 alpha多样性可视化



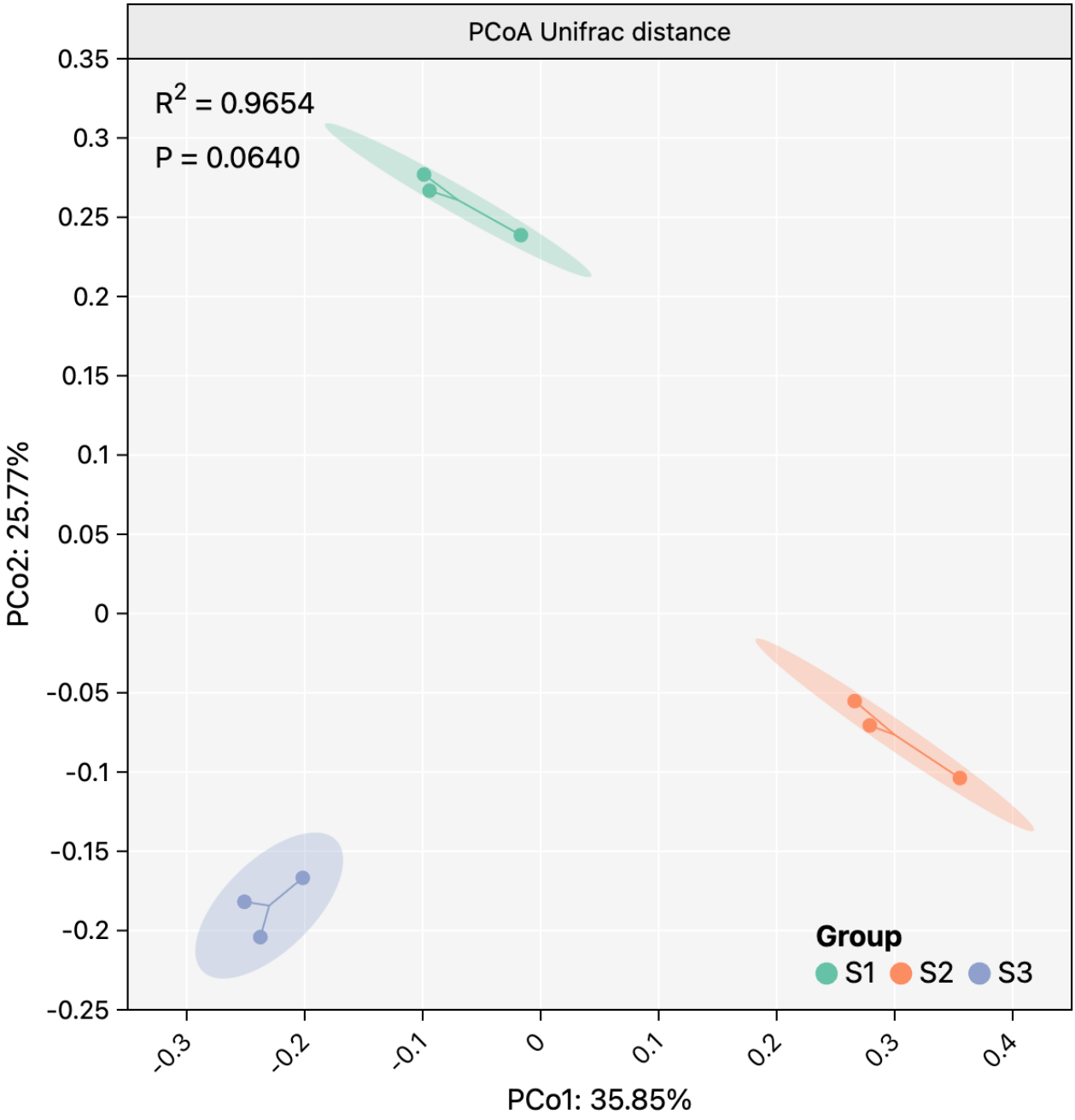
6.3 beta多样性可视化

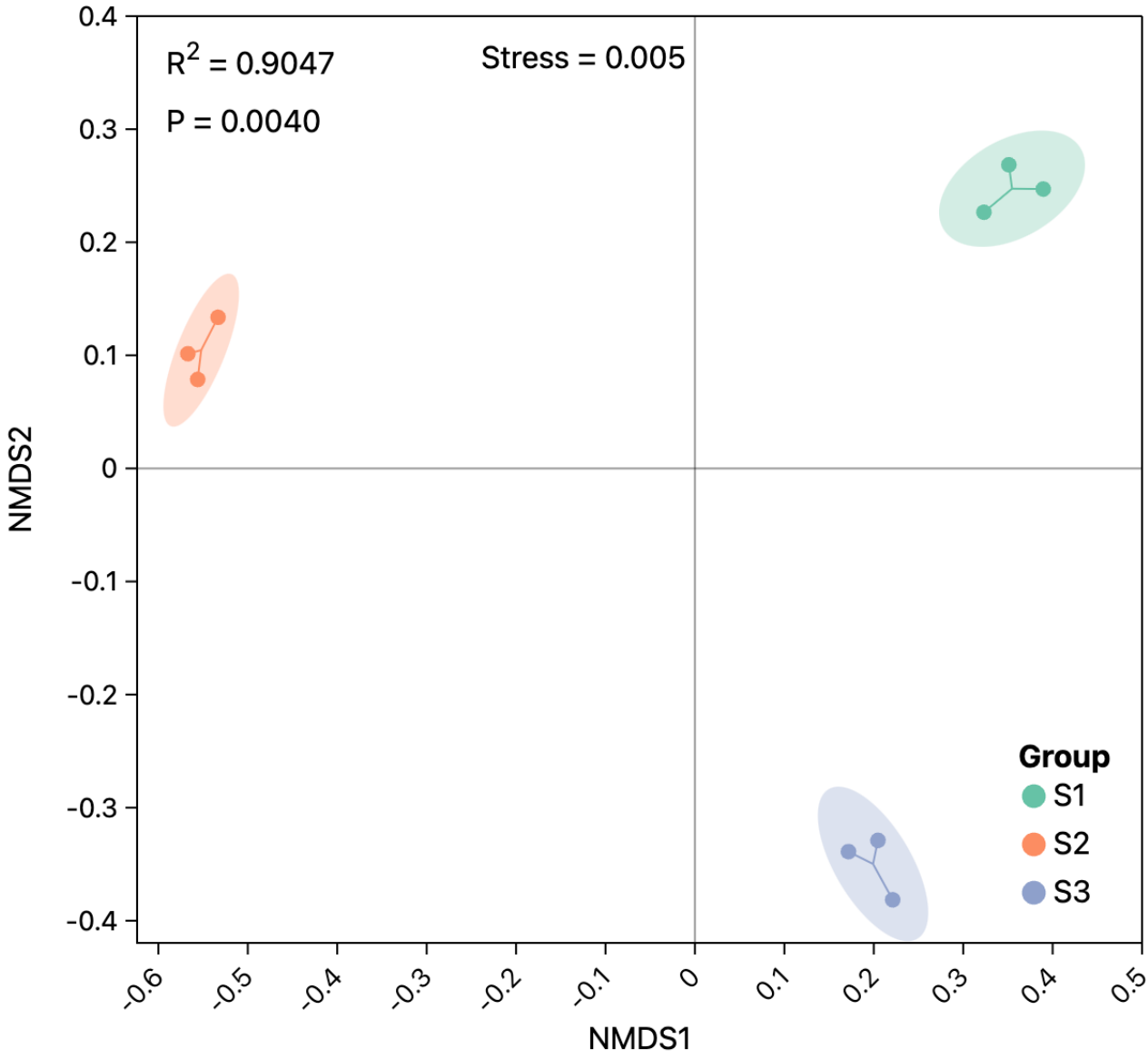
- **PERMANOVA** (Permutational Multivariate Analysis of Variance)

PERMANOVA是一种用于比较多组间差异的统计方法，其中R^2表示解释变量对响应变量的方差解释程度。在PERMANOVA中，R^2值越接近1表示解释变量对样本间差异的解释程度越高，即样本间的差异可以较好地由解释变量来解释。这个值可以帮助我们理解解释变量对于样本结构的影响程度。通常情况下，R^2值越高，说明解释变量对于样本间差异的解释能力越强。

- **NMDS** (Non-metric multidimensional scaling)

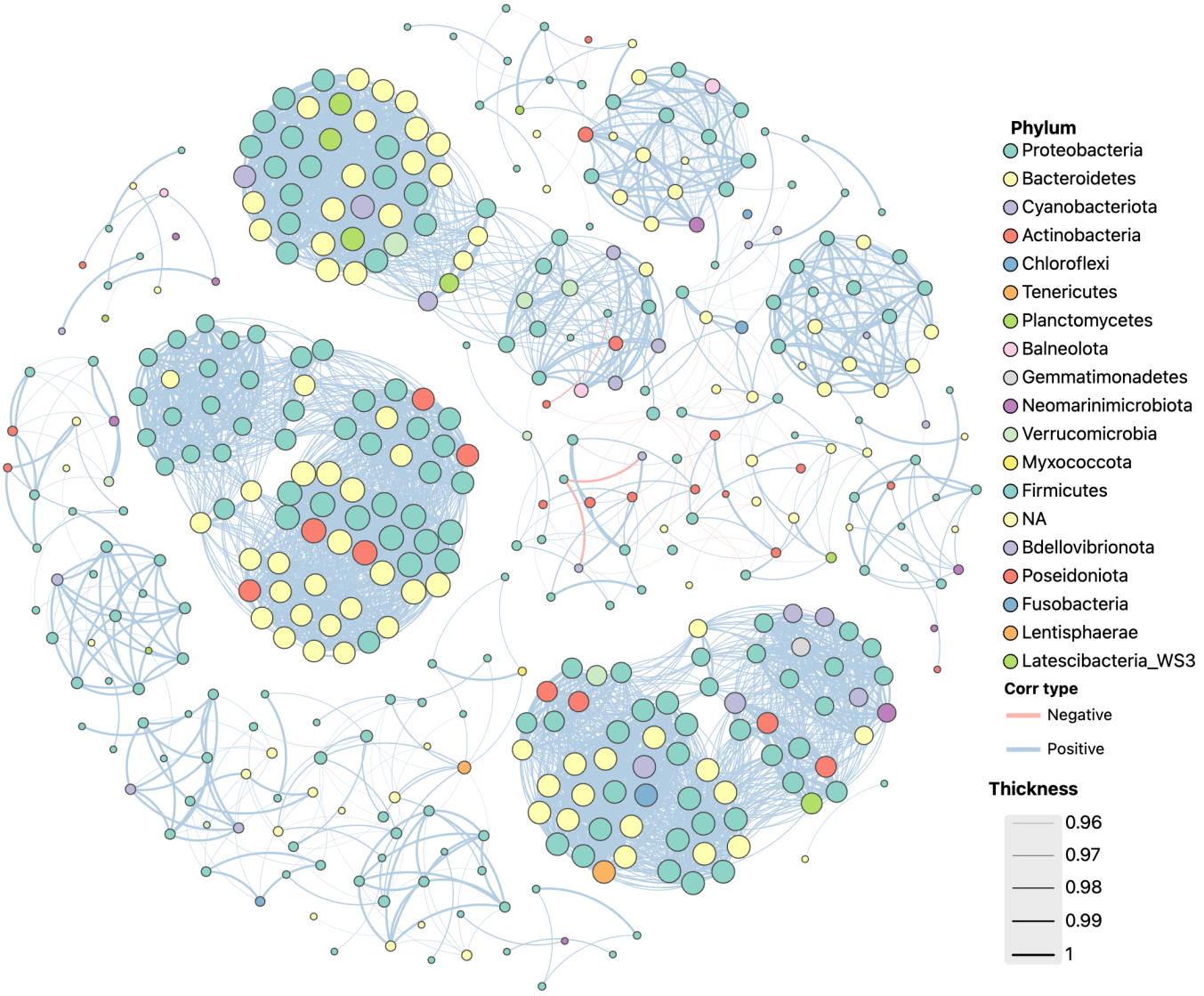
非度量多维尺度分析（NMDS）中的“stress”是一种用来衡量数据在低维空间中的拟合程度的指标。它表示数据在降维过程中失真的程度，即原始数据点之间的距离在降维后是否得到了很好的保留。Stress值越接近0，表示降维后的结果越能够准确地反映原始数据点之间的距离关系。通常来说，Stress值在0-0.2之间被认为是一个比较好的拟合度量。在NMDS分析中，通过调整数据点在低维空间中的位置，以最小化stress值来找到最佳的降维结果。


PCoA Unifrac distance
R^2 = 0.9654
P = 0.0640
0.35
0.3
0.25
0.2
0.15
0.1
0.05
0
-0.05
-0.1
-0.15
-0.2
-0.25
PCo2: 25.77%
-0.3
-0.2
-0.1
0
0.1
0.2
0.3
0.4
PCo1: 35.85%
Group
S1
S2
S3



6.4 微生物共现网络分析

Microbial Co-occurrence Network



6.4.1 常见的微生物共现网络指标

（1）**节点数 (Number of nodes)**：网络中的节点个数

（2）**边数 (Number of edges)**：网络中所有连接微生物的边的数量

（3）**节点度（Node Degree）**：节点的度是指与该节点直接相连的边的数量。在微生物共现网络中，节点通常代表微生物物种。节点度表示了一个微生物与多少个其他微生物有共现关系，高度连接的微生物可能在微生物群落中起着重要的作用。

（4）**平均度（Average Degree）**是一个节点度（Node Degree）的统计性指标，用于描述微生物网络中每个节点（通常代表微生物物种）平均连接到多少个其他节点。更具体地说，平均度是通过计算网络中所有节点的度数（与其他节点相连的边的数量）并取平均值得到的。这个指标提供了关于微生物共现网络的整体连接密度的信息。平均度的值越高，表示微生物物种之间的相互关系更为密集；而平均度的值越低，则表示微生物物种之间的相互关系相对稀疏。

（5）**网络密度（Network Density）**：网络密度衡量了网络中实际存在的边数量与可能存在的最大边数量之间的比例。高密度网络表示微生物之间的相互作用较为密集，低密度网络则反映了微生物之间的相互作用较为稀疏。

（6）**平均最短路径长度（Average Shortest Path Length）**：这个指标衡量了网络中任意两个节点之间的平均最短路径长度。较短的平均路径长度可能表示微生物群落内的信息传递较为迅速。

（7）**网络直径 (Network diameter)**：网络中任意两个节点之间最短路径长度的最大值，即网络中最远的微生物间的距离

（8）**聚类系数 (Clustering coefficient)**：用于描述网络中节点的聚集程度或者说节点之间的紧密程度。在微生物共现网络中，聚类系数是用来衡量微生物物种之间共现关系的密度，即它们是否倾向于与彼此共存形成子图（聚类）。

（9）**网络中心性指标（Centrality Measures）**：包括介数中心性（Betweenness Centrality）、接近度中心性（Closeness Centrality）和特征向量中心性（Eigenvector Centrality）等。这些指标用于识别在网络中具有重要影响力的微生物物种，可能在信息传递或控制微生物群落结构方面发挥关键作用。

（10）**社区结构（Community Structure）**：社区检测算法用于将微生物网络分割成不同的社区或模块，每个社区包含相互密切相关的微生物。社区结构有助于识别共现模式并揭示微生物群落的功能模块。

（11）**网络中心节点（Hubs）**：中心节点是在网络中具有极高节点度的微生物物种，它们通常在微生物群落中具有关键的生态功能。

（12）**共现和互斥关系（Co-occurrence and Co-exclusion）**：共现关系指的是微生物在相同样本中同时出现的情况，而互斥关系指的是微生物在相同样本中很少同时出现的情况。这些关系可以揭示微生物之间的相互作用模式，如协作或竞争。

（13）**模块间联系（Inter-module Connectivity）**：这个指标用于了解不同社区之间的联系程度，有助于理解微生物群落的整体结构。

（14）**模块度（modularity）**：是衡量网络社区结构的一个指标。模块度描述了网络中节点聚集成社区的程度，数值范围通常在-1到1之间。当模块度接近1时，表示网络中存在明显的社区结构；而当模块度接近0时，则意味着网络的社区结构不明显。

6.4.2 共现网络计算

- 安装R包`Hmisc`、`igraph`

```
install.packages("Hmisc")
install.packages("igraph")
```

- 计算代码

```
# 参考贴子1：https://mp.weixin.qq.com/s/Ew7lUaPBQbuT84vYQOMLNA
# 参考贴子2：https://mp.weixin.qq.com/s/AAfrCaVobIfgbRtrM7E5dg
library(Hmisc)
library(igraph)

# ASV表：行为样本，列为ASV
ASV_table =
read.table("/Users/xjm/Documents/sampling/resampling_data_analysis/16S_test/dada2_single_end_output/seq_table_no_chloro_mito.tsv", fill = T, check.names = F, row.names = 1, header=T, sep="\t")
# 物种分类表
tax_table =
read.table("/Users/xjm/Documents/sampling/resampling_data_analysis/16S_test/dada2_single_end_output/ASV_taxon_no_chloro_mito.tsv", fill = T, check.names = F, row.names = 1, header=T, sep="\t")
```

```r

# ASV过滤阈值参考文献：https://doi.org/10.1038/s41396-020-0621-7


# 取流行度比例大于20%的ASV
prev_prop_df = apply(X = ASV_table,
                MARGIN = 2,
                FUN = function(x){sum(x > 0)/dim(ASV_table)[1]})

total_relative_abundance = colSums(ASV_table)/sum(colSums(ASV_table))

is.above.prev = prev_prop_df > 0.2
table(is.above.prev)

# 取总相对丰度大于0.01%的ASV
is.above.abundance = total_relative_abundance > 0.0001
table(is.above.abundance)

table(is.above.prev & is.above.abundance)
target_ASV = colnames(ASV_table)[is.above.prev & is.above.abundance]

# 转换成相对丰度
ASV_table.prop = as.data.frame(t(apply(ASV_table, 1, function(ASV) ASV/sum(ASV))))

target_ASV_table.prop = ASV_table.prop[, target_ASV]

print(rowSums(target_ASV_table.prop))


# Calculate pairwise correlation (Spearman correlation)
correlation_matrix <- rcorr(as.matrix(target_ASV_table.prop), type="spearman")

# 相关性过滤阈值参考文献：https://doi.org/10.1186/s40168-023-01708-6

# Thresholding: Retain only significant correlations
correlation_threshold <- 0.7  # Adjust as needed
p_threshold <- 0.001  # Adjust as needed 0.05 有点太大



CorrDF <- function(cormat, pmat) {
  ut <- upper.tri(cormat) # 由于相关性矩阵是对称的，取上三角矩阵计算即可
  data.frame(
    source = rownames(cormat)[col(cormat)[ut]],
    target = rownames(cormat)[row(cormat)[ut]],
    corr = (cormat)[ut],
    weight = abs((cormat)[ut]), # 相关性系数绝对值作为权重
    p = pmat[ut],
    p.adjust = p.adjust(pmat[ut], method="BH"), #Multiple testing correction using Benjamini-Hochberg standard false discovery rate correction ("FDR-BH"), to minimize false-positive signals before network construction
```

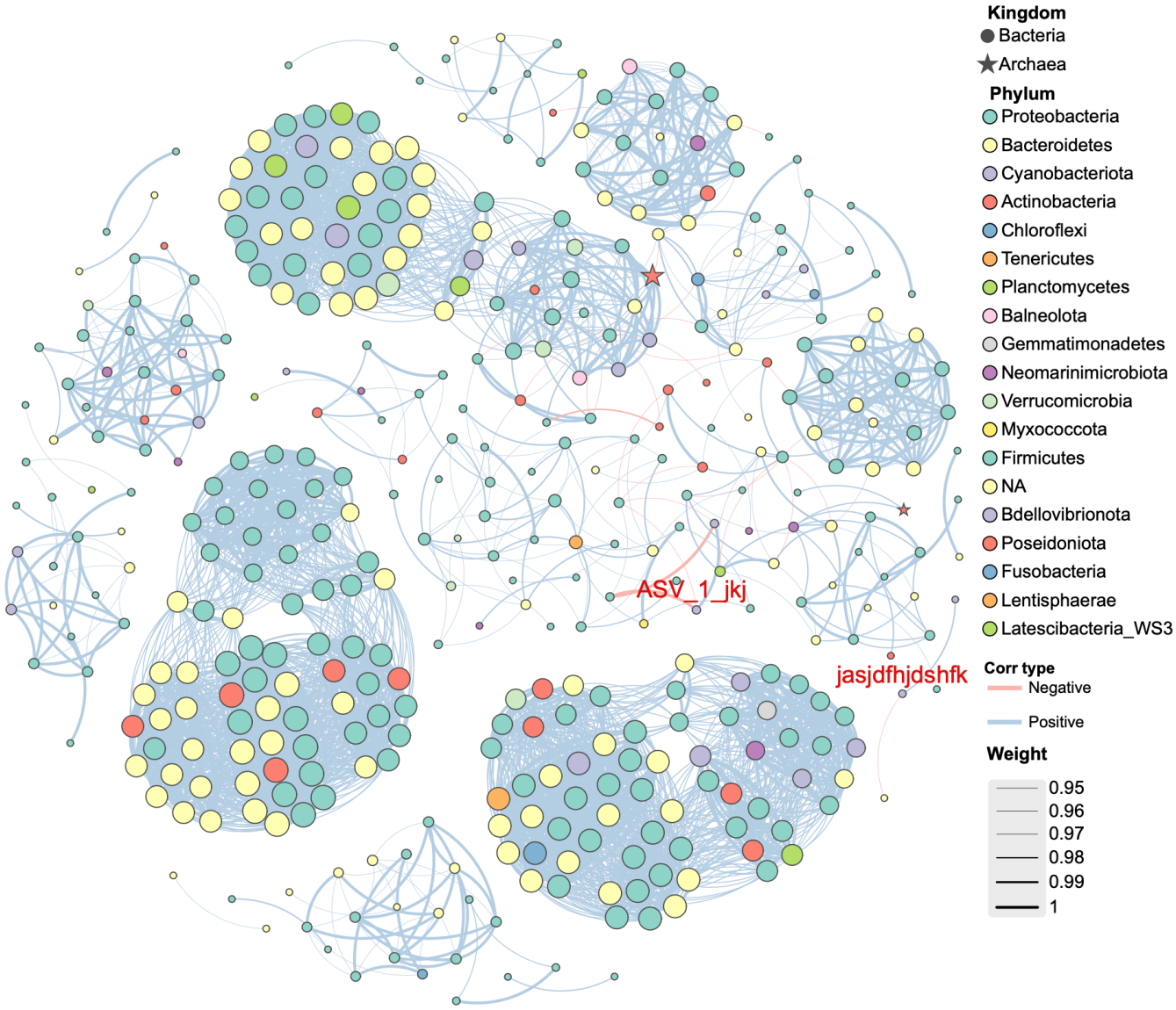
```
    cor_type = ifelse((cormat)[ut]> 0, "Positive", "Negative")
    )
}


cor_df <- CorrDF(correlation_matrix$r , correlation_matrix$P) # 整理ASV之间的连接关系

significant_cor_df <- cor_df[cor_df$weight >= correlation_threshold &
cor_df$p.adjust < p_threshold,] # 保留spearman相关性绝对值>0.7且q-value < 0.001的边

# 非加权无向
g = graph_from_data_frame(significant_cor_df, directed=F)

# plot_network(g, ps.prop_above_prev_abund, type = 'taxa', color = "Phylum")


significant_tax = tax_table[V(g)$name,]
V(g)$Kingdom = significant_tax$Kingdom                              #界
V(g)$Phylum = significant_tax$Phylum                                #门
V(g)$Class = significant_tax$Class                                  #纲
V(g)$Order = significant_tax$Order                                  #目
V(g)$Family = significant_tax$Family                                #科
V(g)$Genus = significant_tax$Genus                                  #属
V(g)$Species = significant_tax$Species                              #种

# 创建节点列表
node_list = data.frame(node_id = names(V(g)),significant_tax)

# 创建边列表
edge_list = data.frame(edge_id = paste0("edge_", c(1:length(E(g)))),
 significant_cor_df)


table(edge_list$cor_type)

# Network analysis

nodes_num = length(V(g))                    #节点数
nodes_num

edges_num = length(E(g))                    #边数
edges_num

positive.corr_num = sum(E(g)$corr>0)         #正相关的数量
positive.corr_num

positive.corr_prop = positive.corr_num/edges_num #正相关的比例
positive.corr_prop

negative.corr_num = sum(E(g)$corr<0)         #负相关的数量
negative.corr_num

```

```r
negative.corr_prop = negative.corr_num/edges_num #负相关的比例
negative.corr_prop

average_degree = mean(degree(g))            #平均度
average_degree

average_shortest_path_length = mean_distance(g, directed = FALSE)      #平均最短路径长度
average_shortest_path_length

network_diameter = diameter(g, directed = FALSE)                    #网络直径
network_diameter

network_density = edge_density(g)                                   #网络密度
network_density


modularity = modularity(g, membership(cluster_louvain(g)), directed = F) # 模块度
modularity

# 传递性：Transitivity measures the probability that the adjacent vertices of a vertex are connected. This is sometimes also called the clustering coefficient.
# 聚类系数
clustering_coefficient <- transitivity(g, type="global")
clustering_coefficient

network_parameter = data.frame(nodes_num,
                               edges_num,
                               positive.corr_num,
                               positive.corr_prop,
                               negative.corr_num,
                               negative.corr_prop,
                               average_degree,
                               average_shortest_path_length,
                               network_diameter,
                               network_density,
                               clustering_coefficient,
                               modularity
)

output_dir = "/Users/xjm/Documents/sampling/resampling_data_analysis/16S_test/summary_output/network"

write.table(node_list, file = file.path(output_dir, "network.node_list.tsv"), sep = "\t", quote = F, row.names = F)
write.table(edge_list, file = file.path(output_dir, "network.edge_list.tsv"), sep = "\t", quote = F, row.names = F)
write.table(network_parameter, file = file.path(output_dir, "network_parameter.tsv"), sep = "\t", quote = F, row.names = F)
write_graph(g, file.path(output_dir, 'network.graphml'), format = 'graphml')  #后续在Gephi中可视化
```

```
154
155
```

6.4.3 共现网络可视化

- ChiPlot



- Gephi：下载地址

b站教程

缺点：没有图注

6.5 Mantel test环境因子与群落组成相关性分析

6.5.1 Mantel test计算

- 安装R包`linkET`

```
install.packages("devtools")
devtools::install_github("Hy4m/linkET", force = TRUE)
```

- 计算代码

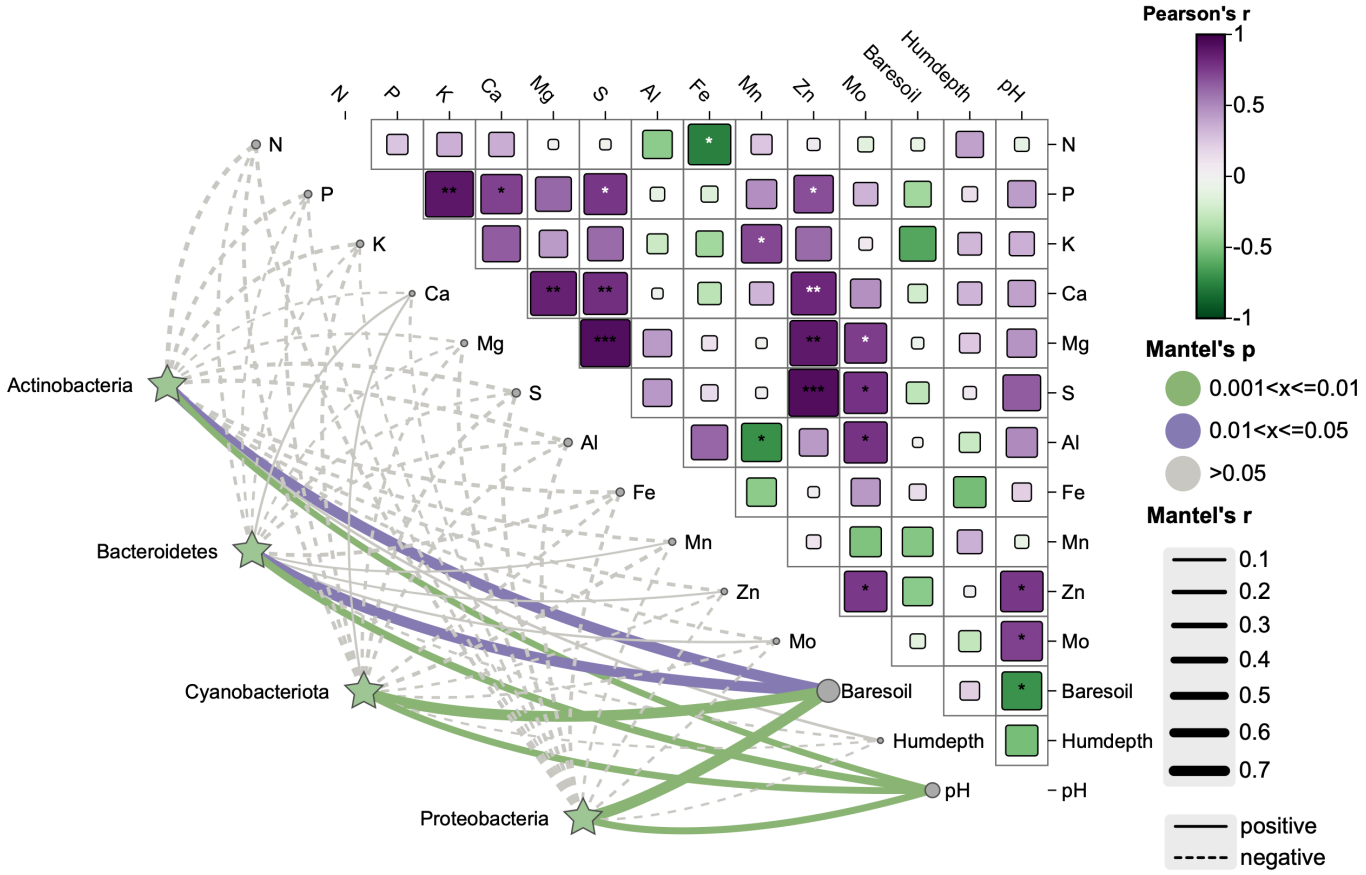
```
library(linkET)

# ASV表：行为样本，列为ASV
ASV_table = read.table("/Users/xjm/Documents/sampling/resampling_data_analysis/16S_test/summary_output/mantel_test/data/seq_table_no_chloro_mito.tsv", fill = T, check.names = F, row.names = 1, header=T, sep="\t")

# 物种分类表
tax_table = read.table("/Users/xjm/Documents/sampling/resampling_data_analysis/16S_test/summary_output/mantel_test/data/ASV_taxon_no_chloro_mito.tsv", fill = T, check.names = F, row.names = 1, header=T, sep="\t")

# 理化因子表
chem_table = read.table("/Users/xjm/Documents/sampling/resampling_data_analysis/16S_test/summary_output/mantel_test/data/varechem.tsv", fill = T, check.names = F, row.names = 1, header=T, sep="\t")

# 转换成相对丰度
ASV_table.prop = as.data.frame(t(apply(ASV_table, 1, function(ASV) ASV/sum(ASV))))

# 计算理化因子相关性
correlation_matrix = correlate(chem_table, engine = "Hmisc", method = "pearson")

corr_p = correlation_matrix$p

# p值校正 Multiple testing correction using Benjamini-Hochberg standard false discovery rate correction ("FDR-BH"), to minimize false-positive signals
correlation_matrix = adjust_pvalue(correlation_matrix, method = "BH")

corr_r = correlation_matrix$r
corr_p_adjust = correlation_matrix$p

# 目标计算分类水平
tax_level = "Phylum"
target_tax_name = c("Actinobacteria", "Bacteroidetes", "Cyanobacteriota", "Proteobacteria")

```

```r
output_dir = "/Users/xjm/Documents/sampling/resampling_data_analysis/16S_test/summary_output/mantel_test/res"

for(target_tax in target_tax_name){
  # 根据目标分类抽取相应的ASV子表
  target_ASV = rownames(tax_table)[tax_table[[tax_level]] %in% target_tax]
  sub_ASV_table.prop = ASV_table.prop[, target_ASV]

  # 计算Mantel test
  mantel_res <- mantel_test(sub_ASV_table.prop, chem_table, mantel_fun = "mantel", spec_dist = "bray", env_dist = "euclidean")
  colnames(mantel_res)[1] <- target_tax

  write.table(mantel_res, file = file.path(output_dir, paste0(target_tax, ".tsv")), sep = "\t", quote = F, row.names = F)


}

# 计算全部物种的Mantel test
mantel_res <- mantel_test(ASV_table.prop, chem_table, mantel_fun = "mantel", spec_dist = "bray", env_dist = "euclidean")

colnames(mantel_res)[1] <- "All species"

write.table(mantel_res, file = file.path(output_dir, "All species.tsv"), sep = "\t", quote = F, row.names = F)


# 导出相关性表格
# 修改第一列名称为rValue，方便chiplot识别
write.table(cbind(rValue=rownames(corr_r),corr_r), file = file.path(output_dir, "rValue.tsv"), sep = "\t", quote = F, row.names = F)

# 修改第一列名称为pValue，方便chiplot识别
write.table(cbind(pValue=rownames(corr_p),corr_p), file = file.path(output_dir, "pValue.tsv"), sep = "\t", quote = F, row.names = F)
write.table(cbind(pValue=rownames(corr_p_adjust), corr_p_adjust), file = file.path(output_dir, "pValue_adjust.tsv"), sep = "\t", quote = F, row.names = F)


```

Mantel test相关性r的生物学意义

- 可以理解为两个矩阵内距离变化的相关性，传统的是两列数值大小变化的相关性
- 就微生物生态方向的研究，一个环境因素对某种生物类群起作用应该是环境因素差异增大时，生物类群差异变大。不存在环境因素差异增大，让生物类群变得更一致的情况。即使结果是负值，也只是基于计算得出的
- 生态领域，一般计算环境距离和群落距离负的情况不多，即使负也是接近0. 所以一般论文那样处理原则性问题没有，r<0.25和|r| <0.25一样的。因为负的都比较小，Mantel基本不会是负，如果是负，也不会显著

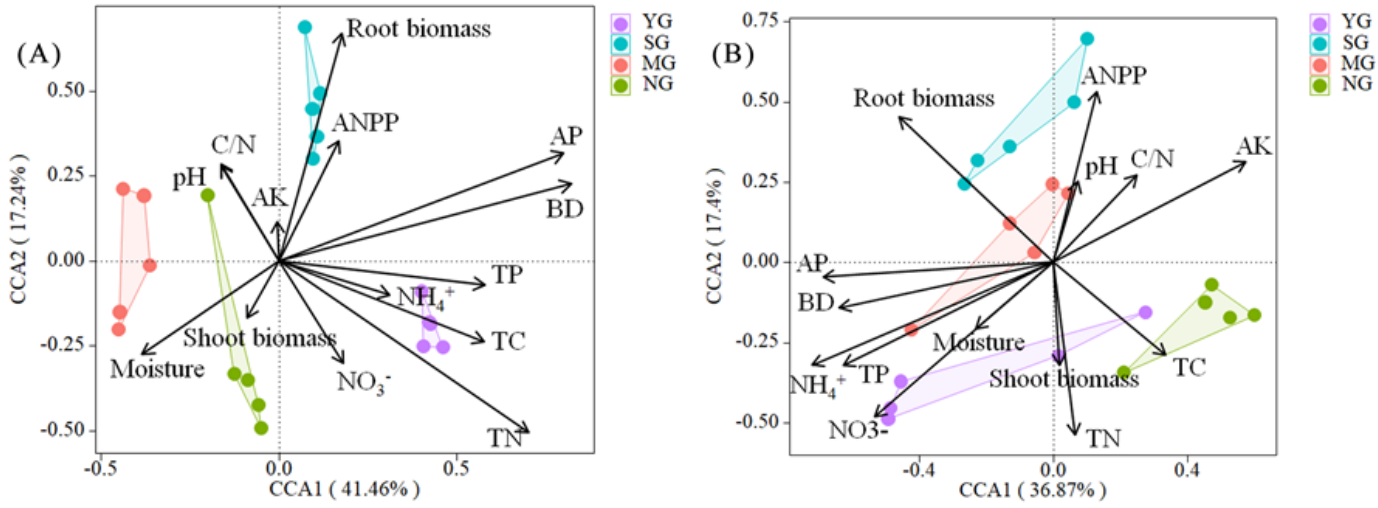
6.5.2 Mantel test可视化



6.6 CCA、RDA环境因子与物种分布相关性分析

6.6.1 CCA、RDA图解读

参考推文



CCA、RDA是一种用于研究生态学数据的多元统计技术。它结合了对物种丰富度和环境因素的分析，帮助研究人员理解生态系统中不同物种与环境条件之间的关系。可以揭示物种分布与环境变量之间的相关性，帮助解释生态系统中的生物多样性模式。该方法常用于揭示环境因素对生物多样性空间分布的影响，以及预测不同环境条件下物种的分布。通过对数据进行排序和约简，有助于识别主要驱动生态系统变化的因素。

> ## 解读
>
> - RDA和CCA的结果图中使用点代表不同的样本，从原点发出的箭头代表不同的环境因子。
> - 箭头的长度代表该环境因子对群落变化影响的强度，箭头的长度越长，表示环境因子的影响越大。
> - 箭头与坐标轴的夹角代表该环境因子与坐标轴的相关性，夹角越小，代表相关性越高。
> - 样本点到环境因子箭头极其延长线的垂直距离表示环境因子对样本的影响强度，样本点与箭头距离越近，该环境因子对样本的作用越强。
> - 样本位于箭头同方向，表示环境因子与样本物种分布的变化正相关，样本位于箭头的反方向，表示环境因子与样本物种分布的变化负相关。

> ## 算法选择
>
> 我们进行RDA或CCA之前先进行DCA分析(Detrended correspondence analysis),根据结果中的Lengths of gradient的数值来进行判断，以下是一条不成文的规则：
>
> 结果会给出4个Lengths of gradient的数值，如果其中最大的数值大于4，则应选择CCA，如果最大的数值小于3，则选择RDA，如果最大的数值在3-4之间，则两种分析方法都可以。
>
> 其实，这只是经验定律，在实际的分析过程中，我们同时进行两个分析，然后比较哪个分析的结果更符合我们的预期，我们就用哪个，毕竟统计是为我们服务的，不能反被统计绑架。

6.6.2 CCA、RDA计算

- 安装R包`vegan`

```
install.packages("vegan")
```

- 计算代码

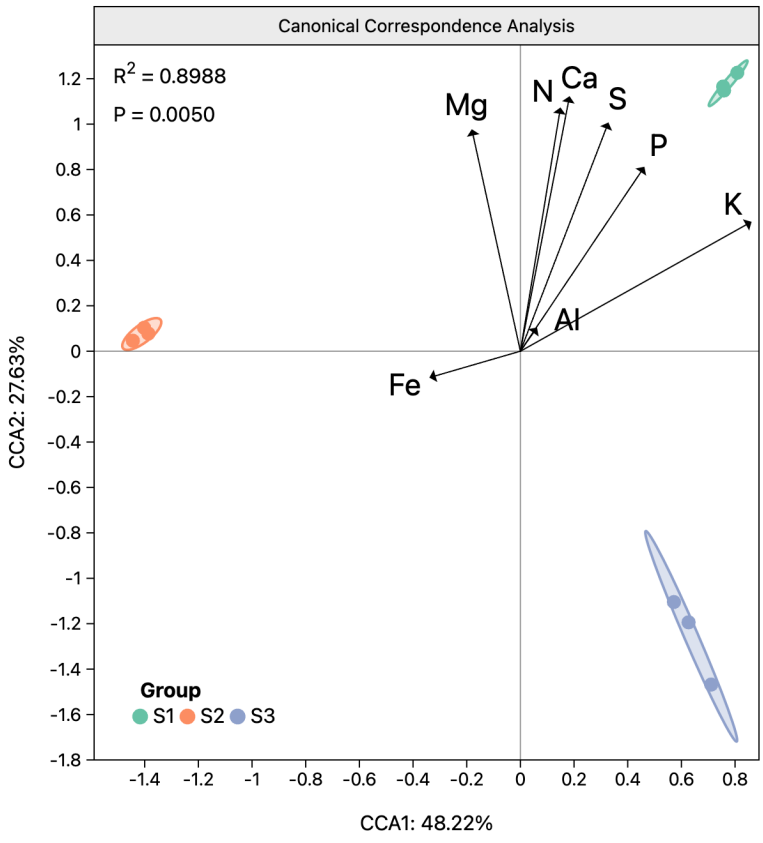
```
#Canonical Correspondence Analysis (CCA)
library(vegan)



# ASV表：行为样本，列为ASV
ASV_table =
read.table("/Users/xjm/Documents/sampling/resampling_data_analysis/16S_test/summary_output/CCA/data/seq_table_no_chloro_mito.tsv", fill = T, check.names = F, row.names = 1, header=T, sep="\t")


# 理化因子表
chem_table =
read.table("/Users/xjm/Documents/sampling/resampling_data_analysis/16S_test/summary_output/CCA/data/varechem.tsv", fill = T, check.names = F, row.names = 1, header=T, sep="\t")

# 标准化成相对丰度
# 标准化参考帖子：https://r.qcbs.ca/workshop09/book-en/transformations.html
```

```r
ASV_table.std.total = decostand(ASV_table, MARGIN = 1, method = "total")

# 标准化成相对丰度的平方根
#ASV_table.std.hellinger = decostand(ASV_table, MARGIN = 1, method = "hellinger")


#先进行DCA分析
dca <- decorana(veg = ASV_table.std.total)

# DCA分析
GL = as.data.frame(apply(dca$rproj, 2, max))

colnames(GL) <- c("Gradient length")

#You do not give a reproducible case, but if you really have 31 sampling units (observations) and 55 predictors, you are overfitting your data. You cannot have more predictors than observations – or you can, but 31 random variables will completely explain 31 observations. Probably the problem is the same with your "only 10 first variables": these were enough to predict exactly your observations and the later ones were dumped (we call that "aliasing"). As a summary: the number of predictor variables cannot be higher than the number of observations. You need either more data or you need to reduce the number of your predictors.
# 环境因子数不能少于样本数，要不只会取前（样本数-1）个环境因子去计算

# 计算RDA
# rda_result <- rda(ASV_table.std.total, chem_table)

# 计算CCA
cca_result <- cca(ASV_table.std.total, chem_table)

plot(cca_result)

cca_score <- scores(cca_result)

# 坐标轴解释比例
cca_summary = summary(cca_result)
explained_proportion1 <- round(cca_summary$cont$importance["Proportion Explained", 1] * 100,2)
explained_proportion2 <- round(cca_summary$cont$importance["Proportion Explained", 2] * 100,2)

output_dir = "/Users/xjm/Documents/sampling/resampling_data_analysis/16S_test/summary_output/CCA/res"

sites_table = cca_score$sites

write.table(cbind(sample=rownames(sites_table), sites_table), file = file.path(output_dir, "cca_sample.tsv"), sep = "\t", quote = F, row.names = F)

envs_table = cca_score$biplot

```

```
write.table(cbind(env=rownames(envs_table), envs_table), file = file.path(output_dir, "cca_env.tsv"), sep = "\t", quote = F, row.names = F)

explained_proportion_table = data.frame(type=c("explained variance ratio (%)"), CCA1=explained_proportion1, CCA2=explained_proportion2)

write.table(explained_proportion_table, file = file.path(output_dir, "explained_proportion.tsv"), sep = "\t", quote = F, row.names = F)


```

6.6.3 CCA、RDA可视化

参考推文



7. 导入qiime2

- 创建目录

```
mkdir -p qiime2_output/qza
mkdir -p qiime2_output/qzv
```

- 导入代表序列

```
# 导入
qiime tools import \
  --input-path dada2_single_end_output/seqs/ASV_seqs_no_chloro_mito.fasta \
  --output-path qiime2_output/qza/rep-seqs.qza \
  --type 'FeatureData[Sequence]'

# 可视化
qiime feature-table tabulate-seqs \
  --i-data qiime2_output/qza/rep-seqs.qza \
  --o-visualization qiime2_output/qzv/rep-seqs.qzv
```

- 导入特征表

```
# 导入
qiime tools import \
  --input-path dada2_single_end_output/tables/seq_table_no_chloro_mito.biom \
  --type 'FeatureTable[Frequency]' \
  --input-format BIOMV100Format \
  --output-path qiime2_output/qza/feature-table.qza
# 可视化
qiime feature-table summarize \
  --i-table qiime2_output/qza/feature-table.qza \
  --o-visualization qiime2_output/qzv/feature-table.qzv
```