1. Download raw sequence

Download from NCBI SRA

- 用SRA-Toolkit从NCBI下载测序数据：

SRA-Toolkit的下载、安装、使用说明请阅读 https://github.com/ncbi/sra-tools/wiki/02.-Installing-SRA-Toolkit。

下载数据的命令如下，可直接复制使用，命令详解请阅读https://trace.ncbi.nlm.nih.gov/Traces/sra/sra.cgi?view=toolkit_doc 查看prefetch的用法。

下载数据的命令：

```
prefetch -O raw-sequence-sra --option-file SRR_Acc_List.txt
```

- Change sequnce format form sra to fastq:

从NCBI下载的数据是SRA格式的，需转换成fastq格式才能用来分析：

数据格式转换的命令如下，可直接复制使用，命令详解请阅读https://trace.ncbi.nlm.nih.gov/Traces/sra/sra.cgi?view=toolkit_doc 查看fastq-dump的用法。

数据格式转换的命令：

```
for i in raw-sequence-sra/*/*.sra;do fastq-dump --split-3 --gzip $i -O raw-
sequence-fastq > change-sequence-format.log;done
```

Download from ENA

- ENA数据下载页面可以获取每个测序数据的下载链接，用wget, axel等下载。ENA下载的数据为fastq格式。

2. metadata

- metadata: 每个样本对应个体的性别，年龄，健康状况，国家等信息，一个样本一条信息（可通俗的理解为一个样本的信息记录在excel表格的一行中）
- 下载：可以从NCBI,补充材料下载
- 整理：如用excel整理，请不要有合并的单元格。
- 整理可能需要注意统一用词，统一用词的目的是能把多个数据集的metadata合并：
    - 列名：每个数据集的表头表示同一信息的列用同一列名，如数据集表示性别的列统一列名用sex或gender，不建议一个数据集用sex,而另一数据集的metadata用gender。
    - 统一大小写：如一个数据集性别中男性用male,其他数据集就不要出现Male。

整metadata很费时间，不要期望一次能整好，常改常新，下图是数据集PRJEB39064的metadata，格式可参考，不同数据集有不同信息可以加列。

	A	B	C	D	E	F	G	H	I	J
	Sampleid	BioProject	Region	Age	Disease_status	Sex	sequencing_method	SRA-Study	Sample_type	Sequencing_region
2	ERR4296609	PRJEB39064	Taiwan	64	Health	male	Illumina MiSeq	ERP122533	Saliva	V3-V4
3	ERR4296610	PRJEB39064	Taiwan	50	Health	male	Illumina MiSeq	ERP122533	Saliva	V3-V4
4	ERR4296611	PRJEB39064	Taiwan	55	Health	male	Illumina MiSeq	ERP122533	Saliva	V3-V4
5	ERR4296612	PRJEB39064	Taiwan	48	Health	male	Illumina MiSeq	ERP122533	Saliva	V3-V4
6	ERR4296613	PRJEB39064	Taiwan	49	Health	male	Illumina MiSeq	ERP122533	Saliva	V3-V4
7	ERR4296614	PRJEB39064	Taiwan	53	Health	male	Illumina MiSeq	ERP122533	Saliva	V3-V4
8	ERR4296615	PRJEB39064	Taiwan	40	Health	male	Illumina MiSeq	ERP122533	Saliva	V3-V4
9	ERR4296616	PRJEB39064	Taiwan	53	Health	male	Illumina MiSeq	ERP122533	Saliva	V3-V4
10	ERR4296617	PRJEB39064	Taiwan	55	Health	male	Illumina MiSeq	ERP122533	Saliva	V3-V4
11	ERR4296618	PRJEB39064	Taiwan	61	Health	male	Illumina MiSeq	ERP122533	Saliva	V3-V4
12	ERR4296619	PRJEB39064	Taiwan	62	Health	male	Illumina MiSeq	ERP122533	Saliva	V3-V4

3. 16s data procesing

- tutorial：https://docs.qiime2.org 可优先看https://docs.qiime2.org/2022.2/tutorials/moving-pictures/
- 以下为16s data procesing的命令，命令详解请阅读教程https://docs.qiime2.org/2022.2/tutorials/moving-pictures
- 注：如果收集的数据集可以做到设置一样的参数一样尽可能每个数据集设置一样的参数，包括序列保留长度，rarefy的深度。

```
# qiime workflow

#generate manifest file
echo -e sample-id'\t'absolute-filepath > manifest
for i in ../raw-sequence-fastq/*_1.fastq.gz; do path=$(readlink -f $i);
id=$(basename $i | cut -d '_' -f 1); echo -e $id'\t'$path >> manifest;done

#import data
qiime tools import \
 --type 'SampleData[SequencesWithQuality]' \
 --input-path manifest \
 --output-path demux.qza \
 --input-format SingleEndFastqManifestPhred33V2
qiime demux summarize \
 --i-data demux.qza \
 --o-visualization demux.qzv

#quality control
qiime quality-filter q-score \
 --i-demux demux.qza \
 --o-filtered-sequences demux-filtered.qza \
 --o-filter-stats demux-filter-stats.qza
qiime metadata tabulate \
  --m-input-file demux-filter-stats.qza \
  --o-visualization demux-filter-stats.qzv


#deblur denoise
qiime deblur denoise-16S --i-demultiplexed-seqs demux-filtered.qza  --p-trim-
length 250 --p-min-reads 0 --p-no-hashed-feature-ids --o-table table-deblur.qza
--o-representative-sequences rep-seqs-deblur.qza --p-sample-stats --o-stats
deblur-stats.qza #注意参数选择


qiime deblur visualize-stats \
  --i-deblur-stats deblur-stats.qza \
```

```
  --o-visualization deblur-stats.qzv
qiime feature-table summarize \
  --i-table table-deblur.qza \
  --o-visualization table-deblur.qzv \

# taxonamy
qiime feature-classifier classify-sklearn \
  --i-classifier /beegfs/db/greengenes/gg-13-8-99-nb-classifier.qza \
  --i-reads rep-seqs-deblur.qza \
  --o-classification taxonomy.qza

qiime metadata tabulate \
  --m-input-file taxonomy.qza \
  --o-visualization taxonomy.qzv

qiime taxa barplot \
  --i-table table-deblur.qza \
  --i-taxonomy taxonomy.qza \
  --m-metadata-file ../metadata.tsv \
  --o-visualization taxa-bar-plots.qzv


qiime phylogeny align-to-tree-mafft-fasttree \
  --i-sequences rep-seqs-deblur.qza \
  --o-alignment aligned-rep-seqs-deblur.qza \
  --o-masked-alignment masked-aligned-rep-seqs-deblur.qza \
 --o-tree unrooted-tree.qza \
  --o-rooted-tree rooted-tree.qza

qiime diversity alpha-rarefaction \
  --i-table table-deblur.qza \
  --p-max-depth 10000 \ #注意参数
  --p-steps 20 \
  --o-visualization alpha-rarefaction.qzv

# diversity
qiime diversity core-metrics-phylogenetic \
  --i-phylogeny rooted-tree-.qza \
  --i-table table-deblur.qza \
  --p-sampling-depth 1314 \ #注意参数
  --m-metadata-file metadata.tsv \
  --output-dir core-metrics-results
```

- 如果序列上有adpter，primer可用如下script 切除。或者用qiime的cut adpter切除。切除后再跑上面qiime流程。

```
# cut adpter, primer
from argparse import ArgumentParser
import sys
import gzip
from skbio.io import read, write
from skbio import DNA
from skbio.alignment import local_pairwise_align_ssw
```

```
def interface(argv):
    parser=ArgumentParser(description='Extract an amplicon region starting from
a given primer')
    parser.add_argument('-i', '--input', help='name of input fasta/fastq file')
    #parser.add_argument('-d','--indir',help='input directory of fastq files
(instead of -i)',default="")
    parser.add_argument('-o', '--output', help='')
    parser.add_argument('-p', '--primer', help='forward primer sequence',
default='ACTCCTACGGGAGGCAGCAG')
    #parser.add_argument('-k', '--keep_primers', help="Don't remove the primer
sequences", action='store_true')

    return parser.parse_args(argv)


def cut_primer(seq, primer):
    alignment, score, start_end_positions = local_pairwise_align_ssw(seq,
primer)
    seq_start, seq_end = start_end_positions[0]
    pirm_start, prim_end = start_end_positions[1]
    match_len = seq_end - seq_start  + 1

    if seq_end + 1 >= len(seq) -3 :
        print( seq.metadata['id'],'align in the end, generate empty line, so
discard the sequence!')
        print(alignment)
        print('raw sequence: ',seq)
        return None

    elif match_len >= len(primer)-3 and prim_end == len(primer)-1:
        print(seq.metadata['id'],'match length enough and primer end also
matched;','match length:',match_len)
        print(alignment)
        return seq[seq_end+1:]

    elif  match_len >= len(primer)-3 and prim_end != len(primer)-1:
        print(seq.metadata['id'], 'match lenth enough but primer end not
match;','match length:', match_len)
        print(alignment)
        n = seq_end + len(primer) - prim_end
        return seq[n:]
        print('raw sequence: ',seq)
        #return None

    else:
        print(seq.metadata['id'],'not found, match length:', match_len)
        print(alignment)
        print('raw sequence: ',seq)
        return None

def main(argv):
    args = interface(argv)
    primer = DNA(args.primer)
    with open(args.input) as fi, open(args.output, 'wb') as fo:
        for seq in read(args.input, format='fastq', variant='illumina1.8',
constructor=DNA):
            after_cut = cut_primer(seq, primer)
            if after_cut is not None:
```

```
                write(after_cut, 'fastq', fo, variant='illumina1.8',
compression='gzip')

if __name__ == '__main__':

    main(sys.argv[1:])
```