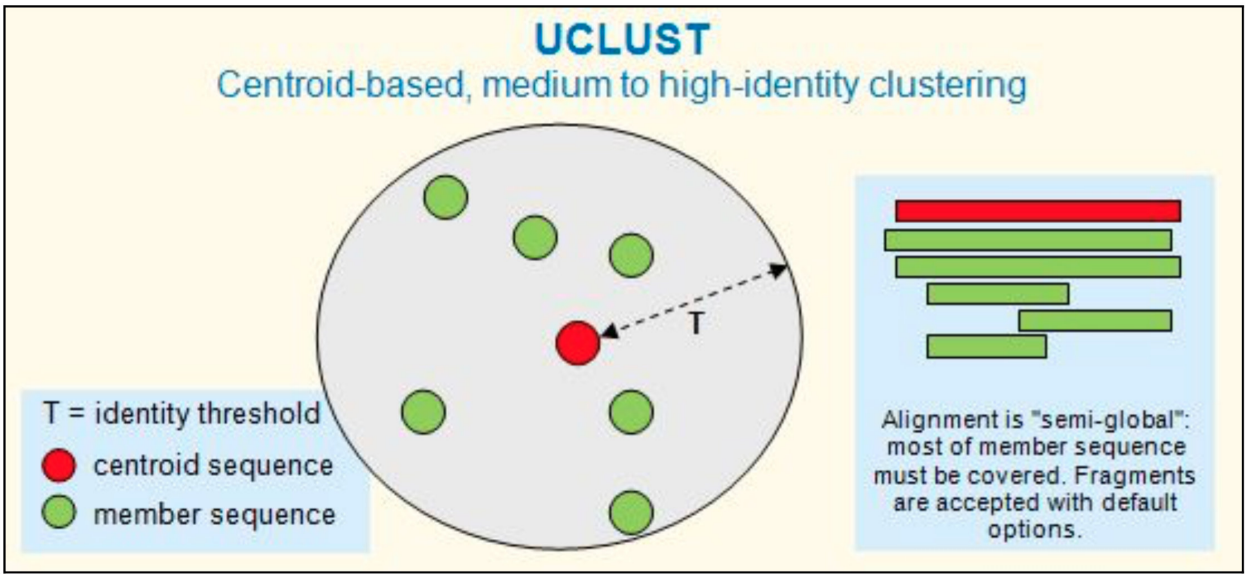
UCLUST算法将一组序列分到不同的cluster中. 但是该算法用于OTU分类.

cluster fast和cluster smalmem 命令基于UCLUST. 一个cluster可被一条序列定义, 命名为centroid或representative序列. 一个cluster中的每一条序列相对于centroid必须拥有超过指定阈值的类似性:



之前版本中, centroids称为seeds, 该名称不再使用了, 避免和比对seeds(matching words)混淆, 例如BLAST/UBLAST. 指定阈值(T)可看作一个cluster的半径. Clustering命令包含cluster fast和cluster smallmeme.

Recommended identity ranges

UCLUST有效用于识别约50%或以上的蛋白序列Clustering, 约75%或以上核酸序列Clustering. 针对较低的阈值, 该命令可能存在问题, 1)比对质量降低 2)根据比对获得的同源性不可信

clustering criteria

UCLUST用于识别一组cluster:

(1) 所有的centroids互相之间的相似性小于T, 同时

(2) 同一个cluster中中的所有序列成员和centroid的相似性大于等于T

实际情况中, 会出现一条序列同时满足多个centrods的情况(相似性都大于T), 那么会选择Clustering到最近的cluster中, 同时若存在多个相同相似性的centroids, 那么会该序列会被任意分配

input sequence order

UCLUST采用贪婪算法, 其输入序列的排序是非常重要的. 在cluster smallmeme 命令中, 输入文件中的序列是按出现的先后顺序处理的. 如果下一个序列满足现存的centroid, 它将会被分配到该cluster, 否则会变成一个新的cluster的centroid. 这意味着, 序列应该先被排序, 使得最恰当的centroids倾向于较早的出现在输入文件中. 其最佳的排序要根据应用: cluster fast 命令用于使用输入序列的顺序(默认), 或根据序列长度排序, 或根据指定的长度或大小(-sort length or -sort size)排序(by size annotation)

searching the centroid database

UCLUST的基本步骤根据识别的centroids比较输入序列. 大多数USEARCH参数, 包括indexing options, 可用于clustering 命令来控制敏感度和内存使用.

clustering fragments

如果全长序列和片段共存于输入文件中, 那么根据降低的序列长度来排序将会有效(sort order, sortbylength). 片段不适合作为centroids, 因为两个全长序列可能有一段类似于一个作为centroid的片段, 但是其序列剩余部分无法匹配. 如果一个片段成为了centroid, 将会使得很多高度差异的全长序列归到同一个cluster中.

OTU clustering

将序列Clustering到Operational Taxonomic Units(OTUs)是一个特殊的问题. 基于长度的分类并不适合, 因为该过程倾向于选择哪些生物学上的outliers作为centroids, 将物种和种属分到不同的clusters中, 带来过多的alpha多样性.

Expected errors predicted by Phred(Q) scores

假设仪器对每个碱基有很好的错误率估计, 那么可以计算一个read的所有期待错误(expected number of errors). 通过Q值(Q20/Q30/Q40)我们并不能获得给定read的所含有的错误碱基数目, 但是, 我们可以通过Q值知道在一个大的read数目中平均的错误数目(average of errors), 这定义为expected number of errors.

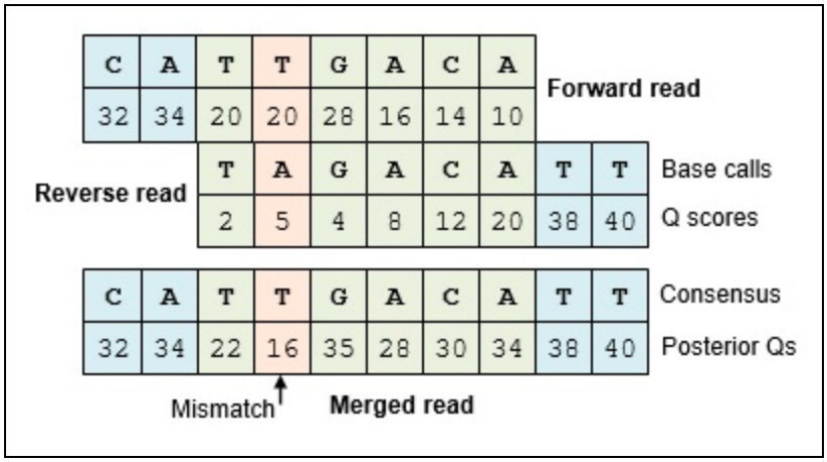
期待错误数目为错误率之和. 首先计算expected errors, sum(P_e), 然后换算成其最接近的整数, 这就为最可能的错误数目. E=floor(E)

更小的E值意味着更高的质量, 较大的E值意味着更低的质量. 因此, 可以设置最大期待错误, 从而过滤掉大于该值的reads(`fastq_filter`).

一般会选择E_max=1, 这样过滤后的reads所包含的错误数目就为0, 也可以设置更严格为0.5或0.25

针对双端reads, 可通过assembler(`fastq_mergepairs`来合并reads并重新计算合并位置的碱基的Q值, 若两碱基一致, 则该合并后碱基的Q值应更大, 反之, 更小(should be calculated as the posterior probability according to Bayes theorem where the P_e's from the forward and reverse read are the prior probabilities).

建议使用USEARCH的[fastq_mergepairs] [https://drive5.com/usearch/manual/merge_pair.html]合并双端reads, 因为其他软件无法正确计算posterior Q值, 包括PANDAseq, COPE, PEAR, fastq-join和FLASH.

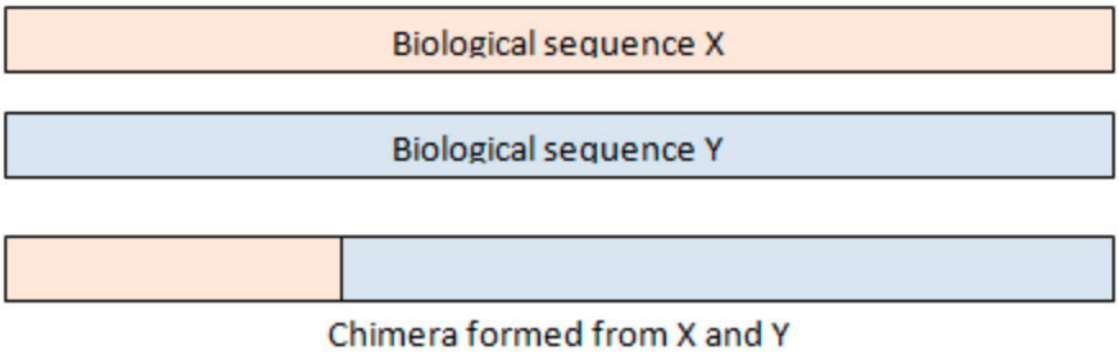


Abundance and amplification bias in amplicon sequencing

read丰度并不一定和物种丰度相关, 首先, 原核基因组包含多个数目的16S基因, 所含数目从1至10, 或更多, 同时包含更多基因的菌种倾向于包含更多的拷贝数; 其次, 若出现引物模版错配其PCR扩增效率将大受影响, 其影响可以是数据级的数目差异. 针对当前流行的V4区域引物, 约9%的物种拥有一个或者多个错配; 再次, 序列组成例如GC含量和均聚物影响聚合酶效率; 最后就是序列长度, 使用降解引物(偏向性扩增)和样本本身组成具有偏向性, 那么经过PCR将会倍增其差异.

Chimeras

Chimeras为来自两个更多个生物学序列的合并. 扩增子的嵌合性序列可来源于PCR过程. 但是shotgun测序时chimeras罕有出现, 当有很相近的序列被扩增时就容易出现嵌合性序列. 尽管原因很多, 最可信的原因机制为不完整的序列延伸. 在PCR循环中, 部分延伸的链能结合到不同的相似序列上, 然后发挥引物作用, 延伸形成嵌合体序列.



在16S测序中, 发现仅有小部分reads是嵌合的, 可能只有1%到5%. 在序列分析中就会出现一个挑战性问题, 是因为这些嵌合体序列常常具有低复杂性, 非常类似于其父源, 因此很难从真正的生物学序列中区分.

[Chimera detection and filtering] [https://www.drive5.com/usearch/manual/cmds_all.html]

annot command

注释来自标记基因测序实验的reads. 可用于mock communities和真实的communities. 分类后的reads称之为sequence, 已知的具有错误, Phix, chimera的序列

`-knowndb`: 包含在reads中的序列数据文件(udb或FASTA), 一般为mock comunity的参考序列

`-db`: 数据文件(udb或FASTA), 应该为基因的最大的可获得的数据文件, SILVA for 16S/UNITE for ITS. 应该提供给mock community用于检测污染

以上两种可提供一个或两个

输出`-tabbedout` / `fastaout` / `-fastqout`

```
usearch -annot mock_reads.fq -knowndb mock_ref.fa -db silva.udb -tabedout annot.txt -fastqout annot.fq
```

uchime2_ref command

采用UCHIME2算法, 根据用户提供的参考数据检测parent sequences

必须通过`-db`选项提供核酸序列数据文件, 其格式可为FASTA/UDB。

由于假阳性, 不可使用UCHIME2对OTU clustering or denoising

参考数据文件, 推荐使用最新的SILVA for 16S/ UNITE for ITS数据文件; 支持多个输出文件:

`-uchimeout out.txt` tabbed text; `-chimeras ch.fa` 包含预测的chimerias的fasta文件; `-notmatched not.fa` 不匹配数据库的fasta文件; `-uchimealnout aln.txt` 比对的文件

`-mode` 该选项也为必须, high_confidence, 报告具有高置信度的预测结果, 同时会带来相对较高的假阴性率; specific, 类似high_confidence, 但是没有那么严格, 其结果类似old UCHIME 算法; balanced, 平衡假阳性和假阴性的结果, 意图减少整体的错误率; sensitive, 在高的假阳性的情况下获得高的敏感性; denoised, 报出所有完美的chimeric models. 最可能用于设计和验证算法使用, 该模型含有使用.

`-strand` 指定搜索的链, 因为目前不支持搜索双链

`-self` 指定忽略匹配query序列的参考序列, 当使用已知不含有chimeras序列的数据文件, 用于评估假阳性率时很有用. 该选型要求query和database为相同文件.

```
usearch -uchime_ref reads.fasta -db 16s_ref.udb -uchimeout out.txt -strand plus -mode sensetive
```

uchime3_denovo command

该命令用于一套已经去噪音后的扩增子序列(a set of denoised amplicons). 输入序列需要拥有[size=nnn; annotations][https://www.drive5.com/usearch/manual/size_annot.html] giving amplicon abundances.

该命令和原始UCHIME2算法最大的改变是默认的丰度倾向(abundance skew)是16而不是2. 根据最近的结果, abskew=2时存在更多的假阳性chimeras. 但是abskew=16时, 假阳性和假阴性之间能够实现很好的平衡.

uchime3_denovo输入文件必须是去噪音后的扩增子(denoised amplicons). 即使为质量过滤后的reads, 如果没有去噪音也是不行的.

支持一下输出:

`-uchimeout` tabbed 文本文件

`-nonchimeras` 不含chimeric序列的fasta文件

`-chimeras` 包含chimeric序列的fasta文件

`-alnout` 可读的比对文件文件

不推荐使用uchime2_ref/uchime3_denovo用于OTU clustering pipeline.
`cluster_otus_command`含有内建的de novo chimera过滤功能.

```
usearch -uchime3_denovo denoised.fa -uchimeout out.txt -chimeras ch.fa -nonchimeras nonch.fa
```

unoise3 command

输入文件为质量过滤后的reads序列, 同时具有size=nnn; abundance annotations. 查看[OUT/denoiseing pipeline][https://www.drive5.com/usearch/manual/uparse_pipeline.html], 了解reads前处理以及其他类型错误的去除.

输入文件必须为decreasing abundance, 例如, 通过降低size=nnn annotation值. 可使用命令[sortbysize command][https://www.drive5.com/usearch/manual/cmd_sortbysize.html]实现. 该算法用于illumina reads, 不适用与454, Ion Torrent, PacBio reads.

-zotus 预测正确的生物序列输出为fasta格式. 标签格式为Zotunnn, Uniqlabel, nnn为1,2,3.., Uniqlabel为输入文件的标签

-ampout 预测正确的扩增序列输出为fasta格式. 这些文件包含chimeras, 标签格式为Ampnnn,uniq=Uniqlable; uniqsize=u; size=s; nnn为1,2,3..., Uniqulabel为输入文件标签, 从第一个逗号分隔, u为来自输入文件的size=annotation, s为源于该amplicon的总的reads数目

-minsize 指定最小的abundance(size=annotation), 默认为8. 具有更低abudances的输入文件舍弃.

-tabbedout 指定tabbed文本文件, 包含每个序列的所有处理, noisy或chimeric

-unoise_alpha 指定alpha参数, 默认为2.0

```
usearch –unoise3 uniques.fa –zotus zotus.fa –tabbedout unoise3.txt
```

[Generating OTUs and ZOTUs] [https://www.drive5.com/usearch/manual/pipe_otus.html]

OTUs为来自reads的选择后的序列. 其目的是为识别一组正确的生物序列([defining and interpreting OTUs][https://www.drive5.com/usearch/manual/otus.html])

根据97%的Clustering标准, 一个OTU序列和其他OTUs应该具有至少3%的差异, 且在其相邻序列中时最丰富的. 可使用[cluster_outs command] [https://www.drive5.com/usearch/manual/cmd_cluster_otus.html]实现, 该命令采用了UPARSE算法.

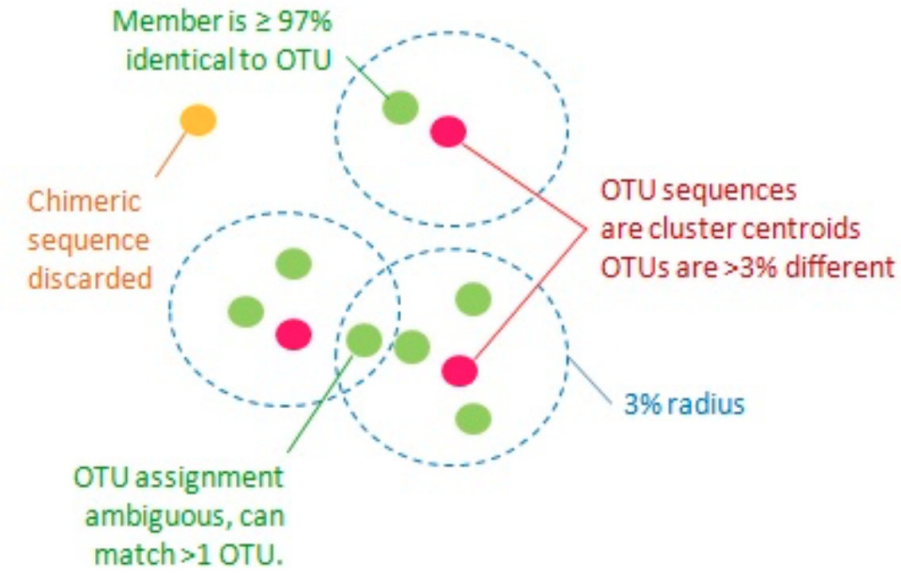
UPARS-OTU 算法

通过cluster_otus命令从NGS扩增reads中构建一套OTU代表性序列. reads需要先重叠双端redas, 去除barcode, 质量过滤和整体过滤([global trimming] [http://www.drive5.com/usearch/manual/global_trimming_and_abundance.html], 确保来自相同模版的reads拥有相同的长度, 也就是确保序列起点和终点位于相同的位置, 若不处理, 那么相同生物序列的两个reads可能又有不同的长度, 这将会在计算唯一序列丰度是带来问题). 后续处理为比对reads到OTU, 构建OTU表格

聚类标准

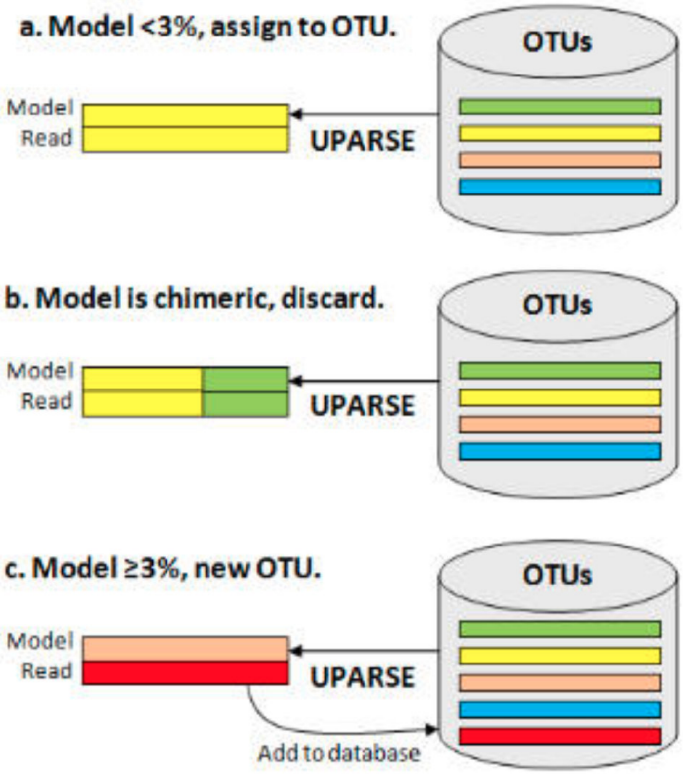
UPARSE-OTU的目的在于发现一套OTU代表性序列, 满足一下标准:

- *所有OTU序列两两之间应小于97%的一致性*
- *OTU应该为其中丰度最大的序列*
- *去除嵌合体序列*
- 所有非嵌合体序列应该匹配至少一个OTU, 大与等于97%一致性



因为高丰度reads倾向于为正确的扩增子序列, 因此UPARSE-OTU根据丰度的降序考虑输入序列. 这意味着OTU centroids倾向从丰度更高的reads中选择.

每一个序列都和当前的OTU数据库比较, 同时使用UPARSE-REF查询序列的最相近模型, 存在三种情况:1), UPARSE-REF模型和现存的OTU满足大与等于97%的一致性; 2), 模型为嵌合体; 3), 模型和任意现存OTU都小于97%一致性. 1), 成为匹配OTU的一员; 2), 舍弃输入序列; 3), 输入序列成为一个新的OTU.



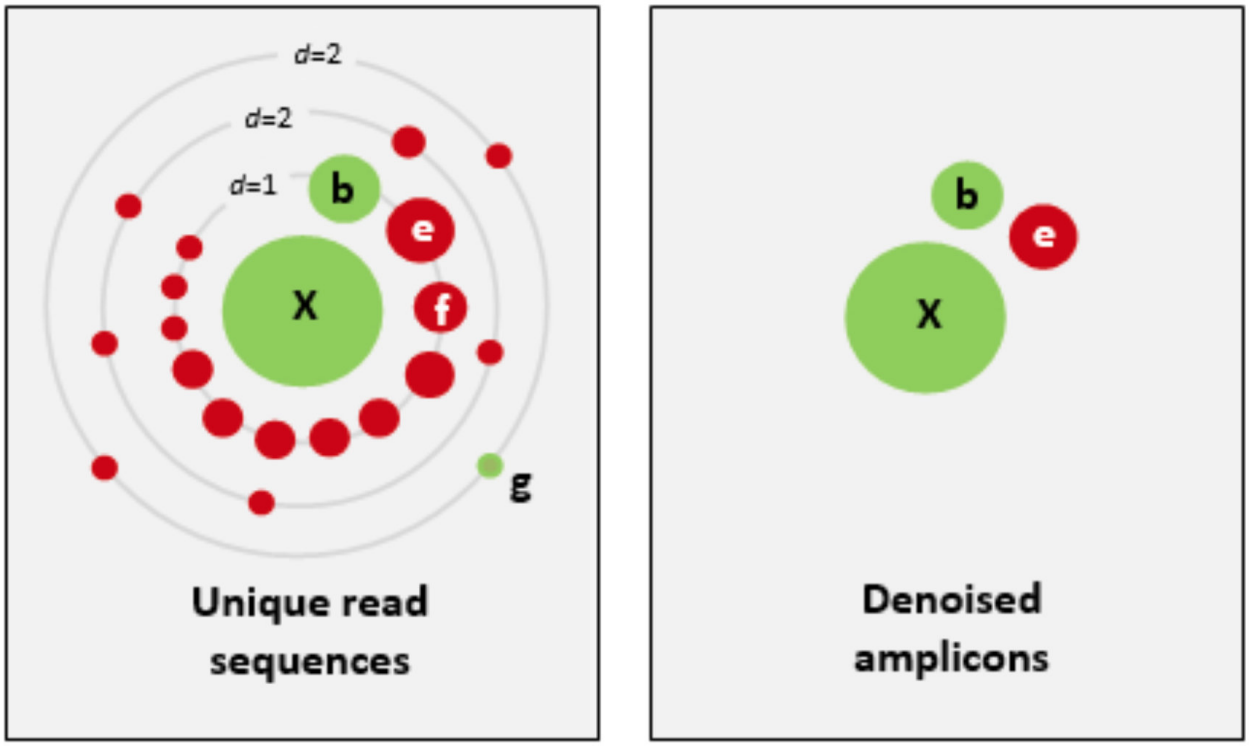
去噪音目的在于识别reads中所有正确的生物序列. 可通过`unoise3`命令实现, 采用UNOISE算法. 去除噪音的序列称之为'ZOTU(zero-radius OTU)'.

UNOISE算法

从reads中检测正确的生物学序列, 分辨序列差异到一个或多个差异(correct biological sequences are recovered from the reads, resolving distinct sequences down to a single difference(often) or two or more differences(almost always))

根据一下步骤错误矫正:

- *识别具有测序或PCR点错误(难以去除, 强烈推荐使用[expected error filtering][http://www.drive5.com/usearch/manual/exp_errs.html]和[discarding singletons][http://www.drive5.com/usearch/manual/singletons.html])的reads并去除*
- *去除嵌合体*



左图显示了靠近高丰度唯一read序列X的分布, 根据其对X的距离分组(d=1;d=2;d=3...). 点表示唯一序列, 点的大小表示其丰度. 绿色为正确的生物学序列, 红色表示有一个或多个的错误. 相对于X拥有小的差异数目或小的丰度则预测为X的错误reads. 右图展示了去噪音后的扩增子. 这里, X和b被正确预测, e为一个具有错误的异常高丰度, 同时被错误的预测为正确, f为一个正确舍弃的但是拥有足够高丰度被认为是假阳性的错误, g为低丰度正确但是被错误舍弃. bef丰度相似, 这里展示了去噪音过程的根本挑战: 如何设置丰度阈值从错误的序列中识别正确的序列

ZOTUs是可用于多样性分析的有效OTUs, 但是其结果解释不同于一般的97% OTUs. 例如, 一个物种可能包含超过一个ZOTU, 同时根据97% OTUs, 一个OTU可能包含超过一个物种.

The input data requirements of cluster_outs and unoise3 are the same: 应该是一套经过trim和质量过滤后的唯一序列, 因此很容易生成97% OTUs和ZOTUs

唯一序列使用[fastx_uniques][https://www.drive5.com/usearch/manual/cmd_fastx_uniques.html]命令生成. -sizeout必须指定, 以使得size annotation包含在标签中. 建议使用-relabel Uniq.

```
usearch -fastx_uniques filtered.fa -fastaout uniques.fa -sizeout -relabel Uniq
```

```
usearch -cluster_otus uniques.fa -otus otus.fa -relabel Otu
```

```
usearch -unoise3 uniques.fa -zotus zotus.fa
```

Defining and interpreting OTUs

see also [Interpreting counts and frequencies in OTU tables] [https://www.drive5.com/usearch/manual/otu_count_interpret.html]

Sneath-Sokal OTUs

Operational Taxonomic Unit(OTU)概念首先被Peter Sneath and Robert Sokal 在1960s引入

OTUs的目的在于, 基于观察的特征开发的量化策略用于分类organisms, 构建一个等级分类来尽可能的真实的反应进化关系. Sneath-Sokal OTUs用于构建观察的特征的表格, 该表格可用于描述多个变量, 例如, 1=present, 0=absent.

根据其共有的特征将其合并进入最相似的群体, 从而构建等级树图. 根据共有的特征数目通过反复将最相似的groups合并到一起来构建等级树状结构.

Historical 97% identity threshold

在16S测序中, OTUs典型地使用97%的一致性构建. 最早, Stackbrandt 和 Goebel 发现16S序列97%的一致性对应70%的DNA重新关联值, 这在之前已经被接受作为细菌物种的可行的定义.

Modern OTU identity thresholds

2014年的研究发现通过对全场16S序列Clustering发现评估物种的最佳阈值为98.5%. 然而, 该研究是使用CD-HIT分析的, 该方法存在[problematic non-standard definition of sequence identity] [https://www.drive5.com/usearch/manual/cd_hit_id.html], 根据SILVA数据库的分类预测, 其中接近1/5是错误的. 在更近的文章中, I showed识别全长序列最佳阈值约为99%, 针对V4区域约为100%.

OTUs constructed by clustering 16S sequences

用于Clustering16S序列的最早软件是FastGroup, 它使用类似于UCLUST的贪婪Clustering算法, 其识别阈值也是97%. **DOTUR和mothur使用一系列一致性阈值构建OTUs, 超过97%阈值的序列分配到相同的species, 具有95%一致性为相同的genus, 拥有80%一致性为相同的phylum, 但是这些区分是有争议的. QIIME使用97%的阈值, 早期的QIIME版本使用CD-HIT作为默认的OTU聚类方法, 后来更改为UCLUST.**

Goals for making OTUs as clusters of reads

operational definiton of speciess-like groups

merging variation within a single strain into one cluster

细菌染色体常包含多个16S基因拷贝(paralogs), paralogs常包含小的序列差异. Clustering倾向于将paralogs合并到同一个OTU中. 如果目的在于每个基因型获得一个OTU, 这将对大多数分析有意义.

merging variation between strains of a single specie into one cluster

给定species的不同stains, 例如E. coli, 在其16S序列中常含有差异. 这在species水平上增加了一层变异, 这使得单个strain由于paralogs仍具有变异. 针对species, Clustering倾向于将这两个水平的变异合并来构建单个OTU, 或至少更少的OTUs. 如果目的在于针对每个基因型获得一个OTU, 那么这并不是一个好的现象, 因为不同的starins在其表型上可拥有重要的差异(e.g. , pathogenic or not). 并且, 当strain的概念被合理完好的定义(minimal variation in the complete genome), 那么species的概念就成为了细菌的问题的, 因此每个specie拥有一个OTU并不能很好的定义或适用.

merging variation due to experimental error into one cluster

PCR测序错误可导致reads中的序列变异. Clustering倾向于将bad序列和正确的生物学序列(correct biological sequences)合并一起. 为实现有效性, 要求序列不能包含大于3%的错误, 因为这样的错误常诱发假的OTUs. 同时, 根据对应正确序列是否靠近cluster的中心或边缘, 具有小于3%的错误的序列也能带来假的OTU.

My definition of OTUs

在USEARCH, 算法设计用于报告具有正确生物序列的OTUs.

UPARSE OTUs

在[UPARSE-OTU][https://www.drive5.com/usearch/manual/uparseotu_algo.html]算法([cluster_otus][https://www.drive5.com/usearch/manual/cmd_cluster_otus.html]), 其目的在于报告正确生物序列(correct biological sequences)的子集, 例如所有OTUs相互都小于97%的一致性, 同时OTU序列是其内最丰富的序列. 这可以解释为Sneath-Sokal OTUs, 针对species给定operational approximation

Denoised OTUs(ZOTUs) 根据UNOISE(unoise3), 其目的在于报告reads中所有正确生物序列(correct biological sequences). 去噪音后称之为zero-radius OTUs, ZOTUs. 存在某些species由于其species内的多样性其reads将会分为多个不同的ZOTUs的情况(differences between paralogs and differences between strains). **ZOTUs的优势在于根据97%的OTUs, 使得具有潜在差异表型的近源的strains能够被划分到相同的cluster.**

OTU abundances

根据这里定义, OTUs为序列, 并不是clusters. 传统的OTU分析常基于[OTU table][https://www.drive5.com/usearch/manual/otu_table.html], 需要每个样本中的每一OTU都具有丰度或频率. 这里通过计算reads数目来得到丰度, 但是需要强调的是, read丰度和species丰度仅有很低的一致性, 同时这些丰度的生物学解释不清晰.

针对去噪音后的序列, 一个ZOTU的丰度为来源唯一的序列(去重)的reads数目, 包含正确的和包含错误的reads.

UPARSE使用97%的OTUs, 一个OTU的丰度就表示为所有来源的大于或等于该OTU序列97%的一致性的reads的数目总和, 包含正确的和具有错误的reads. 该定义并不充分, 因为一个read能满足两个或多个OTUs. 假如存在多个选择, read则分配到最高相似度的OTU中(并不是最高丰度), 因为具有最高相似性的OTU更倾向属于相同species. 假如根据其最高相似度该选择仍然模糊于两个或多个OTUs中, 那么就选选择最先出现在数据中的OTU, 该过程保证了给定strain的reads能够一致性地分配到相同的OTU中.

在实际分析过程中, 对于OTUs和ZOTUs, 使用`otutab`命令在满足97%一致性的条件下计算比对的reads数目. 针对97% OTUs, 使用97%的一致性阈值是self-explanatory, 针对ZOTUs, 其目的在于允许测序或PCR存在3%的错误率.

FAQ: Should you use UPARSE or UNOISE?

两个方法都可以构建OTUs: 97%的clustering和denoising

UPARSE算法构建97% OTUs; UNOISE算法去噪音, 例如矫正错误

如果UPARSE运行完好, 将针对你的reads给出一套正确的生物学序列, 其中任意两个序列比较都大于97%的一致性. 运行命令为`cluster_otu`

如果UNOISE运行完好, 将会给出reads中所有正确的生物学序列. 运行命令为`unoise3`

运行UPARSE和UNOISE的流程是一样的, 唯一不同就是聚类步骤采用cluster_otu还是采用unoise3

建议同时采用两种处理过程, 因为两种过程获得的OTU表格都可以接着进行分析多样性, 无论是通过cluster_otu还是unoise3获得的. 若获得的生物学结论有差异, 那么应该考虑哪一个结论是正确的, 为什么会发生; 若连个结论一致, 则进一步证实分析结论.

97%的clustering的主要缺陷是, 将会舍弃存在reads中一些正确生物学序列. 加入这些代表性的strains或species拥有差异表型, 那么通过97%的聚类将会丢失相关信息, 对应的reads将会汇聚到包含多个表型的OTU中

denoising的主要缺陷是, species在不同个体间常具有差异, 并且paralogs常常不是100%一致. 另一个缺陷是, 更多低丰度的序列将会丢失: UPARSE舍弃单个唯一的序列, 而UNOISE舍弃丰度小于8的唯一序列. 针对一般研究, 该过程不会带来太多差异, 因为样本是混到一起的, 如果一个序列丰度小于8, 那么在一些样本中可能就是单独唯一序列.

denoising的主要优势是, 通过保留所有生物学序列而获得更佳的分辨率. 加入测序的speices中含有species内的变异, 那么针对一个species将会获得两个或更多的OTUs. 它可能有助于检出具有不同表型的strains, 因此若非要推荐一个方法, 那就是denoising.

Interpreting counts and frequencies in OTUs

Estimating microbial diversity

单个样本的多样性([alpha diversity][https://www.drive5.com/usearch/manual/alpha_diversity.html])通常使用metric例如Shannon index和Chao1 estimator计算, 然而成对样本之间的差异([beta diversity][https://www.drive5.com/usearch/manual/beta_diversity.html])则使用来自Jaccard distance或Bray-Curtis dissimilarity计算的metric表示. 这些metric都是通过OTU频率计算而来的, 例如 unweighted UniFrac(called unifrac_binary in usearch)使用presence/absence only, 在存在任何非0的值时将一个count记做1.

OTU frequency dose not correlate with species frequency

实际上, [OTU][https://www.drive5.com/usearch/manual/read_cell_correl.html]频率和species的频率相关性很低. 这就是说, 例如丰度最高的OTU并不包含最丰富的species

Cross-talk degrades presence/absence

有些多样性metric使用OTU的存在与否, 而不是其频率. 在usearch分析中, 这样的metric称之为'binary', 因为其count值会被考虑为0或1. 针对扩增子reads, 由于样本是多重的, 存在与否并不能够真实测量, 因为[cross-talk][https://www.drive5.com/usearch/manual/crosstalk.html]常导致reads不能被真实的分配到样本(假如该OTU不存在时). 当样本来自不同的环境时, 该问题尤为严重(例如, 人类肠道或老鼠肠道).

singleton counts are especially suspect

假如遵循这里的推荐过程, 就会将来自所有样本的reads放到一起同时去除掉[singleton unique][https://www.drive5.com/usearch/manual/singletons.html]序列, 构建97%的OTUs, 同时舍弃掉丰度小于8的unique序列来构建ZOTUs(denoising). 即使这样, 许多OTU表格内的项常为singleton(值为1), 因为所有的count被分配到了多个样本. 小的counts更可能是错误的, 尤其是singletons, 可能是OTU自身错误(例如为检测到的chimera)或由于cross-talk(在拆分index时, 将read分到了错误的样本中).

Misop tutorial: mouse gut and mock community

1. 组装成对reads, 将样本名称加入reads标签

`usearch -fastq_mergepairs`

`-reverse` 反向fastq文件名, 若未指定, 自动使用R2替换R1来指定

`-fastqout/-fastaout` 输出合并后的fastq/fasta文件

`-fastqout_notmerged_fwd/-fastaout_notmerged_rev...` 输出没有合并的fastq/fasta的fwd/rev文件名称

`-report/-tabbedout/-alnout` 输出summary/处理过程的表格文本记录/可读的比对文件

`-relabel @` 使用fastq文件名称前缀作为合并后序列识别符, 第一个下划线前名称

`-fastq_eeout` 添加ee=xxx, 表示合并后序列期待的错误数目

`-fastq_maxdiffs` 最大比对差异数目, 默认为5; 加入重叠数目更长可增加该值

`-fastq_pctid` 最小的一致性比对, 默认为90, 同样在更长重叠时降低该值

`-fastq_minmergelen/-fastq_maxmergelen` 最小合并序列长度

`-fastq_minqual` 若合并的任何Q值小于该值则舍弃该合并后的reads, 默认无

`-fastq_monovlen` 指定最短的合并长度, 默认为16

```
usearch -fastq_mergepairs *_R1_*.fastq -relabel @ -fastqout merged.fq
```

2. 质量过滤, 舍弃可能含有错误的reads

`usearch -fastq_filter`

`-fastqout/-fastqout` fastq/fasta文件输出, 可同时使用

`-fastqout_discarded/-fastaout_discarded` 过滤掉的fastq/fasta文件输出

`-relabel` 输出序列重命名标签, @表示使用第一个下划线前的名称: Mock_S188_L001_R1_oo1.fastq 为 Mock.1...

`-fastq_eeout` 根据Q值标记上期待错误数目: ee=xx

`-fastq_truncqual N` 循环截去第一个位置低于或等于N值的碱基, 使得剩余Q值大于N

`-fastq_maxee N` 在完成截短选项后, 去除含有大于E个期待错误的reads

`-fastq_trunclen L` 在L位置截去碱基, 若序列比L短, 舍弃

`-fastq_minlen L` 舍弃短于L的序列

`-fastq_stripleft N` 删除read中first N bases

`-fastq_maxee_rate E` 在完成截短选项后, 舍弃具有大与E平均错误率的reads: E 0.01, 100bp

, 若期待错误大与1,舍弃

`fastq_maxns k` 若read中包含大与k个N, 舍弃

`usearch -fastq_filter merged.fq -fastq_maxee 1.0 -relabel Filt -fastaout filtered.fa`

3. 识别输入序列中的唯一序列, 又称为去重(dereplication)

`usearch -fastx_uniques`

`-fastaout/-fastqout` 指定唯一序列的fasta文件输出, 根据降低的丰度对序列排序

`-sizeout` 指定输出文件标签加上[size annotations][https://drive5.com/usearch/manual/size_annot.html], 用于表示代表性序列的cluster sizes: size=N

`-relabel` 重命名去重复后的序列: Uniq, Uniq1, Uniq2...

`-minuniquesize` 设置最小的丰度, 具有小于该丰度的唯一序列舍弃, 默认为1

可通过设置`-strand both`来满足反向互补匹配

`usearch -fastx_uniques filtered.fa -sizeout -relabel Uniq -fastaout uniques.fa`

4. 使用[UPARSE-OTU][https://drive5.com/usearch/manual/uparseotu_algo.html]算法构建97%OTUs

UPARSE-OUT的目的在于识别一套OTU代表性序列(在输入序列序列中识别一套子集), 满足一下标准:

- 任意两个OTU序列满足小于97%的一致性
- OTU序列应该是在一个97%一致性的序列中是最丰富的序列
- 舍弃嵌合体序列
- 所有非嵌合体序列应在满足大与等于97%一致性的条件下匹配至少一个OTU

UPARSE-OUT输入序列为一套使用整数标记了丰度的序列(经过质量过滤, 整体过滤和去重), 在实际中为唯一序列所代表的reads数目, 也可以是去噪音步骤后预测的丰度值

`usearch -cluster_otus`

通过该命令过滤掉嵌合体序列. 该嵌合体过滤优于使用UCHIME, 因此不推荐使用参考基因组的嵌合体过滤作为后处理过程(as a post-processing step), 因为容易出现假阳性

针对大多数目的, 推荐使用97%的OTU聚类. 使用[unoise][https://drive5.com/usearch/manual/cmd_unoise3.html]命令来恢复reads中的全套生物学序列会更好. 这时也存在有效的OTUs, 称之为ZOTUs, zero-radius OTUs.

`-minsize` 用于指定最小的丰度, 默认为2, 这会去除单个唯一的序列([singeleton][https://drive5.com/usearch/manual/singletons.html], 仅存在一次)

`-otus` 指定输出fasta文件为OTU代表性序列

`-relabel` 重标记

`-uparseout/-uparsealnout` 指定输出tabbed文本文件描述输入文集如何被分类/比对情况

`usearch -cluster_otus uniques.fa -relabel Otu -otus otus.fa -uparseout otu.reprot`

```
60 Uniq60;size=186;        noisy_chimera   dqt=13;top=Otu48(94.9%);dqm=7;div=2.4;segs=2;parents=Otu48(1-116/6)+Otu15(117-253/1);
61 Uniq61;size=185;        otu50   dqt=10;top=Otu42(96.0%);
62 Uniq62;size=182;        otu51   dqt=28;
63 Uniq63;size=178;        otu52   dqt=55;
64 Uniq64;size=176;        otu53   dqt=17;top=Otu51(93.3%);
65 Uniq65;size=174;        otu54   dqt=16;top=Otu35(93.7%);
66 Uniq66;size=171;        match   dqt=7;top=Otu54(97.2%);
```

dqt: 差异碱基数目; parent: 嵌合体位置

5. 通过去噪音(error-correction)构建ZOTUs

`usearch unoise3`

使用UNOISE算法执行去噪音(error-correction)过程, 通过一下步骤矫正错误:

- 识别reads中的测序错误并矫正
- 去除嵌合体序列

输入为一套过滤后的唯一read序列(size=nn, 丰度注释), 同时按照丰度降序排序, 该算法用于illumina reads, 不能用于454, Ion Torrent或PacBio reads

`-zotus` 指定预测的正确的生物序列fasta文件输出, 标签为Zotunnn, 1,2,3...

`-ampout` 指定预测的正确的扩增序列fasta文件输出, 这些序列包含了嵌合体, 因此该输出文件一般不用于输出流程

`-minsize` 指定最小的丰度(size=annotation), 默认为8. 大部分低丰度输入序列为噪音序列, 无法通过[otutab][https://drive5.com/usearch/manual/cmd_otutab.html]命令比对到ZOTU, 针对更高的敏感度, 可降低该值为4, 尤其是样本单独去噪音而不是f混到一起去噪音. 当然, 在预测的低丰度生物序列中倾向存在更多错误

`-tabbedout` 指定输出每个序列的处理过程, 例如分类为噪音或嵌合体

`-unoise_alpha` 指定alpha参数

`usearch -unoise3 uniques.fa -zotus zotus.fa -tabbedout zotus.table`

```
541 Uniq541;size=8;  denoise amp393
542 Uniq542;size=8;  denoise amp394
543 Uniq543;size=8;  denoise amp395
544 Uniq544;size=8;  denoise bad     dqt=1;top=Uniq1;
545 Uniq545;size=8;  denoise bad     dqt=2;top=Uniq23;
546 Uniq546;size=8;  denoise bad     dqt=1;top=Uniq9;
547 Uniq547;size=8;  denoise bad     dqt=3;top=Uniq1;
548 Uniq548;size=8;  denoise bad     dqt=1;top=Uniq20;
549 Uniq1;size=6641;         chfilter        zotu
550 Uniq2;size=6526;         chfilter        zotu
```

```
631 Uniq83;size=121;        chfilter        chimera dqm=0;dqt=3;div=1.2;top=(L);parentL=Amp1;parentR=Amp2;
632 Uniq84;size=117;        chfilter        zotu
633 Uniq85;size=117;        chfilter        zotu
634 Uniq86;size=115;        chfilter        zotu
635 Uniq87;size=114;        chfilter        zotu
636 Uniq88;size=113;        chfilter        zotu
```

6. 针对97% OTUs构建OTU表格

通过比对reads到OTUs构建OTU表格, 推荐在生成该表格后使用[otutab_rare][http://www.drive5.com/usearch/manual/cmd_otutab_rare.html]命令根据相同数目的reads对所有样本进行标准化(normalize all samples to the same number of reads)

`usearch -otutab`

查询数据集可为fastq或fasta格式, 每个序列必须标记样本识别符号([sample identifier][https://drive5.com/usearch/manual/upp_labels_sample.html], fasta: >name; fastq: @name), 查询序列一般为原始reads, 例如经过合并后的, 如果可行, 也可使用过滤前的reads, 低质量的reads和singleton 也常能够比对到一个OTU.

OTU数据集为一套OTU序列或ZOTU序列(denoised sequences). 数据集必须使用OTU识别符号标记([OTU identifiers][https://drive5.com/usearch/manual/upp_labels_otus.html]), 该数据文件通过`-otus`或`-zotus`选项指定

`-id` 设置最小的一致性比例, 默认为0.97, 对应97%的一致性. 去噪音后的OTUs也使用97%一致性阈值来允许测序或pcr错误.

`-strand both` 来指定搜索双链, 默认假设reads和OTU序列链方向相同

`-mapout` 指定tabbed文本文件, 包含两部分内容, 1, read label; 2, OTU label

`-notmatched` 指定输出包含没有分配到OTU的fasta序列的文件名称

`-notmatchedfq` 指定输出包含没有分配到OTU的fastq序列的文件名称

```
usearch -otutab merged.fa -otus otus.fa -strand plus -otutabout otutab.txt
```

7. 对ZOTUs构建OTU表格

```
usearch -otutab merged.fa -zotus zotus.fa -strand plus -otutabout zotutab.txt
```

8. [otutab_rare][http://www.drive5.com/usearch/manual/cmd_otutab_rare.html] command

该命令对一个OTU表格进行标准化处理, 使得每个样本在不重复取样情况下使用固定的随机数目的reads来重塑OTU表格. 该策略相对于已经舍弃的方法otutab_norm, 更准确地保留了每个样本的丰度分布. 推荐使用命令来标准化样本, 使之互相之间可以比较.

输入和输出都为[QIIME classic format][http://www.drive5.com/usearch/manual/qiime_classic.html], 使用`-sample_size`指定每个样本的reads数目, 没有默认值, 含有小于该sample_size read的样本舍弃.

尽可能保留多的count, 同时尽可能舍弃少的sample

使用`-randseed`指定随机数目seed, 该命令用于获得重复性结果

```
usearch -otutab_rare otutab.txt -sample_size 5000 -output otutab_5k.txt
```

Diversity

> Diversity analysis investigates questions such as "how many species are in a sample?" and "how similar are these two samples?". The diversity in a single sample is called alpha diversity, and the diversity (differences or similarities) between two samples is called beta diversity.

OTU table

OTU中的read counts常被认为在生态中观察到的物种的情况. BIOM格式更加复杂, 但是罕有实际用途. 可使用[otutab2biom][http://www.drive5.com/usearch/manual/cmd_otutab2biom.html]命令构建BIOM文件, 但是QIIME不支持. QIIME classic format:

#OTU ID	F3D0	F3D141	F3D142	F3D143	F3D144	F3D145	F3D146	F3D147
OTU_6	749	535	313	372	607	849	493	2025
OTU_25	29	57	14	2	14	22	16	127
OTU_1	613	497	312	247	472	719	349	1720
OTU_8	426	378	255	237	382	627	330	1417
OTU_31	149	38	10	19	25	21	43	31
OTU_2	366	392	327	185	313	542	248	1367
OTU_7	196	370	92	107	48	155	74	105
OTU_10	46	169	87	109	171	209	120	864
OTU_80	26	6	0	1	4	8	18	11

[Estimating microbial diversity]
[http://www.drive5.com/usearch/manual/otu_count_interpret.html]

单个样本的多样性(alpha diversity)一般使用Shannon index和the Chao1 estimator矩阵来测量, 然而成对样本之间的变异(beta diversity)一般使用Jaccard distance或Bray-Curtis dissimilarity矩阵表示. 许多此类metric, 包括Shannon, Chao1, Jaccard和Bray-Curtis, 都是从OTU频率计算而来的. 其他metric, 例如, unweighted UniFrac(成为unifrac_binary in usearch)仅使用存在/不存在来描述情况.

OTU frequency does not correlate with species frequency

实际上, OTU frequencies have low correlation with species frequencies. 这意味着, 丰度最高的OTU, 常并不包含丰度最高的species.

Cross-talk degrades presence/absence

由于cross-talk, 来自多重index的样本, 使用presence/absence时, 不能可信地表示样本情况, 因为reads可能错误地分配到一个样本

Singleton counts are especially suspect

即使使用97%的OTU或ZOTU, 很多OTU的也常为 singleton, 因为总的count分配到了多个样本中. 小的count更可能是假的, 尤其是singletons, 或者因为OTU本身就是假的, 或着由于cross-talk

Traditonal diversity metrics are invalid or hard to interptet

根据上面描述的问题, 根据OTU计算而来的metric, 其中很多多样性metric是无效的, 没有意义的或很难解释的.

Recommended alpha and beta diversity metrics

Alpha diversity metric

对于扩增子序列, presence/absence尤其半信半疑的结果, 因此使用presence/absence的metric, richness和unweighted UniFrac不应该使用来表示多样性改变. 小的counts相对于高的counts更不可信, 因为它们可能是由于cross-talk或假的OTUs带来的. 因为, 推荐使用具有矩阵with put lower weight on low-frequency OTUs, 例如Shannon entropy 或 Simpson. 使用Simpson时, 其值可能被单个高频率的OTU所主导, 同时这也值得怀疑, 因为高丰度可能是由于偏差(例如, 有些species拥有10个16S拷贝,而其他可能仅有1个或2个拷贝). 相反, 假如没有高丰度的OTU, 这也可能是由于人为错误导致(例如, 样本实际上被单个specie所主导, 同时其引物包含了错误, 其PCR过程很大程度上被抑制了).

这里建议一般使用Shannon entropy作为alpha多样性变化的标准, 即使没有偏差其差异也是很难解释. Shannon entropy的改变可以是因为OTUs的数目的改变, 频率分布形状的改变, 或者两者共同的改变. 其差异的改变可通过查看每个group的频率分布的理解.

Beta diversity metric

UniFrac 作者声称: justification for its tree-based design is that OTUs with higher sequence similarity should be treated as more similar when comparing samples bacause they then to fill similar ecological niches.

QIIME生成的OTUs, 一般是含有很多噪音的, 有必要使用UniFrac来抑制由于噪音带来的差异. 然而, 针对UPARSE或UNOISE生成的更准确的OTUs, 这里认为UniFrac对于是beta多样性分析是一个坏的选择, 因为针对高序列相似度的OTUs而言, 其分辨率很低. 即使OTUs满足类似的niches, 这样的差异也缺失具有生物显著性的. 这里, 推荐使用weighted Jaccard矩阵.

Alpha diversity

Alpha多样性表示为单个生态环境或样本中的多样性. 最近的测量标准为richness, species(或OTUs)在一个样本中的数目. 其他metric考虑OTUs的丰度(频率), 例如, 对于较低丰度的OTUs给予较低的权重. 其丰度分布可使用[octave plot][http://www.drive5.com/usearch/manual/octave_plot.html]查看.

Interpretation

牢记一点很重要, [NGS扩增子测序不能可信的描述OTUs的频率或存在与否][http://www.drive5.com/usearch/manual/otu_count_interpret.html], 因此针对传统生物学而开发的alpha多样性的生物学意义是不明晰/错误/难以解释.

Estimators

因为有些罕有的species无法在数据中观察到. alpha多样性estimator用于通过观察到的reads来推测community中specie的数目. 最为人知NGS OTUs的estimator为Chao1. 这里的观点, 如果使用丰度阈值, estimators不能实际地应用到NGS OTUs中, 因为罕有species会被过滤掉.

Rarefaction

Rarefaction的目的在于获得一个指示, 是否获得了足够的reads从而得到一个好的alpha多样性评估metric. 通过构建[Rarefaction curve][http://www.drive5.com/usearch/manual/rare.html]来展示通过增加reads给meric带来的改变. 如果曲线汇聚到一个水平的渐近线, 表明进一步增加观测(reads)对当前metric没有什么影响.

Units of measurement

让人困惑的是, alpha多样性metric常使用不同单位. 有时其意义不明显(entropy!?), 同时不同单位的metric无法互相比较.

Effective number of OTUs

使用不熟悉单位的metric, 不能通过转换成[effective number of species][http://www.drive5.com/usearch/manual/eff_nr_species.html]来解释和比较

alpha_div command

该命令用于计算OTU表格中的每个样本的alpha多样性metric. 其OTU表格必须为QIIME classic format

同时可食用[alpha_div_sig][http://www.drive5.com/usearch/manual/cmd_alpha_div_sig.html]命令计算不同groups间metric的统计显著性改变

`-metric`选项指定一个或多个metric名称(逗号分隔). 默认为使用多个流行的metric计算

`-output`指定输出表格, 行为样本名称, 列为metric内容

计算默认metric

```
usearch -alpha_div otutable.txt -output alpha.txt
```

计算Gini-Simpson index

```
usearch -alpha_div otutable.txt -output gini.txt -metrics mini_simpson
```

计算Chao1 and Berger-Parker indexes

```
usearch -alpha_div otutable.txt -output alpha.txt -metrics chao1, barger_parker
```

alpha_div_rare command

alpha_div_rare命令用于计算alpha多样性metric的rarefraction curve

需要注意的是, 该命令当前并不用于舍弃sigletons. 其实际作用值得怀疑, 因为其他来源的错误可能更重要, 所以rarefraction分析对于标志基因OTUs的作用尚不明确.

输入OTU表格为QIIME classic format

`-metric`选项指定metric名称, 默认为richness. 指定metric通过从OTU表格中随机抽取样本来计算, 依次为1%, 2%...99%, 100%的reads.

`-method` 指定采用的subsampling方法, 默认为: [fast subsampling]
[http://www.drive5.com/usearch/manual/cmd_otutab_rare.html]

`-output` 指定输出文件tabbed 文本文件

```
usearch -alpha_div_rare otutab.txt -output rare.txt
```

alpha_div_sig command

该命令用于计算一个或多个alpha diversity metrics groups(healthy/sick)之间差异的统计显著性.

`-meta`选项指定[metadata file][http://www.drive5.com/usearch/manual/metadata_file.html], 该选项必须

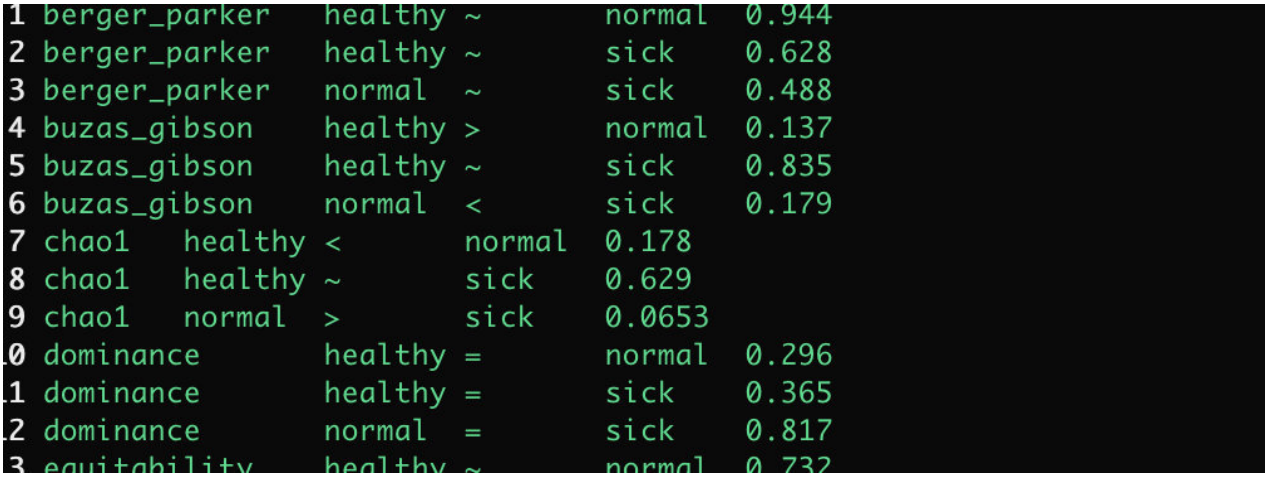
Example

```
SampleA    Sick
SampleB    Sick
SampleC    Healthy
```

`-metrics`指定一个或多个metric名称, 以逗号分隔, 默认为所有metrics

注意, 计算多重metric显著性需要[multiple test correction]
[https://en.wikipedia.org/wiki/Bonferroni_correction]

`-tabbedout` 指定tabbed文本输出文件



```
1 berger_parker   healthy ~       normal  0.944
2 berger_parker   healthy ~       sick    0.628
3 berger_parker   normal  ~       sick    0.488
4 buzas_gibson    healthy >       normal  0.137
5 buzas_gibson    healthy ~       sick    0.835
6 buzas_gibson    normal  <       sick    0.179
7 chao1   healthy <       normal  0.178
8 chao1   healthy ~       sick    0.629
9 chao1   normal  >       sick    0.0653
0 dominance       healthy =       normal  0.296
1 dominance       healthy =       sick    0.365
2 dominance       normal  =       sick    0.817
3 equitability    healthy ~       normal  0.732
```

第一列为metric名称; 第二列metadata分类; 第三列为显著性符号; 第四列为metadata分类; 第五个为P-value

第三列显著性符号意义:

```
<   Metric has lower value for samples in first category, weak significance (P < 0.2)).
<< Metric has lower value for samples in first category, high significant (P < 0.05).
>   Metric has higher value for samples in first category, weak signifiance (P < 0.2).
>> Metric has higher value for samples in first category, high significance (P < 0.05).
~,= Metric is approximately equal in both groups.
```

计算Gini-Simpson index差异显著性

```
usearch -alpah_div_sig otutable.txt -meta meta.txt -tabbedout sig.txt -metrics gini_simpson
```

计算Chao1和Berger-Parker indexes的差异显著性

```
usearch -alpha_div_sig otutable.txt -meta meta.txt -atbbedout sig.txt -metrics chao1, berger_parker
```

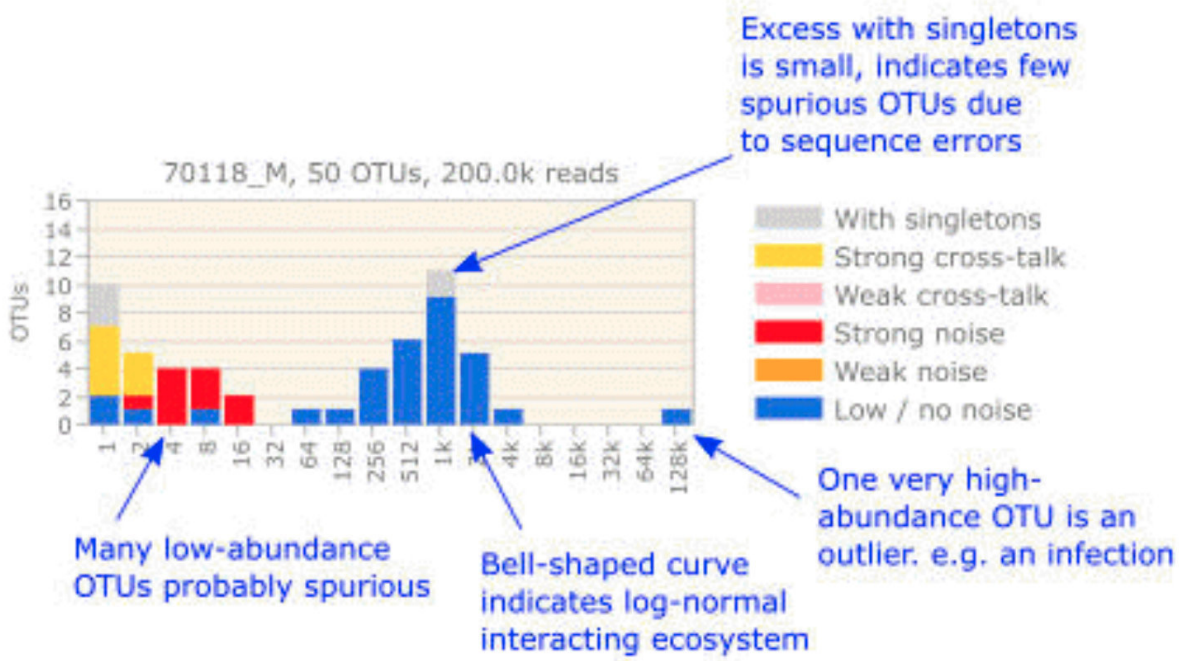
计算所有metric的差异显著性

```
usearch -alpha_div_sig otutable.txt -meta meta.txt -tabbedout sig.txt
```

Octave plots for visualizing alpha diversity

Octave图是用于展示一个或一组样本的OTU丰度分布的直方图, 该方法用于可视查看alpha 多样性分布. 丰度按照OTUs的数目来分不到不同的bins中. 每个bin定义为一定范围的丰度, 且每个bin所包含OTUs范围数目是前一个bin的两倍数目. 例如, 第一bin为丰度为singleton的OTU数目, 第二个bin为丰度为2和3的OTUs数目, 第三个bin为丰度为4到7的OTUs的数目. 这种分布确保了logarithmic scale, bins在图上均匀分布且拥有相同大小.

因此X轴坐标值具有两个意义: 最小丰度值和丰度范围



默认由于cross-talk导致的OTU错误可通过颜色指明出来. Cross-talk颜色通过[UNCROSS2][http://www.drive5.com/usearch/manual/uncross2_algo.html]值来决定, 可指定`-noxtalk`来关闭该值

若提供了OTUs的[distance matrix][http://www.drive5.com/usearch/manual/distmx.html], 那么低丰度的OTUs, 由于非常类似的dominant OTU, 可能通过颜色指示为假的OTU. 该distance matrix可通过[calc_distmx][http://www.drive5.com/usearch/manual/cmd_calc_distmx.html]命令使用OTU序列fasta文件生成.

次级OTU表格可通过`-otutab_with_singles`选项指定. 该表格应使用和第一个OTU表格相同的步骤完成, 除了保留singleton unique sequences但第一个OTU表格舍弃该类序列. 这意味着第一个表格(singletons discarded, otutab_octave选项参数)给出了每个bin中较低的OTUs数目, 第二个表格(singletons included, otutab_with_singles选项参数)给出了每个bin中较高的OTUs数目. 因此, 真实的多样性应该存在于这两者之间. 根据singletons获得的过量多样性由灰色颜色表明.

`-htmlout/-svgout` 输出格式

```
usearch -otutab_octave otutab.txt -distmxin distmx.txt -otutab_with_singles otutab_singles.txt -htmlout octave.html -svgout octave.svg
```

calc_distmx command

根据输入的fasta/fastq格式文件计算[distance matrix][http://www.drive5.com/usearch/manual/distmx.html], 通过`-tabbedout`选项指定输出名

distance value范围从0到1(一致性到没有相似性), 同时Clusters可通过cluster_aggd命令从distance matrix生成. 可设置`-distmx_brute`选项使得所有成对序列进行比对计算. 序列之间的距离计算为:1-[fractional identity][http://www.drive5.com/usearch/manual/identity.html]

USEARCH, BLAST and CD-HIT definitions of %id
An "identity" is a column with two matching letters.

USEARCH and BLAST = Identities / Columns

CD-HIT = Identities / (Length of shorter sequence)

`-maxdist` 指定最大距离输出范围

`-termdist` 指定终止搜索的一致性阈值, 范围从0.0到1.0; 0.0意味着完全的一致性序列

例如, 如果一对序列的distance>maxdist, 该计算终止. 因为U-sorted order并不能和一致性完美匹配, 因此应设置的termdist值略小于最大距离值(maxdist). 若希望所有大于80%一致性的序列对出现在matrix中, 可设置参数`-maxdist 0.2` , `-termdist 0.3`(80%一致性距离, 70%的一致性终止搜索)

```
usearch -calc_distmx seqs.fa -tabbedout mx.txt -maxdist 0.2 -termdist 0.3
```

```
usearch -calc_distmx seqs.fa -tabbedout dist.tree -format phylip_lower_triangular
```

接上文octave plot

1. 生成包含singleton unique sequences的OTU及OUT table

```
usearch -cluster_otus uniques.fa -relabel Otu -otus otus_singleton.fa -uparseout otu.reprot -minsize 1
```

```
usearch -otutab merged.fa -otus otus_singleton.fa -strand plus -otutabout otutab_singleton.txt
```

2. 计算OUTs.fa的distance matrix

```
usearch -calc_distmx uniques.fa -tabbedout mx.txt -maxdist 0.2 -termdist 0.3
```

3. 生成ocatve plots

```
usearch -otutab_octave otutab.txt -distmxin distmx.txt -otutab_with_singles otutab_singleton.txt -htmlout octave.html -svgout octave.svg
```

Beta diversity

beta多样性比较两个样本. 一般而言, 通过计算得到一数值来表明样本间的相似性或差异性. 该数值范围从0到1.

一组样本之间的成对比较可以通过distance matrix展示, 在usearch分析中, beta多样性常为差异测量, 而不是相似性测量值, 因此, 增加值表明更低的相似性和更远的距离.

可通过`beta_div`命令将相似性样本聚集, 自动生成树结构图; 或者通过`cluster_aggd`命令聚集由`beta_div`或其他第三方软件生成的distance matrix来获得样本树状结构图(clac_distmx for sequences/beta_div for OTU table/cluster_aggd for tree from the distance matrix).

`beta_div`命令通过一个OTU表格计算一个或多个beta diversity metrics.

`-metrics` 指定逗号分隔的一个或多个metrics, 默认使用所有支持的metric. 针对每个metric, 生成3个输出文件: distance matrix/sorted distance matrix/tree

Metric	Max value	Cluster	Description
bray_curtis	1	Y	Bray-Curtis
bray_curtis_binary	1	Y	Bray-Curtis
euclidean	(no maximum)	N	Euclidean distance
jaccard	1	Y	Jaccard coefficient
jaccard_binary	1	Y	Jaccard coefficent
manhatten	(no maximum)	N	Manhatten distance
unifrac	1	Y	UniFrac
unifrac_binary	1	Y	UniFrac

`-filename_prefix` 指定输出文件名前缀, 一般为目录名称, 以slash或backslash结尾, 且该目录需已经存在

`-mx_suffix/-sorted_mx_suffix/-tree_suffix` 指定后缀名称; 默认为.txt/.sorted.txt/.tree

计算所有支持的beta metrics并输出到目录'/results/beta'

```
usearch -beta_div otutable.txt -filename_prefix /results/beta
```

计算Jaccard metric且输出当当前目录

```
usearch -beta_div otutable.txt -metrics jaccard
```

Taxonomy

Recommended taxonomy databases

使用小的权威的分类数据库, 推荐使用权威的分类序列, 例如最近的RDP训练集或LTP发布的16S数据库; 大的数据库的分类注释信息不够可信, 例如, SILVA/Greengenes/full RDP databases, 大部分是由16S序列预测而来的. 其中大概5分之1是错误的, probably because the guide trees have pervasive branching order errors.

[SINTAX downloads][http://www.drive5.com/usearch/manual/sintax_downloads.html]

FASTA files reformatted with SINTAX-compatible taxonomy annotations.

16S

rdp_16s_v16.fa.gz RDP training set v16 (13k seqs.). RDP license terms.
rdp_16s_v16_sp.fa.gz RDP training set with species names (not recommended) (can species be predicted?).
gg_16s_13.5.fa.gz Greengenes v13.5 (1.2M seqs.). Greengenes license terms. (not recommended)
silva_16s_v123.fa.gz SILVA v123 (1.6M seqs.). SILVA license terms. (not recommended)
ltp_16s_v123.fa.gz SILVA v123 LTP named isolate subset (12k seqs.) .SILVA license terms

ITS

UNITE (current "utax" version at unite.ut.ee) (53k sequences in v7.1). UNITE license terms.
rdp_its_v2.fa.gz RDP Warcup training set v2 (18k sequences). RDP license terms.

18S

silva_18s_v123.fa.gz SILVA v123 eukaryotic 18S subset (140k seqs.) .SILVA license terms

SINTAX algorithm

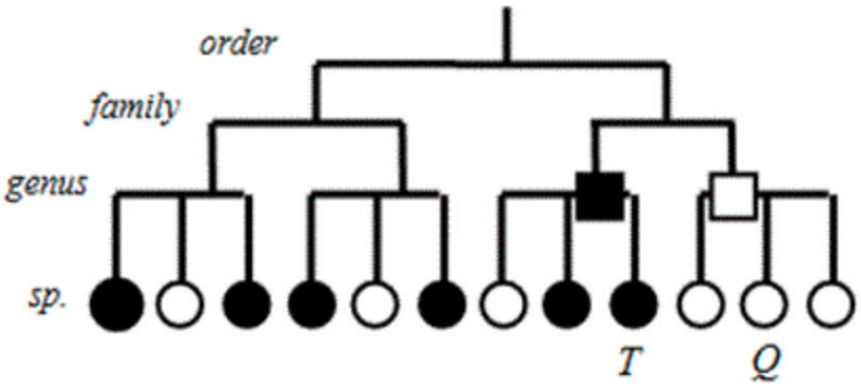
SINTAX算法用于预测标记基因reads的分类信息, 例如16S/ITS. 针对所有预测的ranks, 提供bootstap置信值. 该算法类似[RDP Naive Bayesian Classifier algorithm] [http://www.drive5.com/usearch/manual/nbc_algo.html], 除了使用k-mer相似性来识别top taxonomy, 而不是Bayesian poseriors, 因此无需训练. 不推荐使用SILVA/Greengenes作为分类参考数据库, 因为该数据库拥有高度错误率, 接近1/5的分类注释都是错误.

[Can SINTAX predict species?] [http://www.drive5.com/usearch/manual/sintax_species.html]

根据RDP training set with species names, 若使用`sintax`命令, 将会获得具有bootstrap置信度的species预测结果. 同时应对预测结果持怀疑态度, 以为可能, 也是常见的, 获得一个具有高bootstap值的不正确的预测结果, 该结果也会发生在genus,但是speices上最常见.

若使用短的标签例如V4, 则很容易出现两个或多个species拥有一样的标签序列, 那么就不可能识别species. 但是, 该清醒也不会根据当前数据库检测到, 因为大量species都没有被命名也就不可能出现在RDP training set中.

若根据sparse reference数据库, 使用'top-hit' 分类软件例如SINTAX/RDP, 那么可能遇到over-classification的问题. 例如, 假如top hit为95%的一致性, 那么可能属于不同的species, second-best hit较低, 90%的一致性, 那么SINTAX/RDP的bootstrapping重复选取query序列中一个sub-sample of words(8-mers), 在仅考虑该subsample情况下检测top taxonomy...



A sparse reference database induces over-classification errors. *Q* is the query and *T* is the top hit. Black circles denote known species, white circles novel species. If *Q* belongs to a novel genus (white square), the top few hits will tend to belong to the most similar known genus (black square). Here, SINTAX will tend to give a high bootstrap confidence to the known genus. Other algorithms which explicitly (or effectively) take a consensus of taxonomies of the most similar reference sequences, such as RDP, will similarly tend to make over-classification errors.

[sintax command][http://www.drive5.com/usearch/manual/cmd_sintax.html]

sintax命令使用SINTAX算法预测fasta/fastq格式的query序列的分类注释信息, 使用命令`-sintax_summary`命令获得tabbed文本用于作图

搜索的数据库必须含有[taxonomy annotations][http://www.drive5.com/usearch/manual/tax_annot.html], 使用命令`makeudb_sintax`命令也可用于构建[UDB database][http://www.drive5.com/usearch/manual/udb_files.html]

taxonomy annotation指定参考数据库中的序列的taxonomy名称:

```
>AB008314;tax=d:Bacteria,p:Firmicutes,c:Bacilli,o:Lactobacillales,
  f:Streptococcaceae,g:Streptococcus;
```

名称前的单个letter指定其taxonomic level: k:Kingdom, d:Domain, p:Phylum, c:Class, o:Order, f:Family, g:Genus, s:Species

UDB文件为包含序列和其k-mer index的数据文件. UDB文件由`makeudb_usearch/makeudb_ublast/makeudb_sintax`命令构建, `udb2fasta`命令可用于将UDB文件转换为FASTA文件, [**FASTA文件中的序列标签必须包含taxonomy annotaions**][http://www.drive5.com/usearch/manual/cmd_makeudb_sintax.html]

`-makeudb_sintax`现推荐被`-makeudb_usearch`取代; 其index paramters默认设置用于`usearch_global`和`usearch_local`使用, 也可通过指定[index paramters][http://www.drive5.com/usearch/manual/indexing_options.html]来取代默认设置

```
usearch -makeudb_usearch rdp_16s.fa -output rep_16s.udb
```

或者

```
usearch -makeudb_usearch db.fasta -output db.udb -wordlength 12 -dbstep 4
```

`-tabbedout`指定输出文件名, 文件前三列为query sequence label/predicted bootstrap value/strand, 若指定了预测阈值`-sintax_cutoff`, 则保留足够高的置信度ranks(taxonomic level), 再次打印出来. On V4 reads, using a cutoff of 0.8 gives predictions with similar accuracy to RDP at 80% bootstrap cutoff.

`-stand` 必须指定

```
usearch -sintax reads.fastq -db 16s.udb -tabbedout reads.sintax -strand both -sintax_cutoff 0.8
```

或

```
usearch -sintax otus.fa -db rdp_16s.udp -tabbedout otus.sintax.table -strand plus -sintax_cutoff 0.8
```

```
1 Otu4    d:Bacteria(1.0000),p:"Bacteroidetes"(1.0000),c:"Bacteroidia"(1.0000),o:"Bacteroidale
2 Otu1    d:Bacteria(1.0000),p:"Bacteroidetes"(1.0000),c:"Bacteroidia"(1.0000),o:"Bacteroidale
3 Otu3    d:Bacteria(1.0000),p:"Bacteroidetes"(1.0000),c:"Bacteroidia"(0.9500),o:"Bacteroidale
4 Otu5    d:Bacteria(1.0000),p:"Bacteroidetes"(1.0000),c:"Bacteroidia"(1.0000),o:"Bacteroidale
5 Otu6    d:Bacteria(1.0000),p:"Bacteroidetes"(1.0000),c:"Bacteroidia"(0.9600),o:"Bacteroidale
6 Otu7    d:Bacteria(1.0000),p:"Bacteroidetes"(1.0000),c:"Bacteroidia"(0.9800),o:"Bacteroidale
7 Otu2    d:Bacteria(1.0000),p:"Bacteroidetes"(1.0000),c:"Bacteroidia"(0.9600),o:"Bacteroidale
```

通过`sintax_summary`命令输出sintax预测结果的summary report. 输入为`sintax`输出文件, `-rank`指定rank,例如p为phylum; 同时可指定OTU表格文件, 该OTU identifiers需和sintax输出文件名称一致; 若sintx输出文件发现size annotations, 将会用于计算频率(size=1;size=5;...).

```
usearch -sintax_summary sintax.txt -output phylum_summary.txt -rank p
```

或者

```
usearch -sintax_summary otus.sintax.table -output phylum_summary.txt -rank p -otutabin otutab.txt
```

```
     1 Phylum  F3D0    F3D141  F3D142  F3D143  F3D144  F3D145  F3D146  F3D147  F3D148  F3D149  F3D1
     2 Firmicutes      46.8    34.4    28      34.6    30.1    24.7    43.4    28.8    29.6    39.5
     3 "Bacteroidetes" 49.3    63.7    68.6    62.7    67.7    73.6    55.8    69.1    67.1    59.3
     4 (Unassigned)    1.58    1.21    1.51    1.94    0.868   1.02    0.419   1.21    2.13    0.65
     5 "Actinobacteria"        0.444   0.507   1.37    0.463   0.868   0.196   0.195   0.43    0.91
     6 "Proteobacteria"        0.213   0.0483  0.0472  0.0463  0.167   0.261   0.0279  0.0477  0.07
     7 Candidatus_Saccharibacteria     0.0177  0.121   0.142   0.139   0.267   0.0654  0.0837  0.17
     8 Cyanobacteria/Chloroplast       0.0355  0       0.236   0       0       0.0218  0       0.02
     9 "Deinococcus-Thermus"   0       0.0241  0.0944  0.0463  0       0.0436  0       0       0
    10 "Tenericutes"   1.4     0       0.0472  0       0       0.0654  0       0.248   0       0.01
    11 "Verrucomicrobia"       0.177   0       0       0       0       0       0       0       0
phylum_summary.txt lines 1-11/11 (END)
```

Naive Bayesian Classifier algorithm

RDP Naive Bayesian Classifier(NBC)算法是第一个发布用于自动rRNA taxonomy预测方法. 目前已经公布了多个算法, 包括sintax, 但是没有一个算法的准确性能够明确高于NBC.

nbc_tax command

`nbc_tax`预测非常类似于RDP提供的Java预测程序

`-tabbedout`指定输出文件, 前3列为: query sequence label/prediction with bootstrap values/strand, 若使用`-sintax_cutoff`指定了预测阈值, 那么会根据阈值集ranks再次输出三列高置信度信息. 同时也可使用`sintax_summary`命令来生成summary reprot(需指定strand信息).

```
usearch -nbc_tax otus.fa -db rdp_16s_v16.fa -strand plus -tabbedout nbc_tax.table
```

```
usearch -sintax_summary nbc_tax.table -output phylum_summary.txt -rank p -
otutabin otutab.txt
```

Miscellaneous

Validating merged reads to check for problems

[merge reads programs][https://drive5.com/usearch/manual/merge_bench.html]

[trouble-shooting fastq_mergepairs problems]
[https://drive5.com/usearch/manual/merge_troubleshoot.html]

[reviewing a fastq_mergepairs reprot to check for problems]
[https://drive5.com/usearch/manual/merge_report.html]

成对read的组装会出现两种类型错误: 假阴性和假阳性.

Mergeing errors may not matter

若使用UPARSE/UNOISE流程分析, 合并成对reads导致的错误可能无关紧要. 一般而言, 这类错误非常随机因此需要重视的是[singletons and should therefore be discard]
[https://drive5.com/usearch/manual/singletons.html], 且在构建OTUs前处理掉. 错误比对常拥有很多错配, 导致低的Q值出现, 以至于[expected errors]
[https://drive5.com/usearch/manual/exp_errs.html] > 1, 因此错误比对的reads将会在质量过滤阶段去除掉. 值得关注的错误仅在其频率足够高且拥有足够高的Q值, 因此会在带来错误的OTUs, 这时可以通过[quality control on your sequence][https://drive5.com/usearch/manual/exp_errs.html]来确认.

1. False negatives

假阴性为read对具有有效比对, 但是舍弃了. 假如获得了较低比率的合并reads, 同时有理由相信扩增插入序列足够短使得测序得到的成对reads能够重叠, 那么可能就会遇到假阴性问题. 这时就要查看[merge report][https://drive5.com/usearch/manual/merge_report.html]来确认最常见的舍弃reads对的原因. 最简答方法是通过设置`-fastqout_notmerged_fwd`和`-fastqout_notmerged_rev`来检查未能合并reads输出, 然后(若需要)使用`-fastx_subsample`来获得一小部分reads对去检查. 若出现多个不同原因, 可以采用[use the tabbout file][https://drive5.com/usearch/manual/merge_tabbed_check.html]来发现给定原因下合并失败的reads对, 针对这些example pairs来使用不同参数合并.

可使用[ublast][https://drive5.com/usearch/manual/merge_tabbed_check.html]来做比对独立的检测, 或者比对未成对的reads到全场序列的参考数据库(see [trouble-shooting problems with fastq mergepairs][https://drive5.com/usearch/manual/merge_troubleshoot.html] for more discussion).

2. False positives

假阳性由错误的比对导致, 存在两个原因:

a), 该对reads实际上并不重叠, 但是由于R1末端和R2反向互补序列的开端出现错误的相似性导致错误的比对

b), 该对reads不重叠, 但是比对出错(too long or too short because it is offset compared to the correct position). 这在真正的重叠太短时倾向发生.

检查假阳性的方法就是比对合并后的序列到full-length序列的参考数据库. 不正确的比对将会出现可见的gap. 假如成对的reads实际上不重叠, 那么比对将会在合并的序列中出现gap. 但是需要注意, gaps可能是有效的生物学插入或删除. 最清晰的例子就是, 该片段两短也就是gap两短都有很高的一致性, 尤其针对高度保守基因16S.

在OUT流程中, 可以检查那些低丰度的OTUs, 因为问题常发生在这些序列.