USEARCH software and documentation

http://drive5.com/usearch

robert@drive5.com

Version 5.0
August 22nd, 2011

Contents

Introduction 3
UCHIME implementations 3
Reference database mode 3
De novo mode 3
Basic usage 3
Reference database mode 3
De novo mode 3
FASTA output files 4
Alignment output file 4
Tabbed output 5
UCHIME algorithm 6
Diffs and votes 6
UCHIME scoring function 7
Using UCHIME in practice 8
Single-region sequencing 8
Reference database mode 9
Self mode for database screening 10
De novo mode 11
Consistency check 11
Sensitivity vs. specificity 12
Computational efficiency 12
Paired-end reads 12
Parameter tuning 13
Command-line options 14

Introduction

UCHIME implementations

There are two implementations of the UCHIME algorithm: public domain and USEARCH. The USEARCH implementation uses efficient database search algorithms to achieve speeds 10× faster or more compared to the public domain version.

The command-line usage of both versions is very similar. The public domain version allows both --input and --uchime to specify the query file; the USEARCH version requires --uchime. In the following I will use usearch for the binary file name and --uchime as the option to specify the input file.

Reference database mode

The reference database should contain trusted sequences that are believed to be chimera-free. A good reference database for 16S ribosomal RNA genes is available in the Microbiome Utilities provided by the Broad Institute. An alternative could be the chimera.slayer reference database provided for the use in mothur.

De novo mode

In *de novo* mode, input should be estimated amplicon sequences with integer abundances specified using ;size=*N* in the label, e.g.:

```
>FQ23BBGZ5;size=23
```

Abundance is a measure of how many amplicons with a given unique sequence were present in the sample after amplification by PCR. One way to estimate this is to sum the total number of reads in the cluster(s) used to estimate the given amplicon sequence. UCHIME uses only ratios between pairs of abundances, so the absolute value does not matter. However, using the number of reads is an intuitively useful indicator—for example, a cluster containing one read is likely to be spurious. Amplicon sequences and abundances can be estimated using USEARCH (see Denoising in the USEARCH documentation) or by using another algorithm such as Chris Quince's PyroNoise or AmpliconNoise. When using *de novo* mode, sequences should be estimated amplicon from one sequencing run (strictly, one PCR amplification stage), otherwise abundances may not be directly comparable.

Basic usage

Reference database mode

```
usearch --uchime seqs.fasta --db ref.fasta [--uchimeout output.uchime]
  [--uchimealns alnfile] [--chimeras ch.fasta] [--nonchimeras good.fasta]
```

De novo mode

```
usearch --uchime seqs.fasta --uchimeout output.uchime
  [--uchimealns alnfile] [--chimeras ch.fasta] [--nonchimeras good.fasta]
```

FASTA output files

The --chimeras and --nonchimeras options specify FASTA files where sequences are written if they are classified as chimeras or not. In general, perfect classification is not possible so it should be expected that there may be some false positives in the --chimeras file and some false negatives in the --nonchimeras file.

Alignment output file

The --uchimealns file writes chimeric alignments in a human-readable format, and in the example below.

```
Query   (  300 nt) FQZ123GG6
ParentA ( 1527 nt) 7000004131498057/Pseudomonas syringae pv. syringae B728a
ParentB ( 1418 nt) S000364797/Pseudovibrio ascidiaceicola (T)

A   551 TACTGGGCGTAAAGCGCGCGTAGGTGGTTTGTTAAGTTGaATGTGAAATCCCCGGGCTCAACCTGGGAACTGCATCCAAA 630
Q     1 TACTGGGCGTAAAGCGCGCGTAGGTGGTTTGTTAAGTTGTATGTGAAATCCCCGGGCTCAACCTGGGAACTGCATCCAAA 80
B   471 cACTGGGCGTAAAGaGtaCGTAGGcGGacTGaTAAGTTagggGTGAAATCCCaaGGCTCAACCTtGGAACTGCcTttgAt 550
Diffs   A             A AA      A  AA  A      A?AA          AA          A        A AAA A
Votes                 + ++      +  ++  +      +0++          ++          +        + +++ +
Model   AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA

A   631 ACTGGCAagCTaGAGtatGgtAGAGGgtgGTGGAAtTtCctGTGTAGcGGTGAAATgCGTAGATATagGaAgGAACACCA 710
Q    81 ACTGGCAGTCTTGAGATCGAGAGAGGTGAGTGGAACTCCGAGTGTAGAGGTGAAATTCGTAGATATTCGGAAGAACACCA 160
B   551 ACTGtCAGTCTTGAGATCGAGAGAGGTGAGTGGAACTCCGAGTGTAGAGGTGAAATTCGTAGATATTCGGAAGAACACCA 630
Diffs          BB  B   BBB BB     BBB      B B BB      B        B         BB B B
Votes          ++  +   +++ ++     +++      + + ++      +        +         ++ + +
Model   xxxxxBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBB

A   711 GTGGCGAAGgCGaCcacCTGGactGATACTGACaCTGAGGTgCGAAAGCGTGGGGAGCAAACAGGATTAGATACCCTGGT 790
Q   161 GTGGCGAAGACGGCTCACTGGCTCGATACTGACGCTGAGGTACGAAAGCGTGGGGAGCAAACAGGATTAGATACCCTGGT 240
B   631 GTGGCGAAGgCGGCTCACTGGCTCGATACTGACGCTGAGGTACGAAAGCGTGGGGAGCAAACAGGATTAGATACCCTGGT 710
Diffs            N  B BBB    BBB         B       B
Votes            0  + +++    +++         +       +
Model   BBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBB

A   791 AGTCCACGCCGTAAACGATGtcaaCTAGccGTtgGGagcCTTGagcTcTTaGTGgCGCAGC 851
Q   241 AGTCCACGCCGTAAACGATGAATGCTAGTTGTCAGGTAACTTGCTAT-TTGGTGACGCAGC 300
B   711 AGTCCACGCCGTAAACGATGAATGCTAGTTGTCAGGTAgCTTGCTAT-TTGGTGACGCAGC 770
Diffs                       BBBB    BB  BB  BB?    BBB    B   B
Votes                       ++++    ++  ++  ++0    +++    +   +
Model   BBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBB

Ids.  QA 84.0%, QB 92.3%, AB 77.7%, QModel 99.0%, Div. +7.2%
Diffs Left 19: N 0, A 1, Y 18 (94.7%); Right 48: N 0, A 2, Y 46 (95.8%), Score 5.1416
```

Tabbed output

The --uchimeout file is a tab-separated file with the following 17 fields.

Field	Name	Description
1	Score	Value ≥ 0.0, high score means more likely to be a chimera.
2	Query	Sequence label
3	Parent A	Sequence label
4	Parent B	Sequence label
5	IdQM	%id between query and model made from (A, crossover, B)
6	IdQA	%id between query and parent A.
7	IdQB	%id between query and parent B
8	IdAB	%id between parents (A and B).
9	IdQT	%id between query and closest reference sequence / candidate parent.
10	LY	Yes votes on left
11	LN	No votes on left
12	LA	Abstain votes on left
13	RY	Yes votes on right
14	RN	No votes on right
15	RA	Abstain votes on right
16	Div	Divergence ratio, i.e. IdQM - IdQT
17	YN	Y (yes) or N (no) classification as a chimera. Set to Y if score ≥ threshold set by the --minh option and the --mindiv and --mindiffs criteria are also satisfied.

UCHIME algorithm

UCHIME searches for a chimeric alignment between a query sequence (*Q*) and two candidate parents (*A* and *B*), example below.

```
A     81 CCTTGGTAGGCCGtTGCCCTGCCAACTAGCTAATCAGACGCgggtCCATCtcaCACCaccggAgtTTTtcTCaCTgTacc 160
Q     81 CCTTGGTAGGCCGCTGCCCTGCCAACTAGCTAATCAGACGCATCCCCATCCATCACCGATAAATCTTTAATCTCTTTCAG 160
B     81 TCTTGGTgGGCCGtTaCCCcGCCAACaAGCTAATCAGACGCATCCCCATCCATCACCGATAAATCTTTAAaCTCTTTCAG 160
Diffs    A      A     p A   A      A              BBBB     BBB    BBBBB BB   BBa B  B BBB
Votes    +      +     0 +   +      +              ++++     +++    +++++ ++   ++! +  + +++
Model    AAAAAAAAAAAAAAAAAAAAAAAAAAxxxxxxxxxxxxxxxBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBB
```

Region from a chimeric alignment generated by UCHIME.
Diffs and votes are annotated. The 'Model' row indicates the three segments of the alignment which are closer to A, the crossover (X), and closer to B, respectively. Diffs are 'A'=diff with Q closer to A in the A segment, 'a'=diff with Q closer to A in the B segment, and similarly for 'B' and 'b'. A 'p' diff indicates that the parents agree but are different from Q. Votes are '+' (yes), '!' (no) and '0' (abstain), indicating whether the corresponding diff supports or contradicts the model.

The query sequence is divided into four non-overlapping segments (*chunks*), each of which is used to search a reference database, which is assumed to be chimera-free. The best matches to each chunk are noted, and the two best candidate parents are identified from matches to all chunks. A three-way multiple alignment of the query to these two candidates is constructed. If a pair of segments extracted from these two candidates has identity ≥0.8% closer to the query sequence than either candidate alone, a score is computed from the alignment and a chimera is reported if the score exceeds a predetermined threshold. In reference mode, the user provides a database of trusted sequences. In *de novo* mode, the database is constructed on the fly by considering sequences in order of decreasing abundance. Then, candidate parents must have abundance at least 2× that of the query sequence, assuming chimeras are less abundant than their parents because they undergo fewer rounds of amplification. Sequences not classified as chimeric are added to the reference database.

Diffs and votes

In a typical chimeric alignment, most columns are identities $q=a=b$, where q, a and b are letters from Q, A and B respectively. A column in which at least one sequence differs from the other two is called a *diff*. Diffs can be considered as votes for or against the model. For example, a diff $q=a$, $q\neq b$ increases the distance $d(Q,B)$ while leaving $d(Q,A)$ unchanged. If such a diff is found in the segment that is closer to A, it can be regarded as a "yes" vote supporting the model; if it is found in the segment that is closer to B then it contradicts the model and is regarded as a "no" vote. A diff in which all three sequences differ or in which $a=b$, $q\neq a$, $q\neq b$ increases the distance of Q to both A and B and is regarded as an "abstain" vote that neither supports nor contradicts the model. Let Y_g, N_g and A_g be the total number of yes, no and abstain votes in segment g of the model, where g is L (left) or R (right). If $Y_L > N_L$ and $Y_R > N_R$, the alignment is chimeric and the model is closer to Q than A or B alone. The number of diffs may be very small in more challenging cases. For example, in a 16S experiment using 200nt reads, clusters of radius ~3% might be used in an attempt to identify species. It would then be important to identify chimeras with divergences as low as ~2%, which could have a few as four diffs with their closest parents. In such cases, the small amount of evidence available should increase the uncertainty of the classification.

UCHIME scoring function

Each segment (left and right of the cross-over) is assigned a score:

$$H_g = Y_g / (\beta (N_g + n) + A_g).$$

Intuitively, this can be understood as a generalization of the ratio Y/N, which must be >1 for the alignment to be chimeric. The β parameter (which should be ≥1 and is set to 8 by default) gives a no vote a higher weight than a yes vote, and the n parameter (which should be >0 and is set to 1.4 by default) acts as a pseudo-count prior on the number of no votes. A positive value of n reduces H, especially when Y is small; this models increased uncertainty with reduced evidence. Abstain votes also lower the score as they indicate noise or the use of a step-parent, either of which should increase uncertainty. The query is classified as a chimera if:

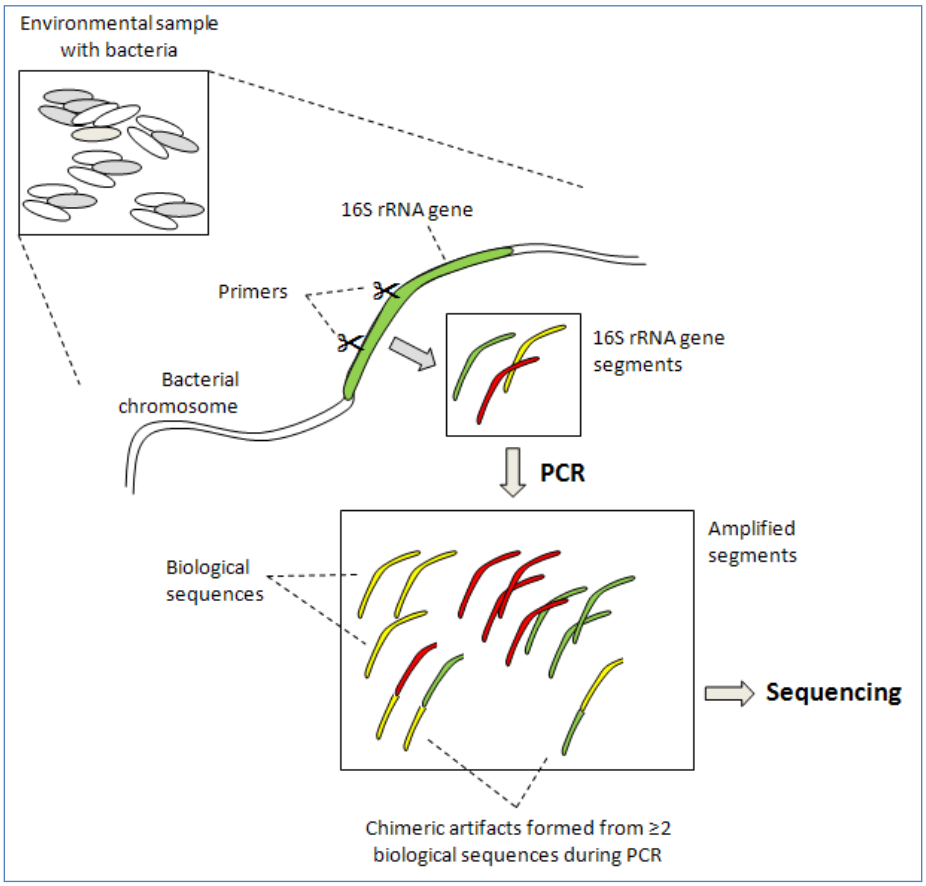
$$H = H_L H_R \geq h.$$

Here, h is the minimum score threshold (0.28 by default).

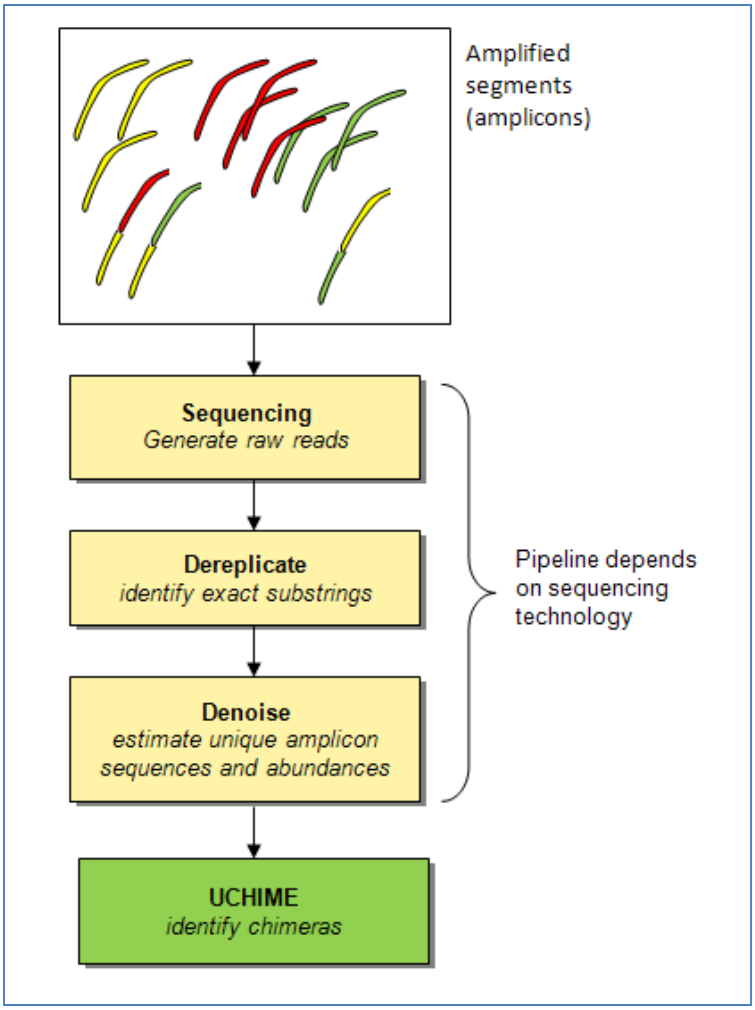
Using UCHIME in practice

Single-region sequencing

UCHIME is designed for experiments that perform community sequencing of a single region such as the 16S rRNA gene or fungal ITS region, as illustrated below.



Schematic summary of a typical 16S environmental sequencing experiment.
Primers are used to extract a short segment of the 16S gene in an environmental sample. Primers are chosen to match highly conserved regions in the gene. The segment between the primers is short enough that the segment can be sequenced with a single read in a current "next-generation" sequencer, and includes variable regions that enable taxonomic identification. Chimeras form in the PCR stage used to amplify segments prior to sequencing.



Schematic summary of a typical 16S sequence analysis pipeline with chimera filtering.
Prior to chimera identification, raw reads are usually quality-filtered, dereplicated and denoised. Input to UCHIME *de novo* mode is a set of estimated amplicon sequences and abundances generated by the denoising stage. In reference database mode, input sequences could be chimera-filtered at any stage in the pipeline and abundances are not needed or used, though in practice chimera filtering would usually be done after denoising as the number of sequences is usually greatly reduced and the computational resources required to run UCHIME are correspondingly less.

Reference database mode

The reference database mode of UCHIME implicitly assumes that the database contains high-quality sequences close to the true biological sequences in the sample. The most common problems with a reference database approach are: (i) the lack of a suitable reference database, (ii) inadequate phylogenetic coverage of the community being studied in available databases, and (iii) poor-quality sequences in available databases.

In practice, reference databases will usually be incomplete, and false negatives should be expected due to missing parents. Unknown species will of course be absent. Even if a given species has a high-quality reference sequence, it may have additional copies of the sequenced

gene due to copies (paralogs, pseudo-genes or segmental duplications) that are absent from the database. Phylogenetic coverage should therefore be not understood not just in terms of species, but also considering of all sequences in the community that are homologous to the gene and match the chosen primers.

A false negative will occur if the query sequence is a chimera and the database contains a sufficiently similar chimera. Noisy reference sequences can cause both false negatives and false positives. Noise can reduce the score of a valid chimeric model below the *h* threshold, creating a false negative. To see how noisy sequences can produce false positives, let X be a correct biological sequences, X_L be a prefix of X, X_R be a suffix of X and X' be a "noisy" copy of X, i.e. a copy of X with spurious substitutions and/or indels. Suppose there are two noisy copies of X^1 and X^2 in the database with asymmetric noise, such that X^1 has more noise on the left and X^2 has more noise on the right, i.e. $X^1 = X'_L X_R$, $X^2 = X_L X'_R$. Then a good copy of X may appear to be a chimera $X = X^2_R X^1_L$ formed from parents X^1 and X^2. If X' and a chimera $C = X_L Y'_R$ are present in the reference database, but not Y, this can cause a false positive identification of Y, which may appear to be a chimera formed as $Y = X'_L C_R$.

Correct sequences in the reference database may give rise to false positives if evolutionary rates in different regions of the gene vary in different lineages. Suppose the gene contains two regions r_1 and r_2, and there are three lineages A, B and C where r_1 evolves faster in A than in B or C, and r_2 evolves faster in B than in A or C. Now suppose the database contains A and B but not C, then C may appear to be a chimera formed from A and B.

These considerations present conflicting goals in the design of a reference database: high phylogenetic coverage and high-quality sequences. Increased phylogenetic coverage generally requires incorporating sequences from unfinished genomes and/or from environmental sequencing studies, both of which tend to have higher error rates than finished genomes. This can be mitigated by using the reference database mode of UCHIME to check a candidate reference database against itself using self mode, as described in the next section.

Self mode for database screening

The self mode (--*self* option) is used when the same file is used for both query and reference. This can be used to screen databases for chimeras. This option causes the query sequence to be excluded as a possible parent, otherwise all sequences would trivially be annotated as non-chimeric due to self-matches. Hits reported using --*self* are 3-way alignments in which either one or two of the sequences are putative chimeras. It should not be assumed that the query sequence is the chimera in this case. Further evidence is required to determine which, if any, of the sequences in the 3-way alignment are PCR artifacts. For example, if two of the sequences are derived from high-quality, finished genomes and the third is from an environmental sequencing study, then the third is most likely to be an artifact and should be discarded from the database. Any remaining sequences found in 3-way alignments can be annotated as unresolved. Hits to experimental data that have an unresolved parent can be treated differently. Whether they should be included or discarded depends on the goals of the study, which will determine the relative importance of sensitivity and specificity of chimera detection. Discarding questionable hits will tend to improve specificity at the expense of sensitivity; including them will tend to improve sensitivity at the expense of specificity.

It is often the case that a reference database contains full-length sequences while a shorter region is sequenced. Here it may be advantageous to trim the database to the shorter region. This can improve computational efficiency because the time required to make a dynamic programming alignment scales with the square of the sequence length (Durbin et al, 1998). This may also reduce the number of false negatives due to failures to identify the correct parent which may be caused by the word-counting heuristic filter (Edgar, 2010) that is used to increase search speed in both the public-domain and USEARCH implementations of UCHIME.

De novo mode

The *de novo* mode of UCHIME assumes (i) input sequences correspond to unique sequences in the amplified sample, (ii) the abundances of those sequences have been estimated with sufficient accuracy, (iii) errors due to amplification and sequencing can be neglected, i.e. are adequately suppressed preprocessing of the sequences and/or by the UCHIME scoring function, and (iv) chimeras have abundance less than their parents, as specified by the abundance skew parameter. At the present time, it is not known how well these assumptions hold in practice, except for the mock communities described in the main text. It is an open research problem to determine how predictive these mock communities are of experiments on natural communities.

An advantage of the *de novo* approach is that we expect most or all parent sequences to be present in the reads, which may enable higher sensitivity to be achieved compared with a pre-existing reference database, which will generally be incomplete. A disadvantage of *de novo* mode is that an estimate of unique amplicon abundances is required, which may not be readily available.

The process of estimating unique amplicon sequences and their abundances from a set of reads is called *denoising*. Denoising is a challenging algorithmic problem in itself, and is a rapidly moving target as sequencing technologies evolve. Currently available methods for denoising include PyroNoise (Quince et al., 2009) or AmpliconNoise (Quince et al., 2011) for 454 flowgrams, or clustering methods such as UCLUST (Edgar, 2010) which can be applied to any set of reads.

Consistency check

Where possible, I recommend that the reference database mode and *de novo* modes be used to check each other. I would consider hits found by both methods to be more reliable than hits reported only by one method, though this assumption has not yet been validated. While the mock communities considered in the main test could potentially have been used to test this idea, it turns out that they have too few false positives to give statistically informative results.

A hit found by the reference database mode but not by *de novo* mode can be investigated by searching the estimated amplicons for the putative parent sequences. If these are present in the reads, then this is probably a false negative by the *de novo* mode, which could be due to poor estimates of amplicon sequences or abundances, a preceding false positive that incorrectly identified a parent as a chimera, or a violation of the assumption that the parents have higher abundance. If the parents are not found in the reads, then this could be a false positive by the reference database mode (see previous discussion of causes of false positives in this mode).

A hit found by *de novo* mode but not by reference database mode may be explained by a missing parent sequence in the reference database, which can be verified by searching the reference database for the parents predicted by *de novo* mode.

Sensitivity vs. specificity

The user can trade sensitivity against specificity by adjusting the score threshold (*h* parameter, --*minh* command line option). It is difficult to predict the sensitivity or specificity that will be obtained for a given experiment with a given score threshold. When considering whether sensitivity or specificity is more important in a given experiment, it should be noted that while chimeric amplicons may be relatively rare in the amplicon pool, they may represent a large fraction of unique amplicon sequences.

Computational efficiency

Community sequencing experiments often produce very large numbers of reads that can be computationally expensive to process. It is generally recommended that the number of sequences be reduced before running UCHIME in order to save computational resources. Preprocessing steps can include dereplication (removing identical sequences), denoising (attempting to correct sequencing error) and data reduction (clustering at, say, 98% identity to reduce experimentally irrelevant variation in the sequences).

In the case of *de novo* mode, preprocessing of raw reads is always required in order to estimate amplicon sequences and abundances. The estimated number of unique amplicons is usually much smaller than the number of reads, reducing the computational cost of downstream stages in an analysis pipeline, such as UCHIME.

Computational cost can also be significantly reduced by using the USEARCH (Edgar, 2010) implementation of the UCHIME algorithm. The most expensive step in UCHIME is generally searching the reference database. The implementation of UCHIME in the usearch package (http://drive5/com/usearch) exploits the highly optimized USEARCH algorithm for the database search step, which often results in significantly improved execution times. As noted in the main text, UCHIME results are generally not sensitive to the details of the database search method, and the USEARCH implementation therefore gives very similar results to the public-domain version.

In reference database mode, execution time for UCHIME scales approximately linearly with the reference database size and number of query sequences, and like the square of the sequence length (due to the dynamic programming step required for alignment). In *de novo* mode, time scales linearly with the number of query sequences, linearly with the number of non-chimeric sequences identified in the input, and with the square of the sequence length.

Paired-end reads

At the time of writing, UCHIME does not explicitly support paired-end reads. Work is in progress to add support for pair-end reads in a future version of the algorithm.

Providing that the gap between the ends is short, a reasonable strategy would be to concatenate the two ends. Longer gaps are likely to result in substantially increased false negative rates. It is not recommended to use the common practice of representing the gap between the ends using the

corresponding number of Ns as UCHIME will consider these to be differences between the chimera and the parents which will usually result in a false negative. An alternative would be to fill the gap with a consensus sequence obtained by a multiple alignment of the top hits to the query sequence. Note that by default, gapped positions are not considered differences, and simple concatenation without using Ns may therefore be more effective.

Parameter tuning

The default parameters of UCHIME were tuned to give lower error rates and higher sensitivity than ChimeraSlayer on the benchmark datasets used in the ChimeraSlayer paper. This strategy was chosen in order to demonstrate that UCHIME has better performance than ChimeraSlayer on a published benchmark (Haas et al., 2011) on which ChimeraSlayer was shown to be superior to previous methods and thereby establish that UCHIME is superior to all previously published methods. I believe that while these parameters probably represent reasonable default settings, different parameters may be optimal in some applications. It should be noted that the ChimeraSlayer validation emphasized sensitivity to closely related parents: the divergence measure used by Haas *et al.* is the distance D between the parents A and B (D = 100% – *id*(A,B)), while in this work we use the identity T between the chimera Q and the closest parent (T = 100% – max { *id*(Q,A), *id*(Q, B) } in the case of bimeras). Generally we expect that $T \leq D/2$ since at last half of the bimera will be identical to the closer parent. In many experiments, it is T rather than D that indicates whether the chimera is experimentally relevant. For example, if the goal is to identify OTUs by clustering at 97%, and a parent is successfully identified as the representative sequence for a cluster, then a chimera with $T \leq 3\%$ should be assigned to the parent cluster and will not create a spurious OTU. Such a chimera could have $D \geq 6\%$, and conversely a chimera with D=6% could have arbitrarily small T and thus fall inside a 3% cluster radius. By default, the minimum T divergence, set by the --*mindiv* option of UCHIME, is set to 0.8% to allow detection of chimeras with small D, which is required to achieve good performance on the ChimeraSlayer benchmark test. Chimeras with divergence $T \gtrsim 0.8\%$ may have very small numbers of diffs and hence be difficult to discriminate from false positives, requiring a higher h threshold to suppress errors. These considerations suggest that in a typical OTU clustering experiment, higher sensitivity to experimentally relevant chimeras could be achieved with acceptable false positive rates by increasing --*mindiv* and reducing h (--*minh* option) and/or β (--*xn* option). In addition, the ChimeraSlayer benchmark has no multimeras and adds simulated noise that is designed to indicate the general impact of sequencing error and natural variation on performance rather than to accurately model errors due to a given sequencing technologies or to model natural biological variation that can cause a reference sequence to differ from the true parent sequence. Ideally, parameters would be re-tuned on a benchmark that is tailored to the details of a particular experiment, including simulated errors based on estimates of error rates of the chosen sequencing technology. Designing and implementing such a benchmark would be challenging. Further work is needed to determine whether and how parameters should be varied according to the details of a particular experiment.

Command-line options

```
--input filename
--uchime filename
    Query sequences in FASTA format.
    If the --db option is not specificed, uchime uses de novo
    detection. In de novo mode, abundance must be given
    by a string ;size=nnn; somewhere in the label, where nnn is
    an integer, e.g. >F00QGH67HG;size=123;. The traling ';'
    may be omitted at the end of the label.

    Both the --input and --uchime options may be used in the
    stand-alone UCHIME program. If you are using USEARCH, then
    you must use the --uchime option.

--db filename
    Reference database of chimera-free sequences in FASTA format.
    Optional, if not specified uchime uses de novo mode.

    ***WARNING*** The database is searched ONLY on the plus strand.
    You MUST include reverse-complemented sequences in the database
    if you want both strands to be searched. Or, in the USEARCH
    implementation, you can use the --rev option.

--abskew x
    Minimum abundance skew. Default 1.9. De novo mode only.
    Abundance skew is:
        min [ abund(parent1), abund(parent2) ] / abund(query).

--chimeras filename
   FASTA file to write sequences that are classified as chimeric.

--nonchimeras filename
   FASTA file to write sequences that are classified as non-chimeric.

--uchimeout filename
    Output in tabbed format with one record per query sequence.
    First field is score (h), second field is query label.
    For details, see manual.

--uchimealns filename
    Multiple alignments of query sequences to parents in human-
    readable format. Alignments show columns with differences
    that support or contradict a chimeric model.

--minh h
    Mininum score to report chimera. Default 0.28. Values from 0.1
    to 5 might be reasonable. Lower values increase sensitivity
    but may report more false positives. If you decrease --xn,
    you may need to increase --minh, and vice versa.

--mindiv div
    Minimum divergence ratio, default 0.8. Div ratio is 100% -
    %identity between query sequence and the closest candidate for
    being a parent. If you don't care about very close chimeras,
    then you could increase --mindiv to, say, 1.0 or 2.0, and
    also decrease --min h, say to 0.1, to increase sensitivity.
```

```
    How well this works will depend on your data. Best is to
    tune parameters on a good benchmark.

--mindiffs n
    Minimum number of diffs in each segment. Default 3.

--xn beta
    Weight of a no vote, also called the beta parameter. Default 8.0.
    Decreasing this weight to around 3 or 4 may give better
    performance on denoised data.

--dn n
    Pseudo-count prior on number of no votes. Default 1.4. Probably
    no good reason to change this unless you can retune to a good
    benchmark for your data. Reasonable values are probably in the
    range from 0.2 to 2.

--xa w
    Weight of an abstain vote. Default 1. So far, results do not
    seem to be very sensitive to this parameter, but if you have
    a good training set might be worth trying. Reasonable values
    might range from 0.1 to 2.

--chunks n
    Number of chunks to extract from the query sequence when searching
    for parents. Default 4.

--[no]ovchunks
    [Do not] use overlapping chunks. Default do not.

--minchunk n
    Minimum length of a chunk. Default 64.

--idsmoothwindow w
    Length of id smoothing window. Default 32.

--minsmoothid f
    Minimum factional identity over smoothed window of candidate parent.
    Default 0.95.

--maxp n
    Maximum number of candidate parents to consider. Default 2. In tests so
    far, increasing --maxp gives only a very small improvement in sensivity
    but tends to increase the error rate quite a bit.

--[no]skipgaps
--[no]skipgaps2
    These options control how gapped columns affect counting of diffs.
    If --skipgaps is specified, columns containing gaps do not found as diffs.
    If --skipgaps2 is specified, if column is immediately adjacent to
    a column containing a gap, it is not counted as a diff.
    Default is --skipgaps --skipgaps2.

--minlen L
--maxlen L
    Minimum and maximum sequence length. Defaults 10, 10000.

--ucl
```

```
    Use local-X alignments. Default is global-X. On tests so far, global-X
    is always better; this option is retained because it just might work
    well on some future type of data.

--queryfract f
    Minimum fraction of the query sequence that must be covered by a local-X
    alignment. Default 0.5. Applies only when --ucl is specified.

--quiet
    Do not display progress messages on stderr.

--log filename
    Write miscellaneous information to the log file. Mostly of interest
    to me (the algorithm developer). Use --verbose to get more info.

--self
    In reference database mode, exclude a reference sequence if it has
    the same label as the query. This is useful for benchmarking by using
    the ref db as a query to test for false positives and also for screening
     a ref db against itself.
```