VSEARCH and Swarm: Tools for analysis of microbiome sequencing data

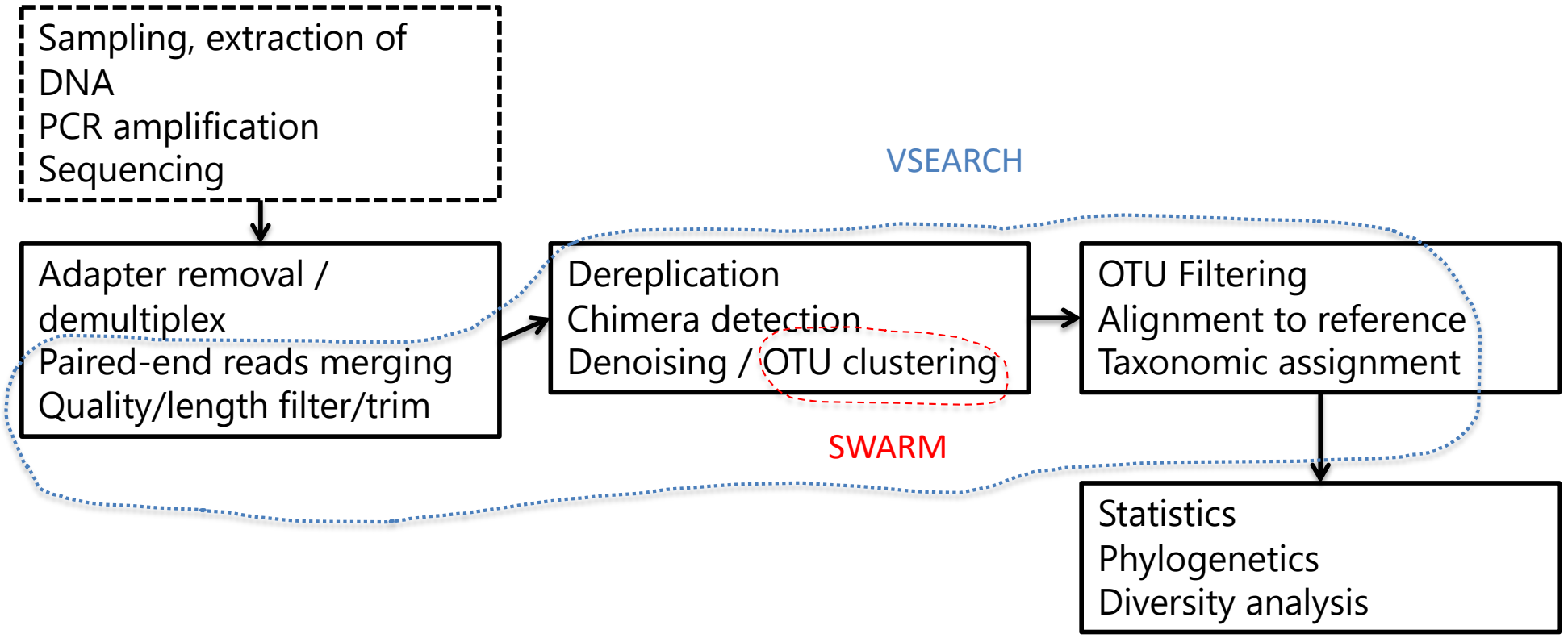
BIO9905MERG1 Course, UiO, 19 April 2023

Torbjørn Rognes
Dept. of Informatics, UiO & Dept. of Microbiology, OUS
torognes@ifi.uio.no

UiO : University of Oslo

Oslo University Hospital

Amplicon sequence analysis pipeline



VSEARCH: a versatile analysis tool

- Versatile command-line tool to analyse DNA sequence data in microbiome projects
- Especially suited for amplicon data
- Free of charge
- Open source software
- Robust, well-tested
- Available on many platforms and architectures
- 64 bit, able to handle very large databases (>4GB)
- A drop-in replacement for USEARCH in many cases (proprietary software, only 32bit version free of charge)

Features of VSEARCH v2.22.1

VSEARCH includes 47 commands with 191 options:

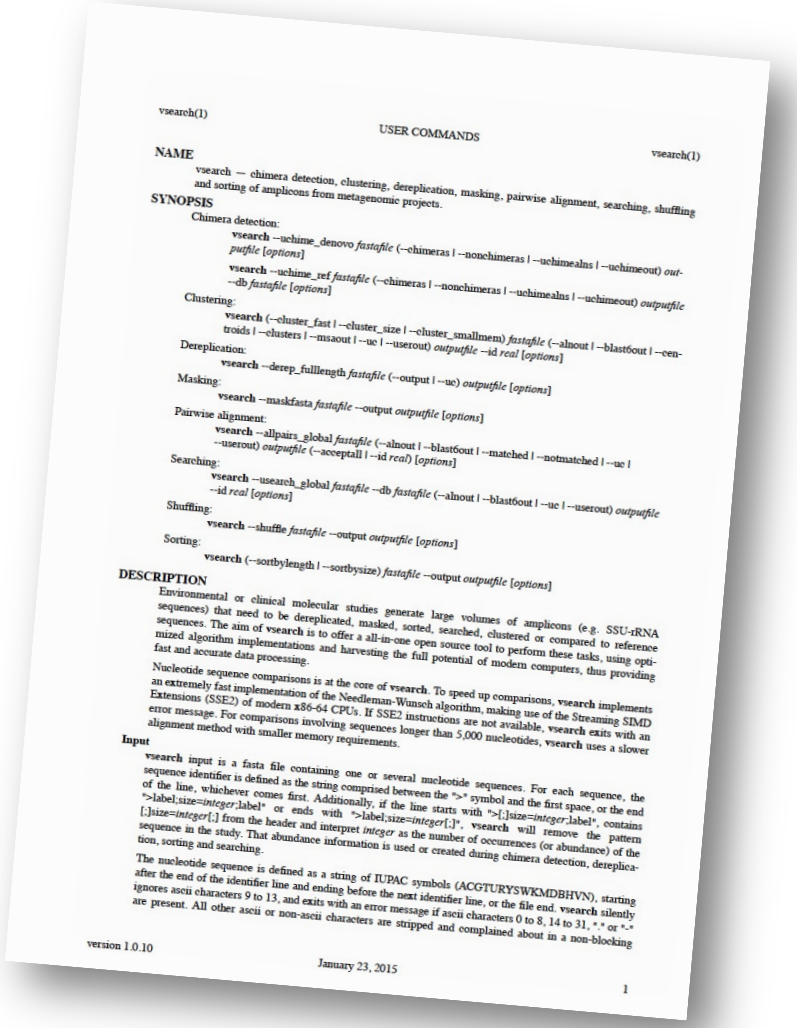
- Chimera detection (*de novo* (uchime, uchime2 or uchime3), and reference)
- Clustering (after sorting by abundance or length, or using initial order, or similar to unoise)
- Detection and decompression of compressed input files (.gz, .bz2)
- Dereplication of sequences (full length, prefix & id) and rereplication
- Extraction of sequences and sub-sequences from large FASTA files
- FASTQ encoding detection and conversion, SFF to FASTQ conversion, FASTA to/from FASTQ
- Masking of low-complexity regions
- Orienting of sequences in same direction as database sequences
- Paired-end reads merging and joining
- Pairwise global sequence alignment (all vs all)
- Restriction site cutting
- Reverse complementation of sequences
- Searching (global alignment and exact matches)
- Sequence quality statistics and filtering
- Shuffling and sorting (abundance and length)
- Subsampling of sequences
- Taxonomic classification (SINTAX, LCA)
- UDB files: Building, extraction and statistics of fast-loading pre-indexed database files

Getting help with VSEARCH

Commands:

```
vsearch
vsearch --help | less
man vsearch
```

Manual also available as PDF:
vsearch_manual.pdf



vsearch(1) USER COMMANDS vsearch(1)

NAME

vsearch — chimera detection, clustering, dereplication, masking, pairwise alignment, searching, shuffling and sorting of amplicons from metagenomic projects.

SYNOPSIS

Chimera detection:

vsearch --uchime_denovo *fastafile* (--chimeras | --nonchimeras | --uchimealns | --uchimeout) *outputfile* [*options*]

vsearch --uchime_ref *fastafile* (--chimeras | --nonchimeras | --uchimealns | --uchimeout) *outputfile* --db *fastafile* [*options*]

Clustering:

vsearch (--cluster_fast | --cluster_size | --cluster_smallmem) *fastafile* (--alnout | --blast6out | --centroids | --clusters | --msaout | --uc | --userout) *outputfile* --id *real* [*options*]

Dereplication:

vsearch --derep_fulllength *fastafile* (--output | --uc) *outputfile* [*options*]

Masking:

vsearch --maskfasta *fastafile* --output *outputfile* [*options*]

Pairwise alignment:

vsearch --allpairs_global *fastafile* (--alnout | --blast6out | --matched | --notmatched | --uc | --userout) *outputfile* (--acceptall | --id *real*) [*options*]

Searching:

vsearch --usearch_global *fastafile* --db *fastafile* (--alnout | --blast6out | --uc | --userout) *outputfile* --id *real* [*options*]

Shuffling:

vsearch --shuffle *fastafile* --output *outputfile* [*options*]

Sorting:

vsearch (--sortbylength | --sortbysize) *fastafile* --output *outputfile* [*options*]

DESCRIPTION

Environmental or clinical molecular studies generate large volumes of amplicons (e.g. SSU-rRNA sequences) that need to be dereplicated, masked, sorted, searched, clustered or compared to reference sequences. The aim of **vsearch** is to offer a all-in-one open source tool to perform these tasks, using optimized algorithm implementations and harvesting the full potential of modern computers, thus providing fast and accurate data processing.

Nucleotide sequence comparisons is at the core of **vsearch**. To speed up comparisons, **vsearch** implements an extremely fast implementation of the Needleman-Wunsch algorithm, making use of the Streaming SIMD Extensions (SSE2) of modern x86-64 CPUs. If SSE2 instructions are not available, **vsearch** exits with an error message. For comparisons involving sequences longer than 5,000 nucleotides, **vsearch** uses a slower alignment method with smaller memory requirements.

Input

vsearch input is a fasta file containing one or several nucleotide sequences. For each sequence, the sequence identifier is defined as the string comprised between the ">" symbol and the first space, or the end of the line, whichever comes first. Additionally, if the line starts with ">label;size=*integer*;label" or ends with ">label;size=*integer*[;]", **vsearch** will remove the pattern [;]size=*integer*[;] from the header and interpret *integer* as the number of occurrences (or abundance) of the sequence in the study. That abundance information is used or created during chimera detection, dereplication, sorting and searching.

The nucleotide sequence is defined as a string of IUPAC symbols (ACGTURYSWKMDBHVN), starting after the end of the identifier line and ending before the next identifier line, or the file end. **vsearch** silently ignores ascii characters 9 to 13, and exits with an error message if ascii characters 0 to 8, 14 to 31, "." or "-" are present. All other ascii or non-ascii characters are stripped and complained about in a non-blocking

version 1.0.10 January 23, 2015 1

Merging paired-end reads

```
vsearch --fastq_mergepairs forward.fastq          \
   --reverse reverse.fastq                        \
   --fastqout merged.fastq                        \
   --fastq_allowmergestagger                      \
   --fastqout_notmerged_fwd notmerged.fwd.fastq \
   --fastqout_notmerged_rev notmerged.rev.fastq
```

Merges overlapping forward and reverse reads into one sequence, if possible. Unmerged reads are written to separate files. Allows for staggered overlapping reads (in case of very short fragments).

Merging paired-end reads

Read 1 (green): Forward sequence
Read 2 (red): Reverse complementary sequence

- Find best overlap between the two sequences
- Take the quality score (error probability) of each base into account

Filtering reads

```
vsearch --fastq_filter input.fastq                \
   --fastq_maxee 1.0                              \
   --fastq_maxns 0                                \
   --fastq_minlen 100                             \
   --fastqout filtered.fastq                      \
   --fastaout filtered.fasta                      \
   --relabel abc
```

Removes or truncates reads that does not satifisfy given requirements. Could be number of N's, length, quality/error, etc.

Dereplication

```
vsearch --fastx_uniques input.fasta                \
   --fastaout derep.fasta                          \
   --minuniquesize 2                               \
   --sizein --sizeout
```

Groups strictly identical sequences into one entry in the FASTA file. Adds a "size" attribute to the FASTA header with the abundance (number of identical copies) of the sequence. In the example, only those with abundance at least 2 are included in the output.

Alternatives: derep_fulllength, derep_prefix, derep_smallmem

```
>abc;size=123
acgtagtcagactgtcagactgtg
```

Sequence similarity clustering

```
vsearch --cluster_size input.fasta                    \
   --id 0.97                                          \
   --centroids centroids.fasta                        \
   --sizein                                           \
   --sizeout
```

Groups similar sequences into clusters (OTUs) using a fast, heuristic and greedy centroid-based algorithm. May specify identity threshold (e.g. 97%).

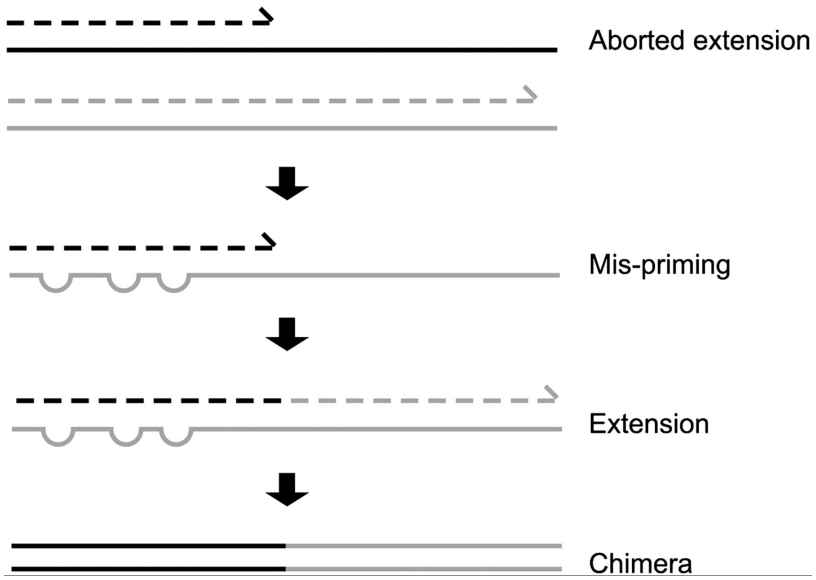
Chimera detection

```
vsearch --uchime_denovo input.fasta                \
   --nonchimeras nonchimeras.fasta                 \
   --chimeras chimeras.fasta                       \
   --sizein --sizeout
```

Detects potential chimeric sequences in the input file and writes the classified sequences into separate files.

Chimera detection

- Chimeric DNA sequences may form during PCR amplification
- Chimeras contain segments from 2 or more parent sequences
- Will inflate the apparent diversity of organisms in the sample unless removed
- VSEARCH implements the algorithms UCHIME (Edgar *et al.*, 2011), UCHIME2 and UCHIME3
- Use either the dataset itself (uchime_denovo) or a reference database (uchime_ref) during analysis
- A new algorithm for long high-quality sequences with potentially more than two parents is in development.



Chimera detection

```
Query   (  250 nt) ch31_m2_90_95/sp8:0-149/sp62:149-249/N=2/top=sp8:92.4%
ParentA (  250 nt) sp8/name=Clostridiummethylpentosum_0_1_2
ParentB (  250 nt) sp62/name=Clostridiumsporogenes_0_3_2

A     1 TGCTGCCTCCCGTAGGAGTCTGGGCCGTGTctcagtcCCAATGTGGCCGTT-CAACCTCTCAGTCCGGCTA-CTGATCGt 78
Q     1 TGCTGCCTCCCGTAGGAGTCTGGGCCGTGTTCAGTCGCCAATGTGGCCGTTCCAACCTCTCAGTCCGGCTAGCTGATCG- 79
B     1 TGCTGCCTCCCGTAGGAGTCTGGaCCGTGTctcagttCCAATGTGGCCGaT-CAcCCTCTCAGgtCGGCTAcgcatcgt- 78
Diffs                          A      NNNNNN?            A    A        AA       AAAAAAA
Votes                          +      0000000            +    +        ++        +++++
Model   AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA

A    79 CGACTTGGTGAGCCATTACCTCACCAACTATcTAATCAGA-CGCGAGCCCATCTTaCAGCGATATAATCTTTGAT-AAcA 156
Q    80 CGACTTGGTGAGCCATTACCTCACCAACTAT-TAATCAGACCGCGAGCCCATCTT-CAGCGATATAATCTTTGATAAAAA 157
B    79 tGcCTTGGTaAGCCgTTACCTtACCAACTAg-ctAatgcgCCGCGgGtCCATCTc-aAagcAataAATCTTTGAT-AAAA 155
Diffs   A A      A    A      A        A AA AAAAA     A A      A A AAA AAA             B
Votes     +      +    +      +           + ++++      + +          +++ +++             +
Model   AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAxxxxxxxxxxxxxBB

A   157 AAAcCATGCGATTCcgTTATgTTATGCGGTATTAgcgTTCgTTTCc--AAacGtTATtCCCctcTgtAAGGCAGGTTgCt 234
Q   158 AAATCATGCGATTCTCTTATATTATGCGGTATTAATCTTCCTTTCG--AA--GCTATCCCCACTTTGAAGGCAGGTTACC 233
B   156 AAATCATGCGATTCTCTTATATTATGCGGTATTAATCTTCCTTTCGgaAg--GCTATCCCCcacTTtgAGGCAGGTTACC 233
Diffs      B          BB    B             BBB   B    B   A   B   B   N?N BNa         B B
Votes      +          ++    +             +++   +            +   +   000 +0!         + +
Model   BBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBB

A   235 CACGTGTTACTCACCC- 250
Q   234 CACGTGTTACTCACCCG 250
B   234 CACGTGTTACTCACCCG 250
Diffs
Votes
Model   BBBBBBBBBBBBBBBBB

Ids.  QA 88.9%, QB 82.7%, AB 80.5%, QModel 94.7%, Div. +6.5%
Diffs Left 34: N 0, A 7, Y 27 (79.4%); Right 19: N 1, A 4, Y 14 (73.7%), Score 0.8952
```

Sequence similarity search

```
vsearch --usearch_global query.fasta          \
   --db database.fasta                        \
   --id 0.99                                  \
   --maxaccepts 3                             \
   --maxrejects 1000                          \
   --alnout alignments.txt                    \
   --otutabout otutable.txt
```

Generic heuristic sequence similarity search using global alignment with many adjustable parameters.

Taxonomic classification with SINTAX

```
vsearch --sintax clusters.fasta                    \
   --db silva.fasta                                \
   --sintax_cutoff 0.8                             \
   --tabbedout classified.tsv
```

Rapid classification of sequences using a taxonomy given in a specially formatted database. Uses the SINTAX algorithm. Results may vary slightly due to randomness in the algorithm.

```
>AY232296.1.1383;tax=k:Archaea,p:Euryarchaeota,c:Hal
obacteria,o:Halobacteriales,f:Halobacteriaceae,g:Nat
rinema,s:Natrinema_sp._HM06;
```

Taxonomic classification with LCA

```
vsearch --usearch_global clusters.fasta          \
   --db silva.fasta                              \
   --id 0.99                                     \
   --maxaccepts 20                               \
   --maxrejects 1000                             \
   --lca_cutoff 0.95                             \
   --lcaout classified.tsv
```

Simple and rapid classification of sequences using a taxonomy given in a specially formatted database. Retrieves top 20 sequences that are at least 99% identical and finds their lowest common ancestor (LCA), allowing for 5% (i.e. 1 of 20) divergent sequences.

General heuristic search algorithm

Used during search, chimera detection, and clustering

For each query sequence:

- Sort target (database) sequences by decreasing number of *k*-mers (words of consecutive nucleotides, default 8) shared with the query sequence
- Consider target sequences in order and align the query to each candidate target sequence
- Stop when *A* targets (*maxaccepts*, default 1) have been accepted, i.e. they satisfy the accept criteria (e.g. id > 97%)
- Stop when *R* targets (*maxrejects*, default 32) have been rejected, i.e. they do not satisfy the accept criteria.

Converting old sequence files

Conversion from SFF to FASTQ:

```
vsearch --sff_convert input.sff                \
   --fastqout output.fastq                     \
   --sff_clip                                  \
   --fastq_asciiout 33
```

Conversion from old phred 64 FASTQ files to Phred 33 FASTQ:

```
vsearch --fastq_convert old.fastq              \
   --fastqout new.fastq                        \
   --fastq_ascii 64                            \
   --fastq_asciiout 33
```

Phred 64 base quality encoding was used in Solexa and early Illumina (before version 1.8) files. Today, phred 33 is almost always used.

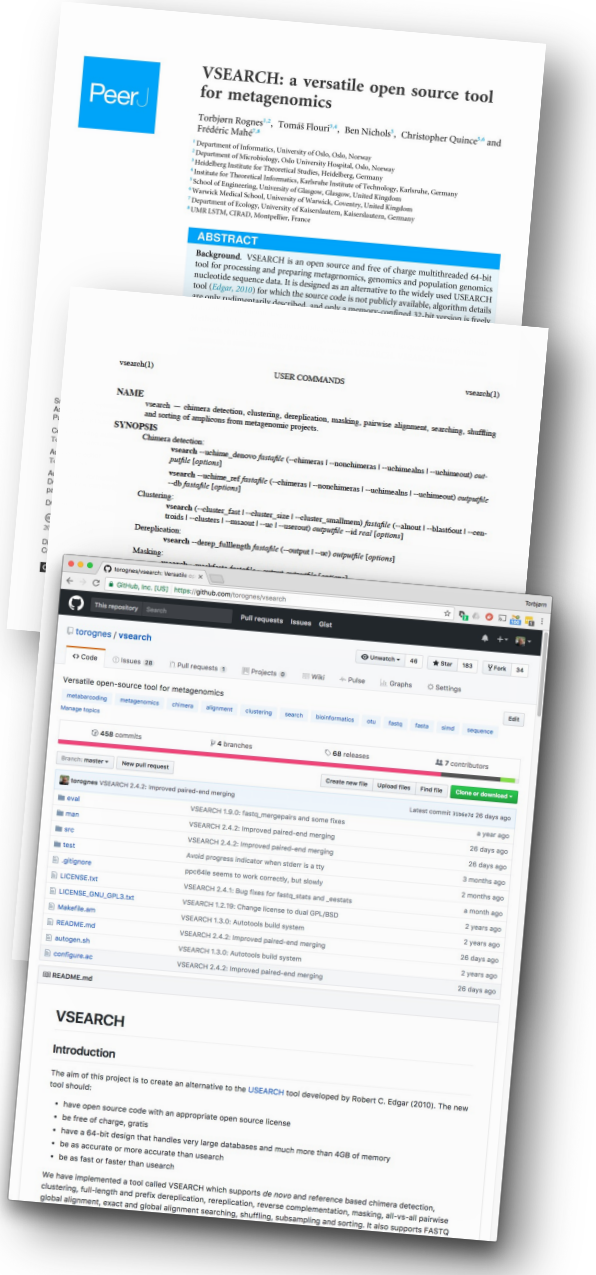
Working with pipes and compressed files

```
cat input.fastq.gz |                                    \
vsearch --fastx_uniques -                               \
  --gzip_decompress                                     \
  --sizeout --quiet --fastaout - |                      \
vsearch --cluster_size -                                \
  --id 0.97 --sizein –sizeout --centroids -             \
> otus.fasta 2> errors.txt
```

- Use shell operators to redirect input (<), redirect output (>), redirect and append output (>>), redirect errors (2>) or set up pipes (|).
- Replace file names with "-" to read from standard input (stdin) or write to standard output (stdout).
- Use "--quiet" to silence messages usually written to stdout.
- Warnings and errors are written to standard error (stderr).
- Use the option "--gzip_decompress" or "--bzip2_decompress" to decompress input from stdin, otherwise decompression is automatic.

VSEARCH availability & documentation

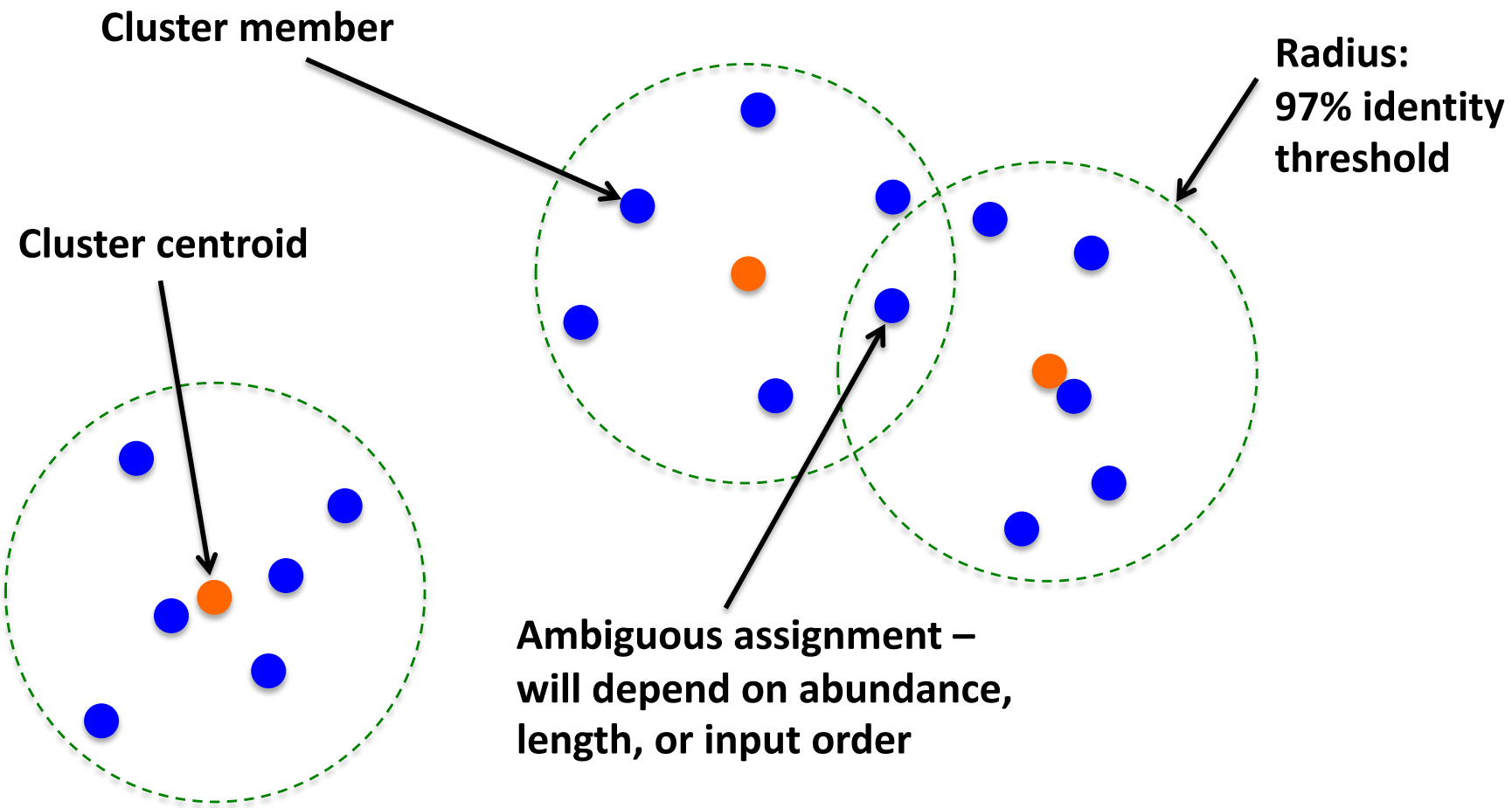
- VSEARCH source code: https://github.com/torognes/vsearch
- 64-bit binary binaries for Linux (Intel x86, ARM, PPC), macOS (Intel, Apple Silicon) and Windows available
- Dual open-source license: GNU AGPL v3 or BSD 2-clause
- May be used directly for clustering and chimera detection in mothur
- QIIME 2 plugin; Conda, Homebrew, Debian packages; Galaxy wrapper
- Publication:
  Rognes T, Flouri T, Nichols B, Quince C, Mahé F. (2016)
  VSEARCH: a versatile open source tool for metagenomics
  PeerJ 4:e2584 doi: 10.7717/peerj.2584 (~6000 citations)
- User manual with command and option details (over 50 pages)
- Extensively tested with >2000 unit tests
- Wiki, issue tracker, online support forum etc
- Suggest new features! Report bugs! Ask for help!



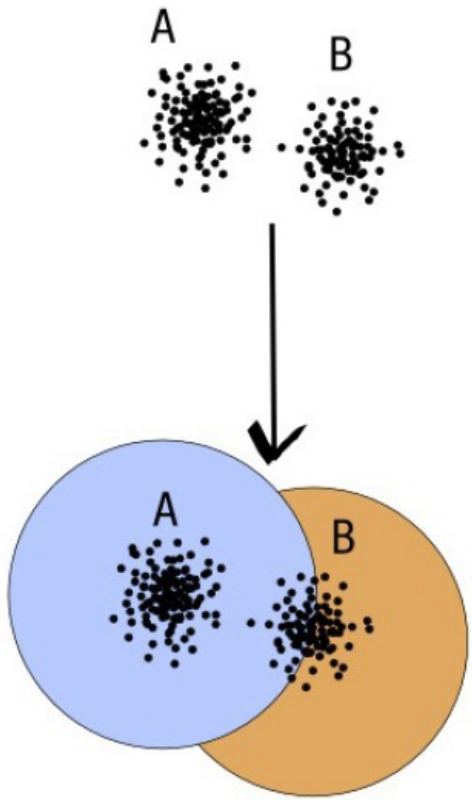
Swarm: an alternative clustering method

- Alternative method for clustering amplicon sequences
- Clusters are groups of sequences that could approximately represent e.g. a species or a genus as we define it
- Single linkage hierarchical clustering approach
- Avoids two problems with traditional methods:
  - input order dependence
  - global clustering threshold
- High resolution
- Very fast: Linear time and space complexity: fast and limited memory demands. Allows huge datasets to be clustered in reasonable time

Heuristic centroid-based clustering

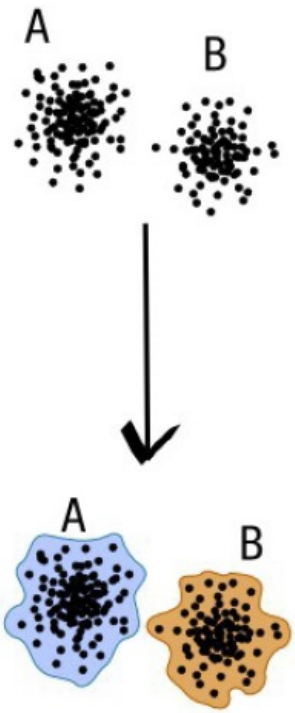


Clustering thresholds



compromise threshold
unadapted threshold

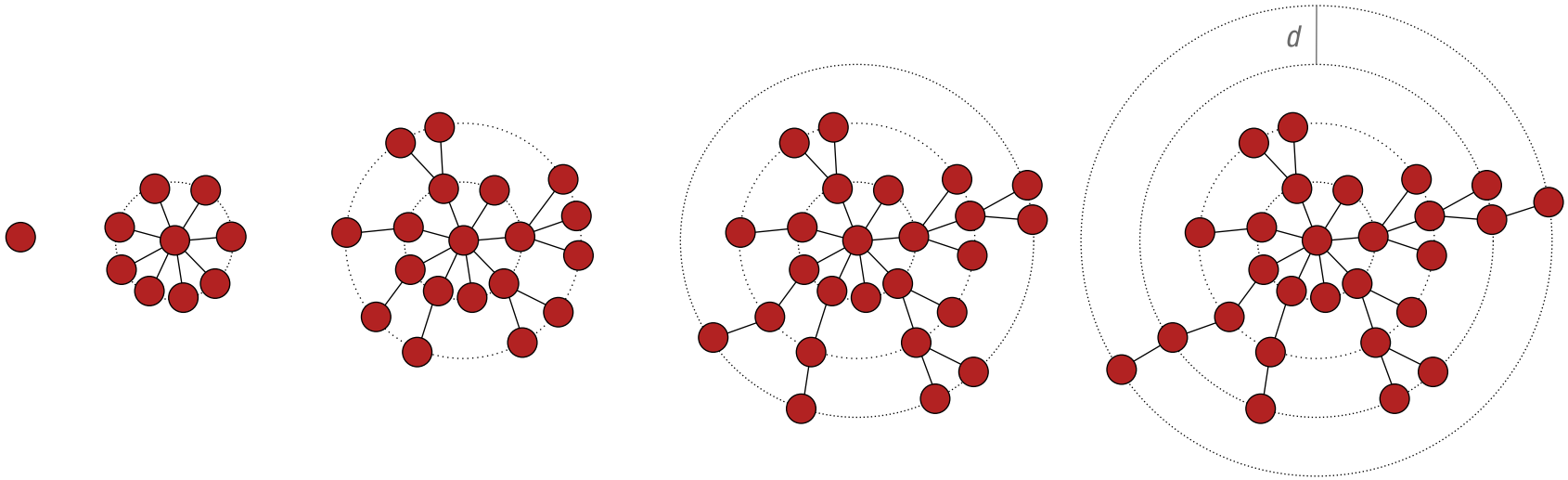
defined by maximum
distance from centroid



natural limits of OTUs

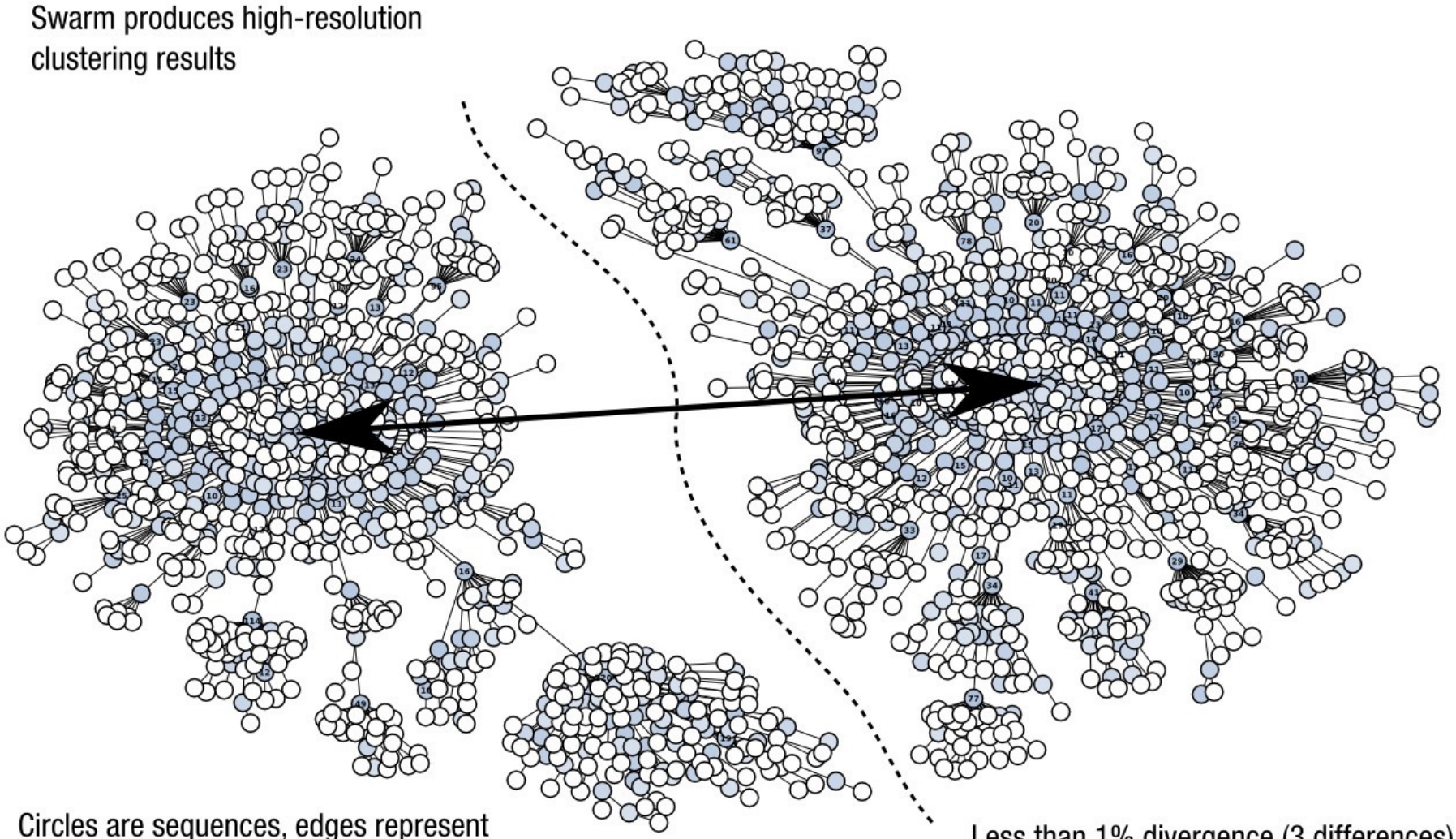
defined by minimum
distance separating clusters

Linking amplicons



- Amplicons (nodes) are linked when distance is less than or equal to *d*.
- Similar to single linkage hierarchical clustering
- Forms a spanning tree
- The adjustable parameter *d* is 1 by default.

Visualising the clustering results



SWARM availability & documentation

- Available on GitHub: https://github.com/torognes/swarm
- Free of charge
- GNU AGPL v3 open source license
- Extensive documentation
- Modern C++ code
- Extensively tested with 732 unit tests
- Precompiled 64-bit binaries for Linux (Intel, PowerPC, ARM), macOS (Intel, Apple Silicon) and Windows
- Publications (>1200 citations in total)
  - Mahé F, Rognes T, Quince C, de Vargas C, Dunthorn M. (2014) **Swarm: robust and fast clustering method for amplicon-based studies**. PeerJ 2, e593.
  - Mahé F, Rognes T, Quince C, de Vargas C, Dunthorn M. (2015) **Swarm v2: highly-scalable and high-resolution amplicon clustering**. PeerJ 3, e1420.
  - Mahé F, Czech L, Stamatakis A, Quince C, de Vargas C, Dunthorn M, Rognes T (2022) **Swarm v3: towards tera-scale amplicon clustering.** Bioinformatics 38, 267-269.

Main international collaborators

Frédéric Mahé
CIRAD, Montpellier, France

Lucas Czech
Heidelberg Inst. for Theoretical Studies, Germany

Tomáš Flouri
Heidelberg Inst. for Theoretical Studies, Germany

Christopher Quince
Warwick & Glasgow Univ., UK


Thank you!