USEARCH Suite
and
UPARSE Pipeline

Susan Huse
Brown University
August 7, 2015
USEARCH
Robert Edgar

USEARCH and UCLUST Edgar (201) Bioinformatics 26(19)

UCHIME Edgar et al. (2011) Bioinformatics 27(16)


UNOISE Edgar et al (2015) Bioinformatics 31(15)

http://drive5.com/usearch/
USEARCH

Search for sequence similarity between your sequence (query) and a database of reference sequences.

Global – match full-length of the query
Local – match sections of the query similar to BLAST
1) Reference database is k-mer indexed (*.udb)
2) k-mer index used to prioritize likely reference sequence matches
3) align query and best matches, calculate identity (% same / total)
4) If ID within threshold = hit, else = reject.
5) Stop when too many rejects
UPARSE

USEARCH-based pipeline for quality- and chimera-filtering raw reads and creating OTUs

http://drive5.com/usearch/manual/uparse_pipeline.html
http://drive5.com/usearch/manual/uparse_cmds.html

Why UPARSE

The UPARSE pipeline has been optimized to reduce spurious OTUs, minimize the effect of sequencing errors, reduce OTU inflation and more closely reflect the true community diversity.

UPARSE dramatically reduces the number of errant OTUs generated when sequencing mock communities.
UPARSE Pipeline

Quality Filter
- Barcode Removal
  - fastq_strip_barcode_relabel2.py
- Maximum Expected Error Filtering
  - fastq_filter
- Paired End Merger
  - fastq_mergepairs

Abundance
- Dereplication
  - derep_fulllength
- Abundance Sort
  - sortbysize
- Singleton Filtering
  - minsize 2

Clustering
- OTU Clustering
  - cluster_otus
- Chimera Filtering
  - uchime_ref
- Map Reads to OTUs
  - usearch_global
- OTU Table
  - uc2otutab.py

http://drive5.com/usearch/manual/uparse_cmds.html
Prep for the lab

Login to your home directory

Load the stamps software:

```
module load stamps
```

Create a subdirectory for this lab:

```
mkdir uparse
cd uparse
cp /class/shared/uparse_commands.sh .
```
DO NOT COPY COMMANDS FROM THIS POWERPOINT!

Copying commands from Word or Powerpoint often introduces newlines and other characters.

Follow the uparse_commands.sh file!!!
usearch
   -fastq_mergepairs /class/shared/fastqfiles/TTAGGC_ACTGC_1_R1.fastq
   -reverse /class/shared/fastqfiles/TTAGGC_ACTGC_1_R2.fastq
   -fastqout s1.merged.fastq
   -fastq_truncqual 3
   -fastq_maxdiffs 3
   -fastq_minovlen 20
   -fastq_minmergelen 200
Merged pair stats

00:26 3.0Mb 100.0% 459.9k recs, 378210 merged (82.2%)

459879 Pairs
378210 Converted (82.2%)
207859 Exact overlaps (45.20%)
11008 Not aligned (2.39%)
70653 Too many diffs (max=3) (15.36%)
324 Gaps
838914 Mismatches
162359 Fwd errs
676555 Rev errs
0 Merged too short (< 200)
8 Staggered
barcodes fasta

$ cat /class/shared/uparse_barcodes.fa

>Sample1
ACTGC
>Sample2
TGACT
Split on barcode, remove primers

fastq_strip_barcode_relabel2.py
s1.merged.fastq
CCAGCAGCAGYGCAGGTAAN
/class/shared/uparse_barcodes.fa
Sample1_ > s1.named.fq
Quality Filter Merged Pairs

text:

usearch
  -fastq_filter s1.named.fq
  -fastaout s1.filtered.fa
  -fastq_maxee 0.5
Filtered Stats

00:08 2.4Mb  100.0% Converting, 378.2k recs, 333093 converted (88.1%)

378210  FASTQ recs (378.2k)
45117   Low qual recs discarded  
         (expected errs > 0.50)
333093  Converted (333.1k, 88.1%)
Multiple samples

Combine the filtered reads from samples before dereplication and further analysis:

```bash
cat s1.filtered.fa s2.filtered.fa > filtered.fa
```
Dereplicate (count dupes)

usearch
-derep_fulllength filtered.fa
-output derep.fa
-sizeout
Size attribute added to fasta file

$ grep "">" derep.fa | head

>Sample1_17;barcodelabel=Sample1;size=33007;
>Sample1_20;barcodelabel=Sample1;size=25186;
>Sample1_21;barcodelabel=Sample1;size=19088;
>Sample1_25;barcodelabel=Sample1;size=17231;
>Sample1_107;barcodelabel=Sample1;size=16512;
>Sample1_15;barcodelabel=Sample1;size=15457;
>Sample1_39;barcodelabel=Sample1;size=14938;
>Sample1_37;barcodelabel=Sample1;size=14197;
>Sample1_75;barcodelabel=Sample1;size=13270;
>Sample1_184;barcodelabel=Sample1;size=13170;
Remove singletons before creating OTU representatives

usearch
-sortbysize derep.fa
-output sorted.min2.fa
-minsize 2
De novo Clustering

usearch
-cluster_otus sorted.min2.fa
-otus otu_reps.init.fa
-uc otu_reps.init.up [-uparseout]
-relabel OTU_
-sizein
-sizeout
Clustering report

01:23  83Mb  100.0%  40 OTUs

Input seqs  21092 (21.1k)
           OTUs  40
         Members  21020 (21.0k)
        Chimeras  32
     Max mem  83Mb
       Time   01:23
Throughput  254.1 seqs/sec.
Reference-based chimera removal

usearch
-uchime_ref otu_reps.init.fa
-db /class/stamps-software/gold.fa
-strand plus
-nonchimeras otu_reps.fa
Chimera detection

00:00 2.4Mb Reading otu_reps.init.fa, 17.7kb
00:00 2.4Mb 40 seqs, min 385, avg 391, max 394nt
00:00 2.4Mb Reading /class/stamps-software/gold.fa, 16Mb
00:00 18Mb 10362 (10.4k) seqs, min 1205, avg 1470, max 1655nt
00:01 19Mb 100.0% Masking
00:02 19Mb 100.0% Word stats
00:02 78Mb 100.0% Building slots
00:03 78Mb 100.0% Build index
00:03 501Mb 100.0% Search 1/40 chimeras found (2.5%)
00:03 501Mb 100.0% Writing 39 non-chimeras
Map reads to OTU reps

usearch
-usearch_global filtered.fa
-db otu_reps.fa
-strand plus
-id 0.97
-uc otu_map.uc
# OTU mapping output

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Record type S, H, C or N (see table below).</td>
</tr>
<tr>
<td>2</td>
<td>Cluster number (0-based).</td>
</tr>
<tr>
<td>3</td>
<td>Sequence length (S, N and H) or cluster size (C).</td>
</tr>
<tr>
<td>4</td>
<td>For H records, percent identity with target.</td>
</tr>
<tr>
<td>5</td>
<td>For H records, the strand: + or - for nucleotides, . for proteins.</td>
</tr>
<tr>
<td>6</td>
<td>Not used, parsers should ignore this field. Included for backwards compatibility.</td>
</tr>
<tr>
<td>7</td>
<td>Not used, parsers should ignore this field. Included for backwards compatibility.</td>
</tr>
<tr>
<td>8</td>
<td><strong>Compressed alignment</strong> or the symbol '=' (equals sign). The = indicates that the query is 100% identical to the target sequence (field 10).</td>
</tr>
<tr>
<td>9</td>
<td>Label of query sequence (always present).</td>
</tr>
<tr>
<td>10</td>
<td>Label of target sequence (H records only).</td>
</tr>
</tbody>
</table>
OTU mapping output

$ less otu_map.uc
Mapped reads

00:00 19Mb Reading otu_reps.fa, 17.2kb
00:00 19Mb 39 seqs, min 385, avg 391, max 394nt
00:00 19Mb 100.0% Masking
00:00 19Mb 100.0% Word stats
00:00 21Mb 100.0% Building slots
00:00 21Mb 100.0% Build index
00:07 500Mb 100.0% Searching, 99.5% matched

$ cut -f 1 otu_map.uc | sort | uniq -c
  590985 H
  2998 N
Create an OTU table

python /class/stamps-software/bin/uc2otutab.py otu_map.uc
> otu_table.txt
## OTU Table

```bash
$ head otu_table.txt
OTUIId  Sample1  Sample2
Sample1_17;barcodelabel=Sample1;size=414314;  226901  237498
Sample1_108;barcodelabel=Sample1;size=22015;  27137  856
Sample1_2727;barcodelabel=Sample1;size=824;1233  52
Sample1_49;barcodelabel=Sample1;size=34987;23173  18698
Sample1_99;barcodelabel=Sample1;size=12579;15922  636
Sample1_50;barcodelabel=Sample1;size=15548;18838  592
Sample1_2952;barcodelabel=Sample1;size=818;1243  4
Sample1_76;barcodelabel=Sample1;size=3216;  4556  82
Sample1_4362;barcodelabel=Sample1;size=1198;  1394  515
```