Supplementary data for "SortMeRNA: Fast and accurate filtering of ribosomal RNAs in metatranscriptomic data" (Kopylova E., Noé L., Touzet H.)

1 Algorithm Overview

In this section, we give further details for the data structures behind SortMeRNA, more notably the Burst trie lookup table introduced in Section 2.2 of the paper, and the universal Levenshtein automaton introduced in Section 2.3 of the paper.

1.1 The Burst trie lookup table

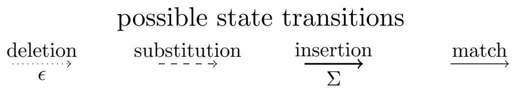
For a read to be accepted by SortMeRNA, it must contain a threshold ratio r of accepted windows (further discussed in *Section 2.3*). Note that a window of an equal length s will have a prefix and a suffix of an equal length $\frac{s}{2}$. Prior to passing a window of length s to the Burst trie, its half window of length $\frac{s}{2}$ must occur at least q number of times in rRNA database. All $\frac{s}{2}$-mer occurrences are stored in the Burst trie lookup table. The purpose of the threshold q is to help avoid false positive window hits with the rRNA database, since infrequent $\frac{s}{2}$-mers generally belong to less conserved regions of a database. The algorithm to compute q ascertains that each read (in the set of reads provided by the user) has at least ratio r windows, of which at least one $\frac{s}{2}$-mer has an occurrence greater than q in the rRNA database.

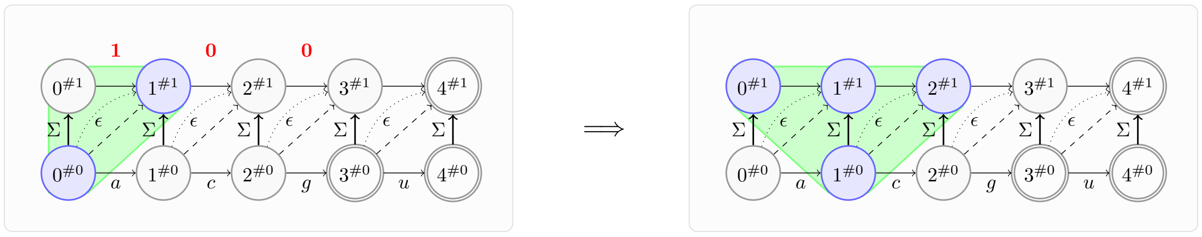
1.2 The universal Levenshtein automaton

The universal Levenshtein automaton for $k = 1$ accepts two words if the number of edit operations (substitution, insertion or deletion) shared by the two words is no more than 1. This automaton is precomputed and can recognize any two words of various lengths, hence the *universal* property.

The mathematics behind constructing this automaton are well described in [Schulz K. and Mihov S., 2002], however here we give a simple example of how the *universal* property can be identified using a classical non-deterministic Levenshtein automaton. Figure 1 illustrates a sequence of snapshots for the active states of a non-deterministic Levenshtein automaton for the word *acgu*, as the input pattern *agu* is introduced into the automaton character by character. The universal Levenshtein automaton determinises the active states in the green triangle, and the changes of transitions are carried out using characteristic bitvectors (in red) rather than individual letters of the input pattern.

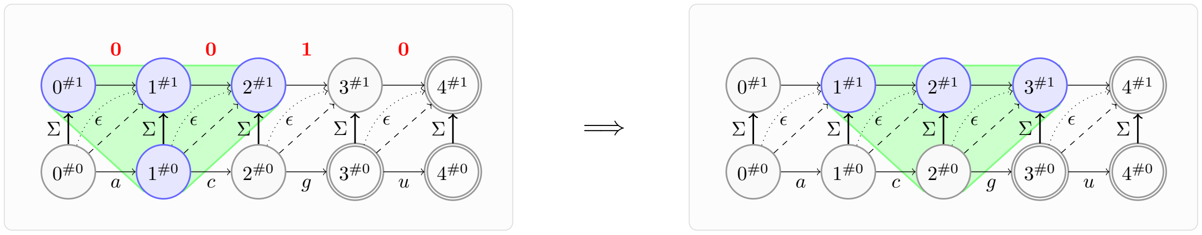
Figure 1: **A non-deterministic Levenshtein automaton for the word 'acgu'.** The $s^{\#e}$ notation for each state corresponds to s number of characters read in the pattern p and e number of errors recorded. The initial state is $0^{\#0}$, the final states are $3^{\#0}$, $4^{\#0}$ and $4^{\#1}$, and the active states are illustrated in blue color. The green triangle represents the boundary of all possible active states after a character is consumed by the automaton. The pattern to be consumed is *agu*. The red binary sequences are the characteristic bitvectors between the input pattern *agu* and the automaton word *acgu*, defined in Section 2.3 of the paper. If a bit of a bitvector is set to 1, the match transition is possible for active states in the adjacent left column of the bit in the automaton. Otherwise, if a bit is set to 0, the match transition is not permitted. Each step corresponds to consuming one character of *agu* by the automaton.





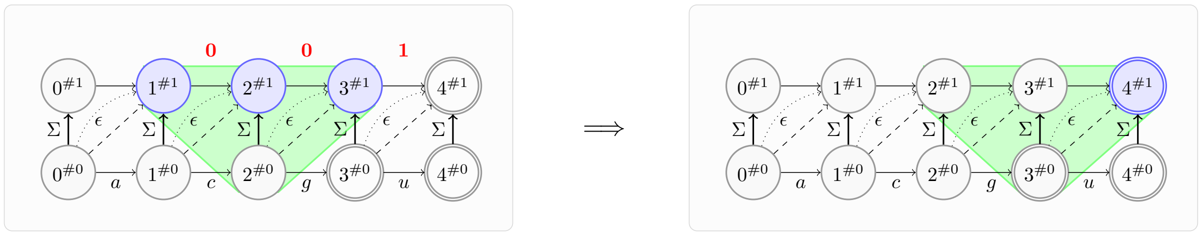
Step 1. $(a, \$acg) = 0100$

no accepting state



Step 2. $(g, acgu) = 0010$

no accepting state



Step 3. $(u, cgu) = 001$

at least one activated accepting state (accept)

2 Parameter Setting

In this section, we describe all data used in Section 2.5 of the paper: The construction of the four rRNA databases Set 1, Set 2, Set 3 and Set 4 (Section 2.1), the construction of the simulated reads (Section 2.2), and the behaviour of SortMeRNA on those databases for varying values of the parameters (Section 2.3). The goal of these experiments is to be able to determine robust values for parameters of SortMeRNA that fit to various kinds of databases.

2.1 Construction of rRNA databases: Set 1, Set 2, Set 3 and Set 4

Website link for Set 1 - Set 4 generated databases: Click-here.

Four different database sets were constructed to measure the behavior of SortMeRNA,

Set 1	80% identity 16S rRNA bacteria & archaea (2262 sequences)
Set 2	80% identity 16S rRNA bacteria & archaea + truncated phylogeny tree (2187 sequences)
Set 3	95% identity 23S rRNA bacteria & archaea (1969 sequences)
Set 4	95% identity 23S rRNA bacteria & archaea + truncated phylogeny tree (1906 sequences)

Each database was constructed by applying the ARB package [Ludwig W. et al., 2004] and UCLUST [Edgar R., 2011] to the small 16S and large 23S subunit databases from SILVA [Pruesse E. et al., 2007].

16S rRNA data source: `http://www.arb-silva.de/no_cache/download/archive/release_108/ARB_files/SSURef_108_SILVA_NR_99_11_10_11_opt_v2.arb.tgz`

23S rRNA data source: `http://www.arb-silva.de/no_cache/download/archive/release_108/ARB_files/LSURef_108_SILVA_16_08_11_opt.arb.tgz`

The procedure is as follows (see also Figures 2,3,4,5 for each Set),

1. use the ARB package to extract the phylogeny trees of 16S rRNA and 23S rRNA bacteria & archaea databases in fasta and xml formats,

2. for Set 2 and Set 4, remove a branch in the phylogenetic tree to induce missing species (see Figures 3 and 5)

3. remove all occurrences of rRNA other than 16S or 23S using description information in the xml format, then apply PRINSEQ [Schmieder R. and Edwards R., 2011] to remove long sequences (1600 for 16S and 5000 for 23S) and those having $> 1\%$ of ambiguous N's,

   command:
   ```
   perl prinseq-lite.pl -log -ns_max_p 1 -max_len [1600 or 5000]
   -fasta <infile> -out_good <outfile> -out_bad null
   ```

4. apply UCLUST on the filtered set of rRNAs to create representative 16S and 23S rRNA databases with identity $x\%$,

   command:
   ```
   usearch -cluster <filtered 16S/23S sorted fasta file> -id 0.x
   -seedsout <representative database file>
   ```

The identity percentage x refers to the definition of clusters used by UCLUST: Each cluster is defined by a representative sequence, and each sequence in a cluster matches the representative sequence according to the identity threshold. Finally, each database is constituted by the set of representative sequences. By construction, every pair of sequences in the representative database has an identity percentage lower than the threshold x.

Website link for bash script to carry out Steps 2-4 above: Click-here.

Figure 2: Construction of Set 1 – 16S rRNA database with 80% identity

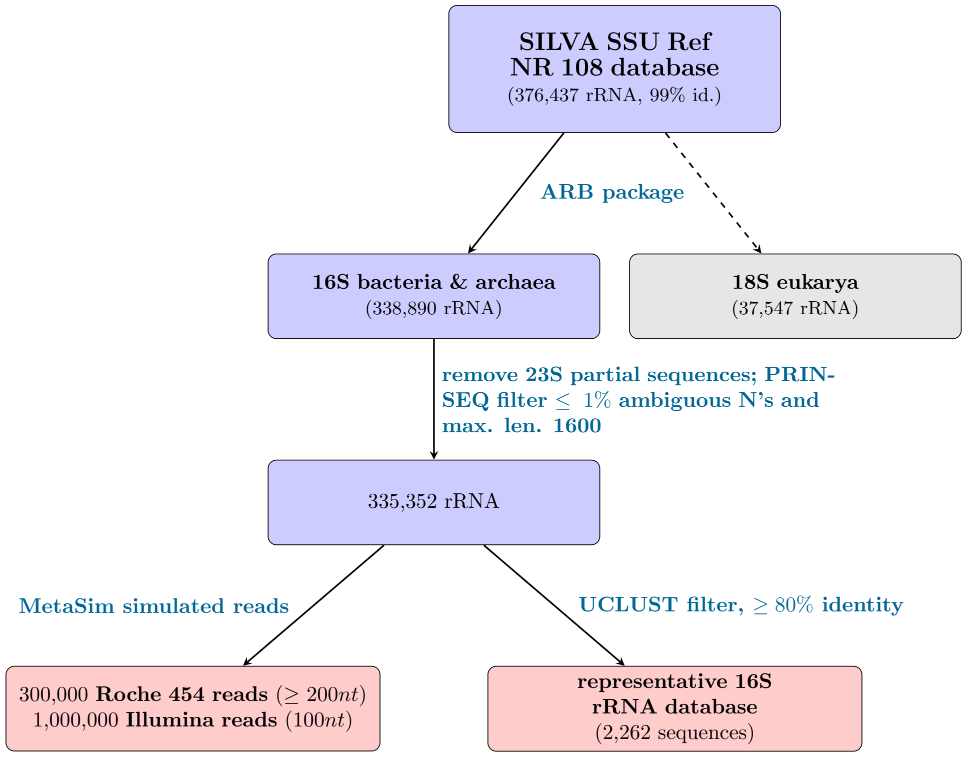

SILVA SSU Ref NR 108 database
(376,437 rRNA, 99% id.)
ARB package
16S bacteria & archaea
(338,890 rRNA)
18S eukarya
(37,547 rRNA)
remove 23S partial sequences; PRINSEQ filter ≤ 1% ambiguous N's and max. len. 1600
335,352 rRNA
MetaSim simulated reads
UCLUST filter, ≥ 80% identity
300,000 Roche 454 reads (≥ 200nt)
1,000,000 Illumina reads (100nt)
representative 16S rRNA database
(2,262 sequences)

Figure 3: Construction of Set 2 – 16S rRNA database with 80% identity + truncated phylo. tree

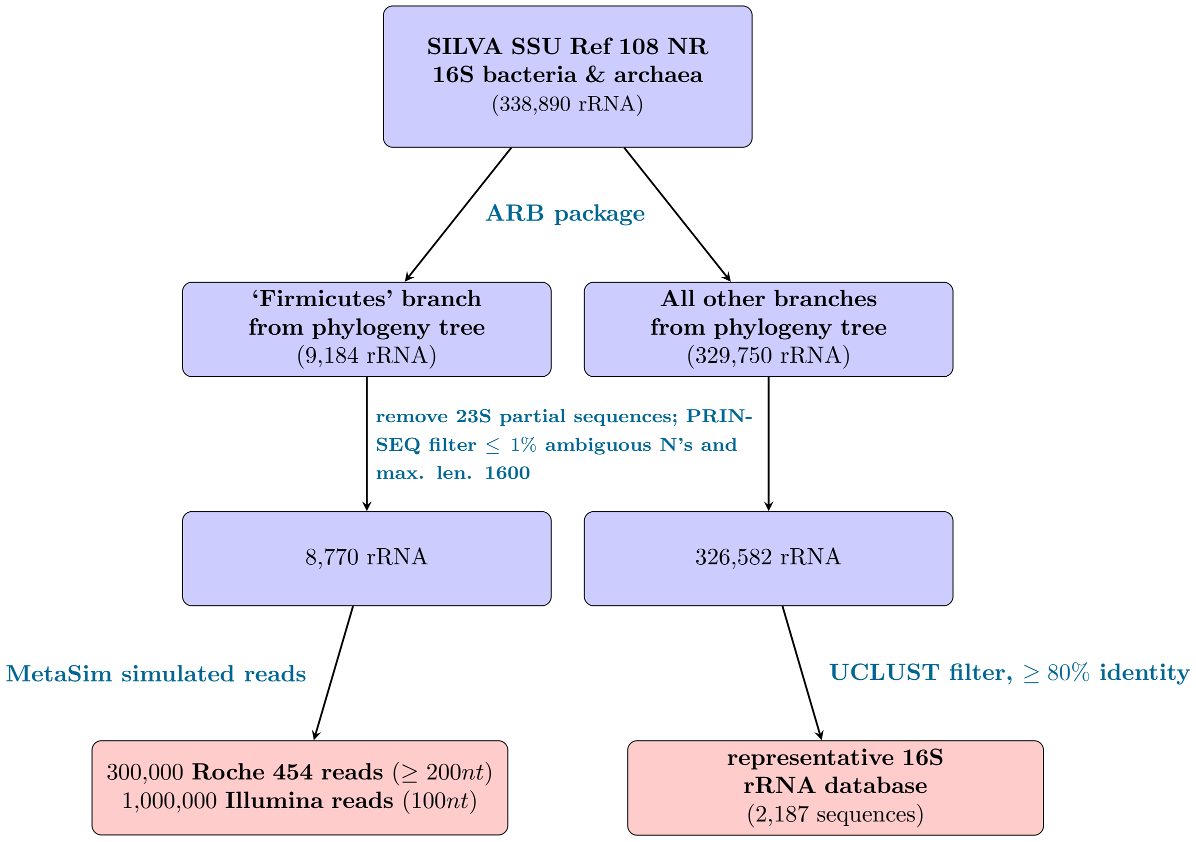

SILVA SSU Ref 108 NR
16S bacteria & archaea
(338,890 rRNA)
ARB package
'Firmicutes' branch
from phylogeny tree
(9,184 rRNA)
All other branches
from phylogeny tree
(329,750 rRNA)
remove 23S partial sequences; PRINSEQ filter $\leq$ 1% ambiguous N's and max. len. 1600
8,770 rRNA
326,582 rRNA
MetaSim simulated reads
UCLUST filter, $\geq$ 80% identity
300,000 **Roche 454 reads** ($\geq 200nt$)
1,000,000 **Illumina reads** ($100nt$)
representative 16S
rRNA database
(2,187 sequences)

Figure 4: Construction of Set 3 – 23S rRNA database with 95% identity

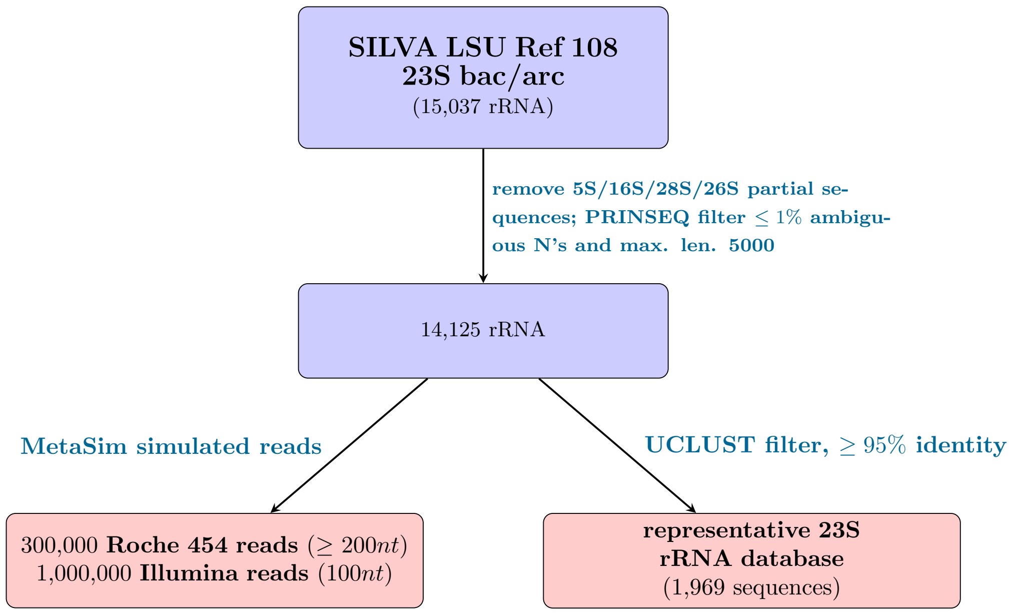
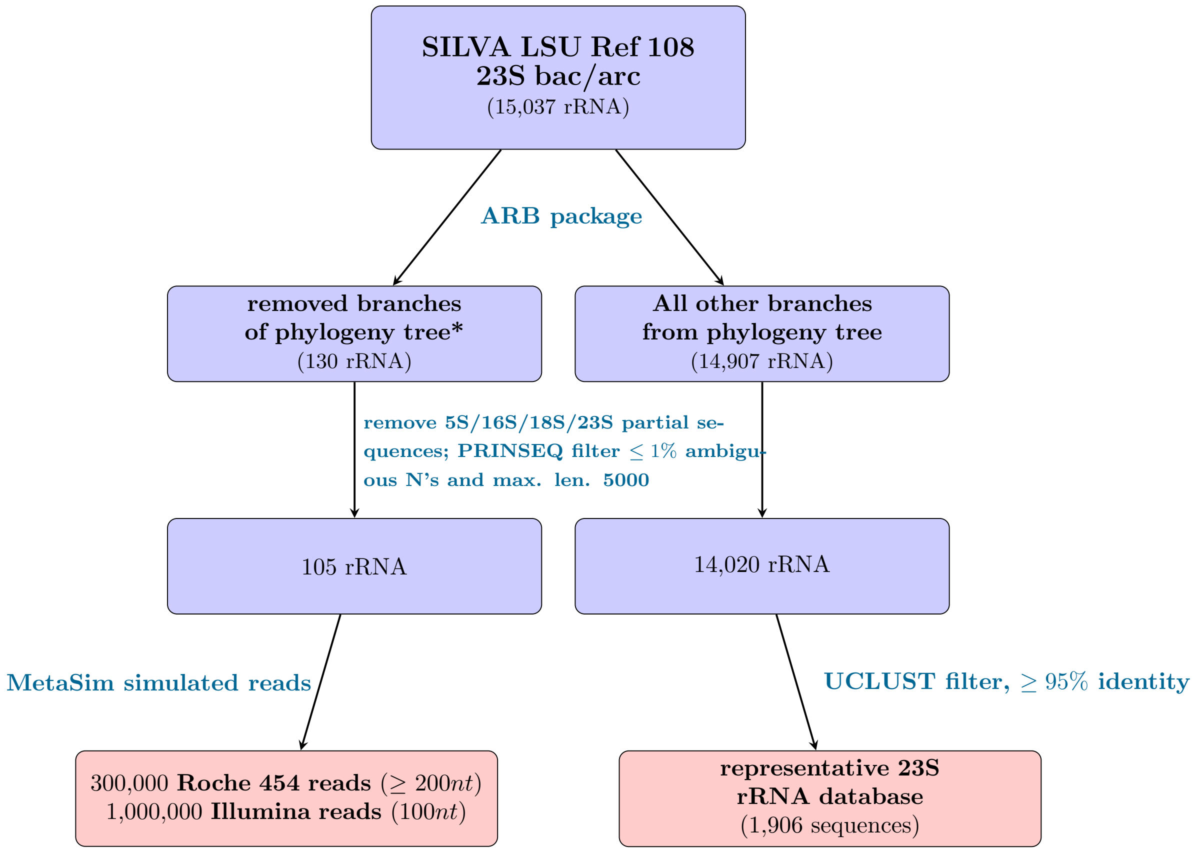

SILVA LSU Ref 108
23S bac/arc
(15,037 rRNA)
remove 5S/16S/28S/26S partial sequences; PRINSEQ filter $\leq 1\%$ ambiguous N's and max. len. 5000
14,125 rRNA
MetaSim simulated reads
UCLUST filter, $\geq 95\%$ identity
300,000 **Roche 454 reads** ($\geq 200nt$)
1,000,000 **Illumina reads** ($100nt$)
representative 23S rRNA database
(1,969 sequences)

Figure 5: Construction of Set 4 – 23S rRNA database with 95% identity + truncated phylo. tree


SILVA LSU Ref 108
23S bac/arc
(15,037 rRNA)
ARB package
removed branches of phylogeny tree*
(130 rRNA)
All other branches from phylogeny tree
(14,907 rRNA)
remove 5S/16S/18S/23S partial sequences; PRINSEQ filter ≤ 1% ambiguous N's and max. len. 5000
105 rRNA
14,020 rRNA
MetaSim simulated reads
UCLUST filter, ≥ 95% identity
300,000 **Roche 454 reads** (≥ 200*nt*)
1,000,000 **Illumina reads** (100*nt*)
representative 23S rRNA database
(1,906 sequences)

*section of phylogeny tree: 36 Planctomycetes, 14 Fibrobacteres, 44 Verrucomicrobia, 21 Chloroflexi_1, 6 Candidate division TM7, and 9 Lentisphaerae.

2.2 Simulated reads

Website link for all simulated reads files: Click-here.

We generated simulated reads to estimate the selectivity and the sensitivity of SortMeRNA. For that, we applied MetaSim 0.9.5 [Richter D.C. et al., 2008] using provided error models: Roche 454 and Illumina. MetaSim's maximum length error model for the Illumina technology is $80nt$, in order to adapt this model for $100nt$ the last probability value of the 80^{th} position was extended by $20nt$. In practice, MetaSim had simulated Roche 454 reads with 2.8-3% sequencing error rate of which approximately 79% were insertions and 21% deletions. Additionally, for Illumina reads, the sequencing error rate was 1.2% of which 100% were substitutions.

Generation of Roche 454 and Illumina rRNA reads

We started from sequences of the SILVA database that have not been already selected in the representative set (Set 1 to Set 4).

1. apply MetaSim on the filtered set of rRNAs of SILVA not belonging to the representative databases to create 300,000 Roche 454 reads of $\geq 200nt$ and 1,000,000 Illumina reads of $100nt$.

   In the MetaSim simulator settings, the parameters ‘Mean’ (mean length of clone) and ‘Second Parameter’ (standard deviation of clone length) as defined in the user manual, were set to 1000 and 100 respectively. The default values are 2000 and 200, however since many rRNA sequences are shorter than 2000 nt, we reduced this value to 1000 and the standard deviation to 100.

2. filter out ‘circular’ reads produced by MetaSim, approximately 10% of the reads. There is no direct method to disable the option for generating ‘circular’ reads for *all* sequences in the database. Other than the suggestions given here: `http://ab.inf.uni-tuebingen.de/software/metasim/welcome.html#FAQ` , a working alternative is to filter them out afterwards by using a simple bash command,

```
awk ‘/^>/ {printf("\n%s\t",$0);next;} {printf("%s",$0);} END {printf("\n");}’ \
< all-reads.fasta \
| egrep -v ‘^$’ \
| awk -F "KEY=" ‘{i=NF-1; if(i==1) print $0}’ \
| tr "\t" "\n" \
> linear-reads.fasta
```

3. apply PRINSEQ [Schmieder R. and Edwards R., 2011] to filter out reads shorter than $200nt$ (only for Roche 454) and reads with $> 1\%$ of ambiguous character N (both Roche 454 & Illumina).

Website link for bash script to carry out Steps 2-3 above (Roche 454): Click-here.

Generation of Roche 454 and Illumina non-rRNA reads

We started from the NCBI complete bacterial genomes :
`ftp://ftp.ncbi.nlm.nih.gov/genomes/Bacteria/{all.gbk.tar.gz,all.fna.tar.gz}`

1. apply FeatureExtract 1.2 [Wernersson R., 2005] to find all locations of rRNAs on the complete NCBI bacterial genomes in the Genbank file format,

   command:
   ```
   python gb2tab.py -f 'rRNA' -q --entryname <infile> > <outfile>
   ```

2. mask the locations of rRNA provided by (1) in the Fasta file format by a contiguous sequence of N's (+150nt to cover misannotation)

   Website link for bash/perl scripts used in Step 2: Click-here.

3. run MetaSim on the NCBI complete bacterial genomes with masked rRNAs to create 1,000,000 Roche 454 reads of $\geq 200nt$ and 1,000,000 Illumina reads of $100nt$,

4. filter out 'circular' reads produced by MetaSim, approximately 10% of the reads.

   In the MetaSim simulator settings, the parameters 'Mean' (mean length of clone) and 'Second Parameter' (standard deviation of clone length) as defined in the user manual, were set to 2000 and 200 respectively. These default values work well since the average length of a bacterial genome exceeds 2000 nt.

5. apply PRINSEQ to filter out reads shorter than $200nt$ (only for Roche 454) and reads with $>$ 1% of ambiguous character N (both Roche 454 & Illumina).

Important note: FeatureExtract can only detect rRNAs which are featured as 'rRNA' in the Genbank files. In several cases, the rRNA molecules are not annotated and therefore missed during the masking process. This explains the lower selectivity for all programs in Table B.

Filtering of metatranscriptomic SRR106861 and SRR013513 read sets

Software such as riboPicker, Meta-RNA, rRNASelector and SSU-ALIGN require longer reads to achieve a higher sensitivity. In general, this length is suggested to be $\geq 200nt$ for Roche 454 reads. We filter the metatranscriptomic SRR106861 and SRR013513 read sets to remove any bias caused by shorter reads ($<200nt$), as well as low-quality reads which have more than 1% of the ambiguous character N. The software tool used to filter the reads was PRINSEQ.

command:
```
perl prinseq-lite.pl -log -ns_max_p 1 -min_len 200 -fasta <infile>
        -out_good <outfile> -out_bad null
```

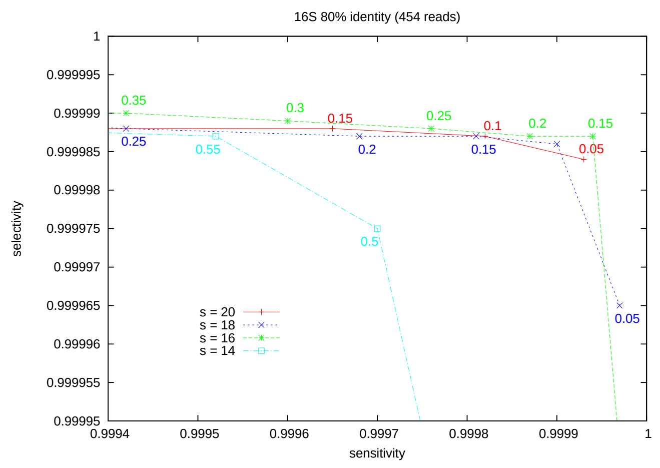
2.3 Results of SortMeRNA with varying parameters

SortMeRNA uses two parameters: The length s of the sliding window and the threshold r of accepted windows to accept a read.

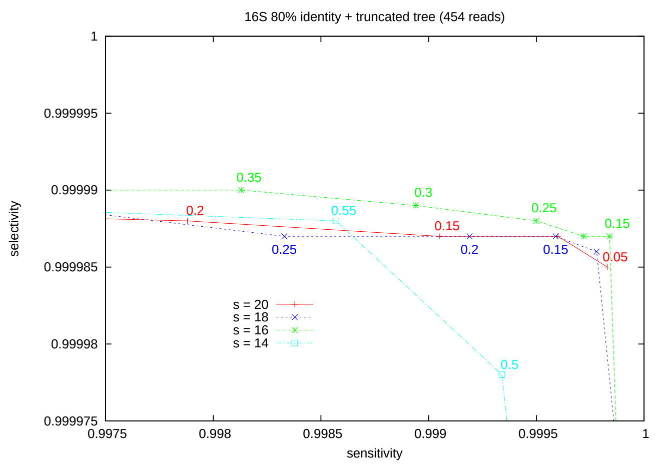
Let ℓ be the length of a read, and ϵ the number of windows accepted by the Burst trie for this read. A read is classified as rRNA if the number of accepted windows divided by the number of total full windows on a read is greater or equal to r,

$$\frac{\epsilon}{\ell - s + 1} \geq r. \tag{1}$$

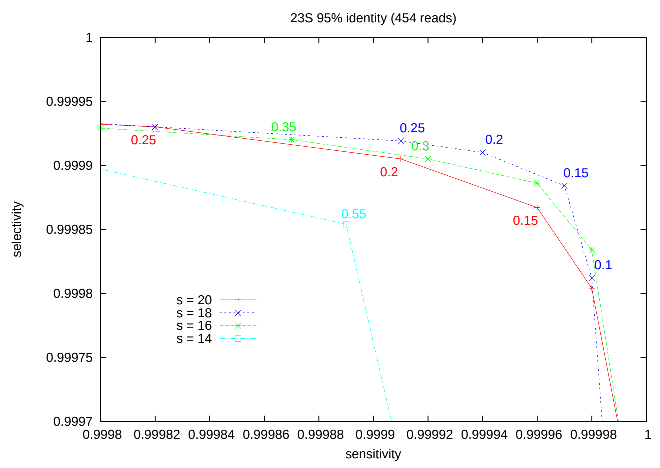
To estimate robust values for s and r, we ran SortMeRNA on Set 1 to Set 4 and varied the parameters $s \in [14, 16, 18, 20]$ and $r \in [0.05, 0.10, 0.15, \ldots, 0.95]$. The results for Roche 454 reads are demonstrated in Figure 6 and those for Illumina reads in Figure 7. From these tests, we have set $s = 18$, $r = 0.15$ for Roche 454 reads and $r = 0.25$ for Illumina reads. However, as shown in the Matthews correlation coefficient tables, short ranges within any direction for r when $s = 18$ render similar results.



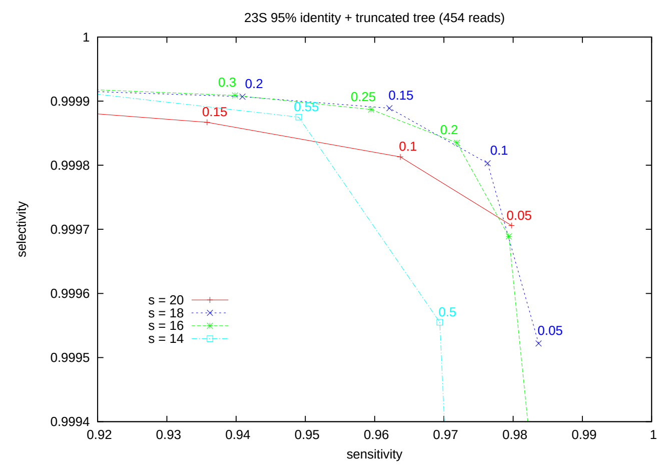
Set 1



Set 2



Set 3

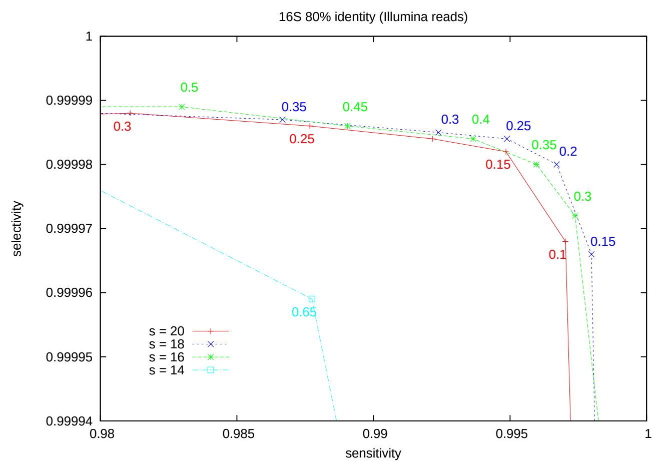


Set 4

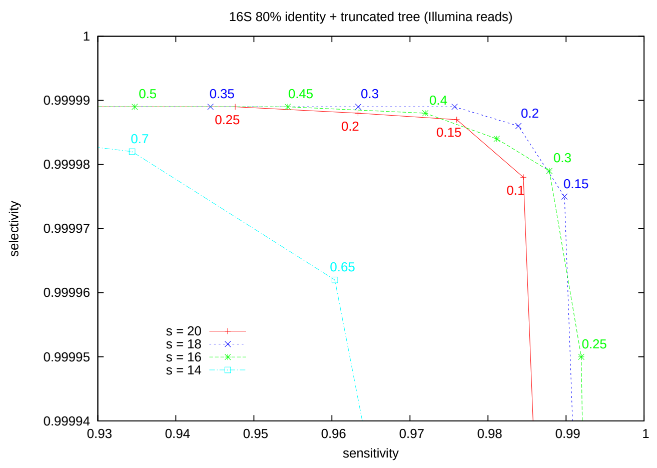
Matthews correlation coefficients for $s = 18$ and various values of r

r	0.05	0.1	0.15	0.2	0.25	0.3	0.35
Set 1	0.9999	0.9999	0.99985	0.99976	0.99959	0.99933	0.99895
Set 2	0.99987	0.99982	0.99971	0.99945	0.99889	0.99773	0.99536
Set 3	0.99895	0.99958	0.99973	0.99977	0.99977	0.99973	0.99970
Set 4	0.98834	0.98417	0.97512	0.96131	0.93811	0.89803	0.84026

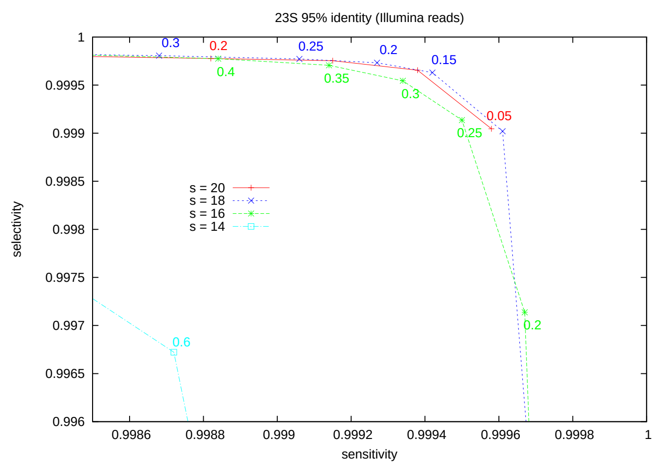
Figure 6: SortMeRNA results by varying parameters s (length of the sliding window) and r (ratio of accepted windows) on Set 1 to Set 4 for Roche 454 simulated reads. For each graph, each curve corresponds to a different value for s: $s = 14, 16, 18, 20$. Each dot on a curve corresponds to a different value for r.



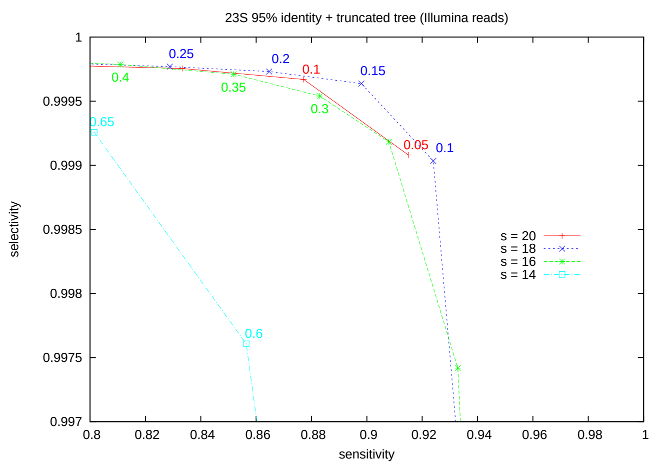
Set 1



Set 2



Set 3



Set 4

Matthews correlation coefficients for $s = 18$ and various values of r

r	0.05	0.1	0.15	0.2	0.25	0.3	0.35
Set 1	0.99598	0.99856	0.99795	0.99670	0.99489	0.99240	0.98674
Set 2	0.99366	0.99395	0.98982	0.98401	0.97600	0.96401	0.94588
Set 3	0.99306	0.99863	0.99905	0.99901	0.99883	0.99849	0.99792
Set 4	0.94040	0.92567	0.90233	0.87242	0.84096	0.80262	0.75228

Figure 7: SortMeRNA results by varying parameters s (length of the sliding window) and r (ratio of accepted windows) on Set 1 to Set 4 for Illumina simulated reads. For each graph, each curve corresponds to a different value for s: $s = 14, 16, 18, 20$. Each dot on a curve corresponds to a different value for r.

3 Experimental Evaluation

In this section, we give information on the experiments conducted in Section 4 of the paper: The parameters and commands used to run the software tools (Section 3.1), the construction of the rRNA databases used (Section 3.2). We also give in Section 3.3 new experimental results for simulated reads against the 23S rRNA database.

3.1 Commands

Website link for bash scripts used to test riboPicker, BLASTN, Meta-RNA and SSU-ALIGN: Click-here.

We list the commands used to measure the performance of SortMeRNA in comparison with riboPicker, BLASTN, Meta-RNA, rRNASelector and SSUALIGN. The following is a legend of common input files required by the software,

`<infile>`: the input reads file in fasta format
`<outfile>`: the matching reads output file
`<db>`: the formatted database file
`<dbinfile>`: the rRNA sequences fasta file
`<alignmentdir>`: output directory for multiple sequence alignment files

riboPicker 0.4.3 [Schmieder R. et al., 2011]

source: `http://sourceforge.net/projects/ribopicker/files/standalone/`
pre-built public databases for 16S and 23S rRNA: `ftp://edwards.sdsu.edu:7009/ribopicker/db/{gg20110531.tar.gz, hmp20111206.tar.gz, ncbibact20120117.tar.gz, rdp1028.tar.gz}`
build personal database: `bwa64 index -p <db> <dbinfile>`
command (using personal database): `perl ribopicker.pl -c 50 -i 75 -z 3 -f <infile> -dbs <db> -id <id>`
command (using pre-built databases 16S): `perl ribopicker.pl -c 50 -i 75 -z 3 -f <infile> -dbs ssr108arcbac,rdp1028,gg20110531,ncbibact2012011716S,hmp2011120616S -id <id>`
command (using pre-built databases 23S): `perl ribopicker.pl -c 50 -i 75 -z 3 -f <infile> -dbs slr108bacarc,ncbibact2012011723S,hmp2011120623S -id <id>`

Both databases 'ssr108arcbac' and 'slr108bacarc' are SILVA 108 databases without 18S rRNA sequences.

BLASTN 2.2.25 [Altschul S.F. et al., 1990]

source: `ftp://ftp.ncbi.nlm.nih.gov/blast/executables/release/2.2.25/`
build personal database: `formatdb -i <dbinfile> -p F -n <db>`
command: `blastall -p blastn -m 8 -a 1 -d <db> -i <infile> -o <outfile>`
parsing: identity 75%, coverage 50%, e-value $< 1e\text{-}5$

Meta-RNA 3 [Huang Y. et al., 2009]

source: `http://weizhong-lab.ucsd.edu/meta_rna/`
command 16S: `python rna_hmm3.py -i <infile> -o <outfile> -k "arc,bac" -m "ssu" -e 1e-5`
command 23S: `python rna_hmm3.py -i <infile> -o <outfile> -k "arc,bac" -m "lsu" -e 1e-5`

rRNASelector 1.0 [Lee J.H. et al., 2010]

source: `http://sw.ezbiocloud.net/sw_detail?uid=rrnaselector`
command: `java -jar rRNASelector.jar`
parameters: e-value: 1e-5; minimum length: 60 (Roche 454), 20 (Illumina)

SSU-ALIGN 1.0 [Narwocki E.P. et al., 2009; Cannone J.J. et al., 2002]

source: `http://selab.janelia.org/software.html`
multiple sequence alignment: `ssu-align <dbinfile> <alignmentdir>`
build personal database: `ssu-build <alignmentdir/multiple.sequence.alignment.stk>`
command: `ssu-align -b 40 -m <db> --no-align <infile> <alignmentdir>`

SortMeRNA 1.0

source: `http://bioinfo.lifl.fr/RNA/sortmerna`
build database command: `buildtrie --input <dbinfile>`
command: `sortmerna [--454 or --I] <infile> --db <db> --accept <outfile>`
parameters: $r = 0.15$ for Roche 454 and $r = 0.25$ for Illumina reads, $s = 18$ (default values)

3.2 rRNA databases

For measuring performance of all software on equal basis, another two databases were created : One for 16S rRNA, and one for 23S rRNA. Note that these two databases are different from those used in *Section 2.3*.

Set 5	85% identity 16S rRNA bacteria & archaea (7659 rRNA)	see Figure 8
Set 6	98% identity 23S rRNA bacteria & archaea (2811 rRNA)	see Figure 9

Figure 8: Construction of Set 5 – representative 16S rRNA database with 85% identity

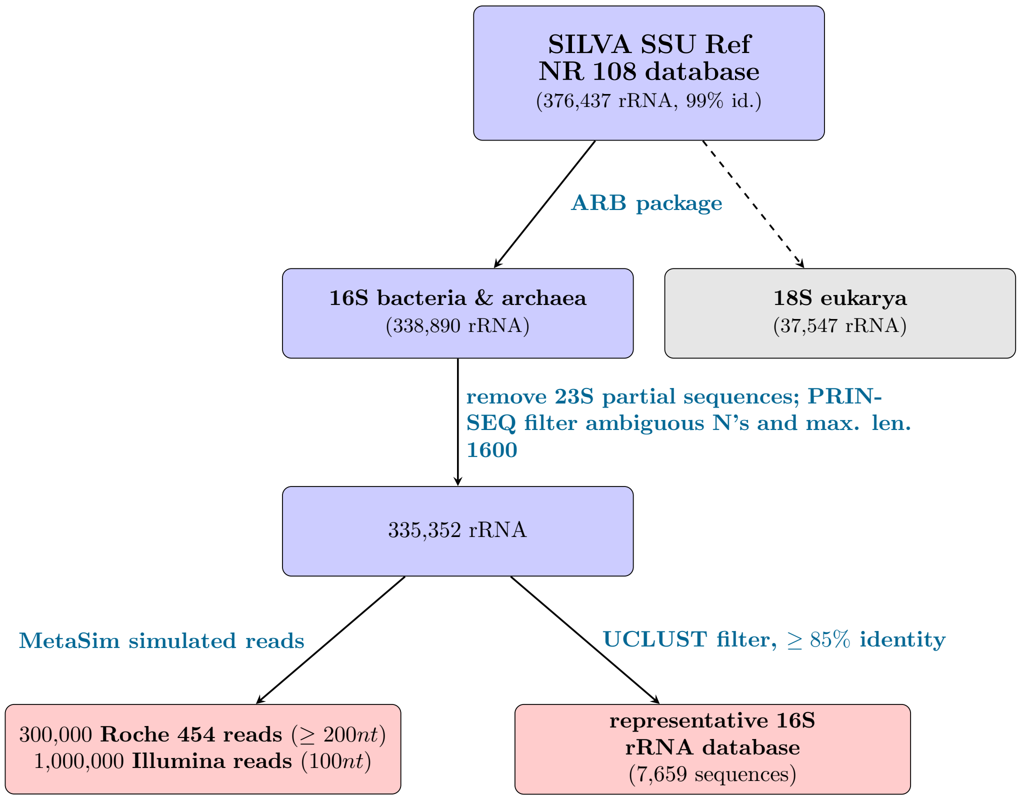
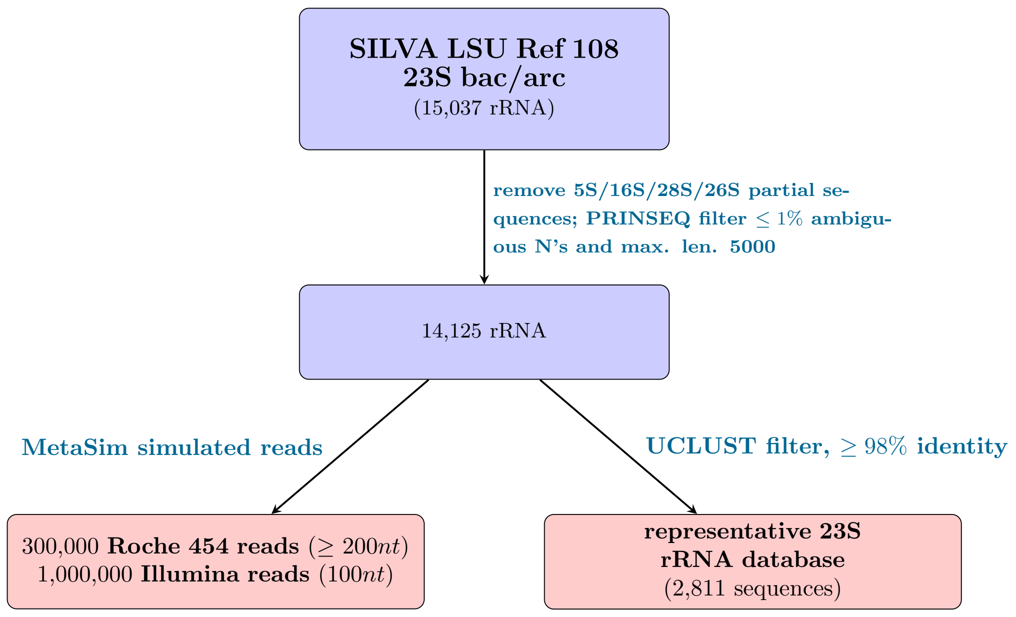

SILVA SSU Ref NR 108 database (376,437 rRNA, 99% id.)
ARB package
16S bacteria & archaea (338,890 rRNA)
18S eukarya (37,547 rRNA)
remove 23S partial sequences; PRINSEQ filter ambiguous N's and max. len. 1600
335,352 rRNA
MetaSim simulated reads
UCLUST filter, ≥ 85% identity
300,000 Roche 454 reads (≥ 200nt)
1,000,000 Illumina reads (100nt)
representative 16S rRNA database (7,659 sequences)

Figure 9: Construction of Set 6 – representative 23S rRNA database with 98% identity


SILVA LSU Ref 108
23S bac/arc
(15,037 rRNA)
remove 5S/16S/28S/26S partial sequences; PRINSEQ filter $\leq 1\%$ ambiguous N's and max. len. 5000
14,125 rRNA
MetaSim simulated reads
UCLUST filter, $\geq 98\%$ identity
300,000 **Roche 454 reads** ($\geq 200nt$)
1,000,000 **Illumina reads** ($100nt$)
representative 23S rRNA database
(2,811 sequences)

3.3 Complementary results for 23S rRNA for simulated reads

In *Section 4* of the original paper are the experimental results for simulated reads compared against Set 5: 16S 85% id. rRNA database (see Table 1 and Table 2 of the paper). Here we present the experimental results for Set 6: 23S 98% id. rRNA database. Reads were simulated using the same protocol, as described in *Section 2.2* of this supplementary file.

Table A below shows the sensitivity for rRNA reads, and Table B gives the selectivity for non-rRNA reads for SortMeRNA, riboPicker, riboPicker*, BLASTN, Meta-RNA and rRNASelector. These experiments confirm the accurate performance of SortMeRNA using a database from an alternate subunit domain.

Table A: SENSITIVITY. 1,000,000 of MetaSim simulated Illumina ($100nt$) and 300,000 Roche 454 ($\geq 200nt$) **rRNA** reads against a representative 98% identity 23S rRNA database of 2,811 sequences.

	Illumina					**Roche 454**				
	rRNA	run time	latency	memory (%)	sensitivity (%)	rRNA	run time	latency	memory (%)	sensitivity (%)
SortMeRNA	999909	1m22s	1x	7.2	99.909	300000	1m24s	1x	4.7	100
riboPicker	659494	19m32s	14x	6.7	65.949	213989	21m23s	15x	5.6	71.329
*riboPicker **	986917	1h26m	63x	8.4	98.691	296584	1h46m	76x	7.3	98.861
BLASTN	999549	15h10m	665x	2.7	99.954	299999	11h25m	489x	1.3	99.999
Meta-RNA	936314	4h25m	194x	31.8	93.631	298918	4h29m	192x	13.1	99.639
rRNASelector	908344	4h7m	181x	16.4	90.834	298733	4h37m	198x	7.2	99.577

* Searching through all 23S rRNA databases provided by SILVA (only 23S), NCBI archaeal and bacterial genomes, and HMP.

Table B: SELECTIVITY. 1,000,000 of MetaSim simulated Illumina ($100nt$) and 300,000 Roche 454 ($\geq 200nt$) **non-rRNA** reads against a representative 98% identity 23S rRNA database of 2,811 sequences

	Illumina					**Roche 454**				
	rRNA	run time	latency	memory (%)	selectivity (%)	rRNA	run time	latency	memory (%)	selectivity (%)
SortMeRNA	243^{+}	1m31s	1x	6.3	99.9757	112^{+}	3m7s	1x	8.9	99.9888
riboPicker	39^{+}	10m18s	7x	6.6	99.9961	24^{+}	31m44s	10x	16.7	99.9976
*riboPicker **	103	30m44s	20x	8.2	99.9897	54	1h32m	29x	18.3	99.9946
BLASTN	310^{+}	26m11s	17x	0.3	99.9690	571^{+}	26m14s	8x	0.2	99.9429
Meta-RNA	36	2m37s	2x	0.1	99.9964	29	6m22s	2x	0.3	99.9971
rRNASelector	34	2m27s	2x	0.1	99.9966	29	5m49s	2x	0.3	99.9971

* Searching through all 23S rRNA databases provided by SILVA (only 23S), NCBI archaeal and bacterial genomes, and HMP. $^{+}$ SortMeRNA, riboPicker and BLASTN use the same database (Set 6). riboPicker* searches through a database with 19,602 23S rRNA sequences, and both Meta-RNA and rRNASelector use prebuilt HMM models. For SortMeRNA, 82% of the 243 Illumina reads and 100% of the 112 Roche 454 reads are in common with the 310 and 571 reads classified by BLASTN. For riboPicker, 97% of the 39 Illumina reads and 100% of the 24 Roche 454 reads are in common with the 310 and 571 reads classified by BLASTN. The majority of these reads map to mRNA. Further investigation showed that due to misannotation, the database for Set 6 was contaminated with several mRNA, and hence the classified reads were correctly spotted in the database.

References

[Altschul S.F. et al., 1990] Altschul S.F. et al., (1990) Basic local alignment search tool, *J. Mol. Biol.*, **215(3)**:403-410, doi:10.1006/jmbi.1990.9999.

[Cannone J.J. et al., 2002] Cannone J.J. et al., (2002) The Comparative RNA Web (CRW) Site: An Online Database of Comparative Sequence and Structure Information for Ribosomal, Intron, and Other RNAs, *BMC Bioinformatics*, **3**:15, doi:10.1186/1471-2105-3-15.

[Edgar R., 2011] Edgar R., (2011) Search and clustering orders of magnitude faster than BLAST, *Bioinformatics*, doi:10.1093/bioinformatics/btr381, url: `http://www.drive5.com/usearch/`.

[Huang Y. et al., 2009] Huang Y. et al., (2009) Identification of ribosomal RNA genes in metagenomic fragments, *Bioinformatics*, **25(10)**:1338-1340, doi:10.1093/bioinformatics/btp161.

[Lee J.H. et al., 2010] Lee J.H. et al., (2010) rRNASelector: a computer program for selecting ribosomal RNA encoding sequences from metagenomic and metatranscriptomic shotgun libraries, *J. Microbiol.*, **49(4)**:689-91, doi:10.1007/s12275-011-1213-z.

[Narwocki E.P. et al., 2009] Nawrocki E.P. et al., (2009) Infernal 1.0: Inference of RNA alignments, *Bioinformatics*, **25**:1335-1337, doi:10.1093/bioinformatics/btp157.

[Pruesse E. et al., 2007] Pruesse E. et al., (2007) SILVA: a comprehensive online resource for quality checked and aligned ribosomal RNA sequence data compatible with ARB, *Nuc. Acids Res.*, **35(21)**:7188-7196, doi: 10.1093/nar/gkm864.

[Richter D.C. et al., 2008] Richter D.C. et al., (2008) A Sequencing Simulator for Genomics and Metagenomics, *PLoS ONE*, **3(10)**, doi:10.1371/journal.pone.0003373.

[Schmieder R. and Edwards R., 2011] Schmieder R. and Edwards R., (2011) Quality control and preprocessing of metagenomic datasets, *Bioinformatics*, **27**:863-864, doi:10.1093/bioinformatics/btr026.

[Schmieder R. et al., 2011] Schmieder R., Lim Y.W. and Edwards R., (2011) Identification and removal of ribosomal RNA sequences from metatranscriptomes, *Bioinformatics*, doi:10.1093/bioformatics/btr669.

[Schulz K. and Mihov S., 2002] Schulz K. and Mihov S., (2002) Fast string correction with Levenshtein automata, *IJDAR*, **5**:67-85.

[Wernersson R., 2005] Wernersson R., (2005) FeatureExtract - extraction of sequence annotation made easy, *Nuc. Acids. Res.*, **33**: Web Server issue W567:W569.

[Wolda H., 1981] Wolda H., (1981) Similarity Indices, Sample Size and Diversity, *Oecologia*, **50**:296-3022.

[Ludwig W. et al., 2004] Wolfgang L. et al., (2004) ARB: a software environment for sequence data, *Nuc. Acids. Res.*, **32(4)**:1363-1371.