

# NanoCLUST: a species-level analysis of 16S rRNA nanopore sequencing data

Héctor Rodríguez-Pérez<sup>1,†</sup>, Laura Ciuffreda<sup>1,†</sup>, Carlos Flores<sup>1,2,3,4\*</sup>

<sup>1</sup>Research Unit, Hospital Universitario N.S. de Candelaria, Universidad de La Laguna, 38010, Santa Cruz de Tenerife, Spain,

<sup>2</sup>CIBER de Enfermedades Respiratorias, Instituto de Salud Carlos III, 28029, Madrid, Spain,

<sup>3</sup>Genomics Division, Instituto Tecnológico y de Energías Renovables (ITER), 38600 Granadilla, Santa Cruz de Tenerife, Spain,

<sup>4</sup>Instituto de Tecnologías Biomédicas (ITB), Universidad de La Laguna, 38200 San Cristóbal de La Laguna, Santa Cruz de Tenerife, Spain.

\*To whom correspondence should be addressed.

†The authors wish it to be known that, in their opinion, the first two authors should be regarded as Joint First Authors.

## Abstract

**Summary:** NanoCLUST is an analysis pipeline for classification of amplicon-based full-length 16S rRNA nanopore reads. It is characterized by an unsupervised read clustering step, based on Uniform Manifold Approximation and Projection (UMAP), followed by the construction of a polished read and subsequent Blast classification. Here we demonstrate that NanoCLUST performs better than other state-of-the-art software in the characterization of two commercial mock communities, enabling accurate bacterial identification and abundance profile estimation at species level resolution.

**Availability and implementation:** Source code, test data and documentation of NanoCLUST is freely available at <https://github.com/genomicsITER/NanoCLUST> under MIT License.

**Contact:** [cflores@ull.edu.es](mailto:cflores@ull.edu.es)

## 1 Introduction

Nanopore sequencing (Oxford Nanopore Technologies, ONT) has emerged as a fast and inexpensive method for long-read DNA/RNA sequencing. Accessing the microbial communities with ONT is feasible using rapid protocols targeting the full-length 16S rRNA gene. Widely used software, such as QIIME (Caporaso *et al.*, 2010), is designed to analyze short-read sequencing that typically allow characterizing the microbial communities at the genus level. However, tools for the analysis of the noisy 16S rRNA long-reads from ONT are scarce (Santos *et al.*, 2020). Among them, the popular Epi2me (<https://www.metrichor.com>) is based on a read-by-read classification strategy that does not cope well with the error rate associated to this technology, resulting in the misclassification of a high percentage of reads and a high uncertainty in the results.

Here we present NanoCLUST, a pipeline for the analysis of ONT 16S rRNA amplicon reads. Besides demultiplexing and quality control (QC) steps, and inspired by the work of Beaulaurier *et al.* (2020), it leverages Uniform Manifold Approximation and Projection (UMAP) (McInnes *et al.*, 2018) and Hierarchical Density-Based Spatial Clustering of Applications with Noise (HDBSCAN) (McInnes *et al.*, 2017) for unsupervised read clustering followed by the construction of a polished sequence for subsequent taxonomic classification. We tested NanoCLUST on ONT data from two commercial mock communities and compared the results to two other popular and accurate classification methods, Kraken2 (Wood *et al.*, 2019) and Bracken (Lu *et al.*, 2017).

## 2 Description and implementation

NanoCLUST is implemented in Nextflow workflow management system, which enables efficient parallel execution in all major systems and computing environments. NanoCLUST development followed the nf-core (Ewels *et al.*, 2020) best practices guidelines and standardized template. Software packages and the dependencies are bundled in the pipeline using built-in integration for conda environments and Docker containers. The general workflow of NanoCLUST is illustrated in Figure 1, and software versions detailed in Table S1. The input data consists of basecalled 16S rRNA ONT sequencing reads, which are internally demultiplexed in case of pooled samples. Then, reads are filtered using fastp to ensure that only near full-length 16S rRNA reads are kept. By default, reads with a Phred score <8.0 and of length below 1,400 base pairs (bp) or above 1,700 bp are discarded. Prior to the clustering step, each read is transformed into a normalized 5-mer frequency vector and stored in tabular format. UMAP projection and posterior clustering using HDBSCAN is applied to the 5-mer frequency vector set and a cluster assignment is given to each read. The clustering step parameters, such as the minimum read number necessary to identify an independent cluster to determine the sensitivity, as well as other HDBSCAN parameters can be manually set by the user. These parameters will be dependent on the input data and the desired level of taxonomy. Therefore, they have a significant practical effect on the clustering and on subsequent analysis.

The next step builds a consensus sequence from the reads belonging to each cluster. For that, the pairwise Average Nucleotide Identity (ANI)

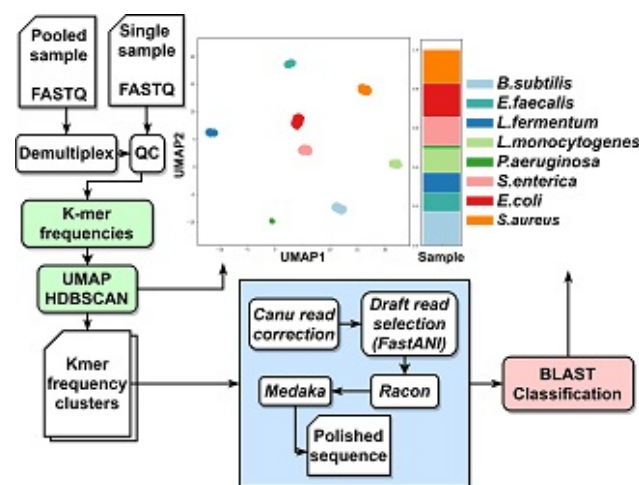


Figure 1. Simplified flowchart of NanoCLUST.

between reads in the same cluster is calculated using FastANI. Then, the read with the highest average intra-cluster ANI is chosen and 100 other reads from the same cluster are selected for polishing the sequence. The polishing stage includes one round each in the Canu read-correction module, in Racon, and in Medaka. The resulting polished sequence is taken as the representative of the cluster and is finally classified using blastn and the NCBI Refseq database (or any other database provided by the user). The resulting taxonomic classification output is assigned to all the reads belonging to the cluster and the results are then merged for relative abundance estimation based on the HDBSCAN cluster assignment and the total number of reads projected by UMAP. The output consists of QC reports of the input data, the consensus sequence representing each cluster, a complete classification report for each cluster, and the relative abundance tables and bar plots at multiple taxonomic levels, both for single and pooled samples.

### 3 Results

We used NanoCLUST to analyze the sequencing data from two commercial mock communities run in various experiments, MOCK1 containing genomic DNA from eight bacterial species (Zymobiomics), and MOCK2 containing genomic DNA from 20 bacterial species (Bioresources) (see Supplementary data). Eight clusters were identified in MOCK1, and classification of the polished sequences successfully detected all taxa present in the sample at the species level (Figure S1). As regards of MOCK2, 19 out of the 20 expected clusters were identified (Figure S2). Of these, only one species, *Bacillus cereus*, was misclassified as *Bacillus thuringiensis*, likely because of the 99.73% similarity in their 16S rRNA sequence (Figure S3). The absence of one species, corresponding to *Actinomyces odontolyticus*, was explained by a low amplicon representation in the sequencing experiment, not reaching the minimum read number set ( $n=100$ ) for UMAP cluster formation.

We then compared the results obtained from NanoCLUST for the two communities with those from Kraken2 (Wood et al., 2019) and Bracken (Lu et al., 2017) (Figure S4, S5). Overall, NanoCLUST performed better than the other classifiers in the analysis of 16S rRNA ONT sequences. Species richness estimated by NanoCLUST in the independent sequencing runs was the most similar to the expected, with eight and 19 species

identified for MOCK1 and MOCK2, respectively, while Kraken2 and Bracken identified a much larger number of species in the two communities (an average of 139 and 214 different species for MOCK1 and MOCK2, respectively). Shannon diversity index was also calculated, and the closer value to the expected for each mock was found when NanoCLUST was used in the analysis, compared to Kraken2 and Bracken (Figure S6).

To compare the expected relative abundance of species in the two mock communities with the estimates provided by the three classification methods, we calculated the mean absolute error (MSA) and the root mean squared error (RMSE). As indicated by the significantly lower MSA and RMSE, relative abundances obtained by NanoCLUST were more similar to the expected than those of Kraken2 and Bracken (Figure S7, S8).

### Acknowledgements

We would like to thank Tamara Hernandez-Beeftink and José M. Lorenzo-Salazar for their help with the experimental setup with the MinION device and the practical advice with UMAP/HDBSCAN parameters, respectively.

### Funding

This work was supported by Instituto de Salud Carlos III [PI14/00844, PI17/00610, and FI18/00230] and co-financed by the European Regional Development Funds, “A way of making Europe” from the European Union; Ministerio de Ciencia e Innovación [RTC-2017-6471-1, AEI/FEDER, UE]; Cabildo Insular de Tenerife [CGIEU0000219140]; Fundación Canaria Instituto de Investigación Sanitaria de Canarias [PIFUN48/18]; and by the agreement with Instituto Tecnológico y de Energías Renovables (ITER) to strengthen scientific and technological education, training, research, development and innovation in Genomics, Personalized Medicine and Biotechnology [OA17/008].

Conflict of Interest: none declared.

### References

- Beaulaurier, J. et al. (2020) Assembly-free single-molecule sequencing recovers complete virus genomes from natural microbial communities. *Genome Res.*, 30, gr.251686.119.
- Caporaso, J.G. et al. (2010) QIIME allows analysis of high-throughput community sequencing data. *Nat. Methods*, 7, 335–336.
- Ewels, P.A. et al. (2020) The nf-core framework for community-curated bioinformatics pipelines. *Nat. Biotechnol.*, 38, 276–278.
- Lu, J. et al. (2017) Bracken: estimating species abundance in metagenomics data. *PeerJ Comput. Sci.*, 3, e104.
- McInnes, L. et al. (2017) hdbSCAN: Hierarchical density based clustering. *J. Open Source Softw.*, 2, 205.
- McInnes, L. et al. (2018) UMAP: Uniform Manifold Approximation and Projection. *J. Open Source Softw.*, 3, 861.
- Santos, A. et al. (2020) Computational methods for 16S metabarcoding studies using Nanopore sequencing data. *Comput. Struct. Biotechnol. J.*, 18, 296–305.
- Wood, D.E. et al. (2019) Improved metagenomic analysis with Kraken 2. *Genome Biol.*, 20, 257.

# NanoCLUST: a species-level analysis of 16S rRNA nanopore sequencing data

Héctor Rodríguez-Pérez<sup>1</sup>, Laura Ciuffreda<sup>1</sup>, Carlos Flores<sup>1,2,3,4</sup>

<sup>1</sup>Research Unit, Hospital Universitario N.S. de Candelaria, Universidad de La Laguna, 38010, Santa Cruz de Tenerife, Spain, <sup>2</sup>CIBER de Enfermedades Respiratorias, Instituto de Salud Carlos III, 28029, Madrid, Spain, <sup>3</sup>Genomics Division, Instituto Tecnológico y de Energías Renovables (ITER), 38600 Granadilla, Santa Cruz de Tenerife, Spain, <sup>4</sup>Instituto de Tecnologías Biomédicas (ITB), Universidad de La Laguna, 38200 San Cristóbal de La Laguna, Santa Cruz de Tenerife, Spain.

## Supplementary information

**Table S1.** Software and algorithms integrated in NanoCLUST\*.

Software/algorithm	Utility	Version
qcat <sup>a</sup>	Demultiplexing	1.1.0
Porechop <sup>b</sup>	Demultiplexing	0.2.3
fastp <sup>c</sup>	QC filtering	0.20.0
fastQC <sup>d</sup>	QC report	0.11.8
multiQC <sup>e</sup>	QC report	1.6
UMAP <sup>f</sup>	Read projection	0.3.10
HDBSCAN <sup>g</sup>	Cluster assignation	0.8.26
seqtk <sup>h</sup>	Cluster read extraction	1.3
FastANI <sup>i</sup>	Draft selection	1.3
Canu <sup>j</sup>	Read correction	2.0
Racon <sup>k</sup>	Read polishing	1.4.13
Medaka <sup>l</sup>	Read polishing	0.11.5
blastn <sup>m</sup>	Taxonomic assignation	2.9.0

<sup>a</sup><https://github.com/nanoporetech/qcat>; <sup>b</sup><https://github.com/rrwick/Porechop>;

<sup>c</sup>Chen *et al.* (2018); <sup>d</sup>Andrews S. (2010); <sup>e</sup>Ewels *et al.* (2016); <sup>f</sup>McInnes *et al.* (2018); <sup>g</sup>McInnes *et al.* (2017); <sup>h</sup><https://github.com/lh3/seqtk>; <sup>i</sup>Jain *et al.* (2018); <sup>j</sup>Koren *et al.* (2017); <sup>k</sup>Vaser *et al.* (2017); <sup>l</sup><https://github.com/nanoporetech/medaka>; <sup>m</sup>Altschul *et al.* (1990). \*Includes the following python libraries for data processing and plotting: Python 3.7, Numpy 1.18.1, Pandas 1.0.3, and Matplotlib 3.2.1.

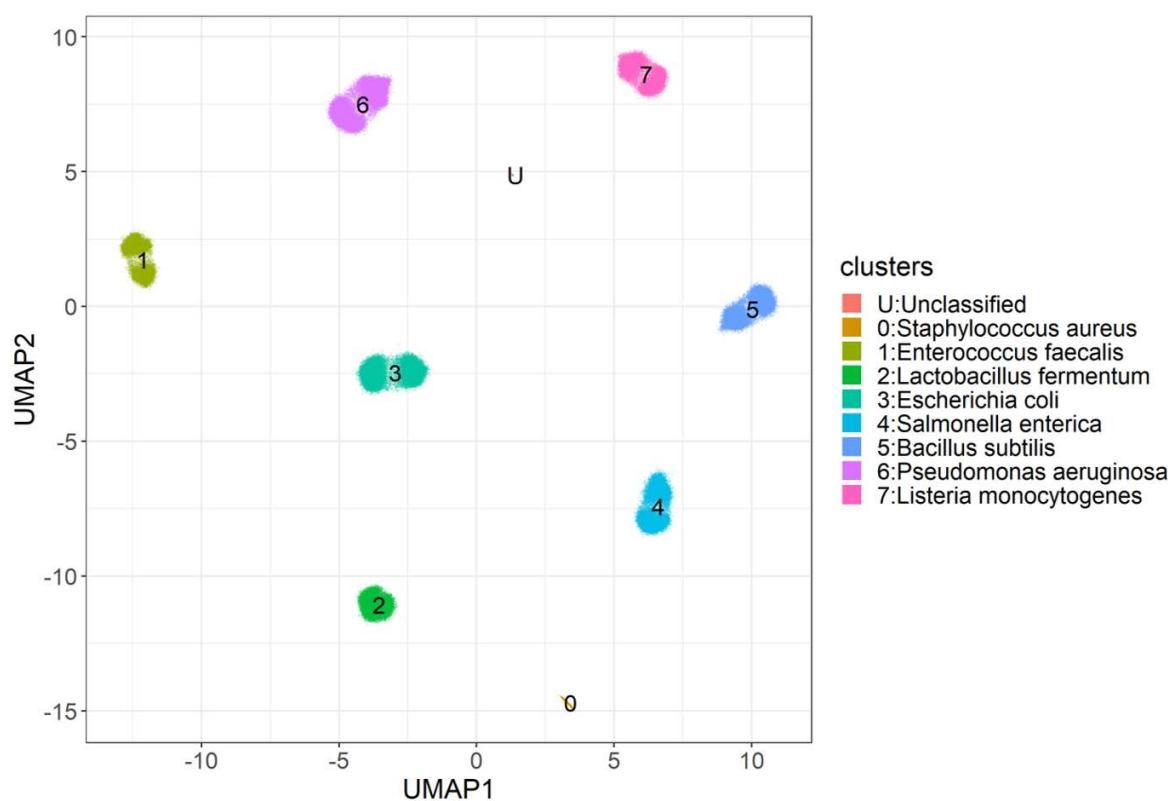
## Library preparation and sequencing

Amplicon-based libraries were prepared using the ONT 16S Barcoding kit (SQK-RAB204) following manufacturer's instructions (ONT). Briefly, 10 ng of DNA from two bacterial mock communities (ZymoBIOMICS™ Microbial Community DNA Standard [Zymo Research]; Microbial Mock Community B (Even, High Concentration) [BEI Resources]) were used as targets in the PCR amplification reaction, performed using the LongAmp Taq 2X MasterMix (New England Biolabs) in a final volume of 50 µl. Negative controls (containing only PCR grade water) were also included in each reaction. PCR products were purified using 30 µl of the AMPure XP beads (Beckman Coulter) and eluted in 10 µl of 10mM Tris-HCl pH 8.0 with 50 mM NaCl. Each barcoded library was quantified using the Invitrogen™ Qubit™ dsDNA HS Assay kit (Thermo Fisher Scientific) and pools of 12-plex barcoded libraries were prepared for each run. Rapid adapters (1 µl) were added to each library pool and followed by an incubation at 23 °C for 5 minutes. Libraries were sequenced using a MinION device (ONT) over a period of 48 h. After priming and loading of the flow cell (R9.4.1), the run was started using MinKNOW software (v19.05). Fast5 files were generated and basecalled using Guppy (v3.1.5) using a local machine with two Xeon 6238T 1.9 GHz, GPU Nvidia RTX 2080TI and 512GB RAM. Demultiplexing and quality controls were carried out either using the NanoCLUST modules for this purpose, or local qcat Python package (ONT) and fastp (v0.20.0) (Chen *et al.*, 2018).

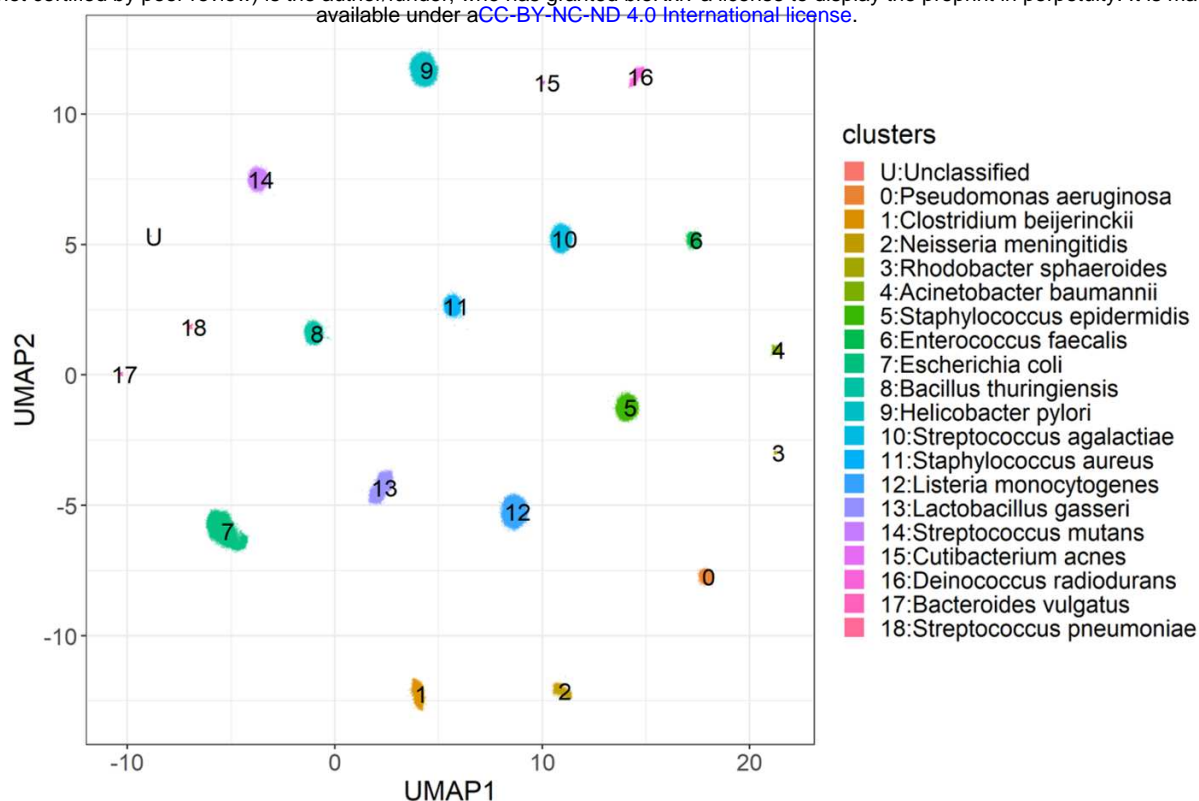
## Comparative analysis of relative abundances

Seven and three sequencing replicates were carried out using the ZymoBIOMICS Microbial DNA Standard (MOCK1) and the Microbial MOCK Community B (MOCK2), respectively. A subset of 100,000 high-quality reads (mean length > 1,400 bp and < 1,700 bp and a Phred quality score > 8.0) from each mock replicate was analyzed using NanoCLUST. The same subset of reads was subsequently analyzed using Kraken2 (v2.0.8-beta) (Wood *et al.*, 2019) and Bracken (v2.5.0) (Lu *et al.*, 2017) against the Refseq complete bacterial genomes database. This subset (only from MOCK1) is included in the repository as an example test file. Bacterial abundance profiles were retrieved from the Kraken2 and Bracken reports and compared to those calculated by NanoCLUST. Species richness and Shannon diversity index were retrieved using the vegan R package (Oksanen *et al.*, 2019). The metrics to assess the deviations from the expected values (the mean absolute error (MAE) and the root mean squared error (RMSE)) were calculated using R software (R Core Team, 2013), and mean values compared between groups using ANOVA followed by the Tukey's multiple comparison test.

## Supplementary figures



**Figure S1.** UMAP plot of reads from MOCK1.

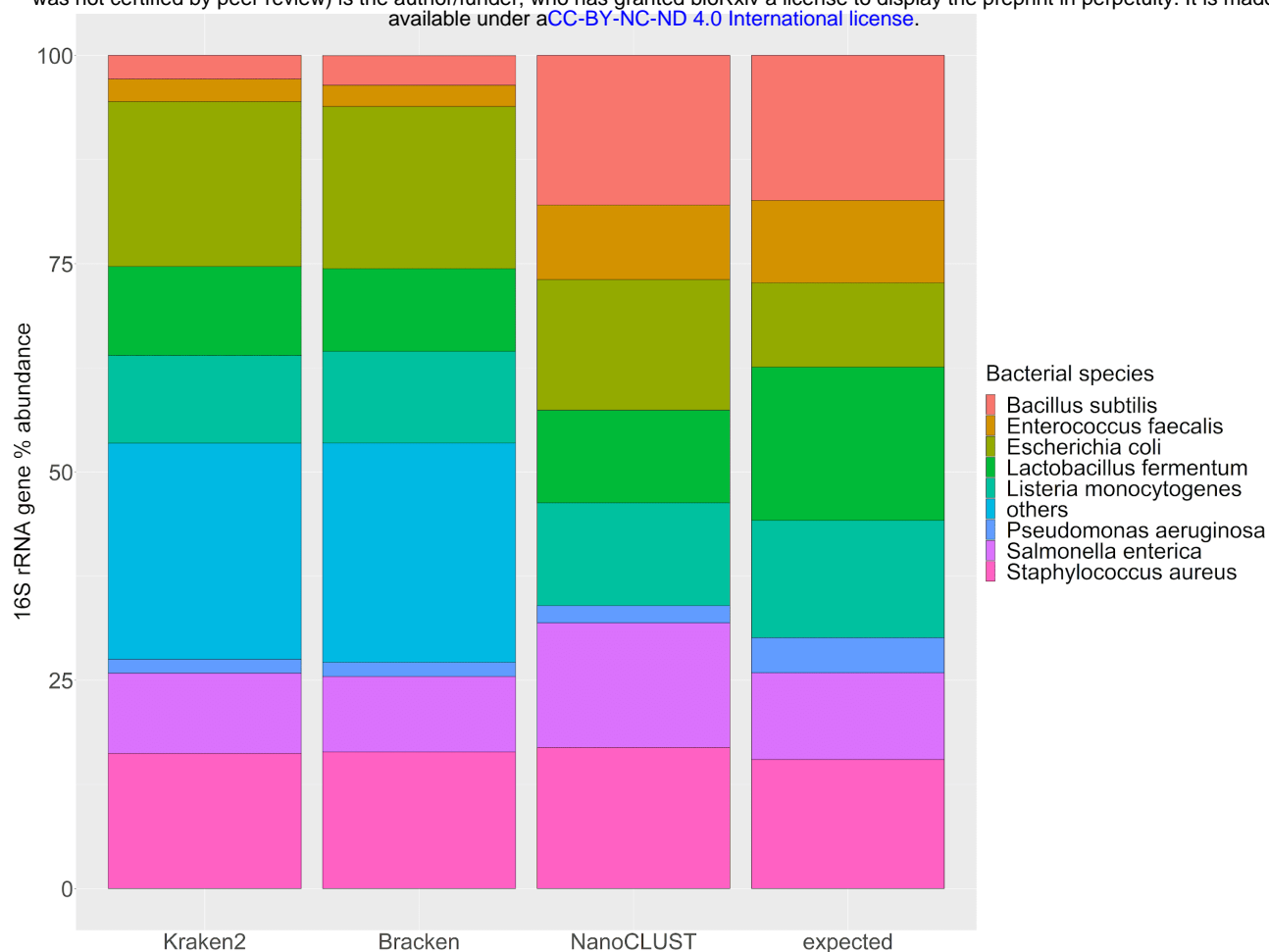


**Figure S2.** UMAP plot of reads from MOCK2. Note that *Bacillus thuringiensis*, supported by NanoCLUST, is not present in this mock community.



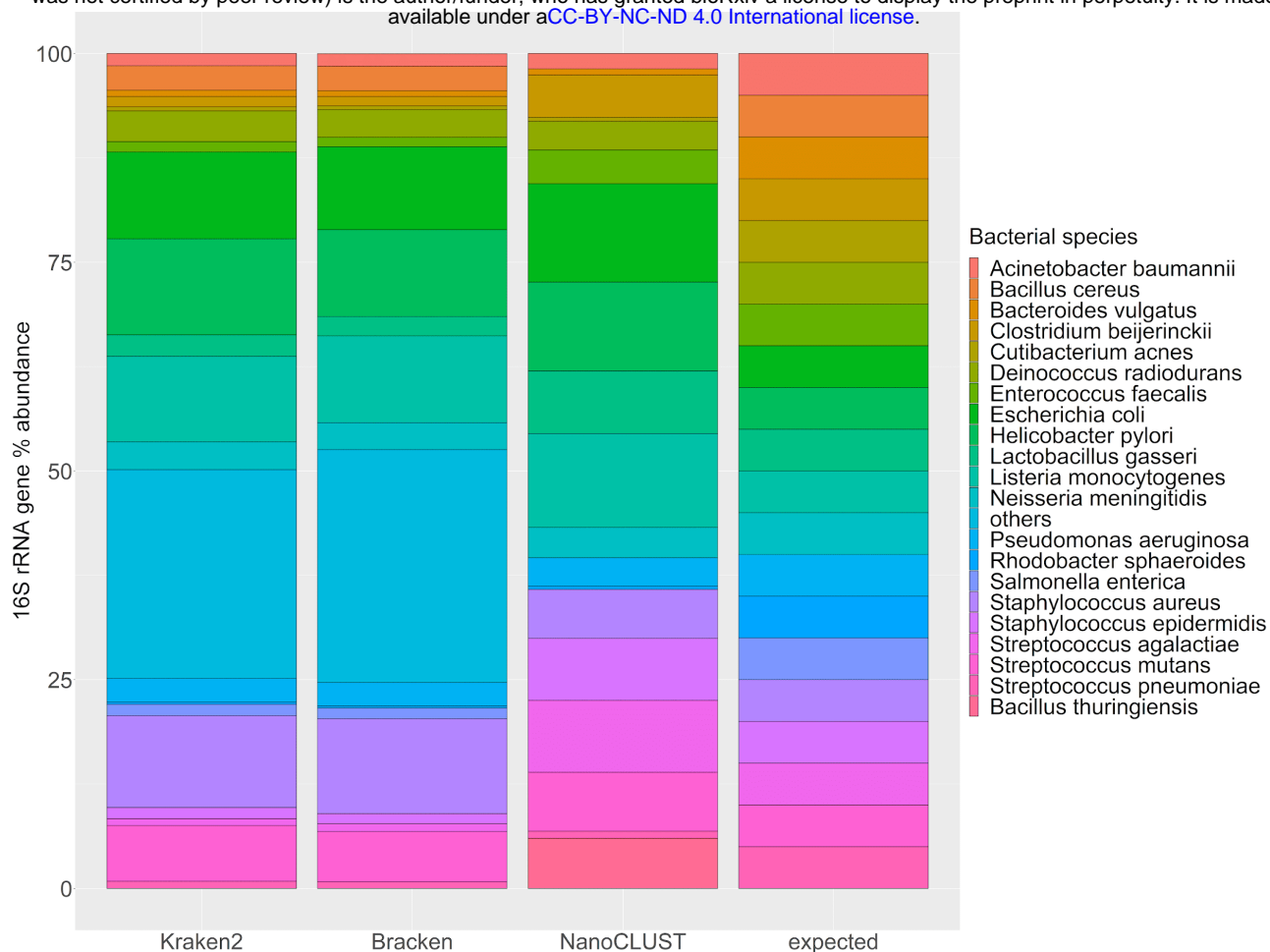
<i>Bacillus cereus</i>	GATGAACGCTGCGGCGTGCCTAATACATGCAAGTCGACGGAATGGATTAGAGCTTGCT	60
<i>Bacillus thuringiensis</i>	GATGAACGCTGCGGCGTGCCTAATACATGCAAGTCGACGGAATGGATTAGAGCTTGCT	60
<i>Bacillus cereus</i>	CTTAAAGTTAGCGGCGGACGGGTGAGTAACACGCGGTAACTGCCCATAAGACTGGG	120
<i>Bacillus thuringiensis</i>	CTTAAAGTTAGCGGCGGACGGGTGAGTAACACGCGGTAACTGCCCATAAGACTGGG	120
<i>Bacillus cereus</i>	ATAACTCCGGAAACCGGGCTAATACCGGATAACATTTTGAACGTCATGGTTCGAAAT	180
<i>Bacillus thuringiensis</i>	ATAACTCCGGAAACCGGGCTAATACCGGATAACATTTTGAACGTCATGGTTCGAAAT	180
<i>Bacillus cereus</i>	GAAAGCGGCTTCGGCTGTCACCTATGGATGACCGCGTCGATTAGCTAGTTGGTGAG	240
<i>Bacillus thuringiensis</i>	GAAAGCGGCTTCGGCTGTCACCTATGGATGACCGCGTCGATTAGCTAGTTGGTGAG	240
<i>Bacillus cereus</i>	GTAAAGGCTCACCAAGGCAACGATGCGTAGCCGACCTGAGAGGGTATCGGCGCACTGG	300
<i>Bacillus thuringiensis</i>	GTAAAGGCTCACCAAGGCAACGATGCGTAGCCGACCTGAGAGGGTATCGGCGCACTGG	300
<i>Bacillus cereus</i>	GACTGAGACACGGCCAGACTCCTACGGAGGCGAGCATAGGGAATCTTCGCAATGGAC	360
<i>Bacillus thuringiensis</i>	GACTGAGACACGGCCAGACTCCTACGGAGGCGAGCATAGGGAATCTTCGCAATGGAC	360
<i>Bacillus cereus</i>	GAAAGTCTGACGGAGCAACGCGCGTGAAGTGAAGGCTTCGGGCTGTAAGACTCTGT	420
<i>Bacillus thuringiensis</i>	GAAAGTCTGACGGAGCAACGCGCGTGAAGTGAAGGCTTCGGGCTGTAAGACTCTGT	420
<i>Bacillus cereus</i>	TGTTAGGGAAGACAAAGTGTAGTTGAATAAGCTGGCACCTTGACGGTACTAACCAGAA	480
<i>Bacillus thuringiensis</i>	TGTTAGGGAAGACAAAGTGTAGTTGAATAAGCTGGCACCTTGACGGTACTAACCAGAA	480
<i>Bacillus cereus</i>	AGCCACGGCTAATACGTGCCAGCAGCGCGTAATACGTAGGTGGCAAGCGTTATCCGG	540
<i>Bacillus thuringiensis</i>	AGCCACGGCTAATACGTGCCAGCAGCGCGTAATACGTAGGTGGCAAGCGTTATCCGG	540
<i>Bacillus cereus</i>	AATTATTGGGCTAAAGCGCGCGAGGTGTTCTTAAGTCTGATGTAAAGCCACGGC	600
<i>Bacillus thuringiensis</i>	AATTATTGGGCTAAAGCGCGCGAGGTGTTCTTAAGTCTGATGTAAAGCCACGGC	600
<i>Bacillus cereus</i>	TCAACCGTGGAGGTCATTGGAACTGGGAGACTTGAGTGCAGAGAGGAAAGTGGAAAT	660
<i>Bacillus thuringiensis</i>	TCAACCGTGGAGGTCATTGGAACTGGGAGACTTGAGTGCAGAGAGGAAAGTGGAAAT	660
<i>Bacillus cereus</i>	CCATGTGTAGCGGTGAAATGCGTAGAGATATGAGGAACACAGTGGCGAAGGCGACTTT	720
<i>Bacillus thuringiensis</i>	CCATGTGTAGCGGTGAAATGCGTAGAGATATGAGGAACACAGTGGCGAAGGCGACTTT	720
<i>Bacillus cereus</i>	CTGGTCTGTAAGTACACTGAGGCGCGAAAGCGTGGGAGCAACAGGATTAGATACCT	780
<i>Bacillus thuringiensis</i>	CTGGTCTGTAAGTACACTGAGGCGCGAAAGCGTGGGAGCAACAGGATTAGATACCT	780
<i>Bacillus cereus</i>	GGTAGTCCACCGCTAAACGATGAGTGTAGTGTAGAGGGTTCCGCGCTTTAGTGCT	840
<i>Bacillus thuringiensis</i>	GGTAGTCCACCGCTAAACGATGAGTGTAGTGTAGAGGGTTCCGCGCTTTAGTGCT	840
<i>Bacillus cereus</i>	GAAATTAAAGCATTAAAGCACTCCGCTGGGAGTACGCGCAAGGCTGAAACTCAAAGG	900
<i>Bacillus thuringiensis</i>	GAAATTAAAGCATTAAAGCACTCCGCTGGGAGTACGCGCAAGGCTGAAACTCAAAGG	900
<i>Bacillus cereus</i>	AATTGACGGGGCCCGACAAAGCGTGGAGCATGTGTTAATTCGAAGCAACGCAAGGA	960
<i>Bacillus thuringiensis</i>	AATTGACGGGGCCCGACAAAGCGTGGAGCATGTGTTAATTCGAAGCAACGCAAGGA	960
<i>Bacillus cereus</i>	ACCTTACAGGTCTTGACATCTCTGAAACCTAGAGTAGGGCTTCTCTTCGGGAGC	1020
<i>Bacillus thuringiensis</i>	ACCTTACAGGTCTTGACATCTCTGAAACCTAGAGTAGGGCTTCTCTTCGGGAGC	1020
<i>Bacillus cereus</i>	AGAGTGACAGGTGGTGCATGGTTGCTGTCAGCTCGTGTGAGATGTTGGTTAAGTCC	1080
<i>Bacillus thuringiensis</i>	AGAGTGACAGGTGGTGCATGGTTGCTGTCAGCTCGTGTGAGATGTTGGTTAAGTCC	1080
<i>Bacillus cereus</i>	CGCAACGAGCGCAACCTTGATCTTAAGTGGCATTAAGTTGGGCACTTAAGGTGAC	1140
<i>Bacillus thuringiensis</i>	CGCAACGAGCGCAACCTTGATCTTAAGTGGCATTAAGTTGGGCACTTAAGGTGAC	1140
<i>Bacillus cereus</i>	TGCCGCTGACAAACCGGAGGAAAGTGGGATGACGTCAAAATCATGCCCCCTTATGACC	1200
<i>Bacillus thuringiensis</i>	TGCCGCTGACAAACCGGAGGAAAGTGGGATGACGTCAAAATCATGCCCCCTTATGACC	1200
<i>Bacillus cereus</i>	TGGGTACACACGTGCTACAATGGACGGTACAAGAGCTGCAAGACCGGAGGTGGAGCT	1260
<i>Bacillus thuringiensis</i>	TGGGTACACACGTGCTACAATGGACGGTACAAGAGCTGCAAGACCGGAGGTGGAGCT	1260
<i>Bacillus cereus</i>	AATCTCATAAACCGTTCTCAGTTGCGATTGTAGGCTGCAACTCGCTACATGAAGCTGG	1320
<i>Bacillus thuringiensis</i>	AATCTCATAAACCGTTCTCAGTTGCGATTGTAGGCTGCAACTCGCTACATGAAGCTGG	1320
<i>Bacillus cereus</i>	AATCGCTAGTAATCGCGATCAGCATGCCCGGTGAATACGTTCCCGGCTTTGTACACA	1380
<i>Bacillus thuringiensis</i>	AATCGCTAGTAATCGCGATCAGCATGCCCGGTGAATACGTTCCCGGCTTTGTACACA	1380
<i>Bacillus cereus</i>	CCGCCCGTACACCAAGAGAGTTGTAAACCCGAAAGTCGGTGGGTAACTTTTGGAG	1440
<i>Bacillus thuringiensis</i>	CCGCCCGTACACCAAGAGAGTTGTAAACCCGAAAGTCGGTGGGTAACTTTTGGAG	1440
<i>Bacillus cereus</i>	CCAGCCGCTAAGGTGGGACAGATGATTGGGGTGAAGTCGTAACA	1485
<i>Bacillus thuringiensis</i>	CCAGCCGCTAAGGTGGGACAGATGATTGGGGTGAAGTCGTAACA	1485

**Figure S3.** Sequence alignment (mismatches in purple) of 16S rRNA genes from *Bacillus cereus* and *Bacillus thuringiensis*. Alignment was performed using Clustal Omega (Sievers *et al.*, 2011).

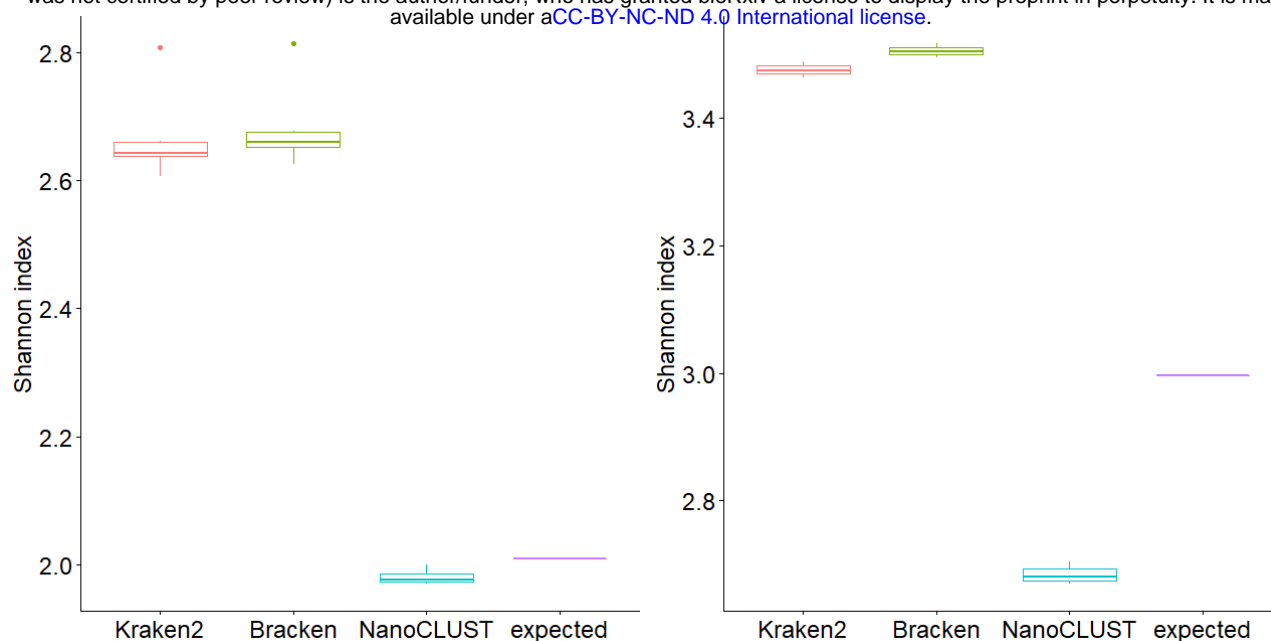


**Figure S4.** Relative abundances of Kraken2, Bracken, and NanoCLUST compared to the expected values for MOCK1.

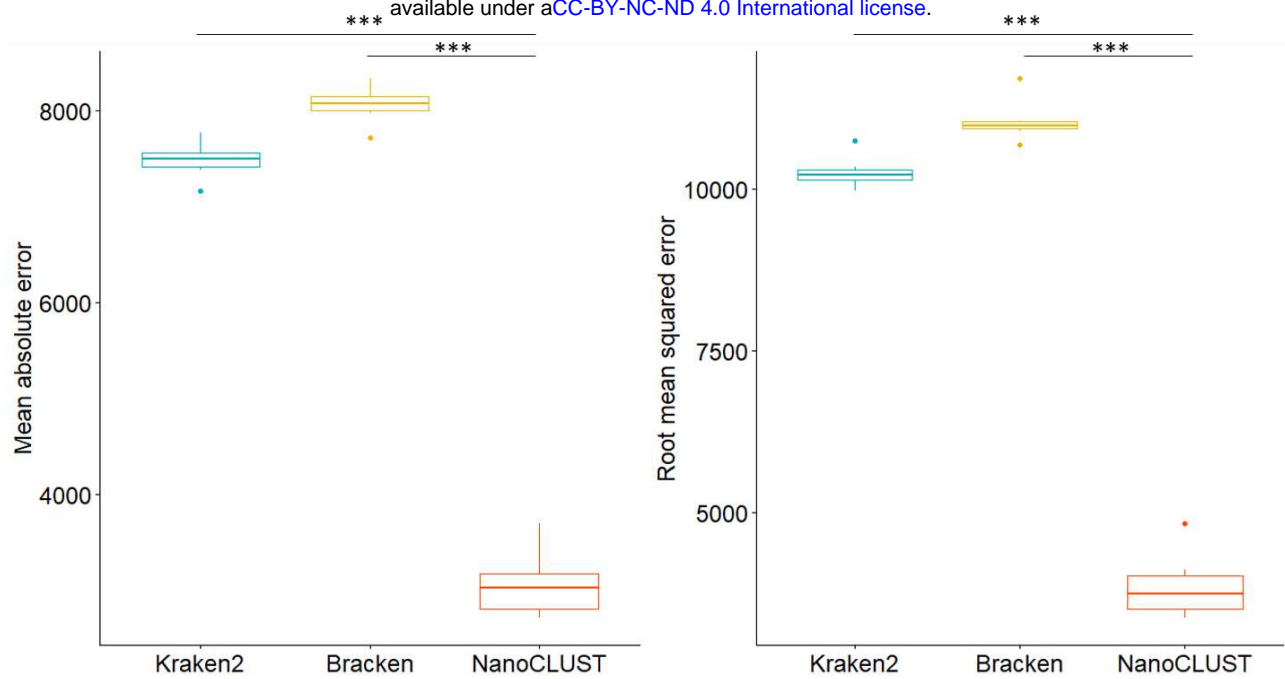




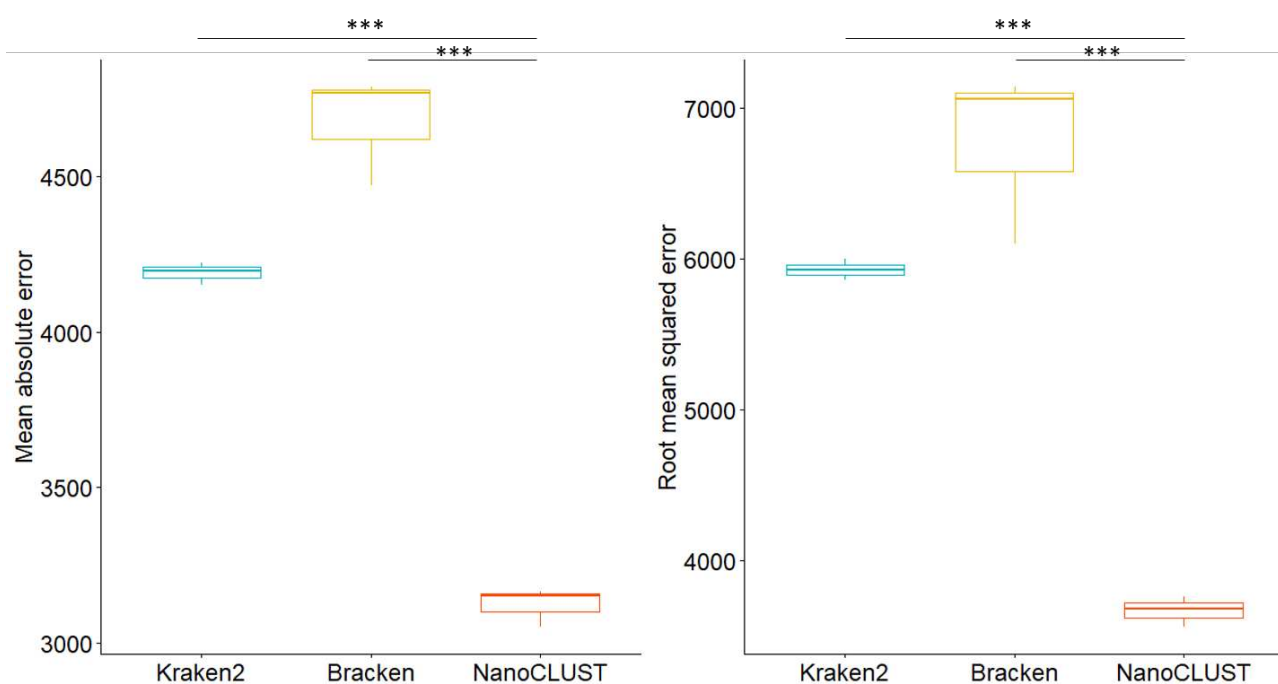
**Figure S5.** Relative abundances of Kraken2, Bracken, and NanoCLUST compared to the expected values for MOCK2. Note that *Bacillus thuringiensis*, supported by NanoCLUST, is not present in this mock community.



**Figure S6.** Shannon diversity index for MOCK1 (left panel) and MOCK2 (right panel).



**Figure S7.** Mean absolute error and root mean squared error against the expected of MOCK1, based on read counts. Tukey multiple comparison test. \*\*\*,  $p < 0.001$ .



**Figure S8.** Mean absolute error and root mean squared error against the expected of MOCK2, based on read counts. Tukey multiple comparison test. \*\*\*,  $p < 0.001$ .

## References

- Altschul,S. et al. (1990) Basic Local Alignment Search Tool. *J. Mol. Biol.*, 215, 403–410.
- Andrews,S. FastQC: a quality control tool for high throughput sequence data. 2010. Available online at: <http://www.bioinformatics.babraham.ac.uk/projects/fastqc>
- Chen,S. et al. (2018) fastp: an ultra-fast all-in-one FASTQ preprocessor. *Bioinformatics*, 34, i884–i890.
- Ewels,P. et al. (2016) MultiQC: summarize analysis results for multiple tools and samples in a single report. *Bioinformatics*, 32, 3047–3048.
- Jain,C. et al. (2018) High throughput ANI analysis of 90K prokaryotic genomes reveals clear species boundaries. *Nat. Commun.*, 9, 5114.
- Koren,S. et al. (2017) Canu: scalable and accurate long-read assembly via adaptive k-mer weighting and repeat separation. *Genome Res.*, 27, 722–736.
- Lu,J. et al. (2017) Bracken: estimating species abundance in metagenomics data. *PeerJ Comput. Sci.*, 3, e104.
- McInnes,L. et al. (2017) hdbscan: Hierarchical density based clustering. *J. Open Source Softw.*, 2, 205.
- McInnes,L. et al. (2018) UMAP: Uniform Manifold Approximation and Projection. *J. Open Source Softw.*, 3, 861.
- Oksanen,J. et al. (2019) vegan: Community Ecology Package.
- R Core Team (2013) R: A Language and Environment for Statistical Computing.
- Vaser,R. et al. (2017) Fast and accurate de novo genome assembly from long uncorrected reads. *Genome Res.*, 27, 737–746.
- Wood,D.E. et al. (2019) Improved metagenomic analysis with Kraken 2. *Genome Biol.*, 20, 257.