OptiFit: a fast method for fitting amplicon sequences to existing OTUs

2021-03-04

Kelly L. Sovacool[1], Sarah L. Westcott[2], M. Brodie Mumphrey[1], Gabrielle A. Dotson[1], Patrick D. Schloss[2]†

1 Department of Computational Medicine and Bioinformatics, University of Michigan, Ann Arbor, MI 48109

2 Department of Microbiology and Immunology, University of Michigan, Ann Arbor, MI 48109

† To whom correspondence should be addressed: pschloss@umich.edu

Abstract

Assigning amplicon sequences to Operational Taxonomic Units (OTUs) is an important step in characterizing the composition of microbial communities across large datasets. OptiClust, a *de novo* OTU clustering method in the mothur program, has been shown to produce higher quality OTU assignments than other methods and at comparable or faster speeds (1, 2). A notable difference between *de novo* clustering and database-dependent methods is that OTU assignments clustered with *de novo* methods are not stable when new sequences are added to a dataset (3). However, in some cases one may wish to incorporate new samples into a previously clustered dataset without performing clustering again on all sequences, such as when deploying a machine learning model where OTUs are features (4). To provide an efficient and robust method to fit amplicon sequence data to existing OTUs, we developed the OptiFit algorithm as a new component of the mothur program.

- **TODO: summarize results & conclusion**

Importance

TODO

Introduction

Amplicon sequencing has become a mainstay of microbial ecology and host-associated microbiome research. Researchers can affordably generate millions of sequences to characterize the composition of hundreds of samples from culture-independent microbial communities. In a typical analysis pipeline, 16S rRNA gene sequences are assigned to Operational Taxonomic Units (OTUs) to facilitate comparison of taxonomic composition between communities. A distance threshold of 3% (or sequence similarity of 97%) is commonly used to cluster sequences into OTUs based on either a reference database or pairwise comparisons of the sequences within the dataset. The method chosen for clustering affects the quality of OTU assignments and thus may impact downstream analyses of community composition (1, 3, 5).

There are three main categories of OTU clustering algorithms: closed reference, open reference, and *de novo* clustering. Closed reference methods assign sequences to a set of pre-made OTUs generated from reference sequences. If a query sequence is not within the distance threshold to any of the reference sequences, it is discarded. While reference-based clustering is generally fast, it is limited by the diversity represented in the reference database. Rare or novel sequences in the sample will be lost if they are not represented by a similar sequence in the database. *De novo* methods cluster sequences based on their distance to each other, without the use of an external reference. *De novo* clustering overcomes the limitations of reference databases by considering only sequences in the dataset, but is more computationally intensive and generates different OTU assignments when new sequences are introduced. Unstable OTU assignments make it difficult to use *de novo* clustering to compare taxonomic composition of communities between studies or to use machine learning models trained with *de novo* OTUs to make predictions on new data. Open reference methods take a hybrid approach, first performing closed reference clustering, then any sequences that cannot be assigned to reference

OTUs are clustered *de novo* to create additional OTUs. Previous studies found that the OptiClust *de novo* clustering algorithm created the highest quality OTU assignments of all clustering methods based on the Matthews correlation coefficient (MCC) (3, 5).

To overcome the limitations of *de novo* clustering while maintaining OTU quality, we developed OptiFit, a reference-based clustering algorithm in the mothur program which takes existing OTUs as the reference to fit new sequences to. **TODO: more words here?** Here, we tested the OptiFit algorithm with the reference as databases or *de novo* OTUs and compared the performance to existing tools.

Results

The OptiFit algorithm

- **TODO: ask Sarah Westcott to check the accuracy of this description**

OptiFit leverages the method employed by OptiClust of iteratively assigning sequences to OTUs to produce the highest quality OTUs possible, and extends this method for reference-based clustering. OptiFit takes as input a list of reference OTUs and their sequences, a list of query sequences to assign to the reference OTUs, the sequence pairs that are within the distance threshold (e.g. 0.03), and the metric to assess clustering quality (default: MCC). Query sequences are randomly seeded in reference OTUs, then for each sequence the algorithm calculates the quality metric based on whether the sequences stays in its current OTU or moves to each of the other OTUs. This process is repeated until the quality metric stabilizes, changing by no more than 0.0001 by default, or until a maximum number of iterations is reached (default: 100). In closed-reference mode, any query sequences that cannot be assigned are thrown out (**TODO: what determines whether a seq can't be assigned?**), and the results only contain OTUs that exist in the original reference. In open-reference mode, un-assigned query sequences are clustered

de novo using OptiClust to generate additional OTUs. The final quality score is reported with the best OTU assignments.

To evaluate the OptiFit algorithm and compare to existing methods, we used four published datasets isolated from soil (6), marine (7), mouse gut (8), and human gut (9) samples. There are two strategies for generating OTUs with OptiFit: 1) fit sequences to reference OTUs of an independent database, or 2) split the dataset into a reference and query fraction, then fit the query sequences to OTUs generated by clustering the reference sequences *de novo*. For each dataset repeated with 100 random seeds, we generated OTUs with OptiFit using both strategies, and also clustered *de novo* OTUs with OptiClust for comparison. All clustering was performed at a sequence distance threshold of 0.03 and OTU quality was evaluated using the Matthews Correlation Coefficient (MCC) as described previously (3, 5). We calculated the fraction of query sequences that were fit to existing OTUs in closed reference mode as an additional measure of quality for this mode.

Reference clustering with public databases

To evaluate reference-based clustering with independent databases, we fit each dataset to reference OTUs generated by *de novo* clustering the Greengenes database (v13_8_99), Silva non-redundant database (v132), and the Ribosomal Database Project (RDP; v16). As a comparison to existing software, vsearch was used to cluster OTUs *de novo* or with reference-based clustering to the greengenes database. In open reference mode, OTU quality was similar between fitting the datasets to reference OTUs with OptiFit and clustering the datasets *de novo* with OptiClust (**TODO: diff. in medians**). This held true for all datasets and reference databases. OptiFit produced higher quality OTUs than vsearch when open reference clustering against the greengenes database, with median MCC scores of **A** and **B%** (respectively). **TODO: open ref runtime of vsearch vs optifit vs opticlust**.

In closed reference mode, OptiFit produced lower quality OTUs than OptiClust by **X%** when fitting sequences to Greengenes and Silva, and **Y%** worse when fitting to RDP. Only up to **Z%** of query sequences were fit to reference OTUs in closed reference mode across any dataset/database combination. OptiFit was able to fit **X** more query sequences to reference OTUs created with the Greengenes and Silva databases than with RDP. vsearch was able to map more query sequences to the greengenes reference than OptiFit in closed reference mode. In terms of run time, closed reference OptiFit outperformed OptiClust by **X%**, while OptiClust outperformed open reference OptiFit by **Y%**. **TODO vsearch closed ref run time**.

For all datasets and clustering methods (*de novo*, open reference, and closed reference), mothur's clustering algorithms produced higher quality OTUs than vsearch. In terms of runtime, OptiFit generally performed faster than vsearch when reference clustering, while vsearch *de novo* clustering outperformed OptiClust.

Reference clustering with split datasets

Datasets were randomly split into a reference fraction and a query fraction. Reference sizes from 10% to 80% of the sequences were created, with the remaining sequences used for the query. Reference sequences were clustered *de novo* with OptiClust, then query sequences were fit to the *de novo* OTUs with OptiFit.

OTU quality from the split dataset strategy with OptiFit was highly similar to that from *de novo* clustering the whole dataset with OptiClust regardless of mode (**TODO: diff in MCC medians**). OTU quality was remarkably stable across 100 different random seeds. In terms of runtime, closed reference OptiFit performed faster than OptiClust on whole datasets by **Z%**. In open reference mode, OptiClust performed **X to Y%** faster than OptiFit only when the OptiFit reference fraction was 30% or less. The split dataset strategy performed just as well as the database strategy in open reference mode regardless of database used, and

outperformed the database strategy in closed reference mode by **W%**.

We also tested three methods for selecting the fraction of sequences to be used as the reference; a simple random sample, weighting sequences by relative abundance, and weighting by similarity to other sequences in the dataset. OTU quality was similar with the simple and abundance-weighted sampling (median MCCs **X** and **Y** respectively), but **Z%** worse with similarity-weighted sampling. In closed reference mode, the fraction of query sequences that can be fit to the reference OTUs decreases as the reference fraction increases from an MCC of **A** with **J** reference sequences to an MCC of **B** with **K** reference sequences.

Discussion

- **TODO:** for these data, we don't see a compelling reason to use reference-based clustering over *de novo*. it was supposed to speed things up. the reason to do reference-based is if you like the ref OTUs – e.g. for ML or downstream tools e.g. PiCrust?
- **TODO:** highlight difference between what we're doing and what previously was done. others use single ref seq to define an OTU, while we use all the ref & query seqs to define an otu. highlight why ours is better than previous methods.

We developed a new algorithm for fitting sequences to existing OTUs and have demonstrated its suitability for reference-based clustering. OptiFit makes the iterative method employed by OptiClust available for tasks where reference-based clustering is required. We have shown that OTU quality is similar between OptiClust and OptiFit in open reference mode, regardless of strategy employed. open reference OptiFit does perform slower than OptiClust due to the additional *de novo* clustering step, so users may prefer OptiClust for tasks that do not require reference OTUs.

When fitting to public databases, OTU quality dropped in closed reference mode to different degrees depending on the database and dataset source, and no more than half of query sequences were able to be fit to OTUs across any dataset/database combination. This may reflect limitations of reference databases, which are unlikely to contain sequences from rare and novel microbes. This drop in quality was most notable with RDP. We recommend users who require an independent reference database opt for Greengenes or Silva instead. Since OptiClust performs faster than open reference OptiFit and creates higher quality OTUs than closed reference OptiFit with the database strategy, we recommend using OptiClust rather than fitting to a database where possible. (**TODO: "if you don't have breadth, closed ref will suck." - Pat**)

The mothur algorithms produced higher quality OTUs than vsearch in open reference, closed reference, or *de novo* modes. However, vsearch was able to fit more sequence into OTUs than OptiFit in closed reference mode. While both mothur and vsearch use a dissimilarity threshold for determining how to assign sequences into OTUs, vsearch is more permissive than mothur. Mothur requires that all pairs of sequences in an OTU are within the dissimilarity threshold without penalizing the MCC, while vsearch only requires sequences to be similar to one other sequence in the OTU. In this way, vsearch sacrifices OTU quality in order to allow more sequences to fit to OTUs. Users who require closed reference clustering may prefer to use vsearch if they wish to maximize the fraction of sequences that can be fit at the cost of OTU quality. However, mothur's OptiClust or OptiFit are preferred for *de novo* or open reference clustering.

When fitting with the split dataset strategy, OTU quality was remarkably similar when reference sequences were selected by a simple random sample or weighted by abundance, but quality was slightly worse when sequences were weighted by similarity. We recommend using a simple random sample since the more sophisticated reference selection methods do not offer any benefit. The similarity in OTU quality between OptiClust and OptiFit with

this strategy demonstrates the suitability of using OptiFit to fit sequences to existing OTUs, such as when using already-trained machine learning models to make predictions on new data.

- **TODO: big picture concluding paragraph**

Materials and Methods

Data Processing Steps

Benchmarking

Data and Code Availability

We implemented the analysis workflow in Snakemake (10) and relied on Python (11), R (12), and GNU bash. Software used includes mothur v1.45.0 (2), vsearch v2.13.3 (13), numpy (14), the Tidyverse metapackage (15), R Markdown (16), and the conda environment manager (17). The complete workflow, manuscript, and conda environment are available at **TODO: UPDATED REPO LINK**.

Acknowledgements

KLS received support from the NIH Training Program in Bioinformatics (T32 GM070449).

PDS received support from **TODO: Pat's grant(s)**.

The funders had no role in study design, data collection and interpretation, or the decision to submit the work for publication.

Author Contributions

KLS wrote the analysis code, evaluated the algorithm, and wrote the original draft of the manuscript. SLW designed and implemented the OptiFit algorithm and assisted in debugging the analysis code. MBM and GAD contributed analysis code. PDS conceived the study, supervised the project, and assisted in debugging the analysis code. All authors reviewed and edited the manuscript.

1. **Westcott SL**, **Schloss PD**. 2017. OptiClust, an Improved Method for Assigning Amplicon-Based Sequence Data to Operational Taxonomic Units. mSphere **2**:e00073–17. doi:10.1128/mSphereDirect.00073-17.

2. **Schloss PD**, **Westcott SL**, **Ryabin T**, **Hall JR**, **Hartmann M**, **Hollister EB**, **Lesniewski RA**, **Oakley BB**, **Parks DH**, **Robinson CJ**, **Sahl JW**, **Stres B**, **Thallinger GG**, **Van Horn DJ**, **Weber CF**. 2009. Introducing mothur: Open-source, platform-independent, community-supported software for describing and comparing microbial communities. Applied and Environmental Microbiology **75**:7537–7541. doi:10.1128/AEM.01541-09.

3. **Westcott SL**, **Schloss PD**. 2015. De novo clustering methods outperform reference-based methods for assigning 16S rRNA gene sequences to operational taxonomic units. PeerJ **3**:e1487. doi:10.7717/peerj.1487.

4. **Topçuoğlu BD**, **Lesniak NA**, **Ruffin M**, **Wiens J**, **Schloss PD**. 2019. Effective application of machine learning to microbiome-based classification problems. bioRxiv 816090. doi:10.1101/816090.

5. **Schloss PD**. 2016. Application of a Database-Independent Approach To Assess the Quality of Operational Taxonomic Unit Picking Methods. mSystems **1**:e00027–16. doi:10.1128/mSystems.00027-16.

6. **Johnston ER**, **Rodriguez-R LM**, **Luo C**, **Yuan MM**, **Wu L**, **He Z**, **Schuur EAG**, **Luo Y**,

Tiedje JM, **Zhou J**, **Konstantinidis KT**. 2016. Metagenomics Reveals Pervasive Bacterial Populations and Reduced Community Diversity across the Alaska Tundra Ecosystem. Front Microbiol **7**. doi:10.3389/fmicb.2016.00579.

7. **Henson MW**, **Pitre DM**, **Weckhorst JL**, **Lanclos VC**, **Webber AT**, **Thrash JC**. 2016. Artificial Seawater Media Facilitate Cultivating Members of the Microbial Majority from the Gulf of Mexico. mSphere **1**. doi:10.1128/mSphere.00028-16.

8. **Schloss PD**, **Schubert AM**, **Zackular JP**, **Iverson KD**, **Young VB**, **Petrosino JF**. 2012. Stabilization of the murine gut microbiome following weaning. Gut Microbes **3**:383–393. doi:10.4161/gmic.21008.

9. **Baxter NT**, **Ruffin MT**, **Rogers MAM**, **Schloss PD**. 2016. Microbiota-based model improves the sensitivity of fecal immunochemical test for detecting colonic lesions. Genome Med **8**:37. doi:10.1186/s13073-016-0290-3.

10. **Köster J**, **Rahmann S**. 2012. Snakemake a scalable bioinformatics workflow engine. Bioinformatics **28**:2520–2522. doi:10.1093/bioinformatics/bts480.

11. **Van Rossum G**, **Drake FL**. 2009. Python 3 Reference Manual | Guide books.

12. **R Core Team**. 2020. R: A language and environment for statistical computing. Manual, R Foundation for Statistical Computing, Vienna, Austria.

13. **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.

14. **Harris CR**, **Millman KJ**, **van der Walt SJ**, **Gommers R**, **Virtanen P**, **Cournapeau D**, **Wieser E**, **Taylor J**, **Berg S**, **Smith NJ**, **Kern R**, **Picus M**, **Hoyer S**, **van Kerkwijk MH**, **Brett M**, **Haldane A**, **del Río JF**, **Wiebe M**, **Peterson P**, **Gérard-Marchant P**, **Sheppard K**, **Reddy T**, **Weckesser W**, **Abbasi H**, **Gohlke C**, **Oliphant TE**. 2020. Array programming

with NumPy. Nature **585**:357–362. doi:10.1038/s41586-020-2649-2.

15. **Wickham H**, **Averick M**, **Bryan J**, **Chang W**, **McGowan LD**, **François R**, **Grolemund G**, **Hayes A**, **Henry L**, **Hester J**, **Kuhn M**, **Pedersen TL**, **Miller E**, **Bache SM**, **Müller K**, **Ooms J**, **Robinson D**, **Seidel DP**, **Spinu V**, **Takahashi K**, **Vaughan D**, **Wilke C**, **Woo K**, **Yutani H**. 2019. Welcome to the Tidyverse. Journal of Open Source Software **4**:1686. doi:10.21105/joss.01686.

16. **Xie Y**, **Allaire JJ**, **Grolemund G**. 2018. R Markdown: The Definitive Guide. Taylor & Francis, CRC Press.

17. 2016. Anaconda Software Distribution. Anaconda Documentation. Anaconda Inc.

References