

**A peer-reviewed version of this preprint was published in PeerJ on 21 August 2014.**

[View the peer-reviewed version](https://peerj.com/articles/545) (peerj.com/articles/545), which is the preferred citable publication unless you specifically need to cite this preprint.

Rideout JR, He Y, Navas-Molina JA, Walters WA, Ursell LK, Gibbons SM, Chase J, McDonald D, Gonzalez A, Robbins-Pianka A, Clemente JC, Gilbert JA, Huse SM, Zhou H, Knight R, Caporaso JG. 2014. Subsampled open-reference clustering creates consistent, comprehensive OTU definitions and scales to billions of sequences. PeerJ 2:e545 <https://doi.org/10.7717/peerj.545>

## **Consistent, comprehensive and computationally efficient OTU definitions.**

We present a performance-optimized algorithm, subsampled open-reference OTU picking, for assigning marker gene (e.g., 16S rRNA) sequences generated on next-generation sequencing platforms to operational taxonomic units (OTUs) for microbial community analysis. This algorithm provides benefits over de novo OTU picking (clustering can be performed largely in parallel, reducing runtime) and closed-reference OTU picking (all reads are clustered, not only those that match a reference database sequence with high similarity). Because more of our algorithm can be run in parallel relative to “classic” open-reference OTU picking, it makes open-reference OTU picking tractable on massive amplicon sequence data sets (though on smaller data sets, “classic” open-reference OTU clustering is often faster). We illustrate that here by applying it to the first 15,000 samples sequenced for the Earth Microbiome Project (1.3 billion V4 16S rRNA amplicons). To the best of our knowledge, this is the largest OTU picking run ever performed, and we estimate that our new algorithm runs in less than 1/5 the time than would be required of “classic” open reference OTU picking. We show that subsampled open-reference OTU picking yields results that are highly correlated with those generated by “classic” open-reference OTU picking through comparisons on three well-studied datasets. An implementation of this algorithm is provided in the popular QIIME software package, which uses uclust for read clustering. All analyses were performed using QIIME’s uclust wrappers, though we provide details (aided by the open-source code in our GitHub repository) that will allow implementation of subsampled open-reference OTU picking independently of QIIME (e.g., in a compiled programming language, where runtimes should be further

reduced). Our analyses should generalize to other implementations of these OTU picking algorithms. Finally, we present a comparison of parameter settings in QIIME's OTU picking workflows and make recommendations on settings for these free parameters to optimize runtime without reducing the quality of the results. These optimized parameters can vastly decrease the runtime of uclust-based OTU picking in QIIME.

# 1 Consistent, comprehensive and computationally 2 efficient OTU definitions

3 Jai Ram Rideout<sup>1,2</sup>, Yan He<sup>3</sup>, Jose A Navas-Molina<sup>4</sup>, William A Walters<sup>5</sup>, Luke K Ursell<sup>6</sup>, Sean M.  
4 Gibbons<sup>7,10</sup>, John Chase<sup>8</sup>, Daniel McDonald<sup>4,9</sup>, Antiono Gonzalez<sup>9</sup>, Adam Robbins-Pianka<sup>4,9</sup>, Jose C.  
5 Clemente<sup>2</sup>, Jack A. Gilbert<sup>10,11</sup>, Susan M. Huse<sup>12</sup>, Hong-Wei Zhou<sup>3</sup>, Rob Knight<sup>9,13</sup>, and J Gregory  
6 Caporaso<sup>1,8,\*</sup>

7 <sup>1</sup> Center for Microbial Genetics and Genomics, Northern Arizona University, Flagstaff, AZ, USA.

8 <sup>2</sup> Department of Genetics and Genomic Sciences, Icahn School of Medicine at Mount Sinai, New York,  
9 NY, USA.

10 <sup>3</sup> State Key Laboratory of Organ Failure Prevention, and Department of Environmental Health, School of  
11 Public Health and Tropical Medicine, Southern Medical University, Guangzhou, Guangdong, China  
12 510515.

13 <sup>4</sup> Department of Computer Science, University of Colorado, Boulder, Colorado, USA.

14 <sup>5</sup> Department of Molecular, Cellular and Developmental Biology, University of Colorado at Boulder,  
15 Boulder, Colorado, USA.

16 <sup>6</sup> Department of Chemistry & Biochemistry, University of Colorado at Boulder, Boulder, CO 80309, USA.

17 <sup>7</sup> Graduate Program in Biophysical Sciences, University of Chicago, Chicago, Illinois, USA.

18 <sup>8</sup> Department of Biological Sciences, Northern Arizona University, Flagstaff, AZ, USA.

19 <sup>9</sup> BioFrontiers Institute, University of Colorado at Boulder, Boulder, Colorado, USA.

20 <sup>10</sup> Institute for Genomics and Systems Biology, Argonne National Laboratory, Lemont, Illinois, USA.

21 <sup>11</sup> Department of Ecology and Evolution, University of Chicago, Chicago, Illinois, USA.

22 <sup>12</sup> Department of Pathology and Laboratory Science, Warren Alpert Medical School, Brown University,  
23 Providence, Rhode Island, USA.

24 <sup>13</sup> Howard Hughes Medical Institute, Boulder, Colorado, USA.

25 \* Corresponding author (gregcaporaso@gmail.com)

## 26 Introduction

27 Three high-level strategies for defining Operational Taxonomic Unit (OTU) cluster centroids  
28 have been widely applied for centroid-based greedy clustering (Li and Godzik 2006; Edgar 2010)  
29 of marker gene (e.g., 16S rRNA) sequences generated on next-generation sequencing platforms  
30 to facilitate microbial community analysis. These are canonically described as *de novo*, closed-  
31 reference, and open-reference OTU picking (Navas-Molina et al. 2013). In each of these  
32 approaches, respectively, centroids are defined internally based only on the sequences being  
33 clustered, based only on an external, predefined database of cluster centroids, or based on a  
34 combination of the two. Each of these methods has benefits and drawbacks.

35 In *de novo* OTU picking, input sequences are aligned against one another, and sequences that  
36 align with greater than a user-specified percent identity are defined as belonging to the same  
37 OTU. There are many variations and free parameters in this process, such as how many  
38 alignments are performed before a sequence is assigned to an OTU or used to define a new OTU,  
39 but the common feature of these methods is that no external reference database is required. This  
40 is also the primary advantage of this method: it is not necessary to have accumulated a collection

41 of reference sequences before working with a new marker gene. However, *de novo* OTU picking  
42 is difficult to parallelize because all processes must be able to use new OTUs that are defined by  
43 other processes. Consequently, this approach cannot scale to modern-sized data sets.

44 In closed-reference OTU picking, input sequences are aligned to pre-defined cluster centroids in  
45 a reference database. If the input sequence does not match any reference sequence at a user-  
46 defined percent identity threshold, that sequence is excluded. The primary advantage of closed-  
47 reference OTU picking is that it is easily parallelizable. Because the cluster centroids are  
48 predefined, the input sequence collection can be partitioned into  $n$  subsets, the assignment  
49 process can be split across  $n$  processors, and the clustering results can be collated when all  
50 processes have completed. This dramatically reduces the “wall time” (i.e., the total time to  
51 completion as you would see it on a clock on the wall, not in terms of CPU  $\times$  hours) of this  
52 method, and makes closed-reference OTU picking a convenient strategy for extremely large  
53 datasets (e.g., as in (Yatsunenkov et al. 2012)). Additionally, it has the convenient feature that,  
54 because OTUs are defined by a pre-existing reference, there are typically high-quality taxonomic  
55 assignments for each OTU, and a high-quality phylogenetic tree, often based on full-length  
56 sequences rather than fragments, exists and describes the relationships among those OTUs.  
57 Furthermore, because input sequences are not compared directly to one another, but rather to an  
58 external reference, the input sequences need not overlap. This is essential, for example, if  
59 performing a meta-analysis including sequences derived from different amplification products of  
60 the same marker gene, such as the V2 and V4 regions of the 16S rRNA (e.g., as in the meta-  
61 analysis performed in (Caporaso et al. 2010)). The major drawback to closed-reference OTU  
62 picking, however, is that it cannot identify novel diversity: if a sequence has no match in the  
63 reference database, it cannot be included in the analysis, restricting analyses to already-known  
64 taxa. (Of course, the importance of this limitation decreases as the reference database increases in  
65 coverage.)

66 Finally, open-reference OTU picking combines the previous protocols. First, input sequences are  
67 clustered against a reference database in parallel in a closed-reference OTU picking process.  
68 However, rather than discarding sequences that fail to match the reference, these “failures” are  
69 clustered *de novo* in a serial process. Open-reference OTU picking offers benefits over both the  
70 *de novo* and closed-reference protocols. Because it includes the parallel closed-reference step, it  
71 will typically run faster than *de novo* OTU picking. And, since it includes *de novo* OTU picking  
72 of the sequences that fail to hit the reference database, all sequences are clustered, so analyses are  
73 not restricted to already-known OTUs. However, because the *de novo* clustering process is run  
74 serially, it can still be prohibitively slow for very large datasets or datasets with a substantial  
75 number of sequences that fail to hit the reference database. Because of these long runtimes, it has  
76 not yet been widely applied despite the benefits it offers.

77 We present a novel strategy for open-reference OTU picking that allows a larger portion of the  
78 computation to be run in parallel, which we call *subsampled open-reference OTU picking*,  
79 allowing open-reference OTU picking on very large datasets. We compare this method to

80 “classic” open-reference OTU picking (as described in the previous paragraph) to confirm that,  
81 despite potentially slightly different OTU definitions, the summary statistics that are often used  
82 derive biological conclusions from application of these different methods to the same data set  
83 would remain the same. To achieve this, we show that alpha diversity, beta diversity, and  
84 taxonomic profiles are highly correlated between the “classic” open-reference OTU picking and  
85 subsampled open-reference OTU picking. We also compare these methods to *de novo* and closed-  
86 reference OTU picking, and explore the effect of dataset and algorithm parameters on runtime  
87 and analysis results. We note that we specifically focus on centroid-based greedy clustering  
88 approaches in this study (e.g., as in *uclust* and *cd-hit* (Li and Godzik 2006; Edgar 2010)), not  
89 approaches that require alignment of all pairs of unique sequences (i.e., the hierarchical methods  
90 described in (Schloss and Westcott 2011)), as the former scale better to larger data sets. However,  
91 because our full evaluation framework (metrics and data sets) and the EMP raw sequence data are  
92 all freely accessible, it is straight-forward for other groups to reproduce these evaluations on  
93 alternative methods.

94 All analyses presented here are performed using the QIIME and pandas python packages. As far  
95 as we know, QIIME contains the only existing implementation of the subsampled open-reference  
96 OTU picking algorithm, but the algorithm is not QIIME-specific. Thus while our comparison is  
97 based on specific QIIME/*uclust*-based implementations of *de novo*, *closed reference*, *classic*  
98 *open reference*, and *subsampled open reference* OTU picking, our findings should be general to  
99 other implementations of these algorithms.

## 100 Materials and Methods

### 101 **Subsampled open-reference OTU picking algorithm**

102 Open-reference OTU picking is preferable to the other methods presented here because it  
103 combines the advantages of closed-reference and *de novo* clustering. However, the *de novo* step  
104 of open-reference OTU picking can only be run serially, and therefore can be time-consuming for  
105 large datasets if many sequences fail to hit the reference database. To improve the runtime of  
106 open-reference OTU picking, we developed *subsampled open-reference OTU picking*, which  
107 incrementally increases the size of the reference database by *de novo* clustering a subset of the  
108 sequences that fail to match the reference database. The remainder of the sequences that fail to hit  
109 the reference database can then be clustered against these new cluster centroids in a parallel  
110 closed-reference OTU picking process. This allows for partial parallelization of the *de novo*  
111 clustering step and can significantly decrease runtime on large datasets, allowing open-reference  
112 OTU picking to scale to billions of input sequences (e.g., as generated in multiple Illumina HiSeq  
113 2000 runs). It can additionally be run iteratively, so that representative sequences for the new  
114 (i.e., non-reference) OTUs can be combined with the reference database for future OTU picking  
115 runs. It is important to note that runtime is not always reduced with subsampled open-reference  
116 OTU picking. Data set and algorithm parameters have a large effect on runtime (discussed further  
117 in *Runtime differences*). This approach is similar to the Buckshot algorithm (Cutting et al. 1992;  
118 Jensen et al. 2002), initially described for semantic clustering of documents in a corpus, though

119 we do not use the parallel hierarchical clustering approach described by (Jensen et al. 2002) for  
120 initial clustering definition.

121 A detailed description of this workflow is illustrated in Figure 1. It is implemented using uclust  
122 v1.2.22q (Edgar 2010) for clustering in QIIME 1.6.0 (Caporaso et al. 2010) and later, though any  
123 sequence clustering software that provides support for *de novo* and closed-reference clustering  
124 could be substituted for uclust in an alternate implementation. The inputs provided to this method  
125 are demultiplexed, quality-filtered sequences, and a reference sequence collection (for example,  
126 the Greengenes 13\_8 97% OTU representative sequences (DeSantis et al. 2006; McDonald,  
127 Price, et al. 2012)). First, sequences are clustered in parallel using a closed-reference OTU  
128 picking workflow, where sequences are queried against the reference database at percent identity  
129  $s$  (default 97%). If a read matches a reference sequence at greater than or equal to  $s\%$  identity, it  
130 is assigned to the OTU defined by that reference sequence. These are referred to as the reference  
131 OTUs. Next, a random subsample of  $n\%$  ( $n$  should be small, the default value in QIIME 1.8.0-  
132 dev and earlier is 0.1%) of the sequences that failed to match the reference sequence collection  
133 are clustered *de novo*, and the cluster centroids for all resulting OTUs are used to define a new  
134 reference sequence collection. Those OTUs are referred to as the new reference OTUs. The  
135 sequences that were not included in the random subsample that was clustered *de novo* then go  
136 through an additional round of parallel closed-reference OTU picking, this time where they are  
137 clustered against the new reference OTUs based on matching a sequence in the new reference  
138 sequence collection at greater than or equal to  $s\%$  identity. This creation of a “new reference  
139 database” allows us to harness the parallelization of our closed-reference OTU picking pipeline,  
140 greatly decreasing the time it takes for sequences that fail to hit the initial reference database to  
141 be clustered into OTUs. In the final clustering step, sequences that fail to hit a reference sequence  
142 during this final closed-reference OTU picking step are clustered *de novo*. These are referred to  
143 as the clean-up OTUs. Finally, the reference OTUs, new reference OTUs, and clean-up OTUs are  
144 combined into a single OTU table (i.e., table of counts of OTUs on a per-sample basis, as  
145 described in (McDonald, Clemente, et al. 2012)), and this table, as well as a filtered table  
146 excluding OTUs with counts less than or equal to a user-defined threshold  $c$ , are provided to the  
147 user. By default,  $c=2$ , so each OTU is observed at least twice (i.e., singleton OTUs are excluded).  
148 Because many more of the sequences can be clustered using closed-reference OTU picking in  
149 this workflow, it can run in far less time than classic open-reference OTU picking (see *Runtime*  
150 *Differences* section below).

### 151 **Evaluation of subsampled open-reference OTU picking**

152 We validated the subsampled open-reference OTU picking workflow by comparing it to *de novo*,  
153 closed-reference, and classic (i.e., non subsampled) open-reference clustering methods on three  
154 different datasets: the Lauber “88 Soils” study (Lauber et al. 2009) (referred to as *88-soils* here),  
155 the Caporaso “Moving Pictures” study (Caporaso et al. 2011) (referred to as *moving-pictures*  
156 here), and the Costello “Whole Body” study (Costello et al. 2009) (referred to as *whole-body*  
157 here) using three metrics. Table 1 provides a description of the OTU picking methods being  
158 compared. First, we tested the correlation between sample alpha diversities (OTU counts, i.e.  
159 QIIME’s *observed species* metric, and Phylogenetic Diversity (PD) (Faith 1992)) based on

160 subsampled open-reference OTU picking and the other OTU picking protocols. Next, we tested  
161 whether beta diversity patterns (as determined by weighted and unweighted UniFrac (Lozupone  
162 and Knight 2005) distances between samples) were consistent across OTU picking protocols,  
163 based on Mantel tests (Mantel 1967) with 1000 Monte Carlo iterations. Finally, we tested  
164 whether the same taxonomic profiles were obtained on a per-sample basis using each of the OTU  
165 picking methods. It is important to note that we are not trying to assess whether one method is  
166 better than another using these metrics. Instead, we are testing whether the methods give highly  
167 correlated results.

## 168 **Data availability**

169 The raw sequence data analyzed in this study is available in the QIIME Database under accession  
170 numbers 103 (88-soils), 449 (whole-body), and 550 (moving-pictures). All analyses were run  
171 with QIIME 1.8.0-dev. All commands, as well as all processed data and IPython Notebooks that  
172 illustrate how to work with that data are available in this project's GitHub repository at  
173 <https://github.com/gregcaporaso/cloaked-octo-ninja>.

## 174 **Results and Discussion**

### 175 **Subsampled versus “classic” open-reference OTU picking**

176 Alpha diversity (Table 2; whole-body PD Pearson  $r=0.989$ ; 88-soils PD Pearson  $r=0.930$ ;  
177 moving-pictures PD Pearson  $r=0.996$ ), beta diversity (Table 3; whole-body unweighted UniFrac  
178 Mantel  $r=0.948$ ; 88-soils unweighted UniFrac Mantel  $r=0.939$ ; moving-pictures unweighted  
179 UniFrac Mantel  $r=0.991$ ) and taxonomic summaries (Table 4; whole-body:  $r=0.999$  at phylum  
180 level,  $0.999$  at species level; 88-soils  $r=0.999$  at phylum level,  $r=0.999$  at species level; moving-  
181 pictures  $r=0.999$  at phylum level,  $r=0.999$  at species level) were highly correlated between classic  
182 and subsampled open-reference OTU picking. Minor differences likely arise from the non-  
183 deterministic step of rarefying all samples to even sampling depth before comparing samples.  
184 These results suggest that subsampled open-reference picking yields the same results as classic  
185 open-reference OTU picking, including identical numbers of sequences failing to hit the  
186 reference database, and therefore is a suitable replacement.

### 187 **Application to the Earth Microbiome Project dataset**

188 In order to evaluate the effectiveness of the subsampled open-reference OTU picking method on  
189 an extremely large data set, the first 15,000 samples (1.3 billion V4 16S rRNA amplicons) from  
190 the Earth Microbiome Project (EMP, (Gilbert et al. 2010)) were processed on the Amazon Web  
191 Services (AWS) EC2 platform. These samples were split across more than 60 studies, which were  
192 clustered iteratively. To the best of our knowledge, this is the largest OTU picking run ever  
193 completed. We created a StarCluster-based (<http://star.mit.edu/cluster/>) virtual cluster on AWS  
194 using between 8 and 18 M2.4xlarge spot instances (the number of instances was varied at  
195 different stages of the run). Each instance (or virtual cluster node) had 69 GB RAM and 8 cores.  
196 A total of 11,242 CPU hours were consumed to complete subsampled open-reference OTU  
197 picking (at 97% nucleotide identity), and the combined input and output files consumed 1.2 TB  
198 of disk space. (This runtime includes the pre-filtering step. The process would have completed



199 much faster if this were disabled.) The resulting OTU table contained 5.6 million non-singleton  
200 OTUs. This is the largest number of OTUs identified, and the most comprehensive survey of  
201 microbial diversity across environment types to date, so it likely suggests the magnitude of the  
202 lower-bound on the microbial diversity of the Earth (although the accuracy is limited because  
203 some of these OTUs may be artifacts of PCR or sequencing: such artifacts, e.g. chimeras, need to  
204 be identified after the OTU picking step).

205 We were next interested in how long the de novo clustering step of classic open-reference OTU  
206 picking would take on the EMP data set, but as we'll illustrate this is an intractable problem in  
207 practice with current computer hardware. We began by applying de novo clustering using the  
208 "fast" uclust parameter settings to the representative sequences from the 5.6 million non-  
209 singleton OTUs from the run described above. These representative sequences represent the full  
210 alpha diversity of the EMP data set (a property known to be important to runtime of de novo and  
211 open reference OTU clustering) but the data set contains only 5.6m sequences, so is feasible to  
212 cluster de novo. We then subsampled this to contain between 10% and 80% of those sequences,  
213 in steps of 10% with 10 iterations at each step, and compiled the runtime for each clustering run.  
214 Figure 2 illustrates the relationship between runtime and input sequence count, along with the  
215 results of a regression analysis presenting median runtime as a function of sequence count  
216 ( $r^2=0.98$ ,  $p=8e-6$ ).

217 In the subsampled open-reference OTU picking run on the EMP dataset, 660 million sequences  
218 failed to hit the reference database, and therefore need to be clustered de novo clustering in open-  
219 reference OTU picking. While it is obviously problematic to use a regression model trained on  
220 5.6 million sequences to extrapolate the runtime on 660 million sequences, we feel that this can  
221 give us an idea of the magnitude of the runtime for the serial de novo clustering of the full  
222 dataset. Our regression model projects that the serial de novo clustering of sequences that fail to  
223 hit the reference data set would require approximately 150 days to run (in wall time). In contrast,  
224 the subsampled open-reference OTU picking run presented here (which included the pre-filtering  
225 step) ran in just under 30 days of wall time. This illustrates that while on relatively small data sets  
226 the performance enhancement of subsampled relative to classic open-reference OTU picking is  
227 either non-existence or modest (discussed in *Run-time differences*), on datasets at the current  
228 upper limit of size, the increased parallelizability of subsampled open-reference OTU picking  
229 makes open-reference OTU picking far more tractable.

### 230 **Run-time differences**

231 The speed improvements of subsampled open-reference OTU picking arise from the fact that a  
232 larger portion of the clustering process can be parallelized. When not run in parallel, or run in  
233 parallel over only a few (e.g., 3) CPUs, classic open-reference OTU picking is likely to be faster.  
234 Similarly, for smaller data sets (e.g., less than a few million sequences), especially if most  
235 sequences have a match in the reference database (e.g., with human gut microbiome data), classic  
236 open-reference OTU picking will achieve similar runtimes to subsampled open-reference  
237 clustering (Table 5). However, in these cases, the results are still highly correlated, so if in doubt  
238 of which method will be faster, subsampled open-reference OTU picking is a reasonable choice

239 as the summary statistics of interest (often alpha diversity, beta diversity and taxonomic profiles)  
240 are very unlikely to be different between the two methods.

241 When more sequences fail to hit the reference database, subsampled open-reference OTU picking  
242 becomes faster than classic open-reference OTU picking (Table 6). To illustrate this, we clustered  
243 the moving-pictures sequences against the 82% and 97% Greengenes reference OTUs at 97%  
244 identity using subsampled and classic open-reference OTU picking on 29 processors. When  
245 clustering against the 82% OTUs, 52.1 million failed to hit the reference, while when clustering  
246 against the 97% OTUs 3.4 million sequences failed to hit the reference. Subsampled open-  
247 reference OTU picking ran in 4000s less wall time than classic open-reference clustering (in a  
248 single run of each on a system dedicated for this run time comparison) against the 82% OTUs,  
249 and in 72s less time against the 97% OTUs, illustrating that as more sequences fail to hit the  
250 reference, subsampled open-reference OTU picking offers more of an advantage. This runtime  
251 difference would be even larger if the job were split over more processors.

252 Another parameter that can affect runtime of subsampled open-reference OTU picking is the size  
253 of the random subsample that is selected. The optimal setting for this parameter is affected by the  
254 size of the dataset being clustered and the diversity of the sequences that fail to match the  
255 reference database. On small datasets, or datasets with a lot of novel diversity, a large fraction  
256 (e.g., 1%) is better than a small fraction (e.g., 0.001%), but as the data set increases in size a large  
257 fraction can result in far more time spent performing *de novo* clustering of the sequences that  
258 initially fail to hit the reference database. We recommend using the default (0.1% in QIIME  
259 1.8.0-dev and earlier), which was chosen to reduce runtime on larger datasets where optimized  
260 runtime is more important. As this parameter setting approaches zero, subsampled open-reference  
261 OTU picking becomes more like classic open-reference OTU picking, in that more of the reads  
262 that fail to hit the reference database are clustered *de novo* serially, and at the limit of 0% of  
263 sequences subsampled, subsampled open reference OTU picking becomes classic open-reference  
264 OTU picking. The summary statistics investigated here are highly correlated between classic and  
265 subsampled open-reference OTU picking, suggesting that this parameter setting will not affect  
266 those statistics, but can affect runtime.

### 267 **Pre-filtering**

268 QIIME's open-reference OTU picking workflow optionally includes a pre-filtering step, where  
269 sequences are searched against the reference database with low percent identity (the default in  
270 QIIME 1.8.0 and earlier is 60%), and sequences that fail to match are discarded from the  
271 analysis. The goal of this process is to discard sequences that are likely not representatives of the  
272 marker gene, such as host genomic sequences or products of non-specific amplification. This  
273 process is functionally similar to closed-reference OTU picking (sequence reads are searched  
274 against a pre-defined reference database), and therefore is easily run in parallel.

275 We show that alpha diversity (Table 2; whole-body PD Pearson  $r=0.991$ ; 88-soils PD Pearson  
276  $r=0.930$ ; moving-pictures PD Pearson  $r=0.996$ ), beta diversity (Table 3; whole-body unweighted  
277 UniFrac Mantel  $r=0.953$ ; 88-soils unweighted UniFrac Mantel  $r=0.940$ ; moving-pictures

278 unweighted UniFrac Mantel  $r=0.990$ ) and taxonomic summaries (Table 4; whole-body:  $r=1.000$   
279 at phylum level,  $r=1.000$  at species level; 88-soils  $r=1.000$  at phylum level,  $r=1.000$  at species  
280 level; moving-pictures  $r=1.000$  at phylum level,  $r=0.999$  at species level) are highly correlated  
281 between the pre-filtered and non-pre-filtered results, when pre-filtering is performed at percent  
282 identity of 60%. Despite nearly identical results, the pre-filtering process results in vastly  
283 increased runtimes. Consequently, we no longer recommend pre-filtering of sequences prior to  
284 open-reference OTU picking. Rather, contaminant sequences should be discarded after OTU  
285 picking. This feature is now disabled by default starting with QIIME 1.8.0-dev.

286 One case where pre-filtering may prove useful is in the preparation of sequence data where there  
287 is a large amount of contamination of non-marker-gene sequence, for example host genomic  
288 contamination. In this case, pre-filtering can be useful to remove those sequences prior to  
289 clustering. Note that if you suspect that your sample may contain human genomic contaminant  
290 sequences, it is important to filter them out before analysis or data deposition due to Institutional  
291 Review Board or other ethical concerns related to release of human DNA sequences.

### 292 **Clustering parameters**

293 We also investigated the effect of clustering parameters on the same summary statistics, as these  
294 can have a considerable effect on runtime. We compared uclust's default settings (referred to in  
295 QIIME as "fast mode") with the default settings in QIIME 1.8.0 and earlier ("slow mode"). We  
296 again compared the methods based on the degree to which they resulted in correlated alpha  
297 diversity (Table 2), beta diversity (Table 3), and taxonomic results (Table 4), and found that all  
298 results were highly correlated between fast and slow modes. This suggests that while fast mode  
299 will occasionally make suboptimal OTU assignments, the effects are subtle enough to be  
300 unnoticeable in downstream ecological analyses. We therefore recommend using the "fast"  
301 settings for decreased runtime, and these are now the default in QIIME 1.8.0-dev.

302 We do recommend using the "slow" settings if clustering sequences to build reference OTUs (for  
303 example, as is performed when building the Greengenes reference OTU collection (McDonald,  
304 Price, et al. 2012)) because suboptimal OTU assignments can have further reaching  
305 consequences. For example, "splitting" an OTU (i.e., defining two sequences that are within  $s\%$   
306 identity of each other as the centroids of two different  $s\%$  OTUs), which is always a possibility in  
307 greedy clustering algorithms, is more common with the "fast" settings than with the "slow"  
308 settings. If this occurs in a single study, the downstream effects are limited to that study and are  
309 likely only to be problematic if the split OTU is of key significance to the system being  
310 investigated. However, a split OTU when defining reference OTUs is more problematic, because  
311 those definitions will be used in many studies, increasing the chance that the split OTU will be  
312 problematic for someone. For this application, the processing step is typically only run once per  
313 database release (which is relatively infrequent). Therefore, the longer runtime is preferable to  
314 less accurate OTU definitions in this particular application. If splitting and lumping of OTUs is of  
315 concern on your dataset, you may want to experiment with the "slow" parameter settings, which  
316 are still accessible in QIIME and we also recommend exploring the use of Oligotyping (Eren et  
317 al. 2013).

### 318 **Consistent OTU definitions across runs: iterative open-reference OTU picking**

319 Subsampled open-reference clustering, as implemented in QIIME, provides new identifiers for  
320 sequences that fail to match the reference database, allowing OTUs to be directly compared  
321 across clustering runs (although sequences clustered against this expanded reference sequence  
322 collection do need to be from the same gene fragment as the sequences used to expand the  
323 reference sequence collection). These OTUs can also be used in iterative OTU picking, which is  
324 useful in studies where sequence data is continuously accumulating, for example in routine  
325 monitoring of microbial communities in human subjects (e.g. patients monitored over time), the  
326 built-environment, or during environmental clean-up.

### 327 **Conclusions**

328 Taken together, the reduced runtime of subsampled open-reference OTU picking relative to  
329 classic open-reference OTU picking on large datasets, and the benefits that open-reference OTU  
330 picking offers over full *de novo* OTU picking (vastly decreased runtime) and closed-reference  
331 OTU picking (all sequences are clustered, not only those that match the reference collection), we  
332 recommend subsampled open-reference OTU picking when a reference collection is available.

333 Because the metrics provided here show that the same summary statistics are derived from the  
334 four OTU picking protocols, an interesting question is whether *de novo* or open-reference OTU  
335 picking offers any benefit over closed-reference OTU picking. The primary motivation for using  
336 methods that incorporate previously unknown OTUs (i.e., those that are not represented in the  
337 reference database) such as *de novo* and open-reference OTU picking is that OTUs not  
338 represented in the reference database might best illustrate a biological pattern of interest. For  
339 example, in the 88-soils data analyzed here, 1 of the top 10 OTUs identified as significantly  
340 different across sample pH is an OTU that is not represented in the reference database (Table 8)  
341 (this OTU was classified as in the *Actinomycetales* order by QIIME's uclust-based taxonomy  
342 classifier). Similarly, for the whole-body data set, 2 of the top 10 OTUs identified as significantly  
343 different across body sites were not represented in the reference database (these were classified as  
344 *Prevotella melaninogenica* and *Veillonella parvula* by QIIME's uclust-based taxonomy  
345 classifier). On the other hand, in the moving-pictures data analyzed here, all of the top 10 OTUs  
346 identified as significantly different across body site were OTUs represented in the reference  
347 database. Table 7 illustrates the fraction of OTUs not represented in the reference database by  
348 environment based on the Earth Microbiome Project dataset. We expect that using OTU picking  
349 methods that incorporate new OTUs is more important in samples where this fraction is higher.

350 In conclusion, this paper presents the performance-optimized subsampled open-reference OTU  
351 picking algorithm, now available in QIIME. This method can be applied iteratively to define  
352 stable OTUs across sequencing runs, and achieves nearly identical results to "classic" open-  
353 reference OTU picking (i.e., not including the subsampling step). It enables massive sequencing  
354 projects such as the Earth Microbiome Project to use open-reference OTU picking in far less time  
355 than is possible with classic open-reference OTU picking, which will facilitate our exploration of  
356 microbial diversity. Further, the iterative nature of the process (which is also possible with classic

357 open-reference OTU picking) enables progressively expanding datasets, as might be generated in  
358 clinical laboratories as microbiome-based medical treatment becomes a reality, to cluster OTUs  
359 using OTU definitions from previous clustering runs as reference sequences. This avoids re-  
360 clustering all sequences every time new sequences are generated, thereby vastly decreasing  
361 computational costs.

## 362 Acknowledgements

363 Sample processing, sequencing and core amplicon data analysis for samples included in the Earth  
364 Microbiome Project analysis were performed by the Earth Microbiome Project  
365 ([www.earthmicrobiome.org](http://www.earthmicrobiome.org)) and all amplicon and metadata has been made public through the  
366 data portal ([www.microbio.me/emp](http://www.microbio.me/emp)).

## 367 References

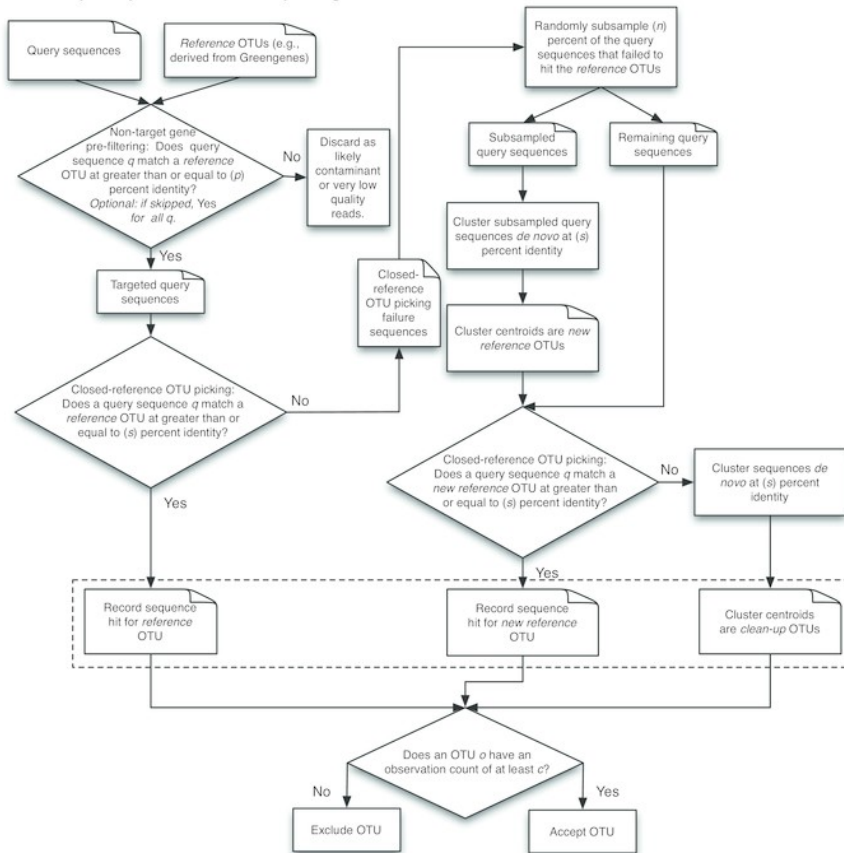
- 368 Caporaso, J Gregory, Justin Kuczynski, Jesse Stombaugh, Kyle Bittinger, Frederic D Bushman, Elizabeth  
369 K Costello, Noah Fierer, et al. 2010. "QIIME Allows Analysis of High-Throughput Community  
370 Sequencing Data." *Nature Methods* 7 (5): 335–36.
- 371 Caporaso, J Gregory, Christian L Lauber, Elizabeth K Costello, Donna Berg-Lyons, Antonio Gonzalez,  
372 Jesse Stombaugh, Dan Knights, et al. 2011. "Moving Pictures of the Human Microbiome." *Genome  
373 Biology* 12 (5): R50.
- 374 Costello, Elizabeth K, Christian L Lauber, Micah Hamady, Noah Fierer, Jeffrey I Gordon, and Rob  
375 Knight. 2009. "Bacterial Community Variation in Human Body Habitats across Space and Time." *Science* 326 (5960): 1694–97.
- 376 Cutting, Douglass R, David R Karger, Jan O Pedersen, and John W Tukey. 1992. "Scatter/Gather: A  
377 Cluster-Based Approach to Browsing Large Document Collections." In *Proceedings of the 15th  
378 Annual International ACM SIGIR Conference on Research and Development in Information  
379 Retrieval*, 318–29. SIGIR '92. New York, NY, USA: ACM. doi:10.1145/133160.133214.
- 380 DeSantis, T Z, P Hugenholtz, N Larsen, M Rojas, E L Brodie, K Keller, T Huber, D Dalevi, P Hu, and G L  
381 Andersen. 2006. "Greengenes, a Chimera-Checked 16S rRNA Gene Database and Workbench  
382 Compatible with ARB." *Applied and Environmental Microbiology* 72 (7): 5069–72.
- 383 Edgar, Robert C. 2010. "Search and Clustering Orders of Magnitude Faster than BLAST." *Bioinformatics*  
384 26 (19): 2460–61.
- 385 Eren, A Murat, Lois Maignien, Woo Jun Sul, Leslie G Murphy, Sharon L Grim, Hilary G Morrison, and  
386 Mitchell L Sogin. 2013. "Oligotyping: Differentiating between Closely Related Microbial Taxa  
387 Using 16S rRNA Gene Data." *Methods in Ecology and Evolution / British Ecological Society* 4 (12).  
388 doi:10.1111/2041-210X.12114.
- 389 Faith, Daniel P. 1992. "Conservation Evaluation and Phylogenetic Diversity." *Biological Conservation* 61  
390 (1): 1–10.
- 391 Gilbert, Jack A, Folker Meyer, Dion Antonopoulos, Pavan Balaji, C Titus Brown, Christopher T Brown,  
392 Narayan Desai, et al. 2010. "Meeting Report: The Terabase Metagenomics Workshop and the Vision  
393 of an Earth Microbiome Project." *Standards in Genomic Sciences* 3 (3): 243–48.
- 394 Jensen, Eric C, Steven M Beitzel, Angelo J Pilotto, Nazli Goharian, and Ophir Frieder. 2002.  
395 "Parallelizing the Buckshot Algorithm for Efficient Document Clustering." In *Proceedings of the  
396 Eleventh International Conference on Information and Knowledge Management*, 684–86. CIKM '02.  
397 New York, NY, USA: ACM. doi:10.1145/584792.584919.
- 398 Lauber, Christian L, Micah Hamady, Rob Knight, and Noah Fierer. 2009. "Pyrosequencing-Based  
399 Assessment of Soil pH as a Predictor of Soil Bacterial Community Structure at the Continental  
400 Scale." *Applied and Environmental Microbiology* 75 (15): 5111–20.
- 401 Li, Weizhong, and Adam Godzik. 2006. "Cd-Hit: A Fast Program for Clustering and Comparing Large  
402 Sets of Protein or Nucleotide Sequences." *Bioinformatics* 22 (13): 1658–59.
- 403

- 404 Lozupone, Catherine, and Rob Knight. 2005. "UniFrac: A New Phylogenetic Method for Comparing  
405 Microbial Communities." *Applied and Environmental Microbiology* 71 (12): 8228–35.
- 406 Mantel, N. 1967. "The Detection of Disease Clustering and a Generalized Regression Approach." *Cancer*  
407 *Research* 27 (2): 209–20.
- 408 McDonald, Daniel, Jose C Clemente, Justin Kuczynski, Jai Ram Rideout, Jesse Stombaugh, Doug  
409 Wendel, Andreas Wilke, et al. 2012. "The Biological Observation Matrix (BIOM) Format or: How I  
410 Learned to Stop Worrying and Love the Ome-Ome." *GigaScience* 1 (1): 7.
- 411 McDonald, Daniel, Morgan N Price, Julia Goodrich, Eric P Nawrocki, Todd Z DeSantis, Alexander  
412 Probst, Gary L Andersen, Rob Knight, and Philip Hugenholtz. 2012. "An Improved Greengenes  
413 Taxonomy with Explicit Ranks for Ecological and Evolutionary Analyses of Bacteria and Archaea."  
414 *The ISME Journal* 6 (3): 610–18.
- 415 Navas-Molina, José A, Juan M Peralta-Sánchez, Antonio González, Paul J McMurdie, Yoshiki Vázquez-  
416 Baeza, Zhenjiang Xu, Luke K Ursell, et al. 2013. "Advancing Our Understanding of the Human  
417 Microbiome Using QIIME." *Methods in Enzymology* 531: 371–444.
- 418 Schloss, Patrick D, and Sarah L Westcott. 2011. "Assessing and Improving Methods Used in Operational  
419 Taxonomic Unit-Based Approaches for 16S rRNA Gene Sequence Analysis." *Applied and*  
420 *Environmental Microbiology* 77 (10): 3219–26.
- 421 Yatsunenko, Tanya, Federico E Rey, Mark J Manary, Indi Trehan, Maria Gloria Dominguez-Bello, Monica  
422 Contreras, Magda Magris, et al. 2012. "Human Gut Microbiome Viewed across Age and  
423 Geography." *Nature* 486 (7402): 222–27.





# Figure 1

Schematic of the subsampled open-reference OTU picking algorithm.

Subsampled open-reference OTU picking workflow



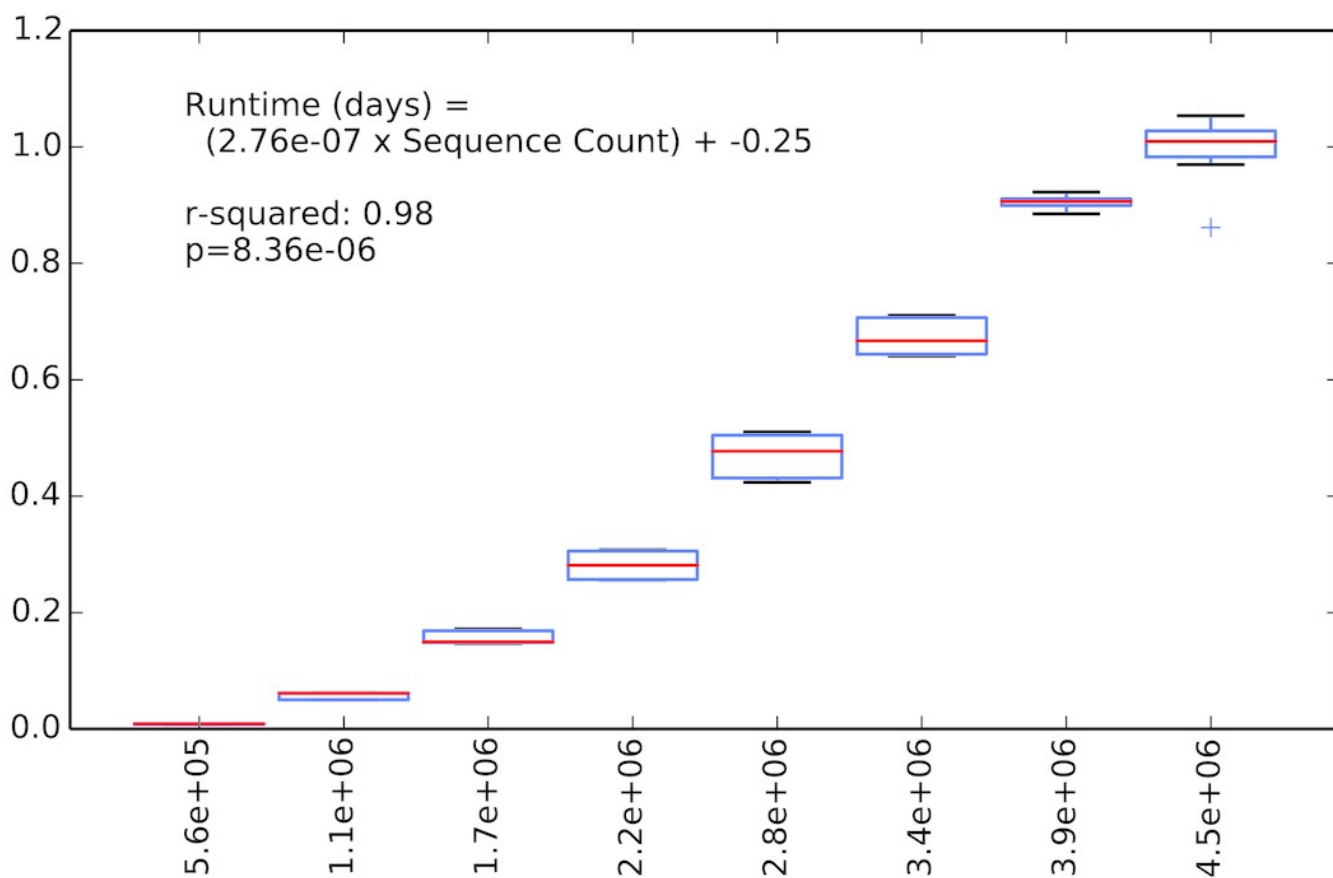
**Legend**

-  Data file (input, intermediate, or output)
-  Per-sequence decision. These are applied individually to all sequences provided as input.
-  Process applied to all sequences provided as input as a collection.
-  Output OTUs
- (p): percent sequence identity threshold used for pre-filtering of sequences (default: 60%)
- (s): percent sequence identity threshold used when clustering sequences either *de novo* or closed-reference (default: 97%)
- (n): percentage of sequences that are randomly subsampled from sequences that failed to hit reference OTUs (default: 0.1%)
- (c): minimum observation count for an OTU to be accepted during post-OTU picking processing (default: 2)

# Figure 2

Runtime comparison.

**Runtime of de novo clustering using “fast” uclust parameters versus number of sequences to be clustered, where sequences are obtained from the EMP subsampled open-reference OTU picking run.**





## **Table 1** (on next page)

Method definitions.

Definitions of the OTU picking methods being compared here, based on the abbreviations used throughout the paper. From here, we refer to each method by its abbreviation for simplicity. We note that the both de novo (uc) and classic open-reference OTU picking (ucr) are accessed through QIIME's `pick_de_novo_otus.py` command. ucr is applied when `pick_otus:otu_picking_method uclust_ref` is specified in the parameters file, and uc is applied when that option is absent. The exact command/parameter combinations used for each OTU picking run are provided in the study's GitHub repository (see Data Availability).

abbreviation	title	command	max_accepts	max_rejects	stepwords	wordlength	prefilter_percent_id	min_otu_size	speed_mode	processors	reference_percent_id	subsample_fraction
<b>uc</b>	De novo	pick_de_novo_otus.py	20	500	20	12 NA	NA		slow	1	0.97 NA	
<b>ucr</b>	Legacy open reference	pick_de_novo_otus.py	20	500	20	12 NA	NA		slow	10	0.97 NA	
<b>ucrC</b>	Closed reference	pick_closed_reference_otus.py	20	500	20	12 NA	NA		slow	10	0.97 NA	
<b>ucrss</b>	Subsampled open reference	pick_open_reference_otus.py	20	500	20	12	0	1	slow	10	0.97	0.001
<b>ucrss_wfilter</b>	Subsampled open reference, filtered	pick_open_reference_otus.py	20	500	20	12	0.6	1	slow	10	0.97	0.001
<b>uc_fast</b>	De novo, fast settings	pick_de_novo_otus.py	1	8	8	8 NA	NA		fast	1	0.97 NA	
<b>ucr_fast</b>	Legacy open reference, fast settings	pick_de_novo_otus.py	1	8	8	8 NA	NA		fast	10	0.97 NA	
<b>ucrC_fast</b>	Closed reference, fast settings	pick_closed_reference_otus.py	1	8	8	8 NA	NA		fast	10	0.97 NA	
<b>ucrss_fast</b>	Subsampled open reference, fast settings	pick_open_reference_otus.py	1	8	8	8	0	1	fast	10	0.97	0.001
<b>ucrss_wfilter_fast</b>	Subsampled open reference, filtered, fast settings	pick_open_reference_otus.py	1	8	8	8	0.6	1	fast	10	0.97	0.001
<b>ucr_fast_O29_r82</b>	Legacy open reference, fast settings, 82% reference OTUs, 29 processors	pick_de_novo_otus.py	1	8	8	8	0	1	fast	29	0.82	0.001
<b>ucr_fast_O29_r97</b>	Legacy open reference, fast settings, 29 processors	pick_de_novo_otus.py	1	8	8	8	0	1	fast	29	0.97	0.001
<b>ucrss_fast_O29_r82</b>	Subsampled open reference, fast settings, 82% reference OTUs, 29 processors	pick_open_reference_otus.py	1	8	8	8	0	1	fast	29	0.82	0.001
<b>ucrss_fast_O29_r97</b>	Subsampled open reference, fast settings, 29 processors	pick_open_reference_otus.py	1	8	8	8	0	1	fast	29	0.97	0.001
<b>ucrss_fast_O29_s1</b>	Subsampled open reference, fast settings, 29 processors, 1% subsample	pick_open_reference_otus.py	1	8	8	8	0	1	fast	29	0.97	0.1

PeerJ Preprints

## **Table 2**(on next page)

Alpha diversity correlation by method and dataset.

**Pearson correlation coefficients (r) of alpha diversity for (a) 88-soils PD, (b) moving-pictures PD, (c) whole-body PD, (d) 88-soils observed species, (e) moving-pictures observed species, and (f) moving-pictures observed species.**

(a)

	uc	ucr	ucrC	ucrss	ucrss_wfilter	uc_fast	ucr_fast	ucrC_fast	ucrss_fast	ucrss_fast_wfilter
uc	1	0.951	0.933	0.934	0.953	0.956	0.936	0.927	0.948	0.947
ucr	0.951	1	0.902	0.931	0.93	0.946	0.94	0.903	0.952	0.944
ucrC	0.933	0.902	1	0.894	0.909	0.905	0.914	0.978	0.902	0.911
ucrss	0.934	0.931	0.894	1	0.929	0.944	0.935	0.894	0.948	0.949
ucrss_wfilter	0.953	0.93	0.909	0.929	1	0.952	0.933	0.903	0.931	0.943
uc_fast	0.956	0.946	0.905	0.944	0.952	1	0.953	0.898	0.956	0.96
ucr_fast	0.936	0.94	0.914	0.935	0.933	0.953	1	0.914	0.95	0.952
ucrC_fast	0.927	0.903	0.978	0.894	0.903	0.898	0.914	1	0.902	0.903
ucrss_fast	0.948	0.952	0.902	0.948	0.931	0.956	0.95	0.902	1	0.962
ucrss_fast_wfilter	0.947	0.944	0.911	0.949	0.943	0.96	0.952	0.903	0.962	1

(b)

	uc	ucr	ucrC	ucrss	ucrss_wfilter	uc_fast	ucr_fast	ucrC_fast	ucrss_fast	ucrss_fast_wfilter
uc	1	0.996	0.993	0.996	0.996	0.995	0.996	0.992	0.996	0.996
ucr	0.996	1	0.993	0.997	0.997	0.995	0.996	0.992	0.996	0.997
ucrC	0.993	0.993	1	0.994	0.991	0.994	0.994	0.998	0.995	0.994
ucrss	0.996	0.997	0.994	1	0.996	0.996	0.997	0.994	0.997	0.997
ucrss_wfilter	0.996	0.997	0.991	0.996	1	0.994	0.995	0.991	0.996	0.996
uc_fast	0.995	0.995	0.994	0.996	0.994	1	0.997	0.994	0.997	0.996
ucr_fast	0.996	0.996	0.994	0.997	0.995	0.997	1	0.994	0.997	0.997
ucrC_fast	0.992	0.992	0.998	0.994	0.991	0.994	0.994	1	0.994	0.994
ucrss_fast	0.996	0.996	0.995	0.997	0.996	0.997	0.997	0.994	1	0.997
ucrss_fast_wfilter	0.996	0.997	0.994	0.997	0.996	0.996	0.997	0.994	0.997	1

(c)

	uc	ucr	ucrC	ucrss	ucrss_wfilter	uc_fast	ucr_fast	ucrC_fast	ucrss_fast	ucrss_fast_wfilter
uc	1	0.985	0.957	0.985	0.985	0.984	0.986	0.961	0.983	0.984
ucr	0.985	1	0.956	0.99	0.989	0.988	0.987	0.96	0.987	0.986
ucrC	0.957	0.956	1	0.961	0.958	0.959	0.961	0.99	0.953	0.961
ucrss	0.985	0.99	0.961	1	0.991	0.988	0.99	0.964	0.989	0.987
ucrss_wfilter	0.985	0.989	0.958	0.991	1	0.985	0.989	0.963	0.987	0.985
uc_fast	0.984	0.988	0.959	0.988	0.985	1	0.986	0.961	0.986	0.985
ucr_fast	0.986	0.987	0.961	0.99	0.989	0.986	1	0.965	0.988	0.989
ucrC_fast	0.961	0.96	0.99	0.964	0.963	0.961	0.965	1	0.957	0.965
ucrss_fast	0.983	0.987	0.953	0.989	0.987	0.986	0.988	0.957	1	0.986
ucrss_fast_wfilter	0.984	0.986	0.961	0.987	0.985	0.985	0.989	0.965	0.986	1

(d)

	uc	ucr	ucrC	ucrss	ucrss_wfilter	uc_fast	ucr_fast	ucrC_fast	ucrss_fast	ucrss_fast_wfilter
uc	1	0.948	0.88	0.909	0.924	0.935	0.934	0.877	0.925	0.913
ucr	0.948	1	0.905	0.946	0.947	0.947	0.953	0.903	0.938	0.932
ucrC	0.88	0.905	1	0.926	0.888	0.882	0.908	0.973	0.91	0.896
ucrss	0.909	0.946	0.926	1	0.932	0.923	0.935	0.915	0.931	0.929
ucrss_wfilter	0.924	0.947	0.888	0.932	1	0.943	0.946	0.884	0.932	0.927
uc_fast	0.935	0.947	0.882	0.923	0.943	1	0.942	0.883	0.941	0.94
ucr_fast	0.934	0.953	0.908	0.935	0.946	0.942	1	0.908	0.943	0.932
ucrC_fast	0.877	0.903	0.973	0.915	0.884	0.883	0.908	1	0.904	0.906
ucrss_fast	0.925	0.938	0.91	0.931	0.932	0.941	0.943	0.904	1	0.953
ucrss_fast_wfilter	0.913	0.932	0.896	0.929	0.927	0.94	0.932	0.906	0.953	1

(e)

	uc	ucr	ucrC	ucrss	ucrss_wfilter	uc_fast	ucr_fast	ucrC_fast	ucrss_fast	ucrss_fast_wfilter
uc	1	0.992	0.984	0.992	0.992	0.989	0.99	0.978	0.989	0.99
ucr	0.992	1	0.994	0.998	0.998	0.992	0.997	0.991	0.997	0.997
ucrC	0.984	0.994	1	0.995	0.995	0.984	0.993	0.997	0.994	0.994
ucrss	0.992	0.998	0.995	1	0.998	0.992	0.997	0.991	0.997	0.997
ucrss_wfilter	0.992	0.998	0.995	0.998	1	0.992	0.997	0.991	0.997	0.997
uc_fast	0.989	0.992	0.984	0.992	0.992	1	0.993	0.981	0.992	0.992
ucr_fast	0.99	0.997	0.993	0.997	0.997	0.993	1	0.992	0.998	0.998
ucrC_fast	0.978	0.991	0.997	0.991	0.991	0.981	0.992	1	0.993	0.992
ucrss_fast	0.989	0.997	0.994	0.997	0.997	0.992	0.998	0.993	1	0.998
ucrss_fast_wfilter	0.99	0.997	0.994	0.997	0.997	0.992	0.998	0.992	0.998	1

(f)

	uc	ucr	ucrC	ucrss	ucrss_wfilter	uc_fast	ucr_fast	ucrC_fast	ucrss_fast	ucrss_fast_wfilter
uc	1	0.986	0.971	0.986	0.986	0.993	0.988	0.972	0.988	0.987
ucr	0.986	1	0.984	0.995	0.995	0.987	0.993	0.98	0.993	0.993
ucrC	0.971	0.984	1	0.985	0.984	0.97	0.981	0.992	0.98	0.979
ucrss	0.986	0.995	0.985	1	0.995	0.987	0.993	0.981	0.993	0.992
ucrss_wfilter	0.986	0.995	0.984	0.995	1	0.986	0.993	0.979	0.992	0.992
uc_fast	0.993	0.987	0.97	0.987	0.986	1	0.989	0.972	0.99	0.988
ucr_fast	0.988	0.993	0.981	0.993	0.993	0.989	1	0.981	0.994	0.994
ucrC_fast	0.972	0.98	0.992	0.981	0.979	0.972	0.981	1	0.982	0.979
ucrss_fast	0.988	0.993	0.98	0.993	0.992	0.99	0.994	0.982	1	0.995
ucrss_fast_wfilter	0.987	0.993	0.979	0.992	0.992	0.988	0.994	0.979	0.995	1

### **Table 3**(on next page)

Beta diversity correlation by method and dataset.

**Mantel correlation coefficients ( $r$ ) of beta diversity for (a) 88-soils unweighted UniFrac, (b) moving-pictures unweighted UniFrac, (c) whole-body unweighted UniFrac, (d) 88-soils weighted UniFrac, (e) moving-pictures weighted UniFrac, and (f) moving-pictures weighted UniFrac.**

(a)

	uc	ucr	ucrC	ucrss	ucrss_wfilter	uc_fast	ucr_fast	ucrC_fast	ucrss_fast	ucrss_fast_wfilter
uc	NA	0.935	0.908	0.944	0.942	0.939	0.945	0.909	0.943	0.941
ucr	NA	NA	0.915	0.94	0.945	0.934	0.942	0.918	0.944	0.949
ucrC	NA	NA	NA	0.917	0.91	0.926	0.913	0.95	0.917	0.92
ucrss	NA	NA	NA	NA	0.94	0.938	0.945	0.914	0.938	0.942
ucrss_wfilter	NA	NA	NA	NA	NA	0.934	0.943	0.907	0.942	0.941
uc_fast	NA	NA	NA	NA	NA	NA	0.938	0.92	0.939	0.941
ucr_fast	NA	NA	NA	NA	NA	NA	NA	0.909	0.946	0.947
ucrC_fast	NA	NA	NA	NA	NA	NA	NA	NA	0.917	0.924
ucrss_fast	NA	NA	NA	NA	NA	NA	NA	NA	NA	0.945
ucrss_fast_wfilter	NA	NA	NA	NA	NA	NA	NA	NA	NA	NA

(b)

	uc	ucr	ucrC	ucrss	ucrss_wfilter	uc_fast	ucr_fast	ucrC_fast	ucrss_fast	ucrss_fast_wfilter
uc	NA	0.992	0.974	0.988	0.988	0.992	0.991	0.977	0.991	0.992
ucr	NA	NA	0.982	0.992	0.991	0.991	0.992	0.984	0.993	0.993
ucrC	NA	NA	NA	0.986	0.985	0.973	0.982	0.994	0.981	0.981
ucrss	NA	NA	NA	NA	0.99	0.988	0.992	0.987	0.992	0.991
ucrss_wfilter	NA	NA	NA	NA	NA	0.986	0.99	0.986	0.99	0.991
uc_fast	NA	NA	NA	NA	NA	NA	0.991	0.976	0.992	0.991
ucr_fast	NA	NA	NA	NA	NA	NA	NA	0.983	0.993	0.992
ucrC_fast	NA	NA	NA	NA	NA	NA	NA	NA	0.982	0.983
ucrss_fast	NA	NA	NA	NA	NA	NA	NA	NA	NA	0.993
ucrss_fast_wfilter	NA	NA	NA	NA	NA	NA	NA	NA	NA	NA

(c)

	uc	ucr	ucrC	ucrss	ucrss_wfilter	uc_fast	ucr_fast	ucrC_fast	ucrss_fast	ucrss_fast_wfilter
uc	NA	0.935	0.891	0.938	0.936	0.93	0.926	0.889	0.933	0.925
ucr	NA	NA	0.899	0.948	0.95	0.934	0.931	0.895	0.941	0.927
ucrC	NA	NA	NA	0.908	0.899	0.878	0.885	0.952	0.897	0.878
ucrss	NA	NA	NA	NA	0.953	0.938	0.936	0.905	0.945	0.928
ucrss_wfilter	NA	NA	NA	NA	NA	0.937	0.94	0.894	0.941	0.932
uc_fast	NA	NA	NA	NA	NA	NA	0.942	0.872	0.939	0.938
ucr_fast	NA	NA	NA	NA	NA	NA	NA	0.888	0.939	0.948
ucrC_fast	NA	NA	NA	NA	NA	NA	NA	NA	0.891	0.879
ucrss_fast	NA	NA	NA	NA	NA	NA	NA	NA	NA	0.933
ucrss_fast_wfilter	NA	NA	NA	NA	NA	NA	NA	NA	NA	NA

(d)

	uc	ucr	ucrC	ucrss	ucrss_wfilter	uc_fast	ucr_fast	ucrC_fast	ucrss_fast	ucrss_fast_wfilter
uc	NA	0.896	0.936	0.951	0.901	0.925	0.937	0.924	0.956	0.902
ucr	NA	NA	0.896	0.889	0.966	0.891	0.939	0.895	0.901	0.947
ucrC	NA	NA	NA	0.919	0.914	0.906	0.928	0.984	0.931	0.896
ucrss	NA	NA	NA	NA	0.9	0.917	0.947	0.903	0.949	0.899
ucrss_wfilter	NA	NA	NA	NA	NA	0.885	0.938	0.911	0.899	0.94
uc_fast	NA	NA	NA	NA	NA	NA	0.909	0.898	0.919	0.874
ucr_fast	NA	NA	NA	NA	NA	NA	NA	0.92	0.952	0.96
ucrC_fast	NA	NA	NA	NA	NA	NA	NA	NA	0.918	0.89
ucrss_fast	NA	NA	NA	NA	NA	NA	NA	NA	NA	0.918
ucrss_fast_wfilter	NA	NA	NA	NA	NA	NA	NA	NA	NA	NA

(e)

	uc	ucr	ucrC	ucrss	ucrss_wfilter	uc_fast	ucr_fast	ucrC_fast	ucrss_fast	ucrss_fast_wfilter
uc	NA	0.971	0.949	0.97	0.973	0.972	0.977	0.949	0.974	0.966
ucr	NA	NA	0.928	0.952	0.952	0.957	0.958	0.928	0.96	0.954
ucrC	NA	NA	NA	0.96	0.94	0.948	0.934	0.999	0.965	0.932
ucrss	NA	NA	NA	NA	0.938	0.965	0.955	0.96	0.98	0.932
ucrss_wfilter	NA	NA	NA	NA	NA	0.946	0.966	0.941	0.951	0.967
uc_fast	NA	NA	NA	NA	NA	NA	0.97	0.948	0.971	0.949
ucr_fast	NA	NA	NA	NA	NA	NA	NA	0.934	0.967	0.967
ucrC_fast	NA	NA	NA	NA	NA	NA	NA	NA	0.965	0.932
ucrss_fast	NA	NA	NA	NA	NA	NA	NA	NA	NA	0.951
ucrss_fast_wfilter	NA	NA	NA	NA	NA	NA	NA	NA	NA	NA

(f)

	uc	ucr	ucrC	ucrss	ucrss_wfilter	uc_fast	ucr_fast	ucrC_fast	ucrss_fast	ucrss_fast_wfilter
uc	NA	0.947	0.896	0.934	0.943	0.96	0.939	0.898	0.904	0.936
ucr	NA	NA	0.9	0.924	0.95	0.951	0.92	0.904	0.871	0.944
ucrC	NA	NA	NA	0.886	0.924	0.907	0.911	0.994	0.831	0.939
ucrss	NA	NA	NA	NA	0.944	0.92	0.917	0.882	0.918	0.911
ucrss_wfilter	NA	NA	NA	NA	NA	0.933	0.918	0.926	0.897	0.932
uc_fast	NA	NA	NA	NA	NA	NA	0.955	0.909	0.889	0.966
ucr_fast	NA	NA	NA	NA	NA	NA	NA	0.91	0.936	0.951
ucrC_fast	NA	NA	NA	NA	NA	NA	NA	NA	0.83	0.94
ucrss_fast	NA	NA	NA	NA	NA	NA	NA	NA	NA	0.866
ucrss_fast_wfilter	NA	NA	NA	NA	NA	NA	NA	NA	NA	NA



## **Table 4**(on next page)

Taxonomic composition correlation by method and dataset.

**Pearson correlation coefficients (r) of taxonomic summaries for (a) 88-soils at phylum level, (b) 88-soils at genus level, (c) moving-pictures at phylum level, (d) moving-pictures at genus level, (e) whole-body at phylum level, and (f) whole-body at genus level.**

(a)

	uc	ucr	ucrC	ucr <sub>ss</sub>	ucr <sub>ss</sub> _wfilter	uc_fast	ucr_fast	ucrC_fast	ucr <sub>ss</sub> _fast	ucr <sub>ss</sub> _fast_wfilter
uc	NA	1	0.983	1	1	1	1	0.981	1	1
ucr	NA	NA	0.983	1	1	1	1	0.981	1	1
ucrC	NA	NA	NA	0.983	0.983	0.983	0.983	0.999	0.983	0.983
ucr <sub>ss</sub>	NA	NA	NA	NA	1	1	1	0.981	1	1
ucr <sub>ss</sub> _wfilter	NA	NA	NA	NA	NA	1	1	0.981	1	1
uc_fast	NA	NA	NA	NA	NA	NA	1	0.981	1	1
ucr_fast	NA	NA	NA	NA	NA	NA	NA	0.981	1	1
ucrC_fast	NA	NA	NA	NA	NA	NA	NA	NA	0.981	0.981
ucr <sub>ss</sub> _fast	NA	NA	NA	NA	NA	NA	NA	NA	NA	1
ucr <sub>ss</sub> _fast_wfilter	NA	NA	NA	NA	NA	NA	NA	NA	NA	NA

(b)

	uc	ucr	ucrC	ucr <sub>ss</sub>	ucr <sub>ss</sub> _wfilter	uc_fast	ucr_fast	ucrC_fast	ucr <sub>ss</sub> _fast	ucr <sub>ss</sub> _fast_wfilter
uc	NA	0.939	0.85	0.939	0.939	1	0.94	0.84	0.94	0.94
ucr	NA	NA	0.821	1	1	0.94	0.998	0.923	0.998	0.998
ucrC	NA	NA	NA	0.821	0.821	0.85	0.82	0.818	0.82	0.82
ucr <sub>ss</sub>	NA	NA	NA	NA	1	0.94	0.998	0.923	0.998	0.998
ucr <sub>ss</sub> _wfilter	NA	NA	NA	NA	NA	0.94	0.998	0.923	0.998	0.998
uc_fast	NA	NA	NA	NA	NA	NA	0.94	0.84	0.94	0.94
ucr_fast	NA	NA	NA	NA	NA	NA	NA	0.921	1	1
ucrC_fast	NA	NA	NA	NA	NA	NA	NA	NA	0.921	0.921
ucr <sub>ss</sub> _fast	NA	NA	NA	NA	NA	NA	NA	NA	NA	1
ucr <sub>ss</sub> _fast_wfilter	NA	NA	NA	NA	NA	NA	NA	NA	NA	NA

(c)

	uc	ucr	ucrC	ucr <sub>ss</sub>	ucr <sub>ss</sub> _wfilter	uc_fast	ucr_fast	ucrC_fast	ucr <sub>ss</sub> _fast	ucr <sub>ss</sub> _fast_wfilter
uc	NA	1	0.997	1	1	1	1	0.997	1	0.998
ucr	NA	NA	0.997	1	1	1	1	0.997	1	0.998
ucrC	NA	NA	NA	0.997	0.997	0.997	0.997	1	0.997	0.998
ucr <sub>ss</sub>	NA	NA	NA	NA	1	1	1	0.997	1	0.998
ucr <sub>ss</sub> _wfilter	NA	NA	NA	NA	NA	1	1	0.997	1	0.999
uc_fast	NA	NA	NA	NA	NA	NA	1	0.997	1	0.998
ucr_fast	NA	NA	NA	NA	NA	NA	NA	0.997	1	0.998
ucrC_fast	NA	NA	NA	NA	NA	NA	NA	NA	0.997	0.997
ucr <sub>ss</sub> _fast	NA	NA	NA	NA	NA	NA	NA	NA	NA	0.998
ucr <sub>ss</sub> _fast_wfilter	NA	NA	NA	NA	NA	NA	NA	NA	NA	NA

(d)

	uc	ucr	ucrC	ucr <sub>ss</sub>	ucr <sub>ss</sub> _wfilter	uc_fast	ucr_fast	ucrC_fast	ucr <sub>ss</sub> _fast	ucr <sub>ss</sub> _fast_wfilter
uc	NA	0.964	0.929	0.964	0.963	0.999	0.923	0.882	0.923	0.92
ucr	NA	NA	0.963	1	0.999	0.967	0.954	0.923	0.954	0.951
ucrC	NA	NA	NA	0.963	0.963	0.934	0.925	0.917	0.925	0.925
ucr <sub>ss</sub>	NA	NA	NA	NA	0.999	0.967	0.954	0.923	0.954	0.951
ucr <sub>ss</sub> _wfilter	NA	NA	NA	NA	NA	0.966	0.953	0.923	0.953	0.952
uc_fast	NA	NA	NA	NA	NA	NA	0.927	0.887	0.927	0.924
ucr_fast	NA	NA	NA	NA	NA	NA	NA	0.885	1	0.997
ucrC_fast	NA	NA	NA	NA	NA	NA	NA	NA	0.885	0.884
ucr <sub>ss</sub> _fast	NA	NA	NA	NA	NA	NA	NA	NA	NA	0.997
ucr <sub>ss</sub> _fast_wfilter	NA	NA	NA	NA	NA	NA	NA	NA	NA	NA

(e)

	uc	ucr	ucrC	ucr <sub>ss</sub>	ucr <sub>ss</sub> _wfilter	uc_fast	ucr_fast	ucrC_fast	ucr <sub>ss</sub> _fast	ucr <sub>ss</sub> _fast_wfilter
uc	NA	1	0.999	1	1	1	1	0.998	1	1
ucr	NA	NA	0.999	1	1	1	1	0.998	1	1
ucrC	NA	NA	NA	0.999	0.999	0.999	0.999	0.999	0.999	0.999
ucr <sub>ss</sub>	NA	NA	NA	NA	1	1	1	0.998	1	1
ucr <sub>ss</sub> _wfilter	NA	NA	NA	NA	NA	1	1	0.998	1	1
uc_fast	NA	NA	NA	NA	NA	NA	1	0.998	1	1
ucr_fast	NA	NA	NA	NA	NA	NA	NA	0.998	1	1
ucrC_fast	NA	NA	NA	NA	NA	NA	NA	NA	0.998	0.998
ucr <sub>ss</sub> _fast	NA	NA	NA	NA	NA	NA	NA	NA	NA	1
ucr <sub>ss</sub> _fast_wfilter	NA	NA	NA	NA	NA	NA	NA	NA	NA	NA

(f)

	uc	ucr	ucrC	ucr <sub>ss</sub>	ucr <sub>ss</sub> _wfilter	uc_fast	ucr_fast	ucrC_fast	ucr <sub>ss</sub> _fast	ucr <sub>ss</sub> _fast_wfilter
uc	NA	0.959	0.9	0.959	0.959	1	0.913	0.879	0.913	0.913
ucr	NA	NA	0.918	1	1	0.957	0.967	0.871	0.967	0.967
ucrC	NA	NA	NA	0.918	0.918	0.896	0.893	0.935	0.892	0.893
ucr <sub>ss</sub>	NA	NA	NA	NA	1	0.957	0.967	0.871	0.967	0.967
ucr <sub>ss</sub> _wfilter	NA	NA	NA	NA	NA	0.957	0.967	0.871	0.967	0.967
uc_fast	NA	NA	NA	NA	NA	NA	0.912	0.876	0.912	0.912
ucr_fast	NA	NA	NA	NA	NA	NA	NA	0.855	1	1
ucrC_fast	NA	NA	NA	NA	NA	NA	NA	NA	0.854	0.855
ucr <sub>ss</sub> _fast	NA	NA	NA	NA	NA	NA	NA	NA	NA	1
ucr <sub>ss</sub> _fast_wfilter	NA	NA	NA	NA	NA	NA	NA	NA	NA	NA

## **Table 5**(on next page)

Runtime comparisons by method and dataset.

**Comparison of runtimes (as seconds of wall time) for each method on each data set.**

	<b>88-soil</b>	<b>moving-picture</b>	<b>whole-body</b>
<b>uc</b>	1220	27748	1095
<b>ucr</b>	1358	46576	1082
<b>ucrC</b>	226	28572	388
<b>ucrss</b>	1493	47207	1212
<b>ucrss_wfilter</b>	1885	76061	2088
<b>uc_fast</b>	914	23510	489
<b>ucr_fast</b>	1052	19371	621
<b>ucrC_fast</b>	44	2428	68
<b>ucrss_fast</b>	1021	23710	707
<b>ucrss_fast_wfilter</b>	1525	52811	1661

**Table 6**(on next page)

Runtime comparisons (parameter variations).

**Comparison of runtimes (as seconds of wall time) for subsampled and “legacy” open-reference OTU picking methods with variations on the default parameters.**

	<b>moving-picture</b>
<b>abbreviation</b>	
ucr_fast_O29_r82	21737
ucr_fast_O29_r97	16241
ucrss_fast_O29_r82	17812
ucrss_fast_O29_r97	16169
ucrss_fast_O29_s1	14911

**Table 7** (on next page)

Novel OTUs by biome.

**Comparison of OTUs with closed-reference and open-reference OTU picking by biome in the Earth Microbiome Project dataset.**



<b>EnvironmentalBiome</b>	<b>Average de novo OTUs (10K sequences per sample)</b>	<b>SD de novo OTUs (10K sequences per sample)</b>	<b>Average Reference OTUs (10k sequences per sample)</b>	<b>SD Reference OTUs (10k sequences per sample)</b>	<b>% novel diversity (10k seqs per sample)</b>	<b>% error novel diversity (10K seqs per sample)</b>	<b>number of samples</b>
<b>mangrove biome</b>	2169	1159	354	73	0.86	0.46	7
<b>tropical humid forests</b>	2398	260	397	35	0.858	0.094	26
<b>tundra biome</b>	1771	403	312	117	0.85	0.201	110
<b>deserts and xeric shrubland biome</b>	3917	127	707	15	0.847	0.028	7
<b>taiga</b>	2598	102	505	35	0.837	0.035	4
<b>marine biome</b>	2040	1048	484	410	0.808	0.446	890
<b>aquatic biome</b>	714	299	177	199	0.801	0.403	762
<b>freshwater biome</b>	768	541	194	120	0.798	0.576	375
<b>warm deserts and semideserts</b>	2386	473	607	147	0.797	0.166	97
<b>tropical and subtropical moist broadleaf forest biome</b>	3072	125	846	18	0.784	0.032	2
<b>temperate needle-leaf forests or woodlands</b>	2836	159	785	132	0.783	0.057	21
<b>polar biome</b>	1721	886	483	218	0.781	0.414	277
<b>tropical and subtropical coniferous forest biome</b>	1993	256	579	94	0.775	0.106	3
<b>mixed island systems</b>	1552	618	511	203	0.752	0.315	124
<b>marginal sea</b>	1795	325	611	225	0.746	0.164	7
<b>temperate coniferous forest biome</b>	2504	1206	885	201	0.739	0.361	19
<b>mediterranean forests, woodlands, and shrub biome</b>	695	361	275	195	0.717	0.424	371
<b>large river biome</b>	1844	629	743	369	0.713	0.282	5
<b>terrestrial biome</b>	2714	222	1138	163	0.705	0.072	627
<b>nest of bird</b>	821	276	355	138	0.698	0.262	313
<b>Temperate broadleaf and mixed forest biome</b>	1910	491	879	235	0.685	0.195	14
<b>temperate grasslands</b>	2745	290	1315	164	0.676	0.082	696
<b>animal-associated habitat</b>	758	329	376	240	0.668	0.359	1036
<b>mammalia-associated habitat</b>	973	357	583	222	0.625	0.27	1918
<b>Cold-winter (continental) deserts and semideserts</b>	847	210	551	215	0.606	0.215	102
<b>Temperate grasslands, savannas, and shrubland biome</b>	1688	272	1497	275	0.53	0.121	85
<b>human-associated habitat</b>	292	242	590	366	0.331	0.498	1597

**Table 8**(on next page)

Differentially represented OTUs by dataset.

**Top 10 OTUs identified as significantly different across (a) binned pH in 88-soils, (b) body site in moving-pictures, and (c) body site in whole-body.**

(a)	taxonomy	Test-Statistic
<b>OTU</b>		
<b>113212</b>	k_Bacteria;p_Acidobacteria;c_DA052;o_Ellin6513;f__;g__;s__	55.859
<b>1123837</b>	k_Bacteria;p_Actinobacteria;c_Rubrobacteria;o_Rubrobacterales;f_Rubrobacteraceae;g_Rubrobacter;s__	50.433
<b>New.ReferenceOTU22</b>	k_Bacteria;p_Actinobacteria;c_Actinobacteria;o_Actinomycetales;f__;g__;s__	49.172
<b>252012</b>	k_Bacteria;p_Proteobacteria;c_Gammaproteobacteria;o_Xanthomonadales;f_Sinobacteraceae;g__;s__	48.65
<b>843189</b>	k_Bacteria;p_Acidobacteria;c_Solibacteres;o_Solibacterales;f_Solibacteraceae;g_Candidatus Solibacter;s__	47.006
<b>1127423</b>	k_Bacteria;p_Acidobacteria;c_Acidobacteriia;o_Acidobacteriales;f_Koribacteraceae;g__;s__	43.87
<b>1129210</b>	k_Bacteria;p_Acidobacteria;c_Acidobacteriia;o_Acidobacteriales;f_Koribacteraceae;g__;s__	43.804
<b>831520</b>	k_Bacteria;p_Actinobacteria;c_Rubrobacteria;o_Rubrobacterales;f_Rubrobacteraceae;g_Rubrobacter;s__	43.625
<b>1139779</b>	k_Bacteria;p_Proteobacteria;c_Alphaproteobacteria	41.863
<b>804187</b>	k_Bacteria;p_Acidobacteria;c_[Chloracidobacteria];o_RB41;f__;g__;s__	41.151

(b)	taxonomy	Test-Statistic
<b>OTU</b>		
<b>368134</b>	k_Bacteria;p_Firmicutes;c_Bacilli;o_Bacillales;f_Staphylococcaceae;g_Staphylococcus;s_epidermidis	1599.696
<b>3154070</b>	k_Bacteria;p_Bacteroidetes;c_Bacteroidia;o_Bacteroidales;f_Bacteroidaceae;g_Bacteroides;s_uniformis	1625.703
<b>1000986</b>	k_Bacteria;p_Actinobacteria;c_Actinobacteria;o_Actinomycetales;f_Corynebacteriaceae;g_Corynebacterium;s__	1630.009
<b>1992</b>	k_Bacteria;p_Bacteroidetes;c_Bacteroidia;o_Bacteroidales;f_Bacteroidaceae;g_Bacteroides;s__	1728.164
<b>4304475</b>	k_Bacteria;p_Bacteroidetes;c_Bacteroidia;o_Bacteroidales;f_Bacteroidaceae;g_Bacteroides;s__	1545.445
<b>191238</b>	k_Bacteria;p_Firmicutes;c_Clostridia;o_Clostridiales;f_Lachnospiraceae;g_Coproccoccus;s__	1546.436
<b>187665</b>	k_Bacteria;p_Firmicutes;c_Clostridia;o_Clostridiales;f_Lachnospiraceae;g__;s__	1474.529
<b>4396297</b>	k_Bacteria;p_Firmicutes;c_Clostridia;o_Clostridiales;f_Lachnospiraceae;g__;s__	1585.015
<b>3903651</b>	k_Bacteria;p_Firmicutes;c_Clostridia;o_Clostridiales;f_Ruminococcaceae;g_Oscillospira;s__	1670.188
<b>3472078</b>	k_Bacteria;p_Bacteroidetes;c_Bacteroidia;o_Bacteroidales;f_Bacteroidaceae;g_Bacteroides;s__	1783.488

(c)	taxonomy	Test-Statistic
<b>OTU</b>		
<b>4326219</b>	k_Bacteria;p_Proteobacteria;c_Epsilonproteobacteria;o_Campylobacterales;f_Campylobacteraceae;g_Campylobac	363.881
<b>v.CleanUp.ReferenceOTU</b>	k_Bacteria;p_Bacteroidetes;c_Bacteroidia;o_Bacteroidales;f_Prevotellaceae;g_Prevotella;s_melaninogenica	358.02
<b>4325533</b>	k_Bacteria;p_Bacteroidetes;c_Bacteroidia;o_Bacteroidales;f_Rikenellaceae;g__;s__	349.852
<b>CleanUp.ReferenceOTU1</b>	k_Bacteria;p_Firmicutes;c_Clostridia;o_Clostridiales;f_Veillonellaceae;g_Veillonella;s_parvula	337.656
<b>316732</b>	k_Bacteria;p_Firmicutes;c_Clostridia;o_Clostridiales;f_Lachnospiraceae;g_Lachnospira;s__	337.309
<b>4346374</b>	k_Bacteria;p_Bacteroidetes;c_Bacteroidia;o_Bacteroidales;f_Bacteroidaceae;g_Bacteroides;s_uniformis	331.433
<b>4458959</b>	k_Bacteria;p_Firmicutes;c_Clostridia;o_Clostridiales;f_Veillonellaceae;g_Veillonella	329.772
<b>3866487</b>	k_Bacteria;p_Firmicutes;c_Clostridia;o_Clostridiales;f_Lachnospiraceae;g_Oribacterium;s__	323.488
<b>4391641</b>	k_Bacteria;p_Proteobacteria;c_Gammaproteobacteria;o_Pasteurellales;f_Pasteurellaceae;g_Haemophilus;s_para	312
<b>175751</b>	k_Bacteria;p_Firmicutes;c_Clostridia;o_Clostridiales;f_Lachnospiraceae;g__;s__	305.531