Evaluating Metagenome Assembly on a Simple Defined Community with Many Strain Variants

Sherine Awad[1], Luiz Irber[1], C. Titus Brown[1*]

1 Department of Population Health and Reproduction

University of California, Davis

Davis, CA 95616 USA

* E-mail: ctbrown@ucdavis.edu

June 15, 2017

Abstract

Foo!

Introduction

Metagenomics refers to sequencing of DNA from a mixture of organisms, often from an environmental or uncultured sample. Unlike whole genome sequencing, metagenomics targets a mixture of genomes, which introduces metagenome-specific challenges in analysis. Most approaches to analyzing metagenomic data rely on mapping or comparing sequencing reads to reference sequence collections. However, reference databases contain only a small subset of microbial diversity (cite: geba), and the much of the remaining diversity is evolutionarily distant and search techniques may not access it.

As sequencing capacity increases and sequence data is generated from many more environmental samples, metagenomics is increasingly using de novo assembly techniques to generate new reference genomes and metagenomes. There are a number of metagenome assemblers that are widely used. However, evaluating the results of these assemblers is challenging due to the general lack of good quality reference metagenomes. Below, we evaluate three commonly assemblers - SPAdes, IDBA, and MEGAHIT - on a mock community containing 64 species of microbes with known genomes.

Moya et al. in [1] evaluated metagenome assembly using two simulated 454 viral metagenome and six assemblers. The assemblies were evaluated based on several metrics including N50, percentages of reads assembled, accuracy when compared to the reference genome. In addition to, chimeras per contigs and the effect of assembly on taxonomic and functional annotations.

Mavromatis et al. in [2] provided a benchmark study to evaluate the fidelity of metagenome process methods. The study used simulated metagenomic data sets constructed at different complexity levels. The datasets were assembled using Phrap v3.57, Arachne v.2 [3] and JAZZ. [4] This study evaluates assembly, gene prediction, and binning methods. However, the study did not evaluate the assembly quality against a reference genome.

Rangwala et al. in [5] presented an evaluation study of metagenome assembly. The study used a de Bruijn graph based assembler ABYSS [6] to assemble simulated metagnome reads of 36 bp. The data set is classified at different complexity levels. The study compares the quality of the assembly of the data sets in terms of quality measures of contigs length, assembly accuracy. The study also took into consideration the effect of kmer size and the degree of chimericity. However, the study evaluated the assembly based on one assembler, and did not evaluate assembly against several assemblers. Also, both previous studies used simulated data, which may lack confounders of assembly such as sequencing artifacts and GC bias.

Shakya et al. (2013) constructed a synthetic community of organisms by mixing DNA isolated from individual cultures of 64 bacteria and archaea, including a variety of strains across a range of nucleotide distances. In addition to performing 16s amplicon analysis and doing 454 sequencing, the authors shotgun sequenced the mixture with Illumina (@cite). While the authors concluded that this metagenomic sequencing generally outperformed amplicon sequencing, they did not conduct an assembly based analysis. a mapping based analysis rather than an assembly based analysis.

More recently, several benchmark studies systematically evaluated metagenome assembly of short reads. The Critical Assessment of Metagenome Interpretation (CAMI) collaboration benchmarked a number of metagenome assemblers on several data sets of varying complexity, evaluating recovery of novel genomes and multiple strain variants (@cite). Notably, CAMI concluded that "The resolution of strain-level diversity represents a substantial challenge to all evaluated programs." Another recent study evaluated eight assemblers on nine environmental metagenomes and three simulated data sets (@cite).

In this paper, we evaluate metagenome assembly on the Illumina data set from Shakya et al. (2013) using three assemblers; IDBA-UD [7], SPAdes [8], and MEGAHIT [9]. These three assemblers were chosen because they are actively used and highly cited, and typically perform well.

Below, we evaluate the performance of these three assemblers using the mock community data from the Shakya et al. study. The performance of each assembler is compared in terms of resource utilization, covered genome fraction, duplication ratio, gene recovery, contig misassembly, and contig length.

In this report, we extend the CAMI study by delving into questions of chimeric misassembly and strain recovery. First, we update the list of reference genomes for Shakya et al. to include updated assemblies as well as plasmids. We then compare IDBA, SPAdes, and MEGAHIT performance on assembling this short-read data set, and explore concordance in recovery between the three assemblers. We also evaluate inter-strain chimerism in the assemblies and explore the poor assemblies caused by "strain confusion" between two *Shewanella baltica* strain. We detect and analyze several previously unreported strains and genomes in the Shakya et al. data set. Our report provides strong guidance on choice of assemblers and significantly extends previous analyses of this low-complexity metagenome benchmarking data set.

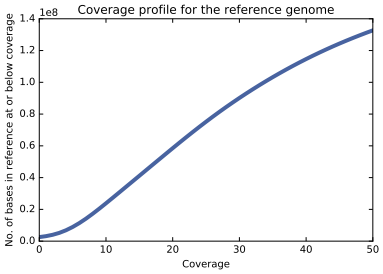

Figure 1: Cumulative coverage profile for the refernce metagenome, based on read mapping.

Datasets

We used a diverse mock community data set constructed by pooling DNA from 64 species of bacteria and archaea and sequencing them with Illumina HiSeq. The raw data set consisted of 109,629,496 reads from Illumina HiSeq 101 bp paired-end sequencing (2x101) with an untrimmed total length of 11.07 Gbp and an estimated fragment size of 380 bp [10].

The original reads are available through the NCBI Sequence Read Archive at Accession SRX200676. We received the 64 reference genomes from the original authors. They consist of 205.6 Mbp of assembled genomes in 64 contigs, and are available for download at https://dx.doi.org/10.6084/m9.figshare.1506873.v2.

We updated the data sets from NCBI etc. etc. The following genomes were updated. Updated data is available for download here (OSF).

Methods

The analysis code and run scripts for this paper are available at: https://github.com/dib-lab/2015-metagenome-assembly/. The scripts and overall pipeline were examined by the first and senior authors for correctness. In addition, the bespoke reference-based analysis scripts were tested by running them on a single-colony *E. coli* MG1655 data set with a high quality reference genome [11].

Quality Filtering

We removed adapters with Trimmomatic v0.30 in paired-end mode with the Truseq adapters [12], using light quality score trimming as recommended in

MacManes, 2014 [13].

Reference Coverage Profile

To evaluate how much of the reference metagenome was contained in the read data, we used `bwa aln` to map reads to the reference genome. We then calculated how many reference bases were covered by mapped reads (custom script `coverage-profile.py`).

Assemblers

We assembled the quality-filtered reads using three different assemblers: IDBA-UD [7], MetaSPAdes [8], and MEGAHIT [9]. For IDBA-UD v1.1.1 [7], we used `--pre_correction` to perform pre-correction before assembly and -r for the pe files.

For MetaSPAdes v3.9.0 [8], we used `--meta --pe1-12 --pe1-s` where `--meta` is used for metagenomic data sets, `--pe1-12` specifies the interlaced reads for the first paired-end library, and `--pe1-s` provides the orphan reads remaining from quality trimming.

For MEGAHIT v1.1.1-2-g02102e1 [9], we used -l 101 `-m 3e9 --cpu-only` where `-l` is for maximum read length, `-m` is for max memory in bytes to be used in constructing the graph, and `--cpu-only` to use only the CPU and no GPUs. We also used `--presets meta-large` for large and complex metagenomes, and `--12` and `-r` to specify the interleaved-paired-end and single-end files respectively. MEGAHIT allows the specification of a memory limit and we used `-M 1e+10` for 10 GB.

All three assemblies were executed on the same high-memory buy-in node on the Michigan State University High Performance Compute Cluster, and we recorded RAM and CPU time of each assembly job using the `qstat` utility at the end of each run.

Unless otherwise mentioned, we eliminated all contigs less than 500 bp from each assembly prior to further analysis.

Mapping

We aligned all quality-filtered reads to the reference metagenome with bwa aln (v0.7.7.r441) [14]. We aligned paired-end and orphaned reads separately. We then used samtools (v0.1.19) [15] to convert SAM files to BAM files for

both paired-end and orphaned reads. To count the unaligned reads, we included only those records with the "4" flag in the SAM files [15].

To extract the reads that contribute to unaligned contigs, we mapped the quality filtered reads to the unaligned contigs using bwa aln (v0.7.7.r441) [14]. Then we used samtools to retrieve the reads that mapped to the unaligned contigs.

k-mer Presence

In order to examine k-mer presence for a k-mer size of 20, we built a k-mer counting table from the given quality filtered reads using `load-into-counting.py` from khmer [**?**]. Then we calculate abundance distribution of the k-mers in the quality filtered reads using the pre-made k-mer counting table using `abundance-dist.py`. We followed the same approach to examine k-mer presence in assemblies.

Assembly analysis using Nucmer

We used the NUCmer tool from MUMmer3.23 [16] to align assemblies to the reference genome with options `-coords -p`. Then we parsed the generated ".coords" file using a custom script `analyze_assembly.py`, and calculated several analysis metrics across all three assemblies at two alignment identities, 95% and 99%.

Reference-based analysis of the assemblies

We analyzed metrics for three different sets of contigs, based on the NUCmer alignments. We used the unfiltered NUCmer alignments for the analyses termed "ambiguous." We also subjected the alignments to two different filtering criteria, "best-hit" and "no-misassemblies." In the best-hit approach, among all alignments of a contig, we took into consideration the longest alignment with an identity above a specified identity threshold (either 95% or 99%). In the no-misassemblies approach, we only counted contigs that have precisely one alignment within the reference.

In all approaches, we flag a base in the reference genome as "covered" if it is contained in a kept alignment. We define the duplication ratio as the percentages of bases in the reference covered by two or more kept alignments. We define misassemblies as those contigs that are divided into different parts when mapped to the reference. The number of misassembled contigs is equal

to the number of aligned contigs (both totally and partially) in the ambiguous approach, minus the number of aligned contigs in the no-misassemblies approach.

All approaches have a non-zero duplication ratio within the reference because we do not explicitly discard contigs that map to the same location in the reference.

Analysis of chimeric misassemblies

We analyzed each assembly for chimeric misassemblies by counting the number of contigs that contained matches to two distinct reference genomes. In order to remove secondary alignments from consideration, we included only the longest non-overlapping NUCmer alignments for each contig at a minimum alignment identity of 99%. We then used the script `analyze_chimeric2.py` to find individual contigs that matched more than one distinct reference genome. As a negative control on our analysis, we verified that this approach yielded no positive results when applied to the alignments of the reference metagenome against itself.

Results

The raw data is high quality.

The reads contains 11,072,579,096 bp (11.07 Gbp) in 109,629,496 reads with 101.0 average length (2x101bp Illumina HiSeq).

Trimming removed 686,735 reads (0.63%). After trimming, we retained 108,422,358 paired reads containing 10.94 Gbp with an average length of 100.9 bases. A total of 46.56 Mbp remained in 520,403 orphan reads with an average length of 89.5 bases. In total, the quality trimmed data contained 10.98 Gbp in 108,942,761 reads. This quality trimmed ("QC") data set was used as the basis for all further analyses.

The reference metagenome is not completely present in the reads.

We next evaluated the fraction of the reference genome covered by at least one read (see Methods for details). Quality filtered reads cover 203,058,414 (98.76%) bases of the reference metagenome (205,603,715 bp total size). Figure 1 shows the cumulative coverage profile of the reference metagenome,

Table 1: Jaccard containment of the reference in the reads

k-mer size	% reference in reads
21	96.8%
31	95.9%
41	94.9%
51	94.1%

and the percentage of bases with that coverage. Most of the reference metagenome was covered at least minimally; only 3.33% of the reference metagenome had mapping coverage <5, and 1.24% of the bases in the reference were not covered by any reads in the QC data set.

In order to evaluate reconstructability with De Bruijn graph assemblers, we next examined k-mer containment of the reference in the reads for k of 21, 31, 41, and 51 (Table 1). The k-mer overlap decreases from 96.8% to 94.1% as the k-mer size increases. This could be caused by low coverage of some portions of the reference and/or variation between the reads and the reference.

Some individual reference genomes are poorly represented in the reads.

Table 2: Top uncovered genomes

Genome	**Read coverage**	**21-mer presence**
B. bronchiseptica	98.2%	97.3%
D. vulgaris DP4	93.2%	82.5%
T. thermophilus HB27	91.1%	79.7%
E. faecalis V583	74.6%	65.6%
F. nucleatum	47.6%	18.2%

To see if specific reference genomes exhibited low coverage, we analyzed read mapping coverage and 21-mer containment for individual genomes. Of the 64 reference genomes used in the metagenome, 59 had a per-base mapping coverage above 95% and a 21-mer containment in the QC reads above 95%. The remaining five varied significantly in both metrics (Table 3), with *F. nucleatum* the lowest – only 47.6% of the bases in the reference genome are covered by one or more mapped reads, and only 18.2% of the 21-mers in the *F. nucleatum* reference genome are present in the reads at

any abundance.

We next did a 51-mer containment analysis of each reference genome in the reads. 99% or more of the constituent 51-mers for 51 of the 64 reference genomes were present in the reads, suggesting that each of the 51 genomes was entirely present at some minimal coverage.

We excluded the remaining 13 genomes from any comparative analysis of assembly quality, because interpreting coverage and misassembly analysis for these genomes would be impossible. (@CTB list or table?)

MEGAHIT is the fastest and lowest-memory assembler evaluated

Table 3: Running Time and Memory Utilization

Assembler	**CPU time**	**Wall time**	**RAM**
MEGAHIT	52hr 25m	4 hr 9m	11.4 GB
IDBA-UD	17h		149.1 GB
SPAdes	94hr 43m	94hr 44m	100.7 GB

We ran three commonly used metagenome assemblers on the QC data set: IDBA-UD, SPAdes, and MEGAHIT. We recorded the time and memory usage of each (Table 3). In computational requirements, MEGAHIT outperformed both SPAdes and IDBA-UD considerably, producing an assembly in four hours – approximately 4 times faster than IDBA and 8 times faster than SPAdes. MEGAHIT used only 11.4 GB of RAM – 1/13th to 1/9th the memory used by IDBA and SPAdes, respectively.

The assemblies contain most of the raw data

Table 4: Read and high-abundance (> 5) k-mer exclusion from assemblies

Assembly	**Unmapped Reads**	**51-mers omitted**
IDBA	3,328,674 (3.05%)	2.4%
SPAdes	3,879,573 (3.56%)	3.2%
MEGAHIT	5,848,494 (5.37%)	2.8%

We assessed read inclusion in assemblies by mapping the QC reads to the length-filtered assemblies and counting the remaining unmapped reads.

Depending on the assembly, between 3.3 million and 5.9 million reads (3.0-5.4%) did not map to the assemblies (Table 4). Here, the MEGAHIT assembly was distinguished by representing 2 million fewer reads than the IDBA and SPAdes assemblies. K-mer inclusion, however, was more closely matched across the assemblies, with all three assemblies containing the large majority of high-abundance 51-mers.

Much of the reference is covered by the assemblies.

Table 5: Contig coverage of reference with "loose" alignment conditions.

Assembly	**bases aligned**	**duplication**	**51-mers**
MEGAHIT	96.2%	0.72%	96.7%
SPAdes	95.8%	0.99%	96.2%
IDBA	95.6%	0.88%	97.2%

We next evaluated the extent to which the assembled contigs recovered the "known/true" metagenome sequence by aligning each assembly to the adjusted reference (Table 5). Each of the three assemblers generates contigs that cover more than 95.6% of the reference metagenome at high identity (99%) with little duplication (0.72-0.99%). All three assemblies contain between 96.2% and 97.2% of the 51-mers in the reference.

At 99% identity with the loose mapping approach, approximately 1.8% of the reference is missed by all three assemblers, while 0.9% is uniquely covered by MEGAHIT, 0.6% is uniquely covered by SPAdes, and 0.4% is uniquely covered by IDBA.

The generated contigs are broadly accurate.

Table 6: Contig accuracy measured by reference coverage with strict alignment.

Assembly	**% covered**
MEGAHIT	93.8%
IDBA	89.5%
SPAdes	87.3%

When counting only the best (longest) alignment per contig at a 99% identity threshold, each of the three assemblies recovers more than 87.3% of

the reference, with MEGAHIT recovering the most – 93.8% of the reference (Table 6).

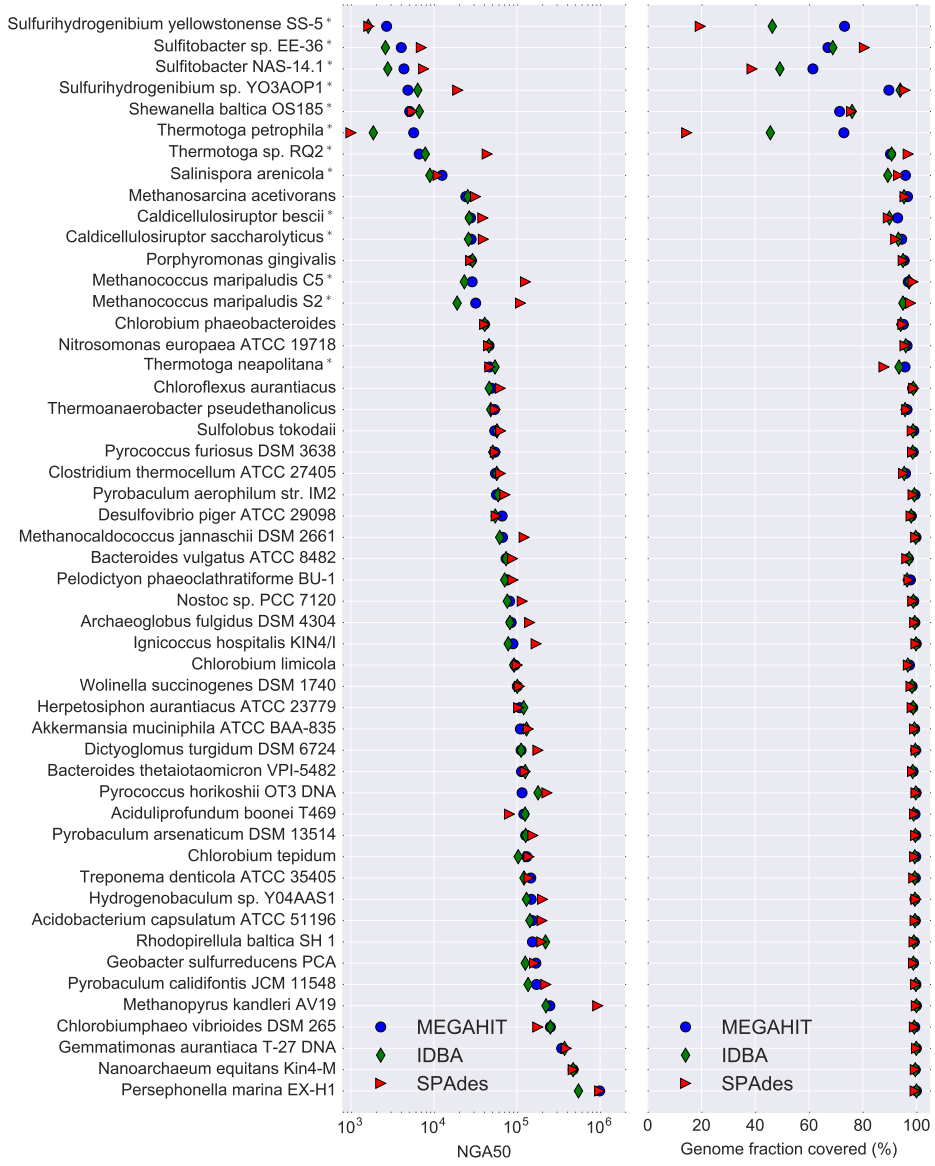

Figure 2: NGA50 by genome and assembler. A '*' after the name indicates the presence of at least one other strain variant in the community.

Individual genome statistics vary widely in the assemblies.

We computed the NGA50 for each individual genome and assembly in order to compare assembler performance on each genome (see left panel of Figure 2). The NGA50 statistics for individual genomes vary widely, but there are consistent assembler-specific trends: IDBA yields the lowest NGA50 for 30 of the 51 genomes, while SPAdes yields the highest NGA50 for 33 of the 51 genomes. Genomes with close strain variants in the defined community (indicated by a '*' suffix in the genome name) tended strongly towards lower NGA50s.

We also evaluated aligned coverage per genome for each of the three assemblies (right panel, Figure 2. We found that a 13 of the 51 genomes were missing 5% or more of bases in at least one assembly, despite all 51 genomes having 99% or higher read- and 51-mer coverage. 12 of the 13 genomes missing 5% or more of their content also had at least one close strain variant in the defined community.

Longer contigs are less likely to be chimeric.

Table 7: Chimeric contigs by contig length.

Assembly	**> 50kb**	**> 5kb**	**> 500 bp**
IDBA	0	1	7
MEGAHIT	1	4	14
SPAdes	0	3	30

Chimerism is the formation of contigs that include sequence from multiple genomes. We evaluated the rate of chimerism in contigs at three different contig length cutoffs: 500bp, 5kb, and 50kb (Table 7). We found that the percentage of contigs that match to the genomes of two or more different species drop as the minimum contig size increases, to the point where only the MEGAHIT assembly had a single chimeric contig longer than 50kb. Overall, chimeric misassemblies were rare, with no assembler generating more than 30 chimeric contigs out of thousands of total contigs.

The unmapped reads contain strain variants of reference genomes.

Approximately 4.8 million reads (4.4%) from the QC data set did not map anywhere in the reference provided by the authors of @cite. We extracted and assembled these reads in isolation using MEGAHIT, yielding 6.5 Mbp of

Table 8: Genbank genomes detected in assembly of unmapped reads

match	**Genbank genome**
44.1%	*Fusobacterium sp. OBRC1*
23.0%	*P. ruminis strain ML2*
18.2%	*Thermus thermophilus HB8*
7.7%	*P. ruminis strain CGMCC*
8.2%	*Enterococcus faecalis M7*
7.3%	*F. nucleatum 13_3C*
3.7%	*F. nucleatum subsp. polymorphum*
2.9%	*Fusobacterium hwasookii*
1.0%	*E. coli isolate YS*
1.7%	*F. nucleatum subsp. polymorphum*
1.9%	*F. nucleatum subsp. vincentii*

assembly in 1711 contigs > 500bp in length. We then did a k-mer analysis of this assembly against all of the Genbank genomes at k=31 using sourmash @cite, and estimated the fraction of the k-mers that belonged to different species (Table 8). We find that 51.1% of the k-mer content of these contigs positively match to a genome present in Genbank but not in the reference metagenome.

To verify these assignments, we aligned the MEGAHIT assembly of unmapped reads to the Genbank genomes in Table 8 with nucmer using "loose" alignment criteria. We found that 1.78 Mbp of the contigs aligned at 99% identity or better to the Genbank genomes. We als confirmed that, as expected, there are no matches to the reference metagenome.

Interestingly, all but the two *P. ruminis* matches and the *E. coli* isolate YS are strain variants of species that are part of the defined community but are not completely present in the reads (see Table 2). The presence of so many 31-mers from the *Proteiniclasticum ruminis* species is intriguing, since there is no closely related species in the mock community design. Very little of the MEGAHIT assembly aligns to known *P. ruminis* genomes via nucmer at 99%, suggesting that this is an unknown species – there are many alignments to *P. ruminis* at 94% or higher, for approximately 2.73 Mbp total.

Discussion

Assembly recovers basic content sensitively and accurately.

All three assemblers performed well in assembling contigs from the content that was fully present in reads and k-mers. After length filtering, all three assemblies contained more than 95% of the reference (Table 5); even with removal of secondary alignments, more than 87% was recovered by each assembler (Table 6). About half the constituent genomes had an NGA50 of 50kb or higher (Figure 2), which, while low for current Illumina single-genome sequencing @cite, is sufficient to recover operon-level relationships for many genes.

The presence of multiple strain variants confounds assembly.

As reported elsewhere, we also find that strain variation causes many assembly problems. This is clearly shown by Figure 2, where 12 of the bottom 14 genomes by NGA50 (left panel) also exhibit poor genome recovery by assembly (right panel). Interestingly, different assemblers handle this quite differently, with e.g. SPAdes failing to recover essentially any of *Thermotoga petrophila*, while MEGAHIT recovers 73%. The presence of strain variation is an almost perfect predictor that one or more assembler will fail to recover 5% or more - of the 13/51 genomes for which less than 95% is recovered, 12 of them have close strain variants in the community.

The *Shewanella baltica* OS185 genome is a case in point: there are two strain variants, OS185 and OS223, present in the defined community. Both are present at more than 99% in the reads, and more than 98% in 51-mers, but only 75% of *S. baltica* OS185 and 50% of *S. baltica* OS223 are recovered by assemblers. This is a clear case of "strain confusion" where the assemblers simply fail to output contigs for a substantial portion of the two genomes.

Another interest of this study was to examine cross-species chimeric assembly, in which a single contig is formed from multiple genomes. In Table 7, we show that there is relatively little cross-species chimerism and that it always occurs between members of the same genus.

MEGAHIT performs best by several metrics.

MEGAHIT is clearly the most efficient computationally, outperforming both SPAdes and IDBA by 5-10x in memory and 17-42x in time (Table 3). While the MEGAHIT assembly had 2m fewer reads mapping to it than the other

assemblies (Table 1), MEGAHIT covered more of the reference genome with both loose and strict alignments (Table 5 and Table 6), with little duplication. This is clearly because of MEGAHIT's superior performance in recovering the genomes of closely related strains (Figure 2, right panel).

Between the assemblers, the assembly content differs by only a small amount when loose alignments are allowed: all three assemblers miss more content (approximately 1.8% of the reference) than they generate uniquely (0.9% or less). In addition to preferring no one assembler over any other, this suggests that combining assemblies may have little value in terms of recovering additional metagenome content.

The missing reference may be present in strain variants of the intended species.

Several individual genomes are missing in measurable portion from the QC reads (Table 2), and many QC reads (4.4% of 108m) did not map to the full reference metagenome. These appear to be related issues: upon analysis of the unmapped reads against Genbank, we find that many of the contigs assembled from the unmapped reads can be assigned to strain variants of the species in the mock community (Table 8). This suggests that the constructors of the mock community may have unintentionally included strain variants of *Fusobacterium nucleatum*, *Thermus thermophilus* HB27, and *Enterococcus faecalis*. In addition, we detect what may be portions of a novel member of the *Proteiniclasticum* genus in the assembly of these reads.

Without returning to the original DNA samples, it is impossible to conclusively confirm that unintended strains were used in the construction of the mock community. In particular, our analysis is dependent on the genomes in Genbank: the genomes we detect in the contigs are clearly more closely related to Genbank genomes other than the species in the reference metagenome, based on k-mer analysis and contig alignment. However, Genbank is unlikely to contain the exact genomes of the included strain variants, rendering conclusive identification impossible.

Conclusions

Overall, assembly of this mock community works well, with good recovery of known genomic sequence for the majority of genomes. All three assemblers that we evaluated recover similar amounts of most genomic sequence, but (recapitulating several other studies) MEGAHIT is computationally most

efficient.

The presence of closely related strains is a major confounder of metagenome assembly, and causes assemblers to drop considerable portions of genomes that (from a read and k-mer perspective) are present. In this relatively simple community, this strain confusion is clearly present but not dominant. However, real microbial communities are likely to have many closely related strains and any resulting loss of assembly will be hard to detect in the absence of good reference genomes. While high polymorphism rates in e.g. animal genomes is known to cause duplication or loss of assembly, some solutions have emerged that make use of assumptions of uniform coverage and diploidy. These solutions cannot however be transferred directly to metagenomes, which have unknown abundance distributions and numbers of strains.

An additional concern is that metagenome assemblies are often performed after pooling data sets to increase coverage; this pooled data is more likely to contain multiple strains, which would then in turn adversely affect assembly of strains. This may not be resolvable within the current paradigm of assembly, which focuses on outputting linear assemblies that cannot properly represent strain variation.

It is unclear how well long error-prone reads (such as those output by Pacific Biosciences SMRT and Oxford Nanopore instruments) will be able to resolve strain variants in metagenomes. High coverage of each individual genome is required to achieve accurate assembly, which may not be easily obtainable for complex communities. Fosmid, single-molecule barcoding, and HiC approaches may work better but these remain untested on well defined communities. (note phase genomics, 10x metagenomes papers.) (Mention Sharon@ paper and moleculo).

Author contributions

SA, LI and CTB developed, tested, and executed the analytical pipeline. SA and CTB created the tables and figures and wrote the paper.

Competing interests

No competing interest to our knowledge.

Grant information

This work is funded by Moore and NIH.

Acknowledgments

We thank Michael R. Crusoe and Phillip T. Brooks for input on analysis and pipeline development.

References

[1] Jorge F Vázquez-Castellanos, Rodrigo García-López, Vicente Pérez-Brocal, Miguel Pignatelli, and Andrés Moya. Comparison of different assembly and annotation tools on analysis of simulated viral metagenomic communities in the gut. *BMC genomics*, 15(1):1, 2014.

[2] Konstantinos Mavromatis, Natalia Ivanova, Kerrie Barry, Harris Shapiro, Eugene Goltsman, Alice C McHardy, Isidore Rigoutsos, Asaf Salamov, Frank Korzeniewski, Miriam Land, et al. Use of simulated data sets to evaluate the fidelity of metagenomic processing methods. *Nature methods*, 4(6):495–500, 2007.

[3] David B Jaffe, Jonathan Butler, Sante Gnerre, Evan Mauceli, Kerstin Lindblad-Toh, Jill P Mesirov, Michael C Zody, and Eric S Lander. Whole-genome sequence assembly for mammalian genomes: Arachne 2. *Genome research*, 13(1):91–96, 2003.

[4] Samuel Aparicio, Jarrod Chapman, Elia Stupka, Nik Putnam, Jer-ming Chia, Paramvir Dehal, Alan Christoffels, Sam Rash, Shawn Hoon, Arian Smit, et al. Whole-genome shotgun assembly and analysis of the genome of fugu rubripes. *Science*, 297(5585):1301–1310, 2002.

[5] Anveshi Charuvaka and Huzefa Rangwala. Evaluation of short read metagenomic assembly. *BMC genomics*, 12(2):1, 2011.

[6] Jared T Simpson, Kim Wong, Shaun D Jackman, Jacqueline E Schein, Steven JM Jones, and Inanç Birol. Abyss: a parallel assembler for short read sequence data. *Genome research*, 19(6):1117–1123, 2009.

[7] Yu Peng, Henry C.M. Leung, S.M. Yiu, and Francis Y.L. Chin. Idba-ud: a de novo assembler for single-cell and metagenomic sequencing data with highly uneven depth. *Bioinformatics*, 28:1420–1428, 2012.

[8] Anton Bankevich, Sergey Nurk, Dmitry Antipov, Alexey A. Gurevich, Mikhail Dvorkin, Alexander S. Kulikov, Valery M. Lesin, Sergey I. Nikolenko, Son

Pham, Andrey D. Prjibelski, Alexey V. Pyshkin, Alexander V. Sirotkin, Nikolay Vyahhi, Glenn Tesler, Max A. Alekseyev, and Pavel A. Pevzner. Spades: A new genome assembly algorithm and its applications to single-cell sequencing. *Journal of Computational Biology*, 19(5):455–477, 2012.

[9] Dinghua Li, Ruibang Luo, Chi-Man Liu, Chi-Ming Leung, Hing-Fung Ting, Kunihiko Sadakane, Hiroshi Yamashita, and Tak-Wah Lam. Megahit v1. 0: A fast and scalable metagenome assembler driven by advanced methodologies and community practices. *Methods*, 102:3–11, 2016.

[10] Shakya Migun, Christopher Quince, James Campbell, Zamin Yang, Christopher Schadt, and Mircea Podar. Comparative metagenomic and rrna microbial diversity characterization using archaeal and bacterial synthetic communities. *Enivromental Microbiology*, 15(6):1882–1899, 2013.

[11] H Chitsaz, JL Yee-Greenbaum, G Tesler, MJ Lombardo, CL Dupont, JH Badger, M Novotny, DB Rusch, LJ Fraser, NA Gormley, O Schulz-Trieglaff, GP Smith, DJ Evers, PA Pevzner, and RS Lasken. Efficient de novo assembly of single-cell bacterial genomes from short-read data sets. *Nat Biotechnol*, 29 (10):915–21, 2011.

[12] Anthony M. Bolger, Marc Lohse, and Bjoern Usadel. Trimmomatic: A flexible trimmer for illumina sequence data. *Bioinformatics*, 30(15):2114–2120, 2014.

[13] Matthew D MacManes. On the optimal trimming of high-throughput mrna sequence data. *Frontiers in genetics*, 5:13, 2014.

[14] Heng Li and Richard Durbin. Fast and accurate short read alignment with burrows–wheeler transform. *Bioinformatics*, 25(14):1754–1760, 2009.

[15] Heng Li, Bob Handsaker, Alec Wysoker, Tim Fennell, Jue Ruan, Nils Homer, Gabor Marth, Goncalo Abecasis, Richard Durbin, et al. The sequence alignment/map format and samtools. *Bioinformatics*, 25(16):2078–2079, 2009.

[16] Stefan Kurtz, Adam Phillippy, Arthur L Delcher, Michael Smoot, Martin Shumway, Corina Antonescu, and Steven L Salzberg. Versatile and open software for comparing large genomes. *Genome biology*, 5(2):1, 2004.